

Linear CCA has an accuracy that is lower than 60% on every test set, and so its results are not shown in the figure.

5 Conclusions and Future Work

This paper focuses on using KCCA to improve the accuracy of location estimation in wireless LANs. We have found that kernel transformation and CCA allow the construction of an accurate mapping between the physical space and the signal space. One advantage is the higher accuracy obtained in localization with much less calibration effort. We use a Gaussian kernel for the signal space to adapt to the noisy characteristic of radio-propagation channels, and a Matérn kernel for the physical space. The advantage of KCCA is shown through extensive experimental tests in a realistic environment. In this paper, we constructed the mapping without considering the additional information of user motion profiles. In the future, we plan to take the user-motion profiles into account, which may lead to higher accuracy with even less calibration effort. Likewise, we wish to experiment more with other environmental and contextual factors to boost the performance.

Acknowledgment

We would like to thank Hong Kong RGC for supporting this work under grant HKUST6187 / 04E. We would also like to thank Jie Yin and Xiaoyong Chai for their effort in collecting the data and discussing the research problems.

References

- [Bahl and Padmanabhan, 2000] P. Bahl and V.N. Padmanabhan. RADAR: An in-building RF-based user location and tracking system. In *Proceedings of the Conference on Computer Communications*, volume 2, pages 775–784, 2000.
- [Bhasker et al., 2004] E.S. Bhasker, S.W. Brown, and W.G. Griswold. Employing user feedback for fast, accurate, low-maintenance geolocation. In *IEEE International Conference on Pervasive Computing and Communications*, pages 111–120, Orlando, Florida, USA, March 2004.
- [Brunato and Battiti, 2004] M. Brunato and R. Battiti. Statistical learning theory for location fingerprinting in wireless LANs. *Computer Networks*, 2004.
- [Bui et al., 2002] H.H. Bui, S. Venkatesh, and G. West. Policy recognition in the abstract hidden Markov model. *Journal of Artificial Intelligence Research*, 17:451–499, 2002.
- [Chai and Yang, 2005] X. Chai and Q. Yang. Reducing calibration effort for location estimation using unlabeled samples. In *Proceedings of the Third IEEE International Conference on Pervasive Computing and Communications*, Kauai Island, Hawaii, USA, 2005.
- [Gentile and Berndt, 2004] C. Gentile and L.K. Berndt. Robust location using system dynamics and motion constraints. In *Proceedings of the IEEE International Conference on Communications*, pages 1360–1364, June 2004.
- [Hardoon et al., 2004] D.R. Hardoon, S. Szedmak, and J. Shawe-Taylor. Canonical correlation analysis; an overview with application to learning methods. *Neural Computation*, 16:2639–2664, 2004.
- [Hashemi, 1993] Homayoun Hashemi. The indoor radio propagation channel. *Proceedings of the IEEE*, 81(7):943–968, 1993.
- [Hotelling, 1936] H. Hotelling. Relations between two sets of variates. *Biometrika*, 28:312–377, 1936.
- [Kaemarungsi and Krishnamurthy, 2004] K. Kaemarungsi and P. Krishnamurthy. Modeling of indoor positioning systems based on location fingerprinting. In *Proceedings of the IEEE International Conference on Computer Communications*, volume 23, pages 1013–1023, 2004.
- [Ladd et al., 2002] A. Ladd, K. Bekris, G. Marceau, A. Rudys, L. Kavraki, and D. Wallach. Robotics-based location sensing using wireless ethernet. In *Proceedings of the Eighth ACM International Conference on Mobile Computing and Networking*, pages 227–238, Atlanta, Georgia, USA, September 2002.
- [Liao et al., 2004] L. Liao, D. Fox, and H. Kautz. Learning and inferring transportation routines. In *Proceedings of the Nineteenth National Conference on Artificial Intelligence*, pages 348–353, San Jose, CA, USA, July 2004.
- [Ni et al., 2003] L.M. Ni, Y. Liu, Y.C. Lau, and A.P. Patil. LANDMARC: Indoor location sensing using active RFID. In *Proceedings of the First IEEE International Conference on Pervasive Computing and Communications*, pages 407–416, Dallas, TX, USA, March 2003.
- [Roos et al., 2002a] T. Roos, P. Myllymäki, and H. Tirri. A statistical modeling approach to location estimation. *IEEE Transactions on Mobile Computing*, 1(1):59–69, 2002.
- [Roos et al., 2002b] T. Roos, P. Myllymäki, H. Tirri, P. Misikangas, and J. Sievanen. A probabilistic approach to WLAN user location estimation. *International Journal of Wireless Information Networks*, 9(3):155–164, 2002.
- [Schwaighofer et al., 2004] A. Schwaighofer, M. Grigoras, V. Tresp, and C. Hoffmann. GPPS: A Gaussian process positioning system for cellular networks. In S. Thrun, L. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems 16*, Cambridge, MA, 2004. MIT Press.
- [Smailagic and Kogan, 2002] A. Smailagic and D. Kogan. Location sensing and privacy in a context-aware computing environment. *IEEE Wireless Communications*, 9(5):10–17, October 2002.
- [Yin et al., 2005] J. Yin, Q. Yang, and L. Ni. Adaptive temporal radio maps for indoor location estimation. In *Proceedings of the 3rd Annual IEEE International Conference on Pervasive Computing and Communications*, Kauai Island, Hawaii, USA, March 2005.
- [Youssef et al., 2003] M. Youssef, A. Agrawala, and U. Shankar. WLAN location determination via clustering and probability distributions. In *Proceedings of the First IEEE International Conference on Pervasive Computing and Communications*, pages 143–150, Fort Worth, Texas, USA, March 2003.

Algebraic Markov Decision Processes

Patrice Perny

LIP6 - Université Paris 6
4 Place Jussieu
75252 Paris Cedex 05, France
patrice.perny@lip6.fr

Olivier Spanjaard

LIP6 - Université Paris 6
4 Place Jussieu
75252 Paris Cedex 05, France
olivier.spanjaard@lip6.fr

Paul Weng

LIP6 - Université Paris 6
4 Place Jussieu
75252 Paris Cedex 05, France
paul.weng@lip6.fr

Abstract

In this paper, we provide an algebraic approach to Markov Decision Processes (MDPs), which allows a unified treatment of MDPs and includes many existing models (quantitative or qualitative) as particular cases. In algebraic MDPs, rewards are expressed in a semiring structure, uncertainty is represented by a decomposable plausibility measure valued on a second semiring structure, and preferences over policies are represented by Generalized Expected Utility. We recast the problem of finding an optimal policy at a finite horizon as an algebraic path problem in a decision rule graph where arcs are valued by functions, which justifies the use of the Jacobi algorithm to solve algebraic Bellman equations. In order to show the potential of this general approach, we exhibit new variations of MDPs, admitting complete or partial preference structures, as well as probabilistic or possibilistic representation of uncertainty.

1 Introduction

In the field of planning under uncertainty, the theory of Markov Decision Processes (MDPs) has received much attention as a natural framework both for modeling and solving complex structured decision problems, see e.g. [Dean *et al.*, 1993; Kaelbling *et al.*, 1999]. In the standard MDP approach, the utilities of actions are given by scalar rewards supposed to be additive, uncertainty in the states of the world and in the consequences of the actions are represented with probabilities, and policies are evaluated using the expected utility (EU) model. Although these choices are natural in various practical situations, many other options are worth investigating for several reasons :

- *Rewards of actions are not necessarily scalar nor additive.* In multi-agent planning or in multicriteria MDPs, the utility of any action is given by a vector of rewards (one per agent or criterion) and actions are compared according to Pareto dominance [Wakuta, 1995]. In qualitative frameworks, rewards are valued on an ordinal scale and therefore are not additive. The sum is then replaced by the min, max or any refinement of them, depending on the context.

- *Non-probabilistic representation of uncertainty might be of interest.* In practice, it is sometimes difficult to quantify precisely the plausibility of states and consequences of actions. Assessing probabilities in such situations seems difficult. For this reason, alternative approaches based on qualitative representations of uncertainty have been proposed (see e.g. [Darwiche and Ginsberg, 1992; Dubois and Prade, 1995; Wilson, 1995]) and might be used in the context of MDPs.

- *Non-EU theories offer also interesting descriptive possibilities.* Despite the appeal of the expected utility model and its theoretical foundations [von Neumann and Morgenstern, 1947; Savage, 1954], recent developments of decision theory have shown the descriptive potential of alternative representations of preferences under uncertainty. For example, the rank dependent expected utility theory (RDEU) is a sophistication of EU theory using probability transforms to better account for actual decision making behaviors under risk [Quiggin, 1993]. Besides this extension, various alternatives to EU have been proposed for decision making in a non-probabilistic setting. Among them, let us mention the qualitative expected utility (QEU) theories proposed in [Dubois and Prade, 1995] and [Giang and Shenoy, 2001], and very recently, the generalized expected utility theory (GEU) proposed in [Chu and Halpern, 2003a; 2003b] which generalizes the notion of expectation for general plausibility measures.

Despite the diversity of models proposed for decision making under uncertainty, very few of them are used in the context of dynamic decision making. This gap can be explained by the inconsistencies entailed by the use of non-EU criteria (typically RDEU) in the dynamic context [Machina, 1989; Sarin and Wakker, 1998]. Indeed, when using a given non-linear utility criterion at each decision stage, the Bellman principle is generally violated, so that backward induction is likely to generate a dominated policy; further, there is in general no operational way to determine an optimal policy. This simple statement largely explains the predominance of the EU model in dynamic decision making under risk.

In the last decade however, some alternative models to EU have been proved to be dynamically consistent, thus providing new possibilities for sequential decision making. This is the case of qualitative expected utility theory [Dubois and Prade, 1995] that has led to a possibilistic counterpart of MDPs [Sabbadin, 1999] with efficient algorithms adapted

from backward induction and value iteration, substituting operations $(+, \times)$ by $(\max, \min)$ in computations. In the same vein, [Littman and Szepesvári, 1996] propose a generalized version of MDPs where $\max$ and $+$ are substituted by abstract operators in Bellman equations, and [Bonet and Pearl, 2002] propose a qualitative version of MDPs. Besides these positive results, very few alternatives to standard MDPs have been investigated.

Among the diversity of possible choices for defining a reward system, a plausibility measure over events and a preference over lotteries on rewards, we need to know which combination of them can soundly be used in the context of MDPs and which algorithm should be implemented to determine an optimal policy. For this reason, we propose in this paper an algebraic generalization of the standard setting, relying on the definition of a semiring structure on rewards, a semiring structure on plausibilities of events, and a generalized expectation model as decision criterion [Chu and Halpern, 2003a]. The generalization power of semirings have been already demonstrated in AI by [Bistarelli *et al.*, 1995] in the context of constraint satisfaction problems. Within this general setting, our aim is to present a unified treatment of MDPs and to provide algorithmic solutions based on the general Jacobi algorithm (initially introduced to solve systems of linear equations).

The paper is organized as follows: in Section 2, we show by example how to recast an MDP as an optimal path problem in a decision graph. Then we introduce an algebraic framework for MDPs, relying on the definition of algebraic structures on rewards, plausibilities and expectations (Section 3). In Section 4 we justify the use of the Jacobi algorithm as a general procedure to determine an optimal policy in an Algebraic MDP (AMDP). Finally, we consider in Section 5 some particular instances of AMDPs, including new probabilistic and possibilistic MDPs using partial preferences over rewards.

2 Decision Rule Graph in MDPs

We briefly recall the main characteristics of a Markov Decision Process (MDP) [Puterman, 1994]. It can be described as a tuple (S, A, T, R) where:

- $S = \{s^1, \dots, s^n\}$ is a finite set of states,
- $A = \{a^1, \dots, a^m\}$ is a finite set of actions,
- $T: S \times A \rightarrow \mathbf{Pr}(S)$ is a transition function, giving for each state and action, a probability distribution over states (in the sequel, we write $T(s, a, s')$ for $T(s, a)(s')$),
- $R: S \times A \rightarrow \mathbb{R}$ is a reward function giving the immediate reward for taking a given action in a given state.

A decision rule is a function from the set of states S to the set of actions A . There are $N = m^n$ available decision rules at each step. We write $\Delta = A^S = \{\delta^1, \dots, \delta^N\}$ the set of decision rules. A policy at step t (i.e., the t^{th} -to-last step) is a sequence of t decision rules. For a policy π and a decision rule δ , we note (δ, π) the policy which consists in applying first decision rule δ and then policy π .

A history is a realizable sequence of successive states and actions. The accumulated reward corresponding to a history $\gamma_t = (s_t, a_t, s_{t-1}, \dots, a_1, s_0)$ (with initial state s_t) is

	δ^1	δ^2	δ^3	δ^4
s^1	a^1	a^1	a^2	a^2
s^2	a^1	a^2	a^1	a^2

Table 1: Decision rules.

$R(\gamma_t) = \sum_{i=1}^t R(s_i, a_i)$. We denote $\Gamma_t(s)$ the set of t -step histories starting from s . For an initial state s , a t -step policy $\pi = (\delta_t, \dots, \delta_1)$ induces a probability distribution $\Pr_t^\pi(s, \cdot)$ over histories. The t -step value of being in state s and executing policy π is given by (expected accumulated reward):

$$v_t^\pi(s) = \sum_{\gamma \in \Gamma_t(s)} \Pr_t^\pi(s, \gamma) R(\gamma)$$

Denoting v_t^π the vector whose i^{th} component is $v_t^\pi(s^i)$, any two t -step policies π, π' can be compared using the componentwise dominance relation $\geq_{\mathbb{R}^n}$ defined by:

$$v_t^\pi \geq_{\mathbb{R}^n} v_t^{\pi'} \iff (\forall s \in S, v_t^\pi(s) \geq v_t^{\pi'}(s))$$

The t -step value of a policy of the form (δ^i, π) is defined recursively by $v_0^{(\delta^i, \pi)} = (0, \dots, 0)$ and $v_t^{(\delta^i, \pi)} = f^i(v_{t-1}^\pi)$, where $f^i: \mathbb{R}^n \rightarrow \mathbb{R}^n$ is the update function which associates to any vector $x = (x_1, \dots, x_n)$ the vector $(f_1^i(x), \dots, f_n^i(x))$ where $f_j^i(x) = R(s^j, \delta^i(s^j)) + \sum_{k=1}^n T(s^j, \delta^i(s^j), s^k) x_k$.

Example 1 Consider an MDP with $S = \{s^1, s^2\}$, $A = \{a^1, a^2\}$, $T(s^i, a^j, s^k) = 1$ if $i = j = k$ and 0.5 otherwise, $R(s^1, a^1) = 8$, $R(s^1, a^2) = 7$, $R(s^2, a^1) = 12$ and $R(s^2, a^2) = 11$. Thus, there are $N = 4$ available decision rules at each step (see Table 1). For instance, decision rule δ^2 consists in applying action a^1 in states s^1 and action a^2 in state s^2 . In this example, the functions f^i are given by:

$$\begin{aligned} f^1(x_1, x_2) &= (8 + x_1, 12 + 0.5x_1 + 0.5x_2) \\ f^2(x_1, x_2) &= (8 + x_1, 11 + x_2) \\ f^3(x_1, x_2) &= (7 + 0.5x_1 + 0.5x_2, 12 + 0.5x_1 + 0.5x_2) \\ f^4(x_1, x_2) &= (7 + 0.5x_1 + 0.5x_2, 11 + x_2) \end{aligned}$$

for all $(x_1, x_2) \in \mathbb{R}^2$.

Given a finite horizon H , the optimal policy π^* can be found thanks to the following Bellman equations:

$$\begin{aligned} v_0^{\pi^*} &= (0, \dots, 0) \\ v_t^{\pi^*} &= \max_{i=1 \dots N} f^i(v_{t-1}^{\pi^*}) \quad t = 1 \dots H \end{aligned} \quad (1)$$

The solution of these equations can be reduced to a vector-weighted optimal path problem in a particular graph, with update functions (the f^i 's) on the arcs allowing the propagation of policy values over nodes. Indeed, consider a graph where each node δ_t^i corresponds to decision rule δ^i at step t , and each arc of the form $(\delta_t^i, \delta_{t-1}^j)$ corresponds to a transition between decision rules. Moreover, nodes δ_1^j , $j = 1 \dots N$ are connected to a sink denoted 0, and a source denoted H is connected to nodes δ_H^j , $j = 1 \dots N$. Hence, any path from node δ_t^i to node 0 corresponds to a t -step policy where decision δ^i is applied first. We name that graph the *decision rule graph*. Note that the Bellman update via f^i 's is nicely separable (f_j^i 's can be computed independently) and therefore the vector value of a path can be obtained componentwise (state by state) as usual in classic MDP algorithms. This property will be exploited later on for Algebraic MDPs. Coming back to Example 1 and assuming that $H = 2$, there are 16 available

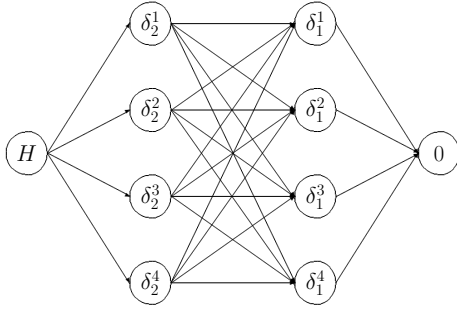


Figure 1: A decision rule graph.

policies (all possible combinations of two successive decision rules). The corresponding graph is pictured on Figure 1.

For all $i = 1 \dots n$, function f^i is associated to every arc issued from δ_t^i , $t = 1 \dots H$. Moreover, the identity function is assigned to every arc issued from H . Thus, the optimal policy can be computed thanks to backward induction on the decision rule graph, by propagating the value $(0, \dots, 0) \in \mathbb{R}^n$ from the sink 0 corresponding to the empty policy. In Example 1, backward induction leads to the labels indicated in Table 2. The optimal policy value can be recovered on node H . The optimal vector value is $(17, 23)$ and the optimal policy (recovered from bolded values in Table 2) is (δ_2^4, δ_1^1) .

t	δ_t^1	δ_t^2	δ_t^3	δ_t^4
1	(8,12)	(8,11)	(7,12)	(7,11)
2	(16,22)	(16,23)	(17,22)	(17,23)

Table 2: Labels obtained during backward induction.

In the next section, we show how to generalize this approach to a wide range of MDPs. In this concern, we introduce the notion of algebraic Markov decision process.

3 Algebraic Markov Decision Process

We now define a more general setting to model rewards and uncertainty in MDPs. Our approach relies on previous works aiming at generalizing uncertainty measurement and expectation calculus. Our rewards take values in a set V and we use plausibility measures¹ [Friedman and Halpern, 1995] to model uncertainty. A plausibility measure Pl is here a function from 2^W (the set of events) to P , where W is the set of worlds, P is a set endowed with two internal operators $\oplus_P$ and $\otimes_P$ (the analogs of $+$ and $\times$ in probability theory), a (possibly partial) order relation $\succeq_P$, and two special elements $\mathbf{0}_P$ and $\mathbf{1}_P$ such that $\mathbf{1}_P \succeq_P p \succeq_P \mathbf{0}_P$ for all $p \in P$. Furthermore, Pl verifies $\text{Pl}(\emptyset) = \mathbf{0}_P$, $\text{Pl}(W) = \mathbf{1}_P$ and $\text{Pl}(X) \succeq_P \text{Pl}(Y)$ for all X, Y such that $Y \subseteq X \subseteq W$. We assume here that Pl is decomposable, i.e. $\text{Pl}(X \cup Y) = \text{Pl}(X) \oplus_P \text{Pl}(Y)$ for any pair of disjoint events X and Y . To combine plausibilities and rewards, we use the generalized expectation proposed by [Chu and Halpern, 2003a]. This generalized expectation is defined on an expectation domain $(V, P, \boxplus, \boxtimes)$ ²

¹This notion must not be confused with the Dempster-Shafer notion of plausibility function.

²In [Chu and Halpern, 2003a], an expectation domain is written $(U, P, V, \boxplus, \boxtimes)$. This structure can be simplified here since $V = U$.

where $\boxplus : V \times V \rightarrow V$ and $\boxtimes : P \times V \rightarrow V$ are the counterparts of $+$ and $\times$ in probabilistic expectation, and the three following requirements are satisfied: $(x \boxplus y) \boxplus z = x \boxplus (y \boxplus z)$, $x \boxplus y = y \boxplus x$, $\mathbf{1}_P \boxtimes x = x$. For any plausibility distribution Pl on V having its support in X , the generalized expectation writes: $\sum_{x \in X} \text{Pl}(x) \boxtimes x$.

An Algebraic MDP (AMDP) is described as a tuple (S, A, T, R) , where T and R are redefined as follows:

- $T : S \times A \rightarrow \text{Pl}(S)$ is a transition function, where $\text{Pl}(S)$ is the set of plausibility measures over S valued in P ,
- $R : S \times A \rightarrow V$ is a reward function giving the immediate reward in V .

Consistently with the standard Markov hypothesis, the next state and the expected reward depend only on the current state and the action taken. In particular, plausibility distributions of type $T(s, a)$ are (plausibilistically) independent of the past states and actions. This plausibilistic independence refers to the notion introduced by [Friedman and Halpern, 1995] and leads to the following algebraic counterpart of the probabilistic independence property: $\text{Pl}(X \cap Y) = \text{Pl}(X) \otimes_P \text{Pl}(Y)$ for any pair X, Y of independent events.

In this setting, rewards and plausibilities take values in two semirings. Roughly speaking, a semiring is a set endowed with two operators allowing the combination of elements (rewards or plausibilities) together. We now recall some definitions about semirings.

Definition 1 A semiring $(X, \oplus, \otimes, \mathbf{0}, \mathbf{1})$ is a set X with two binary operations $\oplus$ and $\otimes$, such that:

- (A₁) $(X, \oplus, \mathbf{0})$ is a commutative semigroup with $\mathbf{0}$ as neutral element (i.e., $a \oplus b = b \oplus a$, $(a \oplus b) \oplus c = a \oplus (b \oplus c)$, $a \oplus \mathbf{0} = a$).
- (A₂) $(X, \otimes, \mathbf{1})$ is a semigroup with $\mathbf{1}$ as neutral element, and for which $\mathbf{0}$ is an absorbing element (i.e., $(a \otimes b) \otimes c = a \otimes (b \otimes c)$, $a \otimes \mathbf{1} = \mathbf{1} \otimes a = a$, $a \otimes \mathbf{0} = \mathbf{0} \otimes a = \mathbf{0}$).
- (A₃) $\otimes$ is distributive with respect to $\oplus$ (i.e., $(a \oplus b) \otimes c = (a \otimes c) \oplus (b \otimes c)$, $a \otimes (b \oplus c) = (a \otimes b) \oplus (a \otimes c)$).

A semiring is said to be *idempotent* when $(X, \oplus)$ is an idempotent commutative semigroup (i.e., a commutative semigroup such that $a \oplus a = a$). The idempotence of $\oplus$ enables to define the following canonical order relation $\succeq_X$:

$$a \succeq_X b \iff a \oplus b = a \quad \forall a, b \in X$$

From now on, we will assume that the rewards are elements of an idempotent semiring $(V, \oplus_V, \otimes_V, \mathbf{0}_V, \mathbf{1}_V)$. Operator $\oplus_V$ is used to select the optimal values in V , whereas operator $\otimes_V$ is used to combine rewards. In classic MDPs with the total reward criterion, $\oplus_V = \max$ and $\otimes_V = +$.

Moreover the structure $(P, \oplus_P, \otimes_P, \mathbf{0}_P, \mathbf{1}_P)$ is also supposed to be a semiring. Operator $\oplus_P$ allows to combine the plausibilities of disjoint events and operator $\otimes_P$ allows to combine the plausibilities of independent events. Note that the assumption that $(P, \oplus_P, \otimes_P, \mathbf{0}_P, \mathbf{1}_P)$ is a semiring is not very restrictive since [Darwiche and Ginsberg, 1992], who use similar properties to define symbolic probability, have shown that it subsumes many representations of uncertainty, such as probability theory, possibility theory and other important calculi used in AI.

Now that the general framework has been defined, we can follow the usual approach in MDPs and define a value function for policies. The accumulated reward for a history $\gamma = (s_t, a_t, s_{t-1}, \dots, a_1, s_0)$ is $R(\gamma) = \bigotimes_{i=1}^t R(s_i, a_i)$. For an initial state s , a t -step policy $\pi = (\delta_t, \dots, \delta_1)$ induces a plausibility measure $\text{Pl}_t^\pi(s, \cdot)$ over histories. Such a policy will be evaluated with respect to the generalized expectation of accumulated reward, which writes:

$$v_t^\pi(s) = \sum_{\gamma \in \Gamma_t(s)} \text{Pl}_t^\pi(s, \gamma) \boxtimes R(\gamma)$$

The policies can be compared with respect to the componentwise dominance relation $\succeq_{V^n}$ between vectors in V^n :

$$x \succeq_{V^n} y \iff (\forall i = 1, \dots, n, x_i \succeq_V y_i)$$

for all $x = (x_1, \dots, x_n), y = (y_1, \dots, y_n) \in V^n$.

Most of the MDPs introduced previously in the literature are instances of our algebraic MDP:

- In standard MDPs, the underlying algebraic structure on P is $(P, \oplus_P, \otimes_P, \mathbf{0}_P, \mathbf{1}_P) = ([0, 1], +, \times, 0, 1)$, and operators $\boxplus = +$ and $\boxtimes = \times$ are used to define the classic expectation operation. When rewards are defined on $(V, \oplus_V, \otimes_V, \mathbf{0}_V, \mathbf{1}_V) = (\mathbb{R}, \max, +, -\infty, 0)$ where $\mathbb{R} = \mathbb{R} \cup \{-\infty\}$, we recognize the total reward criterion. With $(\mathbb{R}, \max, +_\gamma, -\infty, 0)$ (where $x +_\gamma y = x + \gamma y$), we recognize the weighted total reward criterion. With $(\mathbb{R}, \max, +_H, -\infty, 0)$ where $a +_H b = \frac{1}{H}a + b$, we recognize the average reward criterion assuming that there is an initial dummy step with zero reward.

- Qualitative MDPs, introduced in [Bonet and Pearl, 2002], are AMDPs where the rewards and the plausibility measures are defined on the semiring of two-sided infinite formal series, which is a subset of the extended reals [Wilson, 1995].

In order to assign functions to the arcs in the decision rule graph, we first define $f_j^i : V^n \rightarrow V$ (the update function after applying decision rule δ^i in state s^j), for all $x \in V^n$, by:

$$f_j^i(x) = R(s^j, \delta^i(s^j)) \otimes_V \left(\sum_{i=1..n}^{\boxplus} T(s^j, \delta^i(s^j), s^i) \boxtimes x_i \right).$$

Then, for any decision rule δ^i , we define the update function $f^i : V^n \rightarrow V^n$ which associates to any vector $x \in V^n$ the vector $(f_1^i(x), \dots, f_n^i(x))$.

Dynamic consistency in AMDPs is guaranteed by specific properties on functions f^i . The fulfillment of these properties strongly relies on the following conditions:

$$(C_1) p \boxtimes (x \oplus_V y) = (p \boxtimes x) \oplus_V (p \boxtimes y)$$

$$(C_2) x \boxplus (y \oplus_V z) = (x \boxplus y) \oplus_V (x \boxplus z)$$

$$(C_3) p \boxtimes (q \boxtimes x) = (p \otimes_P q) \boxtimes x$$

$$(C_4) \sum_i^{\boxplus} p_i \boxtimes (x \otimes_V y_i) = x \otimes_V \left(\sum_i^{\boxplus} p_i \boxtimes y_i \right)$$

$$(C_5) p \boxtimes (x \boxplus y) = (p \boxtimes x) \boxplus (p \boxtimes y)$$

for all $p, q, p_i \in P, x, y, z, y_i \in V$.

Conditions (C_1) and (C_2) are two distributivity properties entailing a kind of additivity of $\succeq_V$ w.r.t. $\boxtimes$ and $\boxplus$ (i.e., $x \succeq_V y \Rightarrow (z * x \succeq_V z * y)$ for $* \in \{\boxtimes, \boxplus\}$). Condition (C_3) enables the reduction of lotteries. Condition (C_4) enables to isolate a sure reward in a lottery and is similar to the distributivity axiom used in [Luce, 2003]. Condition (C_5) is a distributivity condition as in classic expectation. We now establish a monotonicity result that will be used later (Prop.

3) to justify a dynamic programming approach.

Proposition 1 *If (C_1) and (C_2) hold, then f^i is non-decreasing for all $\delta^i \in \Delta$, i.e*

$$\forall x, y \in V^n, (x \succeq_{V^n} y \Rightarrow f^i(x) \succeq_{V^n} f^i(y))$$

Proof. Let x, y in V^n s.t. $x \succeq_{V^n} y$. For $s_j \in S$, we have $\sum_k^{\boxplus} T(s^j, \delta^i(s^j), s^k) \boxtimes x_k \succeq_V \sum_k^{\boxplus} T(s^j, \delta^i(s^j), s^k) \boxtimes y_k$ by (C_1) and (C_2) . Thanks to distributivity of $\otimes_V$ over $\oplus_V$, we have $f_j^i(x) \succeq_V f_j^i(y)$. Therefore, $f^i(x) \succeq_{V^n} f^i(y)$. ■

Moreover, the value of a policy π can be computed recursively thanks to the following result:

Proposition 2 *Let $\pi = (\delta^i, \pi')$ a $(t+1)$ -step policy, and assume that $v_0^\pi = (\mathbf{1}_V, \dots, \mathbf{1}_V)$. If (C_3) , (C_4) and (C_5) hold, then $v_{t+1}^\pi = f^i(v_t^{\pi'})$ for all $t \geq 0$.*

Proof. Let s be a state and a denote $\delta^i(s)$. We note $\gamma_t = (s_t, a_t, \dots, a_1, s_0)$ and $\gamma_{t+1} = (s, a, s_t, a_t, \dots, a_1, s_0)$. We have:

$$\begin{aligned} v_{t+1}^\pi(s) &= \sum_{\gamma_{t+1} \in \Gamma_{t+1}(s)}^{\boxplus} \text{Pl}_{t+1}^\pi(s, \gamma_{t+1}) \boxtimes R(\gamma_{t+1}) \\ &= \sum_{s' \in S}^{\boxplus} \sum_{\gamma_t \in \Gamma_t(s')}^{\boxplus} (T(s, a, s') \otimes_P \text{Pl}_t^{\pi'}(s', \gamma_t)) \boxtimes (R(s, a) \otimes_V R(\gamma_t)) \\ &= R(s, a) \otimes_V \sum_{s' \in S}^{\boxplus} \sum_{\gamma_t \in \Gamma_t(s')}^{\boxplus} (T(s, a, s') \otimes_P \text{Pl}_t^{\pi'}(s', \gamma_t)) \boxtimes R(\gamma_t) \end{aligned}$$

by (C_4)

$$= R(s, a) \otimes_V \sum_{s' \in S}^{\boxplus} T(s, a, s') \boxtimes \left(\sum_{\gamma_t \in \Gamma_t(s')}^{\boxplus} \text{Pl}_t^{\pi'}(s', \gamma_t) \boxtimes R(\gamma_t) \right)$$

by (C_3) and (C_5)

$$= R(s, a) \otimes_V \sum_{s' \in S}^{\boxplus} T(s, a, s') \boxtimes v_t^{\pi'}(s')$$

Therefore, for all $s^j \in S$, $v_{t+1}^\pi(s^j) = f_j^i(v_t^{\pi'})$. ■

In order to establish the algebraic version of Bellman equations, we define the subset of maximal elements of a set with respect to an order relation $\succeq$ as:

$$\forall Y \subseteq X, M(Y, \succeq) = \{y \in Y : \forall z \in Y, \text{not}(z \succ y)\}$$

Furthermore, we denote $\mathcal{P}^*(X, \succeq)$ the set $\{Y \subseteq X : Y = M(Y, \succeq)\}$. When there is no ambiguity, these sets will be denoted respectively $M(Y)$ and $\mathcal{P}^*(X)$. Besides, for any function $f : V^n \rightarrow V^n$, for all $X \in \mathcal{P}^*(V^n)$, $f(X)$ denote $\{f(x) : x \in X\}$.

We define the semiring $(\mathcal{P}^*(V^n), \oplus, \otimes, \mathbf{0}, \mathbf{1})$ where $\mathbf{0} = \{\mathbf{0}_V, \dots, \mathbf{0}_V\}$, $\mathbf{1} = \{\mathbf{1}_V, \dots, \mathbf{1}_V\}$, and for all $X, Y \in \mathcal{P}^*(V^n)$:

$$X \oplus Y = M(X \cup Y)$$

$$X \otimes Y = M(\{x \otimes_V y : x \in X, y \in Y\})$$

with $x \otimes_V y = (x_1 \otimes_V y_1, \dots, x_n \otimes_V y_n)$. Hence, the algebraic generalization of Bellman equations (1) writes:

$$\begin{aligned} V_0^{\pi^*} &= \mathbf{1} \\ V_t^{\pi^*} &= \bigoplus_{i=1}^N f_M^i(V_{t-1}^{\pi^*}) \quad t = 1 \dots H \end{aligned}$$

where $f_M^i : \mathcal{P}^*(V^n) \rightarrow \mathcal{P}^*(V^n)$ is defined by $f_M^i(X) = M(f^i(X))$ for all $X \in \mathcal{P}^*(V^n)$.

4 Generalized path algebra for AMDPs

Following the construction proposed in Section 2, a decision rule graph can be associated to an AMDP. Clearly, solving algebraic Bellman equations amounts to searching for optimal paths with respect to the canonical order associated to $\oplus$. We now show how to solve this problem.

Consider the set $\mathcal{F}$ of functions from $\mathcal{P}^*(V^n)$ to $\mathcal{P}^*(V^n)$ satisfying for all $f \in \mathcal{F}$, $X \in \mathcal{P}^*(V^n)$, $Y \in \mathcal{P}^*(V^n)$:

$$\begin{aligned} f(X \oplus Y) &= f(X) \oplus f(Y) \\ f(\mathbf{0}) &= \mathbf{0} \end{aligned}$$

The $\oplus$ operation on $\mathcal{P}^*(V^n)$ induces a $\oplus$ operation on $\mathcal{F}$ defined, for all $h, g \in \mathcal{F}$ and $X \in \mathcal{P}^*(V^n)$, by:

$$(h \oplus g)(X) = h(X) \oplus g(X)$$

Let $\circ$ denote the usual composition operation between functions, id the identity function, and (for simplicity) $\mathbf{0}$ the constant function everywhere $\mathbf{0}$. It has been shown by [Minoux, 1977] that the algebraic structure $(\mathcal{F}, \oplus, \circ, \mathbf{0}, id)$ is a semiring. We now prove that update functions f_M^i 's belong to $\mathcal{F}$.

Proposition 3 (C_1) and (C_2) imply $(\forall \delta^i \in \Delta, f_M^i \in \mathcal{F})$.

Proof. For any δ^i , we have $f_M^i(\mathbf{0}) = \mathbf{0}$ since $\sum_i^{\boxplus} p_i \boxtimes \mathbf{0}_V = \sum_i^{\boxplus} p_i \boxtimes (\mathbf{0}_V \otimes_V \mathbf{0}_V) = \mathbf{0}_V \otimes_V (\sum_i^{\boxplus} p_i \boxtimes \mathbf{0}_V) = \mathbf{0}_V$ by (C_4) and absorption. Now, we show that $f_M^i(X \oplus Y) = f_M^i(X) \oplus f_M^i(Y)$, $\forall X, Y \in \mathcal{P}^*(V^n)$. Consider X, Y in $\mathcal{P}^*(V^n)$. First, we have $f^i(M(X \cup Y)) \subseteq f^i(X \cup Y) (*)$. Second, we prove that $M(f^i(X \cup Y)) \subseteq f^i(M(X \cup Y)) (**)$. Let $z \in M(f^i(X \cup Y))$. Then it exists $w \in X \cup Y$ such that $f^i(w) = z$. If $w \in M(X \cup Y)$ then $z \in f^i(M(X \cup Y))$. If $w \notin M(X \cup Y)$ then it exists $w^* \in M(X \cup Y)$ such that $w^* \succ_V w$. As f^i is non-decreasing, we have $f^i(w^*) \succeq_V f^i(w)$. By assumption, $f^i(w) \in M(f^i(X \cup Y))$, therefore $f^i(w^*) = f^i(w)$. Finally, $z \in f^i(M(X \cup Y))$ since $z = f^i(w^*)$. By $(*)$ and $(**)$, we have $M(f^i(X \cup Y)) \subseteq f^i(M(X \cup Y)) \subseteq f^i(X \cup Y)$. Therefore $M(f^i(X \cup Y)) = M(f^i(M(X \cup Y)))$, which means, by definition, that $f_M^i(X \oplus Y) = f_M^i(X) \oplus f_M^i(Y)$. ■

Propositions 2 and 3 prove that the algebraic generalization of Jacobi algorithm solves algebraic Bellman Equations (3) and (4) [Minoux, 1977]. Thanks to the particular structure³ of the decision rule graph, the Jacobi algorithm takes the following simple form:

ALGEBRAIC JACOBI ALGORITHM

1. $V_0 \leftarrow \mathbf{1}; t \leftarrow 0$
2. $Q_t^i = \mathbf{0}, \forall t = 1 \dots H, i = 1 \dots N$
3. **repeat**
4. $t \leftarrow t + 1$
5. **for** $i = 1$ to N **do** $Q_t^i \leftarrow f_M^i(V_{t-1})$
6. $V_t \leftarrow \bigoplus_{i=1}^N Q_t^i$
7. **until** $t = H$

We recognize a standard optimal path algorithm on the decision rule graph valued with functions f_M^i . When the iteration has finished, V_t is the set of generalized expected

³The graph is layered; update functions labelling arcs issued from a same node are identical and invariant from a layer to another.

accumulated rewards at step t associated to optimal paths. The above algorithm is not efficient since lines 5 and 6 require to consider $N = m^n$ decision rules in the computation of the Bellman update. Actually, the complexity of the algorithm can be significantly improved. Indeed, remark that $\bigcup_{i=1}^N \{f_j^i\} = \bigcup_{k=1}^m \{g_j^k\}$, where $g_j^k(x) = R(s^j, a^k) \otimes_V \sum_{i=1 \dots n}^{\boxplus} T(s^j, a^k, s^i) \boxtimes x_i$, and that $\forall v \in V^n$, $\bigcup_{i=1}^N \{f^i(v)\} = \bigcup_{k=1}^m \{g_1^k(v)\} \times \dots \times \bigcup_{k=1}^m \{g_n^k(v)\}$. Since the maxima over a Cartesian product equals the Cartesian product of maxima over components, we have $\bigoplus_{i=1}^N f_M^i(V_{t-1}) = \bigoplus_{v \in V_{t-1}} (\bigoplus_{k=1}^m g_1^k(v) \times \dots \times \bigoplus_{k=1}^m g_n^k(v))$. Hence, lines 5 and 6 can be replaced by:

- 5.1. $V_t \leftarrow \emptyset$
- 5.2. **for** $v \in V_{t-1}$ **do**
- 5.3. **for** $j = 1$ to n **do**
- 5.4. **for** $k = 1$ to m **do** $q_{tj}^k \leftarrow g_j^k(v)$
- 6.1. $V_{tj} \leftarrow \bigoplus_{k=1}^m q_{tj}^k$
- 6.2. **endfor**
- 6.3. $V_t \leftarrow V_t \oplus (V_{t1} \times \dots \times V_{tn})$
- 6.4. **endfor**

This algebraic counterpart of backward induction runs in polynomial time when the reward scale is completely ordered and algebraic operators can be computed in $O(1)$.

5 Examples

To illustrate the potential of the algebraic approach for MDPs, we now consider decision models that have not yet been investigated in a dynamic setting.

• **Qualitative MDPs.** In decision under possibilistic uncertainty, [Giang and Shenoy, 2001] have recently studied a new qualitative decision model (binary possibilistic utility), allowing to handle weak information about uncertainty while improving the discrimination power of qualitative utility models introduced by [Dubois and Prade, 1995]. The latter have been investigated in sequential decision problems by [Sabadin, 1999]. We show here that the former can be exploited in sequential decision problems as well.

Possibilistic uncertainty is measured on a finite qualitative totally ordered set P , endowed with two operators $\vee$ and $\wedge$ (max and min respectively). We denote $\mathbf{0}_P$ (resp. $\mathbf{1}_P$) the least (resp. greatest) element in P . The structure $(P, \vee, \wedge, \mathbf{0}_P, \mathbf{1}_P)$ is a semiring.

In Giang and Shenoy's model, rewards are valued in an ordered scale $(U_P, \succeq)$ where $U_P = \{\langle \lambda, \mu \rangle : \lambda \in P, \mu \in P, \lambda \vee \mu = \mathbf{1}_P\}$ and $\langle \lambda, \mu \rangle \succeq \langle \lambda', \mu' \rangle \Leftrightarrow (\lambda \geq \lambda' \text{ and } \mu \leq \mu')$. The relevant semiring is here $(V, \oplus_V, \otimes_V, \mathbf{0}_V, \mathbf{1}_V)$ where:

$$\begin{aligned} V &= \{\langle \lambda, \mu \rangle : \lambda \in P, \mu \in P\} \\ \langle \alpha, \beta \rangle \oplus_V \langle \lambda, \mu \rangle &= \langle \alpha \vee \lambda, \beta \wedge \mu \rangle \\ \langle \alpha, \beta \rangle \otimes_V \langle \lambda, \mu \rangle &= \langle \alpha \wedge \lambda, \beta \vee \mu \rangle \\ \mathbf{0}_V &= \langle \mathbf{0}_P, \mathbf{1}_P \rangle, \quad \mathbf{1}_V = \langle \mathbf{1}_P, \mathbf{0}_P \rangle \end{aligned}$$

Note that $\oplus_V$ and $\otimes_V$ are operators max and min on U_P .

The binary possibilistic utility model is a generalized expectation with operators $\boxplus$ and $\boxtimes$ taken as componentwise $\vee$ and $\wedge$ respectively. Note that this criterion takes values in U_P . Thanks to distributivity of $\vee$ (resp. $\wedge$) over $\wedge$ (resp. $\vee$), conditions (C_1) to (C_5) hold, which proves that algebraic

backward induction yields the value of optimal policies.

• **Multicriteria MDPs.** In planning problems, different aspects (time, energy, distance...) enter into the assessment of the utility of an action, and often these aspects cannot be reduced to a single scalar reward. This shows the practical interest of investigating MDPs with multicriteria additive rewards and multicriteria comparison models. As an example, consider the following multicriteria decision model. Let Q denote the set of criteria and $\succeq_Q$ a (possibly partial) order relation over Q (reflecting the importance of criteria). Following [Grosf, 1991] and [Junker, 2002], we denote $\succ_G$ the following strict order relation between vectors:

$$x \succ_G y \Leftrightarrow \begin{cases} \exists i = 1 \dots |Q|, x_i \neq y_i \\ \forall i : x_i \neq y_i, ((x_i > y_i) \text{ or } (\exists j \succ_Q i, x_j > y_j)) \end{cases}$$

The interest of such a definition is to unify in a single model both the lexicographic order (when $\succeq_Q$ is a linear order) and the Pareto dominance order (when $\succeq_Q = \emptyset$).

In this case, the relevant semiring structure on rewards is defined by (neutral elements are omitted here):

$$\begin{aligned} V &= \mathcal{P}^*(\mathbb{R}^{|Q|}) \\ X \oplus_V Y &= M(X \cup Y, \succ_G) \\ X \otimes_V Y &= M(\{x_i + y_i : x \in X, y \in Y\}, \succ_G) \end{aligned}$$

Hence, such a reward system can easily be inserted in the classical probabilistic setting. When the generalized expectation is chosen as the componentwise (usual) expectation, conditions (C_1) to (C_5) hold, which proves that algebraic backward induction yields the value of optimal policies.

6 Conclusion

We have introduced a general approach for defining solvable MDPs in various contexts. The interest of this approach is to factorize many different positive results concerning various rewards systems, uncertainty and decision models. Once the structure on rewards, the representation of uncertainty and the decision criteria have been chosen, it is sufficient to check that we have two semirings on V and P and that conditions (C_1) through (C_5) are fulfilled to justify the use of an algorithm “à la Jacobi” to solve the problem. It is likely that this result generalizes to the infinite horizon case, provided a suitable topology is defined on the policy valuation space.

Remark that, despite its generality, our framework does not include all interesting decision theories under uncertainty. For instance, in a probabilistic setting, the RDEU model (as well as other Choquet integrals) cannot be expressed under the form of the generalized expectation used in the paper. Actually, RDEU is known as a dynamically inconsistent model [Machina, 1989; Sarin and Wakker, 1998] and it is unlikely that constructive algorithms like backward induction are appropriate. A further specific study for this class of decision models might be of interest.

References

[Bistarelli et al., 1995] S. Bistarelli, U. Montanari, and F. Rossi. Constraint solving over semirings. In *Proc. of IJCAI*, 1995.

[Bonet and Pearl, 2002] B. Bonet and J. Pearl. Qualitative MDPs and POMDPs: An order-of-magnitude approximation. In *Proc. of the 18th UAI*, pages 61–68, 2002.

[Chu and Halpern, 2003a] F.C. Chu and J.Y. Halpern. Great expectations. part I: On the customizability of generalized expected utility. In *Proc. of the 18th IJCAI*, pages 291–296, 2003.

[Chu and Halpern, 2003b] F.C. Chu and J.Y. Halpern. Great expectations. part II: Generalized expected utility as a universal decision rule. In *Proc. of the 18th IJCAI*, pages 297–302, 2003.

[Darwiche and Ginsberg, 1992] A. Darwiche and M.L. Ginsberg. A symbolic generalization of probability theory. In *Proc. of the 10th AAAI*, pages 622–627, 1992.

[Dean et al., 1993] T. Dean, L.P. Kaelbling, J. Kirman, and A. Nicholson. Planning with deadlines in stochastic domains. In *Proc. of the 11th AAAI*, pages 574–579, 1993.

[Dubois and Prade, 1995] D. Dubois and H. Prade. Possibility theory as a basis of qualitative decision theory. In *Proc. of the 14th IJCAI*, pages 1925–1930, 1995.

[Friedman and Halpern, 1995] N. Friedman and J. Halpern. Plausibility measures: A user’s guide. In *Proc. of the 11th UAI*, pages 175–184, 1995.

[Giang and Shenoy, 2001] P.H. Giang and P.P. Shenoy. A comparison of axiomatic approaches to qualitative decision making using possibility theory. In *Proc. of the 17th UAI*, pages 162–170, 2001.

[Grosf, 1991] B. Grosf. Generalizing prioritization. In *Proc. of the 2nd KR*, pages 289–300, 1991.

[Junker, 2002] U. Junker. Preference-based search and multicriteria optimization. In *Proc. of AAAI*, pages 34–40, 2002.

[Kaelbling et al., 1999] L.P. Kaelbling, M. Littman, and A. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101:99–134, 1999.

[Littman and Szepesvári, 1996] M.L. Littman and C. Szepesvári. A generalized reinforcement-learning model: Convergence and applications. In *Proc. of the 13th ICML*, pages 310–318, 1996.

[Luce, 2003] R.D. Luce. Increasing increment generalizations of rank-dependent theories. *Theory and Decision*, 55:87–146, 2003.

[Machina, 1989] M.J. Machina. Dynamic consistency and non-expected utility models of choice under uncertainty. *Journal of Economic Literature*, 27:1622–1668, 1989.

[Minoux, 1977] M. Minoux. Generalized path algebra. In *Surveys of Mathematical Programming*, pages 359–364. Publishing House of the Hungarian Academy of Sciences, 1977.

[von Neumann and Morgenstern, 1947] J. von Neumann and O. Morgenstern. *Theory of games and economic behavior*. 2nd Ed. Princeton University Press, 1947.

[Puterman, 1994] M.L. Puterman. *Markov Decision Processes - Discrete Stochastic Dynamic Programming*. Wiley & Sons, 1994.

[Quiggin, 1993] J. Quiggin. *Generalized expected utility theory: the rank-dependant model*. Kluwer Academic Publishers, 1993.

[Sabbadin, 1999] R. Sabbadin. A possibilistic model for qualitative sequential decision problems under uncertainty in partially observable environments. In *Proc. of UAI*, pages 567–574, 1999.

[Sarin and Wakker, 1998] R. Sarin and P. Wakker. Dynamic choice and non-EU. *J. of Risk and Uncertainty*, 17:87–119, 1998.

[Savage, 1954] L.J. Savage. *The Foundations of Statistics*. J. Wiley & Sons, 1954.

[Wakuta, 1995] K. Wakuta. Vector-valued markov decision processes and the systems of linear inequalities. *Stochastic Processes and their Applications*, 56:159–169, 1995.

[Wilson, 1995] N. Wilson. An order of magnitude calculus. In *Proc. of the 11th UAI*, pages 548–555, 1995.

Asymptotic Conditional Probability in Modal Logic: A Probabilistic Reconstruction of Nonmonotonic Logic

Riccardo Rosati

Dipartimento di Informatica e Sistemistica
Università di Roma “La Sapienza”
rosati@dis.uniroma1.it

Georg Gottlob

Institut für Informationssysteme
Technische Universität Wien
gottlob@dbai.tuwien.ac.at

Abstract

We analyze the asymptotic conditional validity of modal formulas, i.e., the probability that a formula ψ is valid in the finite Kripke structures in which a given modal formula φ is valid, when the size of these Kripke structures grows to infinity. We characterize the formulas ψ that are almost surely valid (i.e., with probability 1) in case φ is a flat, S5-consistent formula, and show that these formulas ψ are exactly those which follow from φ according to the nonmonotonic modal logic S5_G. Our results provide – for the first time – a probabilistic semantics to a well-known nonmonotonic modal logic, establishing a new bridge between nonmonotonic and probabilistic reasoning, and give a computational account of the asymptotic conditional validity problem in Kripke structures.

1 Introduction

Asymptotic Probabilities in Modal Logic. Asymptotic (or limit) probabilities of classical logic formulas have been investigated in various contexts [Glebskii *et al.*, 1969; Fagin, 1976; Compton, 1988; Kolaitis and Vardi, 1990; Le Bars, 1998]. Halpern and Kapron have analyzed asymptotic probability in modal logic [Halpern and Kapron, 1994], where, instead of relational structures, Kripke structures are considered, and where the size of a structure is measured in terms of the number of its worlds. Among various results,¹ they have shown that every modal formula is either almost surely true or almost surely false in finite Kripke structures. Thus, there is a 0-1 law for modal logic K, analogous to the 0-1 law for function-free first-order logic [Fagin, 1976; Glebskii *et al.*, 1969].

To give a simple example, if p is a propositional letter of a considered finite alphabet $\mathcal{A}$, then the modal formula² Mp is almost surely true (w.r.t. modal logic K). In fact, it can be seen that if $n \rightarrow \infty$, then the cardinality of the set A_n of Kripke structures for $\mathcal{A}$ of size $\leq n$ satisfying Mp (i.e., where at least one world satisfying p is reachable from

each world) grows much faster than the cardinality of the set B_n of Kripke structures which do not satisfy Mp , and thus $\lim_{n \rightarrow \infty} |A_n|/|C_n| = 1$ where C_n is the set of *all* Kripke structures with n worlds over the considered alphabet.

Assigning asymptotic probabilities to modal formulas provides an interesting nonstandard semantics to modal logics and has important connections to philosophy and to artificial intelligence:

PHILOSOPHY: It has been observed [Halpern and Kapron, 1994; Gottlob, 1999] that the modal formulas that are almost surely true in all Kripke models are exactly those formulas which are valid in Carnap’s modal logic, exposed in his well-known foundational treatise *Meaning and Necessity* [Carnap, 1947]. Carnap argued that precisely these formulas are those to be considered logically true (*L-true*). The same logic has since be considered by various philosophers and logicians [Gottlob, 1999].

ARTIFICIAL INTELLIGENCE: Various *nonmonotonic modal logics* have been defined in the literature. Examples are autoepistemic logic [Moore, 1985], nonmonotonic logics K, S4, etc. [Marek and Truszczyński, 1993], the logic MBNF [Lifschitz, 1994] and the logic of minimal knowledge S5_G [Halpern and Moses, 1985]. In these logics, the modal operator K is interpreted as an epistemic operator of knowledge or of belief. As observed in [Gottlob, 1999], the formulas that can be derived from an empty set of premises in *all these logics* precisely coincide with those formulas which are almost surely true in all Kripke structures. This set of formulas is furthermore identical to the *stable set* [Stalnaker, 1993] based on the empty set of formulas, thus, in a sense, to the “absolute” stable set, containing only those formulas that are epistemic consequences of the empty theory, i.e., that can be assumed in the case of total factual ignorance (for a definition of stable sets, see Section 4). We thus retain that in nonmonotonic and epistemic logics, in absence of further knowledge, a modal formula is considered true if and only if this formula is almost surely valid.

Reasoning and Asymptotic Conditional Truth. While almost sure validity provides an appealing probabilistic semantics of truth in the case of total factual ignorance, this does not yet allow us to *reason* on the basis of *premises*, which is the the most important goal of all logical formalisms, and, in particular, of nonmonotonic and epistemic logics. In the setting of limit probabilities, the inference $\varphi \models \psi$ of a (pos-

¹Some of the results of [Halpern and Kapron, 1994] have been corrected in [Le Bars, 2002], but those referred-to in the present paper are perfectly correct in [Halpern and Kapron, 1994].

²We here use K as the symbol for the necessity (= knowledge) operator and $M (\equiv \neg K \neg)$ for the possibility operator.

sibly modal) formula ψ from a knowledge base (theory) φ would most intuitively correspond to the statement that the conditional probability $P(\psi|\varphi)$ is asymptotically equal to 1. Assuming a uniform probability distribution of Kripke structures, this means that for the sets A_n and B_n of Kripke models of size n satisfying $\psi \wedge \varphi$ and φ , respectively, the limit $\lim_{n \rightarrow \infty} |A_n|/|B_n|$ exists and is equal to 1.

There is, in general, no 0-1 law for conditional probabilities in our setting, which immediately follows from previous results (see e.g. [Grove *et al.*, 1996b]) and from the fact that modal logic corresponds to a fragment of function-free first-order logic. The existence and determination of conditional probabilities in first and higher order logics has been the subject of several studies [Grove *et al.*, 1996b; 1996a]. However, to our best knowledge, conditional probabilities for modal logics have never been studied.

Observe that reasoning via conditional limit probabilities as explained above clearly constitutes a form of nonmonotonic reasoning. For example, if p is a propositional letter, then $\models_{\text{lim}} Mp$ but $K\neg p \not\models_{\text{lim}} Mp$, thus, adding a premise may invalidate a consequence (here $\models_{\text{lim}}$ denotes inference of almost sure formulas under conditional limit probabilities). The nonmonotonic behaviour of conditional inference has been pointed out in the context of FO and higher order classical logic (see e.g. [Grove *et al.*, 1996b]). We deem the context of modal logic particularly interesting, because most nonmonotonic logics that have been defined are modal logics. It would thus be very interesting to know how the nonmonotonic modal logic obtained from conditional limit probabilities of Kripke structures relates to well-known and well-studied nonmonotonic modal logics.

Nonmonotonic Modal Logics. For every “classical” monotonic system of modal logic S , a nonmonotonic version NM- S is obtained by the following definition: A set E of modal formulas is an NM- S -expansion of a knowledge base φ iff $E = Cn_S(\varphi \cup \{\neg K\gamma \mid \gamma \in \mathcal{L}_K(\mathcal{A}) - E\})$, where Cn_S is the consequence operator according to modal logic S and $\mathcal{L}_K(\mathcal{A})$ is the underlying modal language, i.e., the set of all possible modal formulas over the alphabet of propositions $\mathcal{A}$ ³. In particular, Moore’s autoepistemic logic corresponds to NM-KD45. For more background, consult [Marek and Truszczyński, 1993].

It was often criticized that the above fixed-point equation is somewhat too liberal, because it allows a theory to have expansions that are not sufficiently “grounded” in the premises and contain positive knowledge that a rational agent should never conclude from the premises. For this reason, *ground nonmonotonic modal logics* have been defined by restricting the introspection of the agent to non-modal sentences. The notion of groundedness has a rather intuitive motivation: in fact, it corresponds to discarding any reasoning based on epistemic assumptions, which, for example, would enable the agent to conclude that something is true in the world, by assuming to know it. The fixed-point equation defining a S_G -expansion of φ is: $E = Cn_S(\varphi \cup \{\neg K\gamma \mid \gamma \in \mathcal{L}(\mathcal{A}) - E\})$, where $\mathcal{L}(\mathcal{A})$ denotes the set of all objective (i.e., non-modal)

formulas of the underlying propositional language. According to this equation, we can associate a logic S_G to each modal logic S . Given $\varphi \in \mathcal{L}_K(\mathcal{A})$, $\psi \in \mathcal{L}_K(\mathcal{A})$, we say that ψ is entailed by φ in S_G (and write $\varphi \models_{S_G} \psi$) iff ψ belongs to all S_G -expansions for φ .

Among all ground nonmonotonic logics, the logic $S5_G$ (which, unlike NM-S5, is a true nonmonotonic logic) has received considerable attention in the literature and is generally referred to as the *logic of minimal knowledge* [Halpern and Moses, 1985] (or the *logic of maximal ignorance*). In fact, independently from its fixed-point characterization, $S5_G$ was characterized on a semantic basis, by means of a preference criterion among the models of an agent’s knowledge selecting just those models in which objective knowledge (i.e. the set of formulas of type $K\varphi$ such that $\varphi \in \mathcal{L}(\mathcal{A})$) is minimal [Halpern and Moses, 1985; Shoham, 1987; Lifschitz, 1994]. As shown in [Shoham, 1987], $S5_G$ has a simple, elegant model theory: The $S5_G$ models of a theory T are precisely those Kripke structures which are universal (i.e., totally connected) and are maximal set of worlds (w.r.t. set-containment). For details on ground nonmonotonic modal logics and $S5_G$, see [Halpern and Moses, 1985; Lifschitz, 1994; Donini *et al.*, 1997].

Main Problems Studied. The investigations reported in this paper were motivated by the following questions:

1. Given a formula φ , which formulas ψ are almost surely true in the Kripke models of φ , i.e., how can we characterize the formulas ψ that are true with limit probability one in the Kripke models of φ , when the size of these Kripke models grows towards infinity?
2. Given that modal inference under almost sure validity constitutes a form of nonmonotonic reasoning, to which of the nonmonotonic modal logics from the above-mentioned plethora of logics does this form of reasoning best correspond?
3. Can we characterize the set of formulas φ that guarantee a 0-1 law for the asymptotic probability $P(\psi|\varphi)$ for arbitrary modal formulas ψ ?
4. What is the complexity of modal reasoning based on asymptotic conditional probabilities?

For studying these questions, we make a very weak assumption on φ , which assures that φ does not contradict some principles of knowledge: we assume that φ is $S5$ -consistent, i.e., we assume that φ is consistent with the axioms of (monotonic) $S5$, which means that φ admits at least one $S5$ -model. This assumption is indeed very weak. It does neither mean that φ contains the $S5$ axioms, or that these axioms should follow from φ , nor that φ has to be interpreted under $S5$ Kripke structures only. Our assumption merely requires that formulas such as $Kp \wedge K\neg p$ which bluntly contradict some axioms of (monotonic) $S5$ cannot be deduced from φ . If φ satisfies this requirement, we say that φ is *knowledge-consistent*. We make this assumption because knowledge-inconsistent theories are well-known to be *inconsistent* for all currently known nonmonotonic modal logics anyway, i.e., in each such nonmonotonic logic, a knowledge-inconsistent formula φ entails a contradiction and thus all formulas of the modal language $\mathcal{L}_K(\mathcal{A})$. In particular, with regard to our goal of comparing asymptotic conditional reasoning to

³This definition is applicable to a large range of modal logics, but a collapse happens at $S5$ because NM- $S5=S5$ [Marek and Truszczyński, 1993].

nonmonotonic modal logics, there is no point to consider knowledge-inconsistent modal premises. Moreover, we characterize asymptotic conditional reasoning when the premise is a *flat* modal formula, i.e., a formula without nested occurrences of the modal operator. The flat fragment of a modal logic of knowledge is certainly a very important (if not the most important) syntactically restricted fragment. It consists of the Boolean closure of *knowledge bases*, i.e., of objective theories under the K operator. In particular, the flat fragment in $S5_G$ is extremely powerful and expressive. As shown in [Rosati, 1998], via appropriate translations, this fragment captures the well-known formalisms of default logic and logic programming under the stable model semantics. In the present paper we limit our attention to this flat fragment. However, we think that (by slightly more involved methods) our results carry over to the fully nested fragment (i.e., the fragment with nesting depth greater than 1).

Results. By first answering the second of the above-stated research questions, we show the following main result:

Answer to Question 2: φ almost surely entails ψ iff ψ is entailed by φ in $S5_G$.

This result gives a fresh probabilistic semantics to the well-known nonmonotonic modal logic $S5_G$, providing a new justification for $S5_G$ based on probabilistic rationality. At the same time, it provides the answers to our questions 1 and 4, too. In fact, as already mentioned, reasoning in $S5_G$ has precise characterizations in terms of model-theory and complexity, hence the same characterizations now apply to asymptotic conditional reasoning over Kripke models:

Answer to Question 1: φ almost surely entails ψ iff ψ is satisfied by all Kripke models of φ that are universal and have a maximal set of worlds.

Answer to Question 4: Deciding whether φ almost surely entails ψ is Π_2^P -complete.⁴

A theory is *honest* [Halpern and Moses, 1985] iff it has exactly one $S5_G$ -model. It has been argued that the epistemic state of a perfectly rational agent is necessarily honest, e.g., it could not be of the form $(KT_1) \vee (KT_2)$ where T_1 and T_2 are theories that contradict each other. Honest premises drastically simplify asymptotic conditional reasoning:

Answer to Question 3: The class of knowledge-consistent premises φ that imply a 0-1 law is exactly the class of honest formulas. In other words, the honest formulas are precisely the formulas φ such that, for all ψ , the asymptotic probability that ψ holds in the structures in which φ is valid is either 0 or 1. Moreover, asymptotic conditional reasoning based on honest theories is only Θ_2^P -complete.

Due to space limitations, we can only include the sketches of some results in the present version of the paper.

2 Preliminaries

We assume familiarity with modal logics K and $S5$. We deal with a propositional alphabet $\mathcal{A}$ such that either $\mathcal{A}$ is finite and fixed, i.e., it is the same for every problem instance, or $\mathcal{A}$ is not bounded, but each problem instance comes along with

a finite alphabet $\mathcal{A}$ as part of the input. We denote by $\mathcal{L}(\mathcal{A})$ the set of propositional (or objective) formulas over $\mathcal{A}$, denote by $\mathcal{L}_K(\mathcal{A})$ the set of modal formulas over $\mathcal{A}$, and denote by $\mathcal{L}_K^F(\mathcal{A})$ the set of *flat* modal formulas, i.e., the subset of $\mathcal{L}_K(\mathcal{A})$ of formulas over $\mathcal{A}$ satisfying the following abstract syntax:

$$\varphi ::= Kf \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2, \text{ where } f \in \mathcal{L}(\mathcal{A})$$

We also use the symbol **true** to denote the formula $a \vee \neg a$, and the symbol **false** to denote the formula $a \wedge \neg a$.

We now recall some auxiliary definitions that we will use in the following sections [Marek and Truszczyński, 1993; Donini *et al.*, 1997]. Given $\varphi \in \mathcal{L}_K(\mathcal{A})$, we denote by $MA(\varphi)$ the set of *modal atoms* from φ , i.e., the set of subformulas of the form $K\psi$ occurring in φ . In the following, we say that an occurrence of a modal atom $K\psi$ in a formula $\varphi \in \mathcal{L}_K(\mathcal{A})$ is *strict* if it does not lie within the scope of a modal operator.

Given a partition (P, N) of the set $MA(\varphi)$ and a formula $\psi \in \mathcal{L}_K(\mathcal{A})$, we denote by $\psi(P, N)$ the formula obtained from ψ by substituting each strict occurrence in ψ of a modal atom in P with **true**, and each strict occurrence in ψ of a modal atom in N with **false**. Notice that, if $P \cup N$ contains $MA(\psi)$, then $\psi(P, N)$ is a propositional formula.

Let $\varphi \in \mathcal{L}_K(\mathcal{A})$ and let (P, N) be a partition of $MA(\varphi)$. We denote by $obj_\varphi(P, N)$ the propositional formula $obj_\varphi(P, N) = \varphi(P, N) \wedge \bigwedge_{K\psi \in P} \psi(P, N)$.

Given a partition (P, N) of the set $MA(\varphi)$, we say that (P, N) is *S5-consistent with φ* if (P, N) satisfies the following conditions: (1) the propositional formula $obj_\varphi(P, N)$ is satisfiable; (2) for each $K\psi \in N$, the propositional formula $obj_\varphi(P, N) \wedge \neg\psi(P, N)$ is satisfiable. It is immediate to see that there exists a partition of $MA(\varphi)$ S5-consistent with φ iff φ is knowledge-consistent, i.e., there exists an S5-structure $\mathcal{S}$ such that $(w, \mathcal{S}) \models K\varphi$ where w is a world of $\mathcal{S}$. Finally, given a structure $\mathcal{S} = \langle W, R, V \rangle$ and a world $w \in W$, we say that (P, N) is the partition of $MA(\varphi)$ *satisfied by $(w, \mathcal{S})$* if, for each $K\psi \in MA(\varphi)$, $(w, \mathcal{S}) \models K\psi$ iff $K\psi \in P$.

3 Strong almost-sure conditional validity

In the work of Halpern and Kapron [Halpern and Kapron, 1994], almost sure structure validity is studied by considering all possible Kripke structures equally likely, i.e., uniformly distributed. This amounts to assume that that every propositional variable is true with probability 1/2 in a randomly chosen world. Under such an assumption, the asymptotic probability of φ w.r.t. a propositional alphabet $\mathcal{A}$ corresponds to the limit $\lim_{n \rightarrow \infty} |\mathcal{W}_n^\varphi|/|\mathcal{W}_n|$, where: (i) $\mathcal{W}_n$ denotes the set of all n -structures over $\mathcal{A}$, i.e., the structures with n worlds of the form $\langle W, R, V \rangle$, where $W = \{1, \dots, n\}$, the accessibility relation R is a binary relation over W , and V is a function mapping each world into a propositional interpretation over $\mathcal{A}$; (ii) $\mathcal{W}_n^\varphi$ denotes the set of all n -structures in which φ holds; (iii) $|S|$ represents the cardinality of a set S . Moreover, under the above uniform probability assumption, asymptotic conditional probability of ψ given φ corresponds to the limit $\lim_{n \rightarrow \infty} |\mathcal{W}_n^{\varphi \wedge \psi}|/|\mathcal{W}_n^\varphi|$.

It would be more appealing to consider the notion of *strong* almost sure validity, where the asymptotic probability of a formula ψ is required to be equal to 1 *for every possible*

⁴ Π_2^P is the complement of $\Sigma_2^P = \text{NP}^{\text{NP}}$; Θ_2^P is the class of problems solved in PTIME by a logarithmic number of calls to an NP-oracle.

probability distribution assigning rational truth probabilities to the propositional variables. It turns out that in the context of [Halpern and Kapron, 1994] both concepts are equivalent, thus all relevant results of [Halpern and Kapron, 1994] extend to strong almost sure validity. In the context of conditional probabilities, these concepts differ, however. In this paper we choose to characterize the notion of *strong* almost sure conditional validity, which is independent of a particular fixed probability distribution. However, we will use (Section 4) the concept of almost sure conditional validity (i.e., the one corresponding to the uniform distribution of Kripke structures) as a tool for establishing (Section 5) our main result on strong almost sure validity.

More formally, we associate to each propositional atom $a \in \mathcal{A}$ a rational probability m such that $0 < m < 1$, which is interpreted as the probability that the proposition a is true. Such an assignment is part of the input. Assignments giving probability 1 (resp. 0) to a are not considered, since in such cases the proposition a is certainly true (resp. false) and all its occurrences in a formula can be eliminated in a simple way. We assume without loss of generality that m is a finite sum of the negative powers of two. Hence, we say that $\mathcal{P}$ is a *probability assignment* over an alphabet $\mathcal{A}$ if $\mathcal{P}$ is a function mapping each propositional symbol from $\mathcal{A}$ to a rational number m such that there exists a finite binary sequence $m_1 \dots m_p$ such that $m = \sum_{i=1}^p \frac{m_i}{2^i}$.

It is immediate to verify the following relationship between the cardinality of a set of interpretations and the number of different n -structures defined over such interpretations.⁵

Lemma 3.1 *Let $\mathcal{I}$ be a set of propositional interpretations, and let $\mathcal{W}_n$ be the set of n -structures defined using the set of interpretations $\mathcal{I}$. Then, $|\mathcal{W}_n| = |\mathcal{I}|^n \cdot 2^{n^2}$.*

Let $\varphi, \psi \in \mathcal{L}_K(\mathcal{A})$, and let $\mathcal{P}$ be a probability assignment over $\mathcal{A}$. Then, $p_n^{\mathcal{P}}(\psi|\varphi)$ denotes the probability that ψ is valid in the n -structures in which φ is valid, under the probability assignment $\mathcal{P}$. We are now ready to define almost-sure and strong almost-sure conditional validity.

Definition 3.2 ($\mathcal{P}$ -almost-sure validity) *Let $\mathcal{P}$ be a probability assignment over $\mathcal{A}$. We say that ψ conditioned by φ is $\mathcal{P}$ -almost-surely valid if $\lim_{n \rightarrow \infty} p_n^{\mathcal{P}}(\psi|\varphi) = 1$, i.e., the asymptotic probability that ψ conditioned by φ is valid is 1 under the probability assignment $\mathcal{P}$. Conversely, if $\lim_{n \rightarrow \infty} p_n^{\mathcal{P}}(\psi|\varphi) < 1$, then we say that ψ conditioned by φ is not $\mathcal{P}$ -almost-surely valid.*

Definition 3.3 (strong-almost-sure validity) *ψ conditioned by φ is strongly almost-surely valid if $\lim_{n \rightarrow \infty} p_n^{\mathcal{P}}(\psi|\varphi) = 1$ for each probability assignment $\mathcal{P}$ over $\mathcal{A}$. Conversely, if there exists a probability assignment $\mathcal{P}$ over $\mathcal{A}$ such that $\lim_{n \rightarrow \infty} p_n^{\mathcal{P}}(\psi|\varphi) < 1$, then we say that ψ conditioned by φ is not strongly almost-surely valid.*

We denote by $\text{SAS}(\varphi, \mathcal{A})$ the set of formulas ψ from $\mathcal{L}_K(\mathcal{A})$ such that ψ conditioned by φ is strongly almost-surely valid.

4 Counting structures

In this section we prove the correspondence between the notion of stable set in nonmonotonic modal logics and strong

almost-sure conditional validity with respect to objective formulas, and establish a first correspondence between S5_G and strong almost-sure conditional validity in modal logic.

From now on, we denote by h the number of propositional interpretations of $\mathcal{A}$, i.e., $h = 2^{|\mathcal{A}|}$. Moreover, given a formula $f \in \mathcal{L}(\mathcal{A})$, we denote by h_f the number of interpretations of $\mathcal{A}$ satisfying f .

We start our analysis by studying the properties of the set of n -structures for f , i.e., the n -structures in which a propositional formula f is valid. First of all, from Lemma 3.1, it follows that the number of n -structures for f is $h_f \cdot 2^{n^2}$.

Then, we recall the definition of *stable set* of modal formulas. Let f be a satisfiable propositional formula over the propositional alphabet $\mathcal{A}$. The *stable set of f in $\mathcal{A}$* (denoted by $\text{Stable}(f, \mathcal{A})$) is the unique set of modal formulas $\mathcal{T} \subset \mathcal{L}_K(\mathcal{A})$ that satisfies the following conditions: (i) for each $\psi \in \mathcal{L}(\mathcal{A})$, $\psi \in \mathcal{T}$ iff $f \supset \psi$ is a tautology; (ii) if $\psi \in \mathcal{T}$ then $K\psi \in \mathcal{T}$; (iii) if $\psi \in \mathcal{L}_K(\mathcal{A}) - \mathcal{T}$ then $\neg K\psi \in \mathcal{T}$; (iv) $\mathcal{T}$ is closed under propositional consequence [Marek and Truszczyński, 1993].

Next, we prove the correspondence between formulas strongly almost-surely valid with respect to an objective formula f and the formulas in $\text{Stable}(f, \mathcal{A})$. The proof is easily obtained by extending an analogous result in [Halpern and Kapron, 1994].

Lemma 4.1 *Let $f \in \mathcal{L}(\mathcal{A})$ be a satisfiable objective formula. Then, $\text{SAS}(f, \mathcal{A}) = \text{Stable}(f, \mathcal{A})$.*

We now introduce two auxiliary lemmas.

Lemma 4.2 *Let φ be a flat and knowledge-consistent formula, let $\mathcal{W}_n^\varphi$ be the set of n -structures for φ and let $\mathcal{D}_n$ be the set of n -structures for φ in which all worlds satisfy the same partition (P, N) of $\text{MA}(\varphi)$ S5-consistent with φ . Then, $\lim_{n \rightarrow \infty} |\mathcal{D}_n|/|\mathcal{W}_n^\varphi| = 1$.*

Proof sketch. The proof is divided in two steps: First, we prove that, if φ is a flat and knowledge-consistent formula, $\lim_{n \rightarrow \infty} |\mathcal{D}'_n|/|\mathcal{W}_n^\varphi| = 1$, where $\mathcal{D}'_n$ denotes the set of n -structures for φ in which no world satisfies an S5-inconsistent partition of $\text{MA}(\varphi)$. Then, we concentrate on the set $\mathcal{D}''_n$: let $\mathcal{D}''_n$ denote the subset of $\mathcal{D}'_n$ in which two worlds satisfy two different S5-consistent partitions of $\text{MA}(\varphi)$ (P_1, N_1) , (P_2, N_2) , let $\mathcal{D}^1_n$ be the subset of $\mathcal{D}'_n$ in which all worlds satisfy (P_1, N_1) and let $\mathcal{D}^2_n$ be the subset of $\mathcal{D}'_n$ in which all worlds satisfy (P_2, N_2) . We prove that either $\lim_{n \rightarrow \infty} |\mathcal{D}''_n|/|\mathcal{D}^1_n| = 0$ or $\lim_{n \rightarrow \infty} |\mathcal{D}''_n|/|\mathcal{D}^2_n| = 0$. $\square$

The following property can be derived by an argument analogous to the proof of Lemma 4.1.

Lemma 4.3 *Let φ be a flat and knowledge-consistent formula and let (P, N) be a partition of $\text{MA}(\varphi)$ consistent with φ . Let $\mathcal{W}_n^\varphi(P, N)$ denote the set of n -structures for φ in which all worlds satisfy the partition (P, N) , and let $\mathcal{D}_n$ be the set of n -structures for $\text{obj}_\varphi(P, N) \wedge \mu(N)$, where $\mu(N) = \bigwedge_{Kf \in \mathcal{N}} \neg Kf$. Then, $\lim_{n \rightarrow \infty} |\mathcal{D}_n|/|\mathcal{W}_n^\varphi(P, N)| = 1$. Moreover, $\lim_{n \rightarrow \infty} |\mathcal{W}_n^\varphi(P, N)| = (h_{\text{obj}_\varphi(P, N)})^n \cdot 2^{n^2}$.*

We now define S5_G -preferred partitions of modal atoms.

Definition 4.4 (S5_G -preferred partition) *Let $\varphi \in \mathcal{L}_K(\mathcal{A})$. A partition (P, N) of $\text{MA}(\varphi)$ is S5_G -preferred for φ if (P, N)*

⁵We adopt the well-known *random worlds* method [Grove et al., 1996b; 1996a].

is $S5$ -consistent with φ and there exists no other partition $(P', N') \neq (P, N)$ of $MA(\varphi)$ such that (i) (P', N') is $S5$ -consistent with φ ; and (ii) the propositional formula $obj_\varphi(P, N) \wedge \neg obj_\varphi(P', N')$ is not satisfiable.

It is immediate to verify that $S5_G$ -preferred partitions of $MA(\varphi)$ are in one-to-one correspondence with $S5_G$ -expansions of φ : in particular, each such partition (P, N) identifies the $S5_G$ -expansion corresponding to $Stable(obj_\varphi(P, N), \mathcal{A})$.

We are now ready to show a first fundamental step towards the correspondence between $S5_G$ and strong almost-sure conditional validity: For each knowledge-consistent modal formula φ , to compute (strong) almost sure conditional validity we can safely consider only the set of n -structures in which all worlds satisfy one of the partitions of $MA(\varphi)$ that are $S5_G$ -preferred for φ .

Theorem 4.5 *Let φ be a flat, knowledge-consistent formula, let $\mathcal{W}_n^\varphi$ denote the set of n -structures for φ , and let $\mathcal{D}_n$ be the union of the sets of n -structures for φ in which all worlds satisfy the same partition (P, N) of $MA(\varphi)$, where (P, N) is $S5_G$ -preferred for φ . Then, $\lim_{n \rightarrow \infty} |\mathcal{D}_n|/|\mathcal{W}_n^\varphi| = 1$.*

Proof. Let $\mathcal{C}'_n$ be the union of the sets of n -structures for $obj_\varphi(P, N) \wedge \mu(N)$ for each partition (P, N) of $MA(\varphi)$ that is $S5$ -consistent with φ . First, by Lemma 4.2 and Lemma 4.3 it follows that $\lim_{n \rightarrow \infty} |\mathcal{C}'_n|/|\mathcal{W}_n^\varphi| = 1$. Moreover, let $\mathcal{C}_n$ be the union of the sets of n -structures for $obj_\varphi(P, N) \wedge \mu(N)$ for each partition (P, N) of $MA(\varphi)$ that is $S5_G$ -preferred for φ : By the same lemmas it follows that $\lim_{n \rightarrow \infty} |\mathcal{C}_n|/|\mathcal{D}_n| = 1$. Thus, we have to prove that $\lim_{n \rightarrow \infty} |\mathcal{C}_n|/|\mathcal{C}'_n| = 1$.

Let (P, N) be a partition (P, N) of $MA(\varphi)$ $S5_G$ -preferred for φ , and consider all the partitions (P', N') of $MA(\varphi)$ such that $obj_\varphi(P, N)$ is satisfied by all propositional interpretations satisfying $obj_\varphi(P', N')$. Let $\mathcal{C}_n^{(P, N)}$ be the set of n -structures for $obj_\varphi(P, N) \wedge \mu(N)$, and let $\mathcal{C}_n^{(P', N')}$ be the union of the set of n -structures for $obj_\varphi(P', N') \wedge \mu(N')$ for each such partition (P', N') . Obviously, $\mathcal{C}_n^{(P, N)} \subseteq \mathcal{C}_n^{(P', N')}$ (since (P, N) is one of such partitions (P', N')).

We now prove that $\lim_{n \rightarrow \infty} \frac{|\mathcal{C}_n^{(P, N)}|}{|\mathcal{C}_n^{(P', N')}|} = 1$. Let $\mathcal{C}''_n = \mathcal{C}_n^{(P', N')} - \mathcal{C}_n^{(P, N)}$. That is, $\mathcal{C}''_n$ is the set of n -structures for $obj_\varphi(P', N') \wedge \mu(N)$ for each partition (P', N') that is $S5$ -consistent with φ and such that $obj_\varphi(P, N)$ is satisfied by all propositional interpretations satisfying $obj_\varphi(P', N')$, and there exists at least an interpretation satisfying $obj_\varphi(P, N) \wedge \neg obj_\varphi(P', N')$. Now let k be the number of interpretations of $\mathcal{A}$ satisfying $obj_\varphi(P, N)$: It is immediate to verify that there can be at most k such different partitions (P', N') . Moreover, for each such partition (P', N') , there exists at least a propositional interpretation $\mathcal{I}$ that satisfies $obj_\varphi(P, N)$ and does not satisfy $obj_\varphi(P', N')$. Therefore, from Lemma 4.3 we have that $\frac{|\mathcal{C}''_n|}{|\mathcal{C}_n^{(P', N')}|} \leq \frac{k \cdot (k-1)^n \cdot 2^{n^2}}{k^n \cdot 2^{n^2}} = k \cdot ((k-1)/k)^n$. Consequently, $\lim_{n \rightarrow \infty} \frac{|\mathcal{C}''_n|}{|\mathcal{C}_n^{(P', N')}|} = 0$, which proves that $\lim_{n \rightarrow \infty} \frac{|\mathcal{C}_n^{(P, N)}|}{|\mathcal{C}_n^{(P', N')}|} = 1$. Hence, $\lim_{n \rightarrow \infty} \frac{|\mathcal{C}_n|}{|\mathcal{C}'_n|} = 1$. $\square$

5 Asymptotic conditional probability and $S5_G$

In this section we prove the main result of the paper, which establishes the correspondence between strong almost-sure conditioned validity and entailment in the logic $S5_G$. To this aim, we need some preliminary definitions and properties.

Let $\mathcal{A}'$ be the set of propositional atoms a^i such that $a \in \mathcal{A}$ and $1 \leq i \leq m$. We define the *canonical probability assignment* $\mathcal{P}_c$ over $\mathcal{A}'$ as follows: $\mathcal{P}_c(a) = \frac{1}{2}$ for each $a \in \mathcal{A}'$. Let $\mathcal{A}$ be a propositional alphabet and let $\mathcal{I}_f$ be the set of propositional interpretations over $\mathcal{A}$ that satisfy f . The *canonical probability* of a propositional formula f , denoted by $cp(f)$, is defined as $cp(f) = \frac{|\mathcal{I}_f|}{2^{|\mathcal{A}|}}$.

The following auxiliary lemma establishes a sufficient condition over the partitions of $MA(\varphi)$ which implies that $\lim_{n \rightarrow \infty} p_n^{\mathcal{P}_c}(\psi|\varphi) < 1$.

Lemma 5.1 *Let $\varphi \in \mathcal{L}_K^F(\mathcal{A})$. If there exists a partition (P, N) of $MA(\varphi)$ such that: (1) (P, N) is $S5_G$ -preferred for φ ; (2) for each partition (P', N') that is $S5_G$ -preferred for φ , $cp(obj_\varphi(P, N)) \geq cp(obj_\varphi(P', N'))$; (3) ψ does not hold in the structures for $obj_\varphi(P, N)$, then $\lim_{n \rightarrow \infty} p_n^{\mathcal{P}_c}(\psi|\varphi) < 1$.*

Let $a \in \mathcal{A}$ and let $\mathcal{P}(a) = k$ where k is a rational number satisfying the definition of probability assignment. Then, we can construct a propositional formula $f^k(a)$ defined over an alphabet $\mathcal{A}^k = \{a^1, \dots, a^{m_k}\}$ (where m_k is a number depending on k) such that $\mathcal{P}(a) = cp(f^k(a))$. Then, given a probability assignment $\mathcal{P}$ over $\mathcal{A}$, we define $\tau_{\mathcal{P}}(\varphi)$ as the formula obtained from φ by replacing, for each $a \in \mathcal{A}$, each occurrence of a in φ with $f^{\mathcal{P}(a)}(a)$. Now, in order to prove our main result, we need some auxiliary lemmas.

Lemma 5.2 *Let $\psi \in \mathcal{L}_K(\mathcal{A})$, and let $f \in \mathcal{L}(\mathcal{A})$. Then, $\psi \in SAS(f, \mathcal{A})$ iff $\tau_{\mathcal{P}}(\psi) \in SAS(\tau_{\mathcal{P}}(f), \mathcal{A}^k)$.*

Lemma 5.3 *Let $\varphi, \psi \in \mathcal{L}_K(\mathcal{A})$. then, for each partition (P, N) that is $S5_G$ -preferred for φ , the partition $(\tau_{\mathcal{P}}(P), \tau_{\mathcal{P}}(N))$ is $S5_G$ -preferred for $\tau_{\mathcal{P}}(\varphi)$, while for each partition $(\tau_{\mathcal{P}}(P), \tau_{\mathcal{P}}(N))$ that is $S5_G$ -preferred for $\tau_{\mathcal{P}}(\varphi)$, the partition (P, N) is $S5_G$ -preferred for φ .*

Lemma 5.4 *Let $\varphi \in \mathcal{L}_K(\mathcal{A})$. If a partition (P, N) of $MA(\varphi)$ is $S5_G$ -preferred for φ , then the stable set of $obj_\varphi(P, N)$ in $\mathcal{A}$ is an $S5_G$ -expansion for φ . Moreover, if a set $T \subset \mathcal{L}_K(\mathcal{A})$ is an $S5_G$ -expansion for φ , then there exists a partition (P, N) of $MA(\varphi)$ such that (P, N) is $S5_G$ -preferred for φ and $Stable(obj_\varphi(P, N), \mathcal{A}) = T$.*

Theorem 5.5 *Let $\varphi \in \mathcal{L}_K^F(\mathcal{A})$, $\psi \in \mathcal{L}_K(\mathcal{A})$, and let φ be a knowledge-consistent formula. Then, ψ conditioned by φ is strongly almost-surely valid iff $\varphi \models_{S5_G} \psi$.*

Proof. First, if $\varphi \models_{S5_G} \psi$, then, from definition of entailment in $S5_G$ and from Lemma 5.4 it follows that, for each partition (P, N) that is $S5_G$ -preferred for φ , ψ belongs to $Stable(obj_\varphi(P, N), \mathcal{A})$, consequently, by Lemma 4.1, $\psi \in SAS(obj_\varphi(P, N), \mathcal{A})$. Hence, by Lemma 5.2 and Lemma 5.3, for each probability assignment $\mathcal{P}$ over $\mathcal{A}$, $\tau_{\mathcal{P}}(\psi) \in SAS(\tau_{\mathcal{P}}(obj_\varphi(P, N)), \mathcal{A}^k)$. Therefore, by Theorem 4.5, $\lim_{n \rightarrow \infty} p_n^{\mathcal{P}_c}(\tau_{\mathcal{P}}(\psi)|\tau_{\mathcal{P}}(\varphi)) = 1$, which implies that ψ conditioned by φ is strongly almost-surely valid.

Conversely, if $\varphi \not\models_{S5_G} \psi$, then by Lemma 5.4 it follows that there exists a partition (P, N) that is $S5_G$ -preferred for φ and such that ψ does not belong to $Stable(obj_\varphi(P, N), \mathcal{A})$, consequently, by Lemma 4.1, $\psi \notin SAS(obj_\varphi(P, N), \mathcal{A})$. Therefore, by Lemma 5.2, $\tau_{\mathcal{P}}(\psi) \notin SAS(\tau_{\mathcal{P}}(obj_\varphi(P, N)), \mathcal{A}^k)$. Moreover, it is immediate to verify the existence a probability assignment $\mathcal{P}'$ over $\mathcal{A}$ such that, for each partition (P', N') that is $S5_G$ -preferred for φ , $cp(\tau_{\mathcal{P}'}(obj_\varphi(P, N))) \geq cp(\tau_{\mathcal{P}'}(obj_\varphi(P', N')))$. Consequently, by Lemma 5.1, it follows that $\lim_{n \rightarrow \infty} p_n^{\mathcal{P}'}(\tau_{\mathcal{P}}(\psi) | \tau_{\mathcal{P}}(\varphi)) < 1$, i.e., ψ conditioned by φ is not strongly almost-surely valid. $\square$

6 Complexity results

As explained above, we consider both the case in which $\mathcal{A}$ is a fixed finite alphabet and the case in which it is considered as finite but not fixed in advance. If $\mathcal{A}$ is finite and fixed, then, based on the algorithm for entailment in $S5_G$ reported in [Donini et al., 1997] it can be easily shown that entailment in $S5_G$ can be decided in polynomial time, and therefore, by Theorem 5.5, strong almost-sure conditional validity can be decided in polynomial time as well. Consider now the case of a finite, non-fixed alphabet $\mathcal{A}$.

Theorem 6.1 *Let $\varphi \in \mathcal{L}_K^F(\mathcal{A})$, $\psi \in \mathcal{L}_K(\mathcal{A})$, and let φ be a knowledge-consistent formula. Deciding whether ψ conditioned by φ is strongly almost-surely valid is a Π_2^P -complete problem.*

Proof. Follows immediately from Theorem 5.5 and from the fact that the entailment problem $\varphi \models_{S5_G} \psi$ when $\varphi \in \mathcal{L}_K^F(\mathcal{A})$ is Π_2^P -complete, which is an immediate consequence of Corollary 4.12 of [Donini et al., 1997] and of Theorem 11 of [Rosati, 1998]. $\square$

Next, we study a subclass of flat modal formulas for which deciding strong almost-sure conditional validity is computationally easier than in the general case: the class of *honest* formulas, introduced by [Halpern and Moses, 1985]. A flat formula φ is honest iff it has exactly one $S5_G$ -model. It is known that deciding whether φ is honest is a Θ_2^P -complete problem. Moreover, if φ is honest, then deciding the entailment $\varphi \models_{S5_G} \psi$ is also Θ_2^P -complete. Therefore, from Theorem 5.5 the following property holds.

Theorem 6.2 *Let $\varphi \in \mathcal{L}_K^F(\mathcal{A})$, $\psi \in \mathcal{L}_K(\mathcal{A})$, and let φ be a honest formula. Deciding whether ψ conditioned by φ is strongly almost-surely valid is Θ_2^P -complete.*

Furthermore, the notion of honesty precisely characterizes the subclass of formulas, among the formulas φ such that $K\varphi$ is $S5$ -consistent, that condition according to a 0-1 law, i.e., such that either $\lim_{n \rightarrow \infty} p_n^{\mathcal{P}}(\psi | \varphi) = 1$ for each probability assignment $\mathcal{P}$ or $\lim_{n \rightarrow \infty} p_n^{\mathcal{P}}(\psi | \varphi) = 0$ for each probability assignment $\mathcal{P}$, which is immediately implied by Theorem 5.5 and by the fact that a honest formula has a single $S5_G$ -model.

7 Conclusions

The present work can be extended in several directions. In particular, we are currently investigating the following issues: (i) strong almost-sure validity for (classes of) conditioning formulas with nested occurrences of modalities; (ii)

extending the study of asymptotic conditional validity to the framework of multimodal logic; (iii) computing $\mathcal{P}$ -almost-sure conditional validity, i.e., the value of the asymptotic conditional validity for a given probability distribution $\mathcal{P}$ of the truth of primitive propositions.

References

- [Le Bars, 1998] J.-M. Le Bars. Fragments of existential second-order logic without 0-1 laws. In *Proc. of LICS*, 1998.
- [Le Bars, 2002] J.-M. Le Bars. The 0-1 law fails for frame satisfiability of propositional modal logic. In *Proc. of LICS*, 2002.
- [Carnap, 1947] R. Carnap. *Meaning and necessity*. University of Chicago Press, 1947.
- [Compton, 1988] K. Compton. 0-1 laws in logic and combinatorics. In *Proc. 1987 Nato Adv. Study Inst. on Algorithms and Order*, pages 353–383. Reidel, Dordrecht, Netherlands, 1988.
- [Donini et al., 1997] F. M. Donini, Daniele Nardi, and Riccardo Rosati. Ground nonmonotonic modal logics. *J. of Log. and Comp.*, 7(4):523–548, August 1997.
- [Fagin, 1976] R. Fagin. Probabilities on finite models. *J. of Symbolic Logic*, 41:50–58, 1976.
- [Glebskii et al., 1969] Y.V. Glebskii, D.I. Kogan, M.I. Liogonki, and V.A. Talanov. Range and degree of realizability of formulas in the restricted predicate calculus. *Cybernetics*, 5, 1969.
- [Gottlob, 1999] G. Gottlob. Review of a Carnapian extension of $S5$. In Ewa Orłowska, editor, *Logic at Work. Essays Dedicated to the Memory of Helena Rasiowa*. Physica Verlag (Springer), 1999.
- [Grove et al., 1996a] A. Grove, J.Y. Halpern, and D. Koller. Asymptotic conditional probabilities: The non-unary case. *J. of Symbolic Logic*, 61(1):250–276, 1996.
- [Grove et al., 1996b] A. Grove, J.Y. Halpern, and D. Koller. Asymptotic conditional probabilities: The unary case. *SIAM J. on Computing*, 25(1):1–51, 1996.
- [Halpern and Kapron, 1994] J. Y. Halpern and B. Kapron. Zero-one laws for modal logic. *Annals of Pure and Applied Logics*, 69:157–193, 1994.
- [Halpern and Moses, 1985] J. Y. Halpern and Y. Moses. Towards a theory of knowledge and ignorance: Preliminary report. In *Logic and models of concurrent systems*. Springer, 1985.
- [Kolaitis and Vardi, 1990] P. G. Kolaitis and M. Y. Vardi. 0-1 laws and decision problems for fragments of second-order logic. *Information and Computation*, 87(1/2):301–337, 1990.
- [Lifschitz, 1994] V. Lifschitz. Minimal belief and negation as failure. *Artificial Intelligence*, 70:53–72, 1994.
- [Marek and Truszczyński, 1993] W. Marek and M. Truszczyński. *Nonmonotonic Logics: Context-Dependent Reasoning*. Springer, 1993.
- [Moore, 1985] R. C. Moore. Semantical considerations on non-monotonic logic. *Artificial Intelligence*, 25:75–94, 1985.
- [Rosati, 1998] R. Rosati. Embedding negation as failure into minimal knowledge. In *Proc. of ECAI'98*, pages 28–32, 1998.
- [Shoham, 1987] Y. Shoham. Nonmonotonic logics: Meaning and utility. In *Proc. of IJCAI'87*, pages 388–392, 1987.
- [Stalnaker, 1993] R. Stalnaker. A note on non-monotonic modal logic. *Artificial Intelligence*, 64(2):183–196, 1993.

Affine Algebraic Decision Diagrams (AADDs) and their Application to Structured Probabilistic Inference

Scott Sanner

Department of Computer Science
University of Toronto
Toronto, ON M5S 3H5, CANADA
ssanner@cs.toronto.edu

David McAllester

TTI at Chicago
1427 East 60th Street
Chicago, IL 60637, USA
mcallester@tti-c.org

Abstract

We propose an affine extension to ADDs (AADD) capable of compactly representing context-specific, additive, and multiplicative structure. We show that the AADD has worst-case time and space performance within a multiplicative constant of that of ADDs, but that it can be linear in the number of variables in cases where ADDs are exponential in the number of variables. We provide an empirical comparison of tabular, ADD, and AADD representations used in standard Bayes net and MDP inference algorithms and conclude that the AADD performs at least as well as the other two representations, and often yields an exponential performance improvement over both when additive or multiplicative structure can be exploited. These results suggest that the AADD is likely to yield exponential time and space improvements for a variety of probabilistic inference algorithms that currently use tables or ADDs.

1 Introduction

Algebraic decision diagrams (ADDs) [1] provide an efficient means for representing and performing arithmetic operations on functions from a factored boolean domain to a real-valued range (i.e., $\mathbb{B}^n \rightarrow \mathbb{R}$). They rely on two main principles to do this:

1. ADDs represent a function $\mathbb{B}^n \rightarrow \mathbb{R}$ as a directed acyclic graph – essentially a decision tree with re-convergent branches and real-valued terminal nodes.
2. ADDs enforce a strict variable ordering on the decisions from the root to the terminal node, enabling a minimal, canonical diagram to be produced for a given function. Thus, two identical functions will always have identical ADD representations under the same variable ordering.

As shown in Figure 1, ADDs often provide an efficient representation of functions with context-specific independence [2], such as functions whose structure is conjunctive (1a) or disjunctive (1b) in nature. Thus,

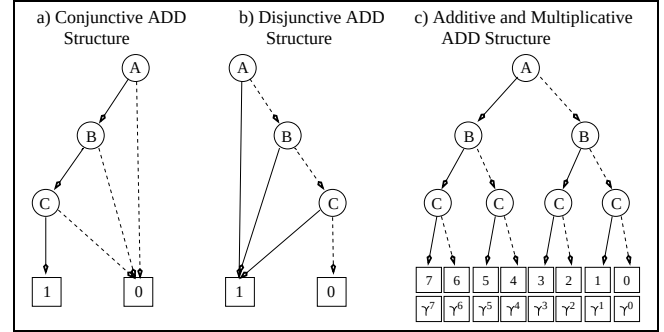


Figure 1: Some example ADDs showing a) conjunctive structure ($f = if(a \wedge b \wedge c, 1, 0)$), b) disjunctive structure ($f = if(a \vee b \vee c, 1, 0)$), and c) additive ($f = 4a + 2b + c$) and multiplicative ($f = \gamma^{4a+2b+c}$) structure (top and bottom sets of terminal values, respectively). The high (true) edge is solid, the low (false) edge is dotted.

ADDs can offer exponential space savings over a fully enumerated tabular representation. However, the compactness of ADDs does not extend to the case of additive or multiplicative independence, as demonstrated by the exponentially large representations when this structure is present (1c). Unfortunately such structure often occurs in probabilistic and decision-theoretic reasoning domains, potentially leading to exponential running times and space requirements for inference in these domains.

2 Affine Algebraic Decision Diagrams

To address the limitations of ADDs, we introduce an affine extension to the ADD (AADD) that is capable of canonically and compactly representing context-specific, additive, and multiplicative structure in functions from $\mathbb{B}^n \rightarrow \mathbb{R}$. We define AADDs with the following BNF:

$$\begin{aligned} G &::= c + bF \\ F &::= 0 \mid if(F^{var}, c_h + b_h F_h, c_l + b_l F_l) \end{aligned}$$

Here, c_h and c_l are real (or floating-point) constants in the closed interval $[0, 1]$, b_h and b_l are real constants in the half-open interval $(0, 1]$, F^{var} is a boolean variable associated with F , and F_l and F_h are non-terminals of type F . We also impose the following constraints:

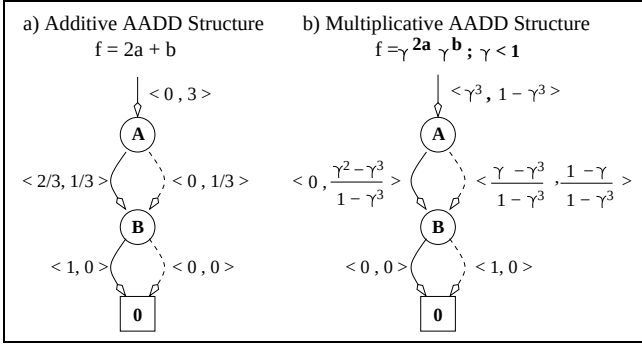


Figure 2: Portions of the ADDs from figure 1c expressed as generalized AADDs. The edge weights are given as $\langle c, b \rangle$.

1. The variable F^{var} does not appear in F_h or F_l .
2. $\min(c_h, c_l) = 0$
3. $\max(c_h + b_h, c_l + b_l) = 1$
4. If $F_h = 0$ then $b_h = 0$ and $c_h > 0$. Similarly for F_l .
5. In the grammar for G , we require that if $F = 0$ then $b = 0$, otherwise $b > 0$.

Expressions in the grammar for F will be called normalized AADDs and expressions in the grammar for G will be called generalized AADDs.¹

Let $Val(F, \rho)$ be the value of AADD F under variable value assignment ρ . This can be defined recursively by the following equation:

$$Val(F, \rho) = \begin{cases} F = 0 : & 0 \\ F \neq 0 \wedge \rho(F^{var}) = true : & c_h + b_h \cdot Val(F_h, \rho) \\ F \neq 0 \wedge \rho(F^{var}) = false : & c_l + b_l \cdot Val(F_l, \rho) \end{cases}$$

Lemma 2.1. *For any normalized AADD F we have that $Val(F, \rho)$ is in the interval $[0, 1]$, $\min_{\rho} Val(F, \rho) = 0$, and if $F \neq 0$ then $\max_{\rho} Val(F, \rho) = 1$.*

We say that F satisfies a given variable ordering if $F = 0$ or F is of the form $if(F^{var}, c_h + b_h F_h, c_l + b_l F_l)$ where F^{var} does not occur in F_h or F_l and F^{var} is the earliest variable under the given ordering occurring in F . We say that a generalized AADD of form $c + bF$ satisfies the order if F satisfies the order.

Lemma 2.2. *Fix a variable ordering. For any non-constant function g mapping $\mathbb{B}^n \rightarrow \mathbb{R}$, there exists a unique generalized AADD G satisfying the given variable ordering such that for all $\rho \in \mathbb{B}^n$ we have $g(\rho) = Val(G, \rho)$.*

Both lemmas can be proved by straightforward induction on n . The second lemma shows that under a given variable ordering, generalized AADDs are canonical, i.e., two identical functions will always have identical AADD representations.

As an example of two AADDs with compact additive and multiplicative structure, Figure 2 shows portions of

¹Since normalized AADDs in grammar F are restricted to the range $[0, 1]$, we need the top-level positive affine transform of generalized AADDs in grammar G to allow for the representation of functions with arbitrary range.

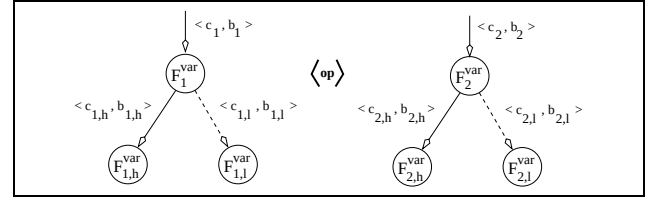


Figure 3: Two AADD nodes F_1 and F_2 and a binary operation op with the corresponding notation used in this paper.

the exponentially sized ADDs from Figure 1c represented by generalized AADDs of linear size.

We let op denote a binary operator on AADDs with possible operations being addition, subtraction, multiplication, division, min, and max denoted respectively as $\oplus$, $\ominus$, $\otimes$, $\oslash$, $\min(\cdot, \cdot)$, and $\max(\cdot, \cdot)$.

3 Related Work

There has been much related work in the formal verification literature that has attempted to tackle additive and multiplicative structure in representation of functions from $\mathbb{B}^n \rightarrow \mathbb{B}^m$. These include *BMDs, K*BMDs, EVBDDs, FEVBDDs, HDDs, and *PHDDs [3]. While space limitations prevent us from discussing all of the differences between AADDs and these data structures, we note two major differences: 1) These data structures have different canonical forms relying on GCD factorization and do not satisfy the invariant property of AADDs that internal nodes have range $[0, 1]$. 2) The fact that these data structures have integer terminals ($\mathbb{B}^m$) requires a rational or direct floating-point representation of values in $\mathbb{R}$. In our experience, this renders them unusable for probabilistic inference due to considerable numerical precision error and computational difficulties.

4 Algorithms

We now define AADD algorithms that are analogs of those given by Bryant [4]² except that they are extended to propagate the affine transform of the edge weights on recursion and to compute the normalization of the resulting node on return.

All algorithms rely on the helper function *GetGNode* given in Algorithm 1 that takes an unnormalized AADD node of the form $if(v, c_h + b_h F_h, c_l + b_l F_l)$ and returns the cached, generalized AADD node of the form $\langle c_r + b_r F_r \rangle$.³

4.1 Reduce

The *Reduce* algorithm given in Algorithm 2 takes an arbitrary ordered AADD, normalizes and caches the internal nodes, and returns the corresponding generalized AADD. One nice property of the *Reduce* algorithm is that one does not need to prespecify the structure that

²Bryant gives algorithms for binary decision diagrams (BDDs), but they are essentially identical to those for ADDs.

³In the algorithms we use the representation $\langle c, b, F \rangle$, while in the text we often use the equivalent notation $\langle c + bF \rangle$ to make the node semantics more clear.

Algorithm 1: $GetGNode(v, \langle c_h, b_h, F_h \rangle, \langle c_l, b_l, F_l \rangle) \rightarrow \langle c_r, b_r, F_r \rangle$

input : $v, \langle c_h, b_h, F_h \rangle, \langle c_l, b_l, F_l \rangle$: Var, offset, mult, and node id for high/low branches

output : $\langle c_r, b_r, F_r \rangle$: Return values for offset, multiplier, and canonical node id

begin

// If branches redundant, return child

if $(c_l = c_h \wedge b_l = b_h \wedge F_l = F_h)$ **then**

return $\langle c_l, b_l, F_l \rangle$;

// Non-redundant so compute canonical form

$r_{min} := \min(c_l, c_h); r_{max} := \max(c_l + b_l, c_h + b_h);$

$r_{range} := r_{max} - r_{min};$

$c_l := (c_l - r_{min})/r_{range}; c_h := (c_h - r_{min})/r_{range};$

$b_l := b_l/r_{range}; b_h := b_h/r_{range};$

if $(\langle v, \langle c_h, b_h, F_h \rangle, \langle c_l, b_l, F_l \rangle \rangle \rightarrow id \text{ is not in node cache})$ **then**

$id :=$ currently unallocated id;

insert $\langle v, \langle c_h, b_h, F_h \rangle, \langle c_l, b_l, F_l \rangle \rangle \rightarrow id$ in cache;

// Return the cached, canonical node

return $\langle r_{min}, r_{range}, id \rangle$;

end

the AADD should exploit. If the represented function contains context-specific, additive, or multiplicative independence, the *Reduce* algorithm will compactly represent this structure uniquely and automatically w.r.t. the variable ordering as guaranteed by previous lemmas.

4.2 Apply

The *Apply* routine given in Algorithm 3 takes two generalized AADD operands and an operation as given in Figure 3 and produces the resulting generalized AADD. The control flow of the algorithm is straightforward: We first check whether we can compute the result immediately, otherwise we normalize the operands to a canonical form and check if we can reuse the result of a previously cached computation. If we can do neither of these, we then choose a variable to branch on and recursively call the *Apply* routine for each instantiation of the variable. We cover these steps in-depth in the following sections.

Terminal computation

The function *ComputeResult* given in the top half of Table 1, determines if the result of a computation can be immediately computed without recursion. The first entry in this table is required for proper termination of the algorithm as it computes the result of an operation applied to two terminal 0 nodes. However, the other entries denote a number of pruning optimizations that immediately return a node without recursion. For example, given the operation $\langle 3 + 4F_1 \rangle \oplus \langle 5 + 6F_1 \rangle$, we can immediately return the result $\langle 8 + 10F_1 \rangle$.

Recursive computation

If a call to *Apply* is unable to immediately compute a result or reuse a previously cached computation, we must recursively compute the result. For this we have two

Algorithm 2: $Reduce(\langle c, b, F \rangle) \rightarrow \langle c_r, b_r, F_r \rangle$

input : $\langle c, b, F \rangle$: Offset, multiplier, and node id

output : $\langle c_r, b_r, F_r \rangle$: Return values for offset, multiplier, and node id

begin

// Check for terminal node

if $(F = 0)$ **then**

return $\langle c, 0, 0 \rangle$;

// Check reduce cache

if $(F \rightarrow \langle c_r, b_r, F_r \rangle \text{ is not in reduce cache})$ **then**

// Not in cache, so recurse

$\langle c_h, b_h, F_h \rangle := Reduce(c_h, b_h, F_h);$

$\langle c_l, b_l, F_l \rangle := Reduce(c_l, b_l, F_l);$

// Retrieve canonical form

$\langle c_r, b_r, F_r \rangle := GetGNode(F^{var}, \langle c_h, b_h, F_h \rangle, \langle c_l, b_l, F_l \rangle);$

// Put in cache

insert $F \rightarrow \langle c_r, b_r, F_r \rangle$ in reduce cache;

// Return canonical reduced node

return $\langle c + b \cdot c_r, b \cdot b_r, F_r \rangle$;

end

cases (the third case where both operands are 0 terminal nodes having been taken care of in the previous section):

F_1 or F_2 is a 0 terminal node, or $F_1^{var} \neq F_2^{var}$: We assume the operation is commutative and reorder the operands so that F_1 is the 0 node or the operand whose variable comes *later* in the variable ordering.⁴ Then we propagate the affine transform to each of F_2 's branches and compute the operation applied separately to F_1 and *each* of F_2 's high and low branches. We then build an *if* statement conditional on F_2^{var} and normalize it to obtain the generalized AADD node $\langle c_r, b_r, F_r \rangle$ for the result:

$$\begin{aligned} \langle c_h, b_h, F_h \rangle &= Apply(\langle c_1, b_1, F_1 \rangle, \langle c_2 + b_2 c_{2,h}, b_2 b_{2,h}, F_{2,h} \rangle, op) \\ \langle c_l, b_l, F_l \rangle &= Apply(\langle c_1, b_1, F_1 \rangle, \langle c_2 + b_2 c_{2,l}, b_2 b_{2,l}, F_{2,l} \rangle, op) \\ \langle c_r, b_r, F_r \rangle &= GetGNode(F_2^{var}, \langle c_h, b_h, F_h \rangle, \langle c_l, b_l, F_l \rangle) \end{aligned}$$

F_1 and F_2 are non-terminal nodes and $F_1^{var} = F_2^{var}$: Since the variables for each operand match, we know the result $\langle c_r, b_r, F_r \rangle$ is simply a generalized *if* statement branching on F_1^{var} ($= F_2^{var}$) with the true case being the operator applied to the high branches of F_1 and F_2 and likewise for the false case and the low branches:

$$\begin{aligned} \langle c_h, b_h, F_h \rangle &= Apply(\langle c_1 + b_1 c_{1,h}, b_1 b_{1,h}, F_{1,h} \rangle, \\ &\quad \langle c_2 + b_2 c_{2,h}, b_2 b_{2,h}, F_{2,h} \rangle, op) \\ \langle c_l, b_l, F_l \rangle &= Apply(\langle c_1 + b_1 c_{1,l}, b_1 b_{1,l}, F_{1,l} \rangle, \\ &\quad \langle c_2 + b_2 c_{2,l}, b_2 b_{2,l}, F_{2,l} \rangle, op) \\ \langle c_r, b_r, F_r \rangle &= GetGNode(F_1^{var}, \langle c_h, b_h, F_h \rangle, \langle c_l, b_l, F_l \rangle) \end{aligned}$$

⁴We note that the first case prohibits the use of the non-commutative $\ominus$ and $\oslash$ operations. However, a simple solution would be to recursively descend on either F_1 or F_2 rather than assuming commutativity and swapping operands to ensure descent on F_2 . To accommodate general non-commutative operations, we have used this alternate approach in our specification of the *Apply* routine given in Algorithm 3.

Canonical caching

If the AADD *Apply* algorithm were to compute and cache the results of applying an operation directly to the operands, the algorithm would provably have the same time complexity as the ADD *Apply* algorithm. Yet, if we were to compute $\langle 0+1F_1 \rangle \oplus \langle 0+2F_2 \rangle$ and cache the result $\langle c_r + b_r F_r \rangle$, we could compute $\langle 5+2F_1 \rangle \oplus \langle 4+4F_2 \rangle = \langle (2c_r + 9) + 2b_r F_r \rangle$ without recursion.

This suggests a canonical caching scheme that normalizes all cache entries to increase the chance of a cache hit. The actual result can then be easily computed from the cached result by reversing the normalization. This ensures optimal reuse of the *Apply* operations cache and can lead to an exponential reduction in running time over the non-canonical caching version.

We introduce two additional functions to perform this caching: *GetNormCacheKey* to compute the canonical cache key, and *ModifyResult* to reverse the normalization in order to compute the actual result. These algorithms are summarized in the *bottom half* of Table 1.

4.3 Other Operations

While space limitations prevent us from covering all of the operations that can be performed (efficiently) on AADDs, we briefly summarize some of them here:

min and max computation: The min and max of a generalized AADD node $\langle c + bF \rangle$ are respectively c and $c + b$ due to $[0, 1]$ normalization of F .

Restriction: The restriction of a variable x_i in a function to either *true* or *false* (i.e. $F|_{x_i=T/F}$) can be computed by returning the proper branch of nodes containing that test variable during the *Reduce* operation.

Sum out/marginalization: A variable x_i can be summed (or marginalized) out of a function F simply by computing the sum of the restricted functions (i.e. $F|_{x_i=T} \oplus F|_{x_i=F}$).

Negation/reciprocation: While it may seem that negation of a generalized AADD node $\langle c + bF \rangle$ would be as simple as $\langle -c + -bF \rangle$, we note that this violates our normalization scheme which requires $b > 0$. Consequently, negation must be performed explicitly as $0 \ominus \langle c + bF \rangle$. Likewise, reciprocation (i.e., $\frac{1}{\langle c + bF \rangle}$) must be performed explicitly as $1 \oslash \langle c + bF \rangle$.

Variable reordering: A generalization of Rudell's ADD variable reordering algorithm [5] that recomputes edge weights of reordered nodes can be applied to AADDs without loss of efficiency.

4.4 Cache Implementation

If one were to use a naive cache implementation that relied on exact floating-point values for hashing and equality testing, one would find that many nodes which *should* be the same under exact computation often turn out to have offsets or multipliers differing by $\pm 1e-15$ due to numerical precision error. This can result in an exponential explosion of nodes if not controlled. Consequently, it is better to use a hashing scheme that considers equality within some range of numerical precision error ϵ . While

Algorithm 3: *Apply*($\langle c_1, b_1, F_1 \rangle, \langle c_2, b_2, F_2 \rangle, op$) $\rightarrow \langle c_r, b_r, F_r \rangle$

```

input   :  $\langle c_1, b_1, F_1 \rangle, \langle c_2, b_2, F_2 \rangle, op$  : Nodes and op
output  :  $\langle c_r, b_r, F_r \rangle$  : Generalized node to return
begin
  // Check if result can be immediately computed
  if (ComputeResult( $\langle c_1, b_1, F_1 \rangle, \langle c_2, b_2, F_2 \rangle, op$ )  $\rightarrow \langle c_r, b_r, F_r \rangle$  is not null) then
     $\perp$  return  $\langle c_r, b_r, F_r \rangle$ ;
  // Get normalized key and check apply cache
   $\langle \langle c'_1, b'_1 \rangle, \langle c'_2, b'_2 \rangle \rangle :=$ 
    GetNormCacheKey( $\langle c_1, b_1, F_1 \rangle, \langle c_2, b_2, F_2 \rangle, op$ );
  if ( $\langle \langle c'_1, b'_1, F_1 \rangle, \langle c'_2, b'_2, F_2 \rangle, op \rangle \rightarrow \langle c_r, b_r, F_r \rangle$  is not in apply cache) then
    // Not terminal, so recurse
    if ( $F_1$  is a non-terminal node) then
      if ( $F_2$  is a non-terminal node) then
        if ( $F_1^{var}$  comes before  $F_2^{var}$ ) then
           $var := F_1^{var}$ ;
        else
           $var := F_2^{var}$ ;
        else
           $var := F_1^{var}$ ;
      else
         $var := F_2^{var}$ ;
    else
       $var := F_2^{var}$ ;
    // Propagate affine transform to branches
    if ( $F_1$  is non-terminal  $\wedge var = F_1^{var}$ ) then
       $F_l^{v1} := F_{1,l}; F_h^{v1} := F_{1,h};$ 
       $c_l^{v1} := c'_1 + b'_1 \cdot c_{1,l}; c_h^{v1} := c'_1 + b'_1 \cdot c_{1,h};$ 
       $b_l^{v1} := b'_1 \cdot b_{1,l}; b_h^{v1} := b'_1 \cdot b_{1,h};$ 
    else
       $F_{l/h}^{v1} := F_1; c_{l/h}^{v1} := c'_1; b_{l/h}^{v1} := b'_1;$ 
    if ( $F_2$  is non-terminal  $\wedge var = F_2^{var}$ ) then
       $F_l^{v2} := F_{2,l}; F_h^{v2} := F_{2,h};$ 
       $c_l^{v2} := c'_2 + b'_2 \cdot c_{2,l}; c_h^{v2} := c'_2 + b'_2 \cdot c_{2,h};$ 
       $b_l^{v2} := b'_2 \cdot b_{2,l}; b_h^{v2} := b'_2 \cdot b_{2,h};$ 
    else
       $F_{l/h}^{v2} := F_2; c_{l/h}^{v2} := c'_2; b_{l/h}^{v2} := b'_2;$ 
    // Recurse and get cached result
     $\langle c_l, b_l, F_l \rangle := \text{Apply}(\langle c_l^{v1}, b_l^{v1}, F_l^{v1} \rangle, \langle c_l^{v2}, b_l^{v2}, F_l^{v2} \rangle, op);$ 
     $\langle c_h, b_h, F_h \rangle := \text{Apply}(\langle c_h^{v1}, b_h^{v1}, F_h^{v1} \rangle, \langle c_h^{v2}, b_h^{v2}, F_h^{v2} \rangle, op);$ 
     $\langle c_r, b_r, F_r \rangle := \text{GetGNode}(var, \langle c_h, b_h, F_h \rangle, \langle c_l, b_l, F_l \rangle);$ 
    // Put result in apply cache and return
    insert  $\langle c'_1, b'_1, F_1, c'_2, b'_2, F_2, op \rangle \rightarrow \langle c_r, b_r, F_r \rangle$ 
    into apply cache;
  return ModifyResult( $\langle c_r, b_r, F_r \rangle$ );
end

```

it is difficult to guarantee such an exact property for an *efficient* hashing scheme, we next outline an approximate approach that we have found to work both efficiently and nearly optimally in practice.

The node cache used in *GetGNode* and the operation result cache used in *Apply* both use cache keys containing four floating-point values (i.e., the offsets and multipliers

$ComputeResult(\langle c_1, b_1, F_1 \rangle, \langle c_2, b_2, F_2 \rangle, op) \longrightarrow \langle c_r, b_r, F_r \rangle$	
Operation and Conditions	Return Value
$\langle c_1 + b_1 F_1 \rangle \langle op \rangle \langle c_2 + b_2 F_2 \rangle; F_1 = F_2 = 0$	$\langle \langle c_1 \langle op \rangle c_2 \rangle + 0 \cdot 0 \rangle$
$\max(\langle c_1 + b_1 F_1 \rangle, \langle c_2 + b_2 F_2 \rangle); c_1 + b_1 \leq c_2$	$\langle c_2 + b_2 F_2 \rangle$
$\max(\langle c_1 + b_1 F_1 \rangle, \langle c_2 + b_2 F_2 \rangle); c_2 + b_2 \leq c_1$	$\langle c_1 + b_1 F_1 \rangle$
$\langle c_1 + b_1 F_1 \rangle \oplus \langle c_2 + b_2 F_2 \rangle; F_1 = F_2$	$\langle (c_1 + c_2) + (b_1 + b_2) F_1 \rangle$
$\max(\langle c_1 + b_1 F_1 \rangle, \langle c_2 + b_2 F_1 \rangle); F_1 = F_2,$ $(c_1 \geq c_2 \wedge b_1 \geq b_2) \vee (c_2 \geq c_1 \wedge b_2 \geq b_1)$	$c_1 \geq c_2 \wedge b_1 \geq b_2 : \langle c_1 + b_1 F_1 \rangle$ $c_2 \geq c_1 \wedge b_2 \geq b_1 : \langle c_2 + b_2 F_1 \rangle$
Note: for all max operations above, return opposite for min	
$\langle c_1 + b_1 F_1 \rangle \langle op \rangle \langle c_2 + b_2 F_2 \rangle; F_2 = 0, op \in \{\oplus, \ominus\}$	$\langle \langle c_1 \langle op \rangle c_2 \rangle + b_1 F_1 \rangle$
$\langle c_1 + b_1 F_1 \rangle \langle op \rangle \langle c_2 + b_2 F_2 \rangle; F_2 = 0, c_2 \geq 0, op \in \{\otimes, \oslash\}$	$\langle \langle c_1 \langle op \rangle c_2 \rangle + (b_1 \langle op \rangle c_2) F_1 \rangle$
Note: above two operations can be modified to handle $F_1 = 0$ when $op \in \{\oplus, \otimes\}$	
other	<i>null</i>

$GetNormCacheKey(\langle c_1, b_1, F_1 \rangle, \langle c_2, b_2, F_2 \rangle, op) \longrightarrow \langle c'_1, b'_1 \rangle \langle c'_2, b'_2 \rangle$ and $ModifyResult(\langle c_r, b_r, F_r \rangle) \longrightarrow \langle c'_r, b'_r, F'_r \rangle$		
Operation and Conditions	Normalized Cache Key and Computation	Result Modification
$\langle c_1 + b_1 F_1 \rangle \oplus \langle c_2 + b_2 F_2 \rangle; F_1 \neq 0$	$\langle c_r + b_r F_r \rangle = \langle 0 + 1 F_1 \rangle \oplus \langle 0 + (b_2/b_1) F_2 \rangle$	$\langle \langle c_1 + c_2 + b_1 c_r \rangle + b_1 b_r F_r \rangle$
$\langle c_1 + b_1 F_1 \rangle \ominus \langle c_2 + b_2 F_2 \rangle; F_1 \neq 0$	$\langle c_r + b_r F_r \rangle = \langle 0 + 1 F_1 \rangle \ominus \langle 0 + (b_2/b_1) F_2 \rangle$	$\langle \langle c_1 - c_2 + b_1 c_r \rangle + b_1 b_r F_r \rangle$
$\langle c_1 + b_1 F_1 \rangle \otimes \langle c_2 + b_2 F_2 \rangle; F_1 \neq 0$	$\langle c_r + b_r F_r \rangle = \langle (c_1/b_1) + F_1 \rangle \otimes \langle (c_2/b_2) + F_2 \rangle$	$\langle b_1 b_2 c_r + b_1 b_2 b_r F_r \rangle$
$\langle c_1 + b_1 F_1 \rangle \oslash \langle c_2 + b_2 F_2 \rangle; F_1 \neq 0$	$\langle c_r + b_r F_r \rangle = \langle (c_1/b_1) + F_1 \rangle \oslash \langle (c_2/b_2) + F_2 \rangle$	$\langle (b_1/b_2) c_r + (b_1/b_2) b_r F_r \rangle$
$\max(\langle c_1 + b_1 F_1 \rangle, \langle c_2 + b_2 F_2 \rangle);$ $F_1 \neq 0$, Note: same for min	$\langle c_r + b_r F_r \rangle = \max(\langle 0 + 1 F_1 \rangle, \langle (c_2 - c_1)/b_1 + (b_2/b_1) F_2 \rangle)$	$\langle \langle c_1 + b_1 c_r \rangle + b_1 b_r F_r \rangle$
any $\langle op \rangle$ not matching above: $\langle c_1 + b_1 F_1 \rangle \langle op \rangle \langle c_2 + b_2 F_2 \rangle$	$\langle c_r + b_r F_r \rangle = \langle c_1 + b_1 F_1 \rangle \langle op \rangle \langle c_2 + b_2 F_2 \rangle$	$\langle c_r + b_r F_r \rangle$

Table 1: Input and output summaries of the *ComputeResult*, *GetNormCacheKey*, and *ModifyResult* routines.

for two AADD nodes). If we consider this 4-tuple of floating-point values to be a point in Euclidean space, then we can measure the error between two 4-tuples $\langle u_1, u_2, u_3, u_4 \rangle$ and $\langle v_1, v_2, v_3, v_4 \rangle$ as the L_2 (Euclidean) distance between these points. As a consequence of the triangle inequality for four dimensions, we know that $\sqrt{(u_1 - v_1)^2 + (u_2 - v_2)^2 + (u_3 - v_3)^2 + (u_4 - v_4)^2} \leq \epsilon \Rightarrow |\sqrt{u_1^2 + u_2^2 + u_3^2 + u_4^2} - \sqrt{v_1^2 + v_2^2 + v_3^2 + v_4^2}| \leq \epsilon$. This is shown graphically for two dimensions in Figure 4.

Based on this inequality, we can use the following approximate hashing scheme for $\langle v_1, v_2, v_3, v_4 \rangle$: Compute the L_2 distance between $\langle v_1, v_2, v_3, v_4 \rangle$ and the origin; For hashing, extract only the bits from the floating-point base representing a fractional portion greater than ϵ and use this as the integer representation of the hash key (we are effectively discretizing the distances into buckets of width ϵ); For equality, test that the true L_2 metric between $\langle u_1, u_2, u_3, u_4 \rangle$ and $\langle v_1, v_2, v_3, v_4 \rangle$ is less than ϵ .

While this hashing scheme does not guarantee that all 4-tuples having distance from the origin $\langle 0, 0, 0, 0 \rangle$ within ϵ hash to the same bucket (some 4-tuples within ϵ could fall over bucket boundaries), we found that with bucket width $\epsilon = 1e-9$ and numerical precision error generally less than $1e-13$, there was only a small probability of two nodes equal within numerical precision straddling a bucket boundary. For the empirical results described in Section 6, this hashing scheme was sufficient to prevent any uncontrollable cases of numerical precision error.

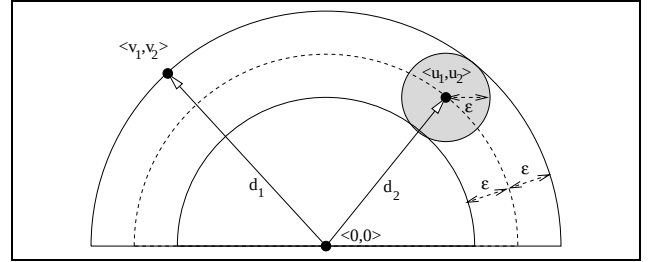


Figure 4: A 2D visualization of the hashing scheme we use. All points within ϵ of $\langle u_1, u_2 \rangle$ (the dark circle) lie within the ring having outer and inner radius $\sqrt{u_1^2 + u_2^2} \pm \epsilon$. Thus, a hashing scheme which hashes all points within ϵ of $\langle u_1, u_2 \rangle$ to the same bucket guarantees that all points within ϵ of $\langle u_1, u_2 \rangle$ also hash to the same bucket. We can use squared distances to avoid a $\sqrt{}$, thus the hash key for $\langle v_1, v_2 \rangle$ can be efficiently computed as the squared distance from $\langle v_1, v_2 \rangle$ to the origin.

5 Theoretical Results

Here we present two fundamental results for AADDs:

Theorem 5.1. *The time and space performance of Reduce and Apply for AADDs is within a multiplicative constant of that of ADDs in the worst case.*

Proof Sketch. Under the same variable ordering, an ADD is equivalent to a non-canonical AADD with fixed edge weights $c = 0, b = 1$. Thus the ADD *Reduce* and *Apply* algorithms can be seen as analogs of the AADD algorithms without the additional constant-time

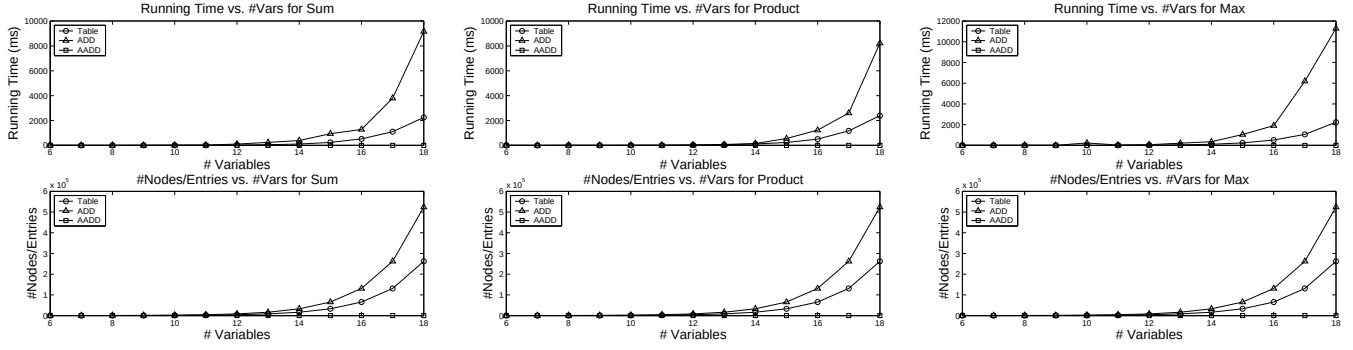


Figure 5: Comparison of *Apply* operation running time (top) and table entries/nodes (bottom) for tables, ADDs and AADDs. Left to Right: $(\sum_i 2^i x_i) \oplus (\sum_i 2^i x_i)$, $(\gamma^{\sum_i 2^i x_i}) \otimes (\gamma^{\sum_i 2^i x_i})$, $\max(\sum_i 2^i x_i, \sum_i 2^i x_i)$. Note the linear time/space for AADDs.

and constant-space overhead of normalization. Since normalization can only increase the number of *Reduce* and *Apply* cache hits and reduce the number of cached nodes, it is clear that an AADD must generate equal or fewer *Reduce* and *Apply* calls and have equal or fewer cached nodes than the corresponding ADD. This allows us to conclude that in the worst case where the AADD generates as many *Reduce* and *Apply* calls and cache hits as the ADD, the AADD is still within a multiplicative constant of the time and space required by the ADD.

Theorem 5.2. *There exist functions F_1 and F_2 and an operator op such that the running time and space performance of $Apply(F_1, F_2, op)$ for AADDs can be linear in the number of variables when the corresponding ADD operations are exponential in the number of variables.*

Proof Sketch. Two functions and *Apply* operation examples where this holds true are $\sum_{i=1}^n 2^i x_i \oplus \sum_{i=1}^n 2^i x_i$ and $\prod_{i=1}^n \gamma^{2^i x_i} \otimes \prod_{i=1}^n \gamma^{2^i x_i}$. (Examples of these operands as ADDs and AADDs were given in Figures 1c and 2.) Because these computations result in a number of terminal values exponential in n , the ADD operations must require time and space exponential in n . On the other hand, it is known that the operands can be represented in linear-sized AADDs. Due to this structure, the *Apply* algorithm will begin by recursing on the high branch of both operands to depth n . Then, at each step as it returns and recurses down the low branch, the respective additive and multiplicative differences will be normalized out due to the canonical caching scheme, thus yielding cache hits for all low branches. This results in n cached nodes and $2n$ *Apply* calls for the AADD operations.

6 Empirical Results

6.1 Basic Operations

Figure 5 demonstrates the relative time and space performance of tables, ADDs, and AADDs on a number of basic operations. These verify the exponential to linear space and time reductions proved in Theorem 5.2.

6.2 Bayes Nets

For Bayes nets (BNs), we simply evaluate the variable elimination algorithm [6] (under a fixed, greedy tree-width minimizing variable ordering) with the conditional

probability tables (CPTs) P_j and corresponding operations replaced with those for tables, ADDs, and AADDs:

$$\sum_{x_i \notin \text{Query}} \prod_{P_1 \dots P_j} P_1(x_1 | \text{Parents}(x_1)) \dots P_j(x_j | \text{Parents}(x_j))$$

Table 2 shows the total number of table entries/nodes required to represent the original network and the total running time of 100 random queries (each consisting of one query variable and one evidence variable) for a number of publicly available BNs (<http://www.cs.huji.ac.il/labs/compbio/Repository>) and two *noisy-or* and *noisy-max* CPTs with 15 parent nodes.

Note that the intermediate probability tables were often too large for the tables or ADDs, but not the AADDs, indicating that the AADD was able to exploit additive or multiplicative structure in these cases. Also, the AADD yields an exponential to linear reduction on the *Noisy-Or-15* problem and a considerable computational speedup on the *Noisy-Max-15* problem by exploiting the additive and multiplicative structure inherent in these special CPTs [7].

6.3 Markov Decision Processes

For MDPs, we simply evaluate the value iteration algorithm [8] using a tabular representation and its extension for decision diagrams [9] to factored MDPs from the *SysAdmin* domain [10].⁵ Here we simply substitute tables, ADDs, and AADDs for the reward function R , value function V , and transition model dynamics P_i in the following factored MDP value iteration update:

$$V^{t+1}(s_1, \dots, s_n) = R(s_1, \dots, s_n) + \gamma \max_a \sum_{s'_1 \dots s'_n} \left[\prod_{P_1 \dots P_n} P_i(s'_i | \text{Parents}(s'_i), a) \right] V^t(s'_1, \dots, s'_n)$$

Figure 6 shows the relative performance of value iteration until convergence within 0.01 of the optimal value

⁵In these domains, there is a network of computers whose operational status (i.e., running or crashed) at each time step is a stochastic function of the previous status of computers with incoming connections. At each time step the task of the system administrator is to choose the computer to reboot in order to maximize the sum of operational computers over a discounted reward horizon.

Bayes Net	Table		ADD		AADD	
	# Table Entries	Running Time	# ADD Nodes	Running Time	# AADD Nodes	Running Time
Alarm	1,192	2.97 s	689	2.42 s	405	1.26 s
Barley	470,294	<i>EML*</i>	139,856	<i>EML*</i>	60,809	207 m
Campo	636	0.58	955	0.57 s	360	0.49 s
Hailfinder	9045	26.4 s	4511	9.6 s	2538	2.7 s
Insurance	2104	278 s	1596	116 s	775	37 s
Noisy-Or-15	65566	27.5 s	125356	50.2 s	1066	0.7 s
Noisy-Max-15	131102	33.4 s	202148	42.5 s	40994	5.8 s

Table 2: Number of table entries/nodes in the original network and variable elimination running times using tabular, ADD, and AADD representations for inference in various Bayes nets. **EML* denotes that a query exceeded the 1Gb memory limit.

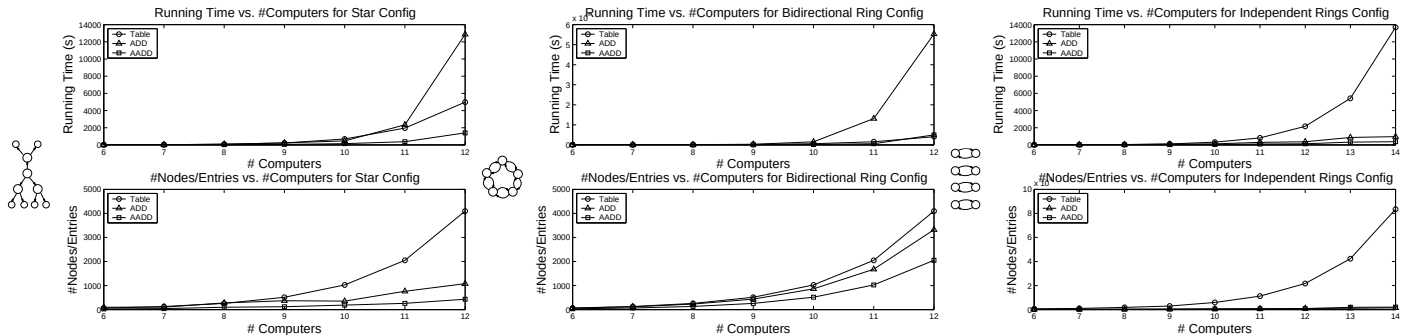


Figure 6: MDP value iteration running times (top) and number of entries/nodes (bottom) in the final value function using tabular, ADD, and AADD representations for various network configurations in the *SysAdmin* problem.

for networks in a star, bidirectional, and independent ring configuration. While the reward and transition dynamics in the *SysAdmin* problem have considerable additive structure, we note that the exponential size of the AADD (as for all representations) indicates that little additive structure survives in the *exact* value function. Nonetheless, the AADD-based algorithm still manages to take considerable advantage of the additive structure during computations and thus performs comparably or exponentially better than ADDs and tables.

7 Concluding Remarks

We have presented the AADD and have proved that its worst-case time and space performance are within a multiplicative constant of that of ADDs, but can be linear in the number of variables in cases where ADDs are exponential in the number of variables. And we have provided an empirical comparison of tabular, ADD, and AADD representations used in Bayes net and MDP inference algorithms, concluding that AADDs perform at least as well as the other two representations, and often yield an exponential time and space improvement over both.

In conclusion, we should emphasize that we have not set out to show that variable elimination and value iteration with AADDs are the best Bayes net and MDP inference algorithms available – many other approaches propose to handle similar structure efficiently via special-purpose problem-structure or algorithm modifications. Rather, we intended to show that transparently substituting AADDs in two diverse probabilistic inference algorithms that used tables or ADDs could yield exponen-

tial performance improvements over both by exploiting context-specific, additive, and multiplicative structure in the underlying representation. These results suggest that the AADD is likely to yield exponential time and space improvements for a variety of probabilistic inference algorithms that currently use tables or ADDs, or at the very least, a gross simplification of these algorithms.

References

- [1] R.I. Bahar, E.A. Frohm, C.M. Gaona, G.D. Hachtel, E. Macii, A. Pardo, F. Somenzi: Algebraic Decision Diagrams and Their Applications. ICCAD. (1993)
- [2] Boutilier, C., Friedman, N., Goldszmidt, M., Koller, D.: Context-specific independence in Bayesian networks. UAI. (1996)
- [3] Drechsler, R., Sieling, D.: BDDs in theory and practice. Software Tools for Technology Transfer. (2001)
- [4] Bryant, R.E.: Graph-based algorithms for Boolean function manipulation. IEEE Trans. on Computers. (1986)
- [5] Rudell, R.: Dynamic variable ordering for ordered binary decision diagrams. ICCAD. (1993)
- [6] Zhang, N.L., Poole, D.: A simple approach to bayesian network computations. CCAI. (1994)
- [7] Takikawa, M., D’Ambrosio, B.: Multiplicative factorization of noisy-max. UAI. (1999)
- [8] Bellman, R.E.: Dynamic Programming. Princeton University Press, Princeton, NJ (1957)
- [9] Hoey, J., St-Aubin, R., Hu, A., Boutilier, C.: SPUDD: Stochastic planning using decision diagrams. UAI. (1999)
- [10] Guestrin, C., Koller, D., Parr, R.: Max-norm projections for factored MDPs. IJCAI. (2001)

Probabilistic Reasoning with Hierarchically Structured Variables

Rita Sharma

Department of Computer Science
University of British Columbia
<http://www.cs.ubc.ca/spider/rsharma/>
rsharma@cs.ubc.ca

David Poole

Department of Computer Science
University of British Columbia
<http://www.cs.ubc.ca/spider/poole/>
poole@cs.ubc.ca

Abstract

Many practical problems have random variables with a large number of values that can be hierarchically structured into an abstraction tree of classes. This paper considers how to represent and exploit hierarchical structure in probabilistic reasoning. We represent the distribution for such variables by specifying, for each class, the probability distribution over its immediate subclasses. We represent the conditional probability distribution of any variable conditioned on hierarchical variables using *inheritance*. We present an approach for reasoning in Bayesian networks with hierarchically structured variables that dynamically constructs a *flat Bayesian network*, given some evidence and a query, by collapsing the hierarchies to include only those values necessary to answer the query. This can be done with a single pass over the network. We can answer the query from the flat Bayesian network using any standard probabilistic inference algorithm such as variable elimination or stochastic simulation. The domain size of the variables in the flat Bayesian network is independent of the size of the hierarchies; it depends on how many of the classes in the hierarchies are directly associated with the evidence and query. Thus, the representation is applicable even when the hierarchy is conceptually infinite.

1 Introduction

Many problem domains have discrete variables with a large number of values that can be represented *a priori* as an *abstraction* hierarchy (or an *is-a* hierarchy or *taxonomic* hierarchy) [Pearl, 1986; Mackworth *et al.*, 1985]. We call these variables *hierarchical variables*. Taxonomic hierarchies allow us to manage effectively the large state space of a variable because they allow facts and regularities to be revealed both at high and low levels of abstraction. In taxonomic hierarchies the information from high-level abstractions are automatically inherited by more specific concepts. Abstraction hierarchies also allow us to answer more abstract queries. As an example of a hierarchical variable, consider a random variable *LT* that describes the species of living things. The large number of

values of *LT* (i.e., millions of species) can be classified according to Linnaean taxonomy hierarchy.¹ Living things are divided into kingdoms (e.g., *plantae*, *animalia*), classes (e.g., mammals, birds, fish), all the way down to species.

In this paper, we look at two related problems with hierarchical variables in Bayesian networks. The first is the compact representation of conditional probability distributions. The second is how to exploit that representation in probabilistic inference for computational gain. We assume that the tree hierarchy of the values is fixed and does not change with the context.

To compute a posterior probability, given some evidence in a Bayesian network that has both simple and hierarchical variables, we construct a flat Bayesian network by collapsing the hierarchies and including only those values necessary to answer the query. We can answer the query from the flat Bayesian network using any standard inference algorithm. The domain size of the variables in the flat Bayesian network is independent of the size of the hierarchies; it depends on how many of the classes in the hierarchy are directly associated with the evidence and query.

2 Hierarchical Variables

We divide the discrete random variables into two categories: **simple variables** and **hierarchical variables**. We call Bayesian networks that have both simple and hierarchical variables *hierarchical Bayesian networks*.² We use upper case letters to denote simple random variables (e.g., X_1, X_2, X) and the actual value of these variables by small letters (e.g., a, b, x_1). The domain of X , written $Val(X)$, is the set of values that X can take on.

A hierarchical variable is a variable in which subsets of the values of the variable are represented hierarchically in a tree [Pearl, 1986]. We refer to a node in the tree as a *class*. The nodes in the tree represent subsets of the values of the variable. The root of the tree represents the set of all the values of the variable. The children of a node correspond to a partitioning of the set of values of the node. Thus, the subsets represented by the children of a class must be mutually disjoint and any class represents the union of its children in the tree.

¹http://www.wikipedia.org/wiki/Linnaean_taxonomy

²The term hierarchical Bayesian network used in this paper is different than the term hierarchical Bayesian model used in statistics.

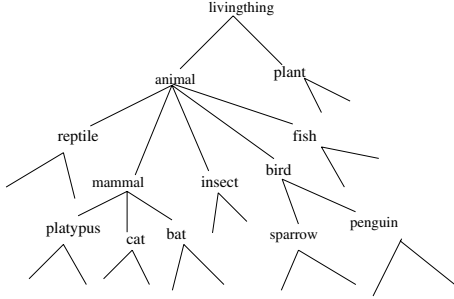


Figure 1: Part of a taxonomic hierarchy of living things

If a class C_j is a child of C_i , we say that C_j is an *immediate subclass* of C_i . The *subclass* relation is the symmetric transitive closure of the immediate subclass relation. That is, C_j is a subclass of C_i , written $C_j \leq C_i$, if C_j is a descendant of C_i in the abstraction hierarchy (or is C_i). The inverse of subclass is *superclass*. The conjunction of two classes C_j and C_k , written as $C_j \wedge C_k$, denotes the set intersection of C_j and C_k . This is empty unless one is a superclass of the other.

We denote the hierarchical variables using boldface upper-case letters (e.g., **LT**). For example, a part of the tree hierarchy of values of variable **LT** is shown in Figure 1. $tree(\mathbf{LT})$ denotes the tree hierarchy of **LT**.

3 Conditional Probabilities of Hierarchical Bayesian Networks

There are two main issues to be considered to represent hierarchical Bayesian networks:

- C1:** specifying the conditional probability for hierarchical variables;
- C2:** specifying how variables are conditioned on hierarchical parents.

The simplest case of **C1** is how to specify the prior probability of a hierarchical variable. The simplest case of **C2** is how to specify the conditional probability of a simple variable conditioned on a hierarchical variable. The more complicated cases are built from these simpler cases.

3.1 Prior Probability for Hierarchical Variables

The prior probability of a hierarchical variable can be specified in a top-down manner. For each class that has subclasses, we specify a probability distribution over its immediate subclasses. For example, suppose we want to specify the prior over the hierarchical variable **LT** as shown in Figure 1. The root of the hierarchy is the class *livingthing*. We specify the probability distribution over its immediate subclasses:

$$\begin{aligned} P(\text{animal}|\text{livingthing}) &= 0.4 \\ P(\text{plant}|\text{livingthing}) &= 0.6 \end{aligned}$$

We can specify the probability distribution over the immediate subclasses of *animal*. Suppose we have as part of this distribution:

$$\begin{aligned} P(\text{mammal}|\text{animal}) &= 0.3 \\ P(\text{bird}|\text{animal}) &= 0.2, \text{ etc.} \end{aligned}$$

We can specify the probability of an immediate subclass of *mammal* given *mammal*, with probabilities such as:

$$P(\text{bat}|\text{mammal}) = 0.1$$

We can compute the prior probability of any class in a recursive manner by multiplying the probability of class given its immediate superclass and the probability of its immediate superclass. The root has probability 1 (as it represents the set of all values). For example, given the probabilities as above, $P(\text{bat})$ can be computed as follows:

$$\begin{aligned} P(\text{bat}) &= P(\text{bat}|\text{mammal}) \times P(\text{mammal}|\text{animal}) \times \\ &\quad P(\text{animal}|\text{livingthing}) \\ &= 0.012 \end{aligned}$$

In this representation, computing the probability of any class is linear in the depth of the class in the hierarchy and otherwise is not a function of the hierarchy's size.

3.2 Hierarchical Variable Is the Parent of a Simple Variable

Suppose that **H** is a hierarchical variable, F is a simple variable, and **H** is the only parent of F . We can exploit any hierarchical structure by assuming that the influence on F is local in the hierarchy of **H**. To specify the conditional probability distribution (CPD) $P(F|\mathbf{H})$ we use the notion of a *default probability distribution*. The default probability distribution of F for class C_j , written $P_d(F|C_j)$, is assumed to be inherited by all subclasses of C_j , unless overridden by a more specific default probability distribution. We can represent $P(F|\mathbf{H})$ by specifying $P_d(F|C_j)$ for some classes C_j in the hierarchy of **H**. We call such classes C_j the *exceptional classes* of **H**. The idea of using “inheritance” with abstraction hierarchies is similar to the use of “context specific independence” for the compact representation of the CPTs in a Bayesian network [Boutilier et al., 1996].

To make this properly defined, we need to ensure that every terminal node in the tree hierarchy of **H** is covered by some default probability distribution. The union of all of the sets of values associated with the exceptional classes must equal $Val(\mathbf{H})$. This condition holds trivially if the root class is an exceptional class for F . If the root class is not an exceptional class, there must be an exceptional class along every path from the root.

The conditional probability of F for any class C_k of **H**, $P(F|C_k)$, can be computed as follows:

- if C_k does not have any exceptional subclasses, $P(F|C_k)$ is inherited from the default probability distribution of F for the lowest exceptional superclass C_j of C_k . Thus,

$$P(F|C_k) = P_d(F|C_j)$$

- otherwise, $P(F|C_k)$ can be derived from its children

$$P(F|C_k) = \sum_{\forall C_i \in \text{children}(C_k)} P(F|C_i) \times P(C_i|C_k)$$

Example: Consider a Bayesian network in which a hierarchical variable **LT** is the only parent of a simple variable *FLYING* with domain $\{\text{flying}, \neg\text{flying}\}$. To represent $P(\text{FLYING}|\mathbf{LT})$,

we need to define the default probability distributions over *FLYING* for the exceptional classes of **LT**. For example, we can state that *livingthings* have a low probability of *flying* by default, but *bird* and *insect* are exceptional because they have a high probability of flying. From the children of class *mammal*, a *bat* is exceptional because it has a high probability of flying. From the children of class *bird*, a *penguin* is exceptional because it has a very low probability of flying. Thus, to represent $P(\text{FLYING}|\text{LT})$ we could have:

$$\begin{aligned} P_d(\text{flying}|\text{livingthing}) &= 0.00001 \\ P_d(\text{flying}|\text{bird}) &= 0.5 \\ P_d(\text{flying}|\text{bat}) &= 0.3 \\ P_d(\text{flying}|\text{penguin}) &= 0.000001 \\ P_d(\text{flying}|\text{insect}) &= 0.4 \end{aligned}$$

Note that $P(\text{flying}|\text{livingthing}) \neq 0.00001$ as the class *livingthing* contains *bird* and *bat* that each have a much higher probability of flying.

3.3 The General Case

The general case is when a random variable (hierarchical or simple) can have any number and any kind of variables as parents. To represent the CPD of a hierarchical variable **H** conditioned on its parents, we assume that each class in the hierarchy of **H** has a probability distribution over its immediate subclasses that is conditioned on (some of) the parents of **H**. We can treat each of these classes as a simple variable. Thus, the problem of representing a (conditional) probability distribution over a hierarchical variable reduces to the problem of representing a (conditional) probability distribution over simple variables.

Let *F* be a simple variable that has both simple and hierarchical parents. Suppose $pa^s(F)$ denotes the simple parents of *F* and suppose $pa^h(F)$ denotes the hierarchical parents of *F*. Let **X** denote an assignment of a class to each hierarchical parent of *F*; we call **X** a *parent context*. $\mathbf{X} = (c_x^1, \dots, c_x^n)$, where c_x^i is a class in the tree hierarchy of the i^{th} hierarchical parent of *F*.

To represent $P(F|pa^s(F) \wedge pa^h(F))$ we specify the default probability distribution over *F* for different combinations of values for simple parents, $pa^s(F)$, and some parent contexts. The given parent contexts (parent contexts for which the default distribution over *F* has been defined) can be represented in a concept hierarchy C_p of partial ordering $C_p = (Z_p, \leq)$, where Z_p is the set of given parent contexts. A parent context $\mathbf{X} \in Z_p$ is a *super parent context* of parent context **Y**, written $\mathbf{Y} \leq \mathbf{X}$, if $\forall i, c_y^i \leq c_x^i$. The conditional distribution over *F* for a parent context **X** is inherited from the default probability distribution of *F* for the³ super parent context **Z** of **X** in Z_p that is not overridden by a more specific default probability distribution of *F* for parent context **Y** such that $\mathbf{X} \leq \mathbf{Y} < \mathbf{Z}$.

Example: Consider the Bayesian network shown in Figure 2

³We assume that there is only one most-specific super parent context. If there are more than one most-specific compatible parent contexts, with different default probability distributions, there must be a default distribution that disambiguates the two default distributions. We are thus forcing the user to disambiguate cases in which there would otherwise be a problem of multiple inheritance.

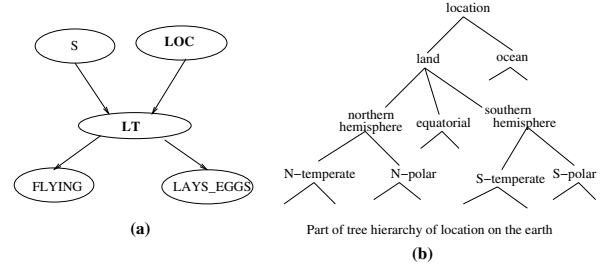


Figure 2: (a) A Bayesian network with simple and hierarchical variables (b) A part of the tree hierarchy for **LOC**

(a). The hierarchical variable **LOC** represents the location on earth. The simple variable *S* represents the season. Figure 2 (b) shows the part of the hierarchy of the location. We consider that *S* has only two values: (northern hemisphere) *summer* (May – October) and *winter* (November – April). The probability distribution of class *animal* of **LT** over its immediate subclasses can be conditioned on some of its parents. Suppose the distribution of class *animal* in the *equatorial* region is independent of season, so for parent context **X** = (*equatorial*) we could have a default distribution such as:

$$\begin{aligned} P_d(\text{reptile}|\text{animal} \wedge \mathbf{X}) &= 0.1 \\ P_d(\text{insect}|\text{animal} \wedge \mathbf{X}) &= 0.4 \\ P_d(\text{mammal}|\text{animal} \wedge \mathbf{X}) &= 0.2, \text{ etc.} \end{aligned}$$

In the polar regions the distribution of class *animal* over its children depends on the season, so for parent context **X** = (*N – polar*) we could have a default distribution such as:

$$\begin{aligned} P_d(\text{reptile}|\text{animal} \wedge S = \text{summer} \wedge \mathbf{X}) &= 0.01 \\ P_d(\text{mammal}|\text{animal} \wedge S = \text{summer} \wedge \mathbf{X}) &= 0.04 \\ P_d(\text{insect}|\text{animal} \wedge S = \text{summer} \wedge \mathbf{X}) &= 0.7, \text{ etc.} \end{aligned}$$

4 Construction of a Flat Bayesian Network

An observation about a hierarchical variable **H** can be *positive* or *negative*. An observation about **H** is positive when we observe that **H** is C_{obs} , where C_{obs} is a class in the hierarchy of **H**. This means that **H** has a value v_h such that $v_h \in C_{obs}$. An observation about **H** is negative when we observe that **H** is $\neg C_{obs}$. This means that **H** has a value v_h such that $v_h \notin C_{obs}$. Without loss of generality, we can assume that there is always one positive observation about **H**, denoted by C_{pos} . For example, suppose we have two positive observations $C1$ and $C2$ about **H**. If classes $C1$ and $C2$ are disjoint then observations about **H** are not consistent. If $C1$ is a subclass of $C2$, $C1 \wedge C2$ equals $C1$. If there are only negative observations about **H**, we assume root class is the positive observation. There can be multiple negative observations about **H** that are descendants of C_{pos} in the hierarchy of **H**.

We assume a query asks either for a distribution over a simple variable or for the probability of a particular class in a hierarchical variable.

Given some evidence (a set of observations) and a query, we construct a flat Bayesian network, B^f , from a hierarchical Bayesian network B^h by replacing each hierarchical variable with a simple variable whose domain includes only those values necessary to answer the query. The state space of a hi-

erarchical variable $\mathbf{H}$ can be abstracted because, for a given problem, only certain classes of $\mathbf{H}$ are supported directly by the evidence or relevant to the query. To construct B^f from B^h given some evidence and a query, we traverse B^h from the leaves upwards, prune those variables irrelevant to answer the query, and abstract the hierarchical variables. We abstract the hierarchical variables as part of a pass to remove variables that are irrelevant to the query. From the bottom-up, you can recursively prune any variable that has no children and is not observed or queried [Geiger *et al.*, 1990; Pearl, 1988] as well as abstract hierarchical variables when their children have been abstracted. We can then answer the query from the flat Bayesian network using any standard probabilistic inference algorithm.

4.1 Abstraction of Hierarchical Variables

Let $\mathbf{H}$ be a hierarchical variable with values $Val(\mathbf{H})$. The *abstraction* of the domain of $\mathbf{H}$, denoted by $part(\mathbf{H})$, is a partition of $Val(\mathbf{H})$. That is, $part(\mathbf{H})$ is a set of subsets of $Val(\mathbf{H})$ such that all sets in $part(\mathbf{H})$ are mutually disjoint and the union of all the sets equals $Val(\mathbf{H})$. We refer to a set in $part(\mathbf{H})$ as an *abstract value* of $\mathbf{H}$. An *abstraction of hierarchical variable* $\mathbf{H}$, denoted by H^a , is a simple variable with domain $Val(H^a) = part(\mathbf{H})$.

4.2 Finding a Safe Abstraction

In this section we describe a simple algorithm *Flat_Bayes* for constructing a flat Bayesian network B^f from a hierarchical Bayesian network B^h given some evidence and a query. A safe abstraction has the same posterior distribution over the query variable in B^f as in B^h . To develop a safe abstraction for efficient inference, the following constraints are followed in *Flat_Bayes*: 1. For every evidence or query, there must be an abstract value that directly associates with that evidence/query and conveys its effect. 2. All abstract values must be mutually disjoint and exclusive. The algorithm *Flat_Bayes* consists of three phases:

Phase0:

If a query asks for the probability of a particular class in hierarchical variable $\mathbf{H}$, we create an extra binary child Q of $\mathbf{H}$, which is true exactly when the query is true (i.e., $P_d(Q = true|C_{root}) = 0$ and $P_d(Q = true|C_q) = 1$, where C_{root} is the root class and C_q is the query class in the tree hierarchy of $\mathbf{H}$). The variable Q has the same probability as the query class. We thus reduce the problem to one of computing the probability of a simple variable.

Phase1: Abstract

In this phase *Flat_Bayes* traverses B^h from the leaves upwards. It prunes a variable that is not queried or observed and does not have any children [Geiger *et al.*, 1990; Pearl, 1988]. For each unpruned hierarchical variable $\mathbf{H}$, the algorithm computes its abstraction H^a as follows:

Case1: $\mathbf{H}$ is not observed

The hierarchical variable $\mathbf{H}$ can be a parent of several simple variables.⁴ We need to find the exceptional classes C_k of $\mathbf{H}$

⁴Note that because of the bottom-up nature of the algorithm, $\mathbf{H}$ can only have simple children at this stage of the algorithm.

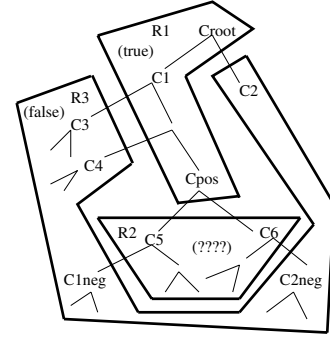


Figure 3: The abstraction hierarchy of $\mathbf{H}$ is divided into three regions $R1$, $R2$, and $R3$ by positive (C_{pos}) and negative ($C1_{neg}$, $C2_{neg}$) observations about $\mathbf{H}$

with respect to each of these children (i.e., classes that are associated with the evidence or the query). The domain of H^a is the set of non-empty abstract values for each exceptional class. The algorithm computes the abstract value, v^a , for every exceptional class C_k as follows:

- $v^a = C_k$, if C_k does not have any exceptional strict subclasses.
- Otherwise, $v^a = C_k - C_1 - \dots - C_m$, where $C_1, \dots, C_m$ are the highest exceptional strict subclasses of C_k . This set difference represents the set of all of the values that are in C_k and are not covered by other abstract values.

The abstract values that are empty are removed from the domain of H^a . To recursively abstract the parents of $\mathbf{H}$, the algorithm finds those strict super classes of the exceptional classes C_k , which are affected by $\mathbf{H}$'s parents.

Case2: $\mathbf{H}$ is observed

The observations about $\mathbf{H}$ divide its tree hierarchy into three regions $R1$, $R2$, and $R3$. $R1$ includes the classes that are true for the observation (superclasses of C_{pos}), $R2$ includes the classes that we know nothing about, and $R3$ includes the classes that we know are false. For example, suppose we have one positive (C_{pos}) and two negative ($C1_{neg}$, $C2_{neg}$) observations about $\mathbf{H}$ as shown in Figure 3. The regions $R1$, $R2$, and $R3$ are shown by the bold lines in Figure 3. We do not need to distinguish between the values of $\mathbf{H}$ that we know are false. The values of $\mathbf{H}$ can be partitioned into two disjoint and exclusive subsets: one (v_{false}) corresponds to all the values that are false and another ($v_{notknow}$) corresponds to all the values we know nothing about. Thus,

$$\begin{aligned} v_{false} &= \{(C_{root} - C_{pos}) \cup (\text{negative observations})\} \\ v_{notknow} &= \{C_{pos} - (\text{negative observations})\} \\ Val(H^a) &= \{v_{notknow}, v_{false}\} \end{aligned}$$

If $\mathbf{H}$ also has children, we need to partition its abstract value $v_{notknow}$ into subsets based on the evidence from its children. We need to find the exceptional classes C_k of $\mathbf{H}$ with respect to its children that are in region $R2$ in the hierarchy of $\mathbf{H}$. We can compute the abstract value vo^a that corresponds to each exceptional class C_k in the same way as described in Case1, but we need to remove all the false values. Thus, $vo^a = v^a - \{\text{negative observations that are subclasses of } C_k\}$

Let $vo_1^a, \dots, vo_k^a$ be the non-empty abstract values that correspond to exceptional classes C_k . Let $v_r = vo_1^a \cup \dots \cup vo_k^a$. Then, $Val(H^a) = \{vo_1^a, \dots, vo_k^a, v_{notknow} - v_r, v_{false}\}$

To recursively abstract the parents of $\mathbf{H}$, the algorithm finds those strict super classes of exceptional classes C_k and C_{pos} that are affected by $\mathbf{H}$'s parents.

Phase2: Construct tables

The algorithm constructs the CPT for variables X^a of a flat network. Let $pa^a(X^a)$ denote the abstracted parents of X^a . We compute $P(X^a = v^a | pa^s(X^a) = V_s \wedge pa^a(X^a) = \mathbf{V})$ for each value v^a of X^a and for each assignment V_s of $pa^s(X^a)$ and $\mathbf{V}$ of $pa^a(X^a)$ for the following two cases:

Case1: X^a is not abstracted but it has abstracted parents

As discussed in Section 3.3, $P(X^a = v^a | pa^s(X^a) = V_s \wedge pa^a(X^a) = \mathbf{V})$ is inherited from the default distribution of X^a for the parent context $\mathbf{C}$, $\mathbf{C} \in Z_p$, $\mathbf{V} \leq \mathbf{C}$ and $\exists \mathbf{Y} \in Z_p$ such that $\mathbf{V} \leq \mathbf{Y} < \mathbf{C}$, where Z_p is the set of parent contexts for which the default distribution of X^a has been defined.

Case2: X^a is an abstracted variable

Let Δ denote the assignment $pa^s(X^a) = V_s \wedge pa^a(X^a) = \mathbf{V}$. As discussed in Phase1, $v^a = C_k - C_1 - \dots - C_m$. Then,

$$P(X^a = v^a | \Delta) = P(C_k | \Delta) - \dots - P(C_m | \Delta)$$

As discussed in Section 3.1, to compute $P(C_k | \Delta)$ we need the probability distribution of all the super classes of C_k over their immediate subclasses for parent context $\mathbf{V}$. As discussed in Section 3.3, we can treat each of these classes as a simple variable. Thus, the probability distribution of a class over its immediate subclasses for parent context $\mathbf{V}$ can be computed in the same way as described in Case1.

The evidence for observed hierarchical variable $\mathbf{H}$ is translated into the corresponding abstract variable H^a of B^f . The evidence E_h for H^a in B^f is the disjunction of all those abstract values of $\mathbf{H}$ that are not false for the observation.

The domain size of the variables in B^f is independent of the size of the hierarchies; it depends on how many classes in their hierarchy are *exceptional* with respect to their relevant⁵ children in B^h . The running time to construct B^f depends on both the depth and number of the exceptional classes. After constructing B^f we can answer the query from B^f using any standard probabilistic inference algorithm, for example, variable elimination algorithm [Zhang and Poole, 1994] or stochastic simulation.

4.3 Correctness of Flat Bayesian Network

Let B^h be a hierarchical Bayesian network that has n discrete random variables, $\mathbf{X} = \{X_1, \dots, X_n\}$, and we want to answer some probabilistic query, $P(Q|E)$, where $Q, E \subseteq \mathbf{X}$. Q denotes the *query variables*. We observed that E is in the set e , $e \subseteq \text{domain}(E)$, $E \in e$ is the evidence. From B^h we can recursively prune any variable that has no children, and is not observed or queried [Geiger *et al.*, 1990; Pearl, 1988].

$$P(Q|E \in e) = \frac{P(Q \wedge E \in e)}{P(E \in e)}$$

⁵Variables irrelevant to the query are pruned by *Flat_Bayes*.

Let $X_1, \dots, X_s$ be the non-query non-observed random variables of B^h . $pa(X_i)$ denotes the parents of variable X_i . The marginal $P(Q \wedge E)$ can be computed as follows:

$$P(Q \wedge E \in e) = \sum_{X_s} \dots \sum_{X_1} \prod_{i=1}^n P(X_i | pa(X_i))_{\{E \in e\}}$$

Lemma 4.1 The posterior probability $P(Q|E)$ computed from a Bayesian network after replacing its variables X_k by their abstractions X_k^a is the same as if it is computed without replacing X_k , if the X_k^a are constructed from the partitions $part(X_k)$ that have the following property :

$$\forall X_k, \forall Y_j \in \text{children}(X_k), \forall V \in \text{part}(pa(Y_j)), \\ \forall v_1, v_2 \in V, P(Y_j | pa(Y_j) = v_1) = P(Y_j | pa(Y_j) = v_2) \quad (1)$$

Proof Sketch Summing over all the values of X_k is equivalent to summing over the partition of the values of X_k . Thus,

$$\sum_{v \in \text{Val}(X_k)} P(X_k = v | pa(X_k)) \times \prod_{l=1}^j P(Y_l | pa(Y_l)) \\ = \sum_{V \in \text{part}(X_k)} \sum_{v \in V} P(X_k = v | pa(X_k)) \times \prod_{l=1}^j P(Y_l | pa(Y_l)) \quad (2)$$

If (1) is true, we can distribute the product out of the innermost sum from the RHS of the above equality, leaving the term $\sum_{v \in V} P(X_k = v | pa(X_k))$, which is equal to $P(V | pa(X_k))$. Thus (2) is equal to

$$\sum_{V \in \text{part}(X_k)} P(X_k^a = V | pa(X_k)) \times \prod_{l=1}^j P(Y_l | pa(Y_l))$$

It is straightforward to incorporate evidence E and query Q into the proof.

Lemma 4.2 The algorithm *Flat_Bayes* partitions the values of the hierarchical variables X_k of B^h such that all its children satisfies the Equation (1).

Proof Sketch Let $part(X_k)$ denote the abstraction of the domain of hierarchical variable X_k computed by *Flat_Bayes*. X_k^a denotes the abstraction of X_k . If X_k is observed, *Flat_Bayes* removes a set V , $V \in \text{part}(X_k)$, such that all values in V are false. *Flat_Bayes* constructs the conditional probability tables for variables X_k^a of a flat Bayesian network using "inheritance" for the following two cases:

Case1: X_k^a is not abstracted but some of its parents are

As discussed in Phase2, $P(X_k^a = v^a | pa^s(X_k^a) = V_s \wedge pa^a(X_k^a) = \mathbf{V})$, $\mathbf{V} \in \text{part}(pa^h(X_k))$, is inherited from the default distribution of X_k^a for the parent context $\mathbf{C} \in Z_p$, $\mathbf{V} \leq \mathbf{C}$ and $\exists \mathbf{Y} \in Z_p$ such that $\mathbf{V} \leq \mathbf{Y} < \mathbf{C}$. Thus, $\forall v \in \mathbf{V}, P(X_k^a | pa^s(X_k^a) = V_s \wedge pa^a(X_k^a) = v) = P_d(X_k | pa^s(X_k) = V_s \wedge \mathbf{C})$. This implies Equation (1).

Case2: X_k^a is abstracted

As discussed in Phase2, to compute $P(X_k^a = v^a | pa^s(X_k^a) = V_s \wedge pa^a(X_k^a) = \mathbf{V})$ we need the conditional distribution of classes C_j , $C_j \in \text{tree}(X_k)$, over their immediate subclasses. We can treat a class C_j , $C_j \in \text{tree}(X_k)$, as a simple variable. Thus, the proof follows from Case1.

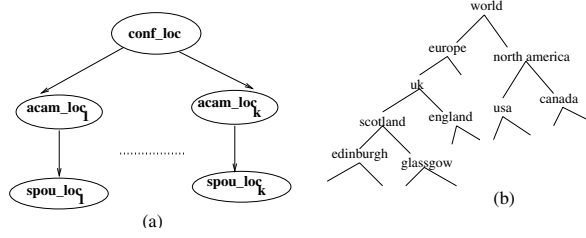


Figure 4: (a) Bayesian network representation of conference academics domain (b) A part of the tree hierarchy for location in the world

Theorem 4.1 The posterior probability $P(Q|E)$ computed from a flat Bayesian network as constructed by *Flat_Bayes* is equal to the posterior probability computed from a hierarchical Bayesian network.

Proof Sketch The theorem follows from Lemmas 4.1 and 4.2.

5 Experiments

We implement our *Flat_Bayes* algorithm on top of the variable elimination (VE) algorithm, we call it *Flat_Bayes + VE*. To get an idea of the performance of *Flat_Bayes + VE* compared to *VE*, we investigate how the performance of both algorithms vary with the number of hierarchical variables in the Bayesian network, with the depth of the tree hierarchies of the hierarchical variables, and with the depth of the exceptional classes.

To test the algorithms, we consider a simple conference domain. In this domain, the location of an academic is influenced by the location of the conference and the location of the spouse of the academic is influenced by the location of the academic. Let there be k academics. We assume that the locations of the different academics are not independent; they depend on the conference location. We consider that all location variables (conference location or person locations) have the same domain and are hierarchical variables. The Bayesian network representation of this domain is shown in Figure 4 (a). The part of the tree hierarchy for the location is shown in Figure 4 (b). Here we have $2k + 1$ hierarchical random variables. The hierarchical random variable **conf_loc** denotes the conference location. The hierarchical random variables **acam_loc**₁, ..., **acam_loc**_k denote the locations of the academics, and **spou_loc**₁, ..., **spou_loc**_k denote the locations of their spouses respectively.

To specify the conditional probability $P(\text{acam_loc}_k | \text{conf_loc})$, we need to specify the default distribution of each class in the hierarchy of **acam_loc**_k over its immediate subclasses for some parent contexts (for some classes in the hierarchy of **conf_loc**). We assume the conference location influences each academic's location very locally; we need to specify the default distribution of the classes over their immediate subclasses for very few parent contexts.

Example: Suppose we know that the academic is in the UK but don't know whether he is in Scotland, Wales or England. Now, knowing that the conference is taking place in the UK but not where in the UK, doesn't provide us any information

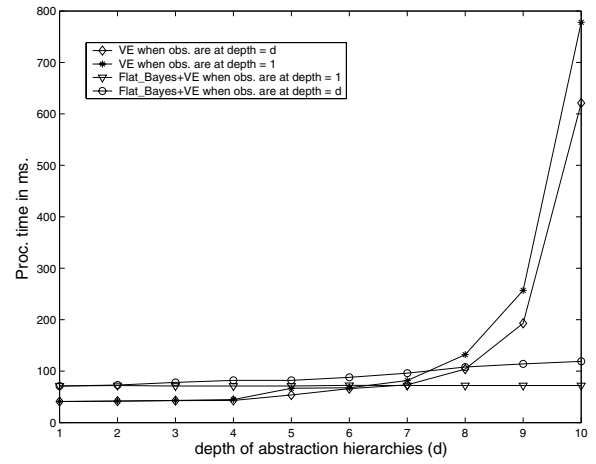


Figure 5: The average inference time of applying *VE* and *Flat_Bayes + VE* for $k = 2$ as a function of the depth of the hierarchy, and depth of the observations

about the location of the academic in the UK. However, if we know that the conference is taking place somewhere in Scotland, we can state that there is a high probability that the academic is in Scotland. We assume that knowing the conference location in Scotland (e.g., in Edinburgh), doesn't provide us any extra information about where they may be if not in Scotland. Thus, we could have a default probability distribution of the class **uk** in the hierarchy of **acam_loc**_k over its immediate subclasses conditioned on **conf_loc** such as:

$$\begin{aligned} P_d(\text{scotland} | \text{uk} \wedge \text{conf_loc} = \text{scotland}) &= 0.8 \\ P_d(\text{scotland} | \text{uk} \wedge \text{conf_loc} = \text{england}) &= 0.3 \\ P_d(\text{scotland} | \text{uk} \wedge \text{conf_loc} = \text{world}) &= 0.4 \end{aligned}$$

For the experiments, we observe the locations of the spouses and we want to compute where the conference is taking place. To determine how the inference time depends on the number of hierarchical variables, we vary k from 1 to 8. We represent the abstraction hierarchy of all location variables by a binary tree of depth d where d is a parameter. To determine how inference time depends on the depth of the hierarchy (d) we vary d from 1 to 10 with a step of 1. To determine how the inference time depends on the depth of the exceptional classes, for each value of d we vary the depth of the observation for the observed variables from 1 to d . We compute the posterior distribution of **conf_loc** by applying only *VE* and *Flat_Bayes + VE*.

Figure 5 shows the average inference time of both *VE* and *Flat_Bayes + VE* for the belief network that has only 2 academics as a function of the depth of the tree, and depth of the observations. Figure 6 shows the average inference time of both *VE* and *Flat_Bayes + VE* as a function of the number of academics and the depth of observations; here we consider the depth of the hierarchy is 8. Figure 5 shows that the inference time for *Flat_Bayes + VE* doesn't depend on the depth of the hierarchy but increases linearly as the depth of the observations increases. This increase is because the depth of the observations for **spou_loc**_k variables changes the depth and the number of exceptional classes for **acam_loc**_k and **conf_loc**.

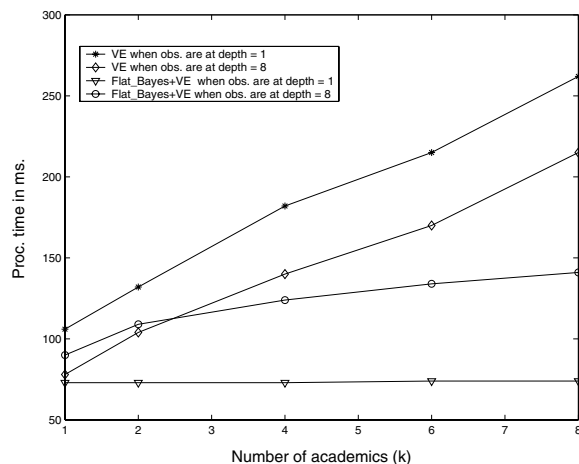


Figure 6: The average inference time of applying *VE* and *Flat_Bayes + VE* for $d = 8$ as a function of the number of academics and the depth of the observations

Note that *VE* takes less time as the depth of observation increases. This is because the domain of `spou_lock` variable reduces as the depth of the observation increases. The results shown in Figures 5 and 6 give us some indication of the overhead of constructing a flat Bayesian network and cases in which it may be more effective to use it rather than *VE* alone.

6 Related Work

There is very limited work on combining taxonomic hierarchies and probabilistic reasoning. The problem of evidential reasoning in a taxonomic hierarchy of hypotheses was first studied by Gordon and Shortliffe [1985] using the Dempster-Shafer (D-S) theory. Shenoy and Shafer [1986] improved the algorithm proposed by Gordon and Shortliffe and showed that belief functions can be propagated using local computations. Pearl [1986] provides a method for the same problem using Bayesian formalism. In all of these, a very restricted naive Bayes classifier network form was considered. In this paper we concentrate on general Bayesian networks that have both hierarchical and simple variables. Pearl's method involves traversing the whole hierarchy for each observation. We show in this paper that we do not need to do this. In more recent work, Koller and Pfeffer [1998] describe a representation language for combining frame-representation systems and Bayesian networks for representing complex structured domains. They are looking at different aspects of the problem, concentrating on multiple objects, combining first-order logic representation with probabilities.

7 Conclusion

Having the values of a variable hierarchically structured can make reasoning more efficient than reasoning over the set of all values. We show how, given a query and observations, we can dynamically construct a flat Bayesian network from the given hierarchical Bayesian network that can be used in any standard probabilistic inference algorithm. The size of

the flat network is independent of the size of the hierarchies; it depends on how many of the classes in the hierarchy are exceptional with respect to children that are observed or have observed descendants in a Bayesian network. Thus it is possible to have hierarchical variables with unbounded or infinite domains. For example, with a spatial hierarchy, as long as we have a way to compute the probability distribution over subclasses, there is no reason not to have a hierarchy that is infinitely detailed. It is only when we ask a query of a class or make observations that we need to restrict the size of the hierarchy.

Acknowledgments

This work was supported by NSERC Discovery Grant OG-POO44121. Thanks to Valerie McRae, Mark Crowley and Ben D'Andrea for proofreading.

References

- [Boutilier *et al.*, 1996] C. Boutilier, N. Friedman, M. Goldszmidt, and D. Koller. Context-specific independence in Bayesian networks. In *Proceeding of Twelfth Conf. on Uncertainty in Artificial Intelligence (UAI-96)*, pages 115–123, 1996.
- [Geiger *et al.*, 1990] D. Geiger, T. Verma, and J. Pearl. d-separation: From theorems to algorithms. In *Proceeding of Fifth Conf. on Uncertainty in Artificial Intelligence (UAI-90)*, pages 139–148, 1990.
- [Gordon and Shortliffe, 1985] J. Gordon and E.H. Shortliffe. A method of managing evidential reasoning in a hierarchical hypothesis space. *Artificial Intelligence*, 26:323–57, 1985.
- [Koller and Pfeffer, 1998] D. Koller and A. Pfeffer. Probabilistic frame-based systems. In *Proceedings of the 15th National Conference on Artificial Intelligence (AAAI)*, pages 580–587, 1998.
- [Mackworth *et al.*, 1985] A.K. Mackworth, J.A. Mulder, and W.S. Havens. Hierarchical arc consistency: exploiting structured domains in constraint satisfaction problems. In *Computational Intelligence*, volume 1(3), 1985.
- [Pearl, 1986] J. Pearl. On evidential reasoning in a hierarchy of hypotheses. *Artificial Intelligence*, 28:9–15, 1986.
- [Pearl, 1988] J. Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann Publishers Inc., 1988.
- [Shenoy and Shafer, 1986] P.P. Shenoy and G. Shafer. Propagating belief functions with local computations. *IEEE Expert*, 1:43–52, 1986.
- [Zhang and Poole, 1994] N.L. Zhang and D. Poole. A simple approach to Bayesian network computation. In *Proc. of the 10th Canadian Conference on Artificial Intelligence*, 1994.

Self Adaptive Particle Filter

Alvaro Soto

Pontificia Universidad Catolica de Chile
Department of Computer Science
Vicuna Mackenna 4860 (143), Santiago 22, Chile
asoto@ing.puc.cl

Abstract

The particle filter has emerged as a useful tool for problems requiring dynamic state estimation. The efficiency and accuracy of the filter depend mostly on the number of particles used in the estimation and on the propagation function used to re-allocate these particles at each iteration. Both features are specified beforehand and are kept fixed in the regular implementation of the filter. In practice this may be highly inappropriate since it ignores errors in the models and the varying dynamics of the processes. This work presents a self adaptive version of the particle filter that uses statistical methods to adapt the number of particles and the propagation function at each iteration. Furthermore, our method presents similar computational load than the standard particle filter. We show the advantages of the self adaptive filter by applying it to a synthetic example and to the visual tracking of targets in a real video sequence.

1 Introduction

The particle filter is a useful tool to perform dynamic state estimation via Bayesian inference. It provides great efficiency and extreme flexibility to approximate any functional non-linearity. The key idea is to use samples, also called particles, to represent the posterior distribution of the state given a sequence of sensor measurements. As new information arrives, these particles are constantly re-allocated to update the estimation of the state of the system.

The efficiency and accuracy of the particle filter depend mainly on two key factors: the number of particles used to estimate the posterior distribution and the propagation function used to re-allocate these particles at each iteration. The standard implementation of the filter specifies both factors beforehand and keeps them fixed during the entire operation of the filter. In this paper, we present a self adaptive particle filter that uses statistical methods to select an appropriate number of particles and a suitable propagation function at each iteration.

This paper is organized as follows. Section 2 provides background information about the standard particle filter.

Section 3 presents our method to adaptively estimate the number of particles. Section 4 presents our method to adaptively improve the propagation function. Section 5 shows the results of applying the self adaptive filter to a visual tracking task. Finally, Section 6 presents the main conclusions of this work.

2 Particle Filter

In Bayesian terms, the posterior distribution of the state can be expressed as:

$$p(x_t/\vec{y}_t) = \beta p(y_t/x_t) p(x_t/\vec{y}_{t-1}) \quad (1)$$

where β is a normalization factor; x_t represents the state of the system at time t ; and $\vec{y}_t$ represents all the information collected until time t . Equation (1) assumes that x_t totally explains the current observation y_t .

The particle filter provides an estimation of the posterior in Equation (1) in 3 main steps: sampling, weighting, and re-sampling. The sampling step consists of taking samples (particles) from the so-called dynamic prior distribution, $p(x_t/\vec{y}_{t-1})$. Next, in the weighting step, the resulting particles are weighted by the likelihood term $p(y_t/x_t)$. Finally, a re-sampling step is usually applied to avoid the degeneracy of the particle set. The key point that explains the efficiency of the filter comes from using a Markovian assumption and expressing the dynamic prior by:

$$p(x_t/\vec{y}_{t-1}) = \int p(x_t/x_{t-1}) p(x_{t-1}/\vec{y}_{t-1}) dx_{t-1}. \quad (2)$$

This expression provides a recursive implementation of the filter that allows it to use the last estimation $p(x_{t-1}/\vec{y}_{t-1})$ to select the particles x_{t-1}^i in the next iteration. These particles are then propagated by the dynamics of the process $p(x_t/x_{t-1}^i)$ to complete the sampling step.

At each iteration the operation of the particle filter can be seen as an importance sampling process [Tanner, 1996]. Importance sampling provides an efficient way to obtain samples from a density $p(x)$, in cases where this function can be evaluated, but it is not affordable or possible to sample from it directly. The basic idea in importance sampling is to use a proposal distribution $q(x)$, also called importance function, to obtain the samples x_i , and then weigh each sample using a compensatory term given by $p(x_i)/q(x_i)$. It is possible to

show [Tanner, 1996] that under mild assumptions the set of weighted-samples resembles the target distribution $p(x)$.

The sampling and weighting steps of the particle filter correspond to the basic steps of an importance sampling process. In this case, given that the true posterior $p(x_t/\bar{y}_t)$ is not known, the samples are drawn from an importance function that corresponds to the dynamic prior $p(x_t/\bar{y}_{t-1})$. Using this importance function, the compensatory terms are exactly the non normalized weights used in the weighting step of the particle filter. The methods presented in this paper use results from the theory of importance sampling to provide a self adaptive version of the particle filter.

3 Adaptive Selection of the Number of Particles

The selection of the number of particles is a key factor in the efficiency and accuracy of the particle filter. The computational load and the convergence of the filter depend on this number. Most applications select a fixed number of particles in advance, using ad hoc criteria, or statistical methods such as Monte Carlo simulations or some standard statistical bound [Boers, 1999]. Unfortunately, the use of a fixed number of particles is often inefficient. The dynamics of most processes usually produces great variability in the complexity of the posterior distribution¹. As a consequence, the initial estimation of the number of particles can be much larger than the real number of particles needed to perform a good estimation of the posterior distribution or, worse, at some point, the selected number of particles can be too small causing the filter to diverge.

The effect of the number of particles on the accuracy of the filter is determined by two factors: the complexity of the true density and how closely the proposal density mimics the true density. Both factors are very intuitive; the estimation of a more complex pdf requires a greater number of samples to correctly represent the less predictable shape of the function. Also, a greater mismatch between the proposal and the true densities produces many wasted samples located in irrelevant parts of the true distribution. Previous works to adaptively determine an adequate number of particles have failed to consider these two factors together [Fox *et al.*, 1999; Koeller and Fratkina, 1998; Fox, 2001].

Here, we propose two methods based in the theory of Statistics that can be used to adaptively estimate the number of particles to represent the target posterior distribution without adding a significant load to the normal operation of the filter. At each cycle of the particle filter, these techniques estimate the number of particles that, with a certain level of confidence, limits the maximum error in the approximation.

3.1 KLD-Sampling Revised

The KLD-Sampling algorithm [Fox, 2001] is a method to adaptively estimate the number of samples needed to bound the error of the particle filter. The error is measured by the Kullback-Leibler divergence (KL-divergence) between

the true posterior distribution and the empirical distribution, which is a well known nonparametric maximum likelihood estimate. KLD-Sampling is based on the assumption that the true posterior can be represented by a discrete piecewise constant distribution consisting of a set of multidimensional bins. This assumption allows the use of the χ^2 asymptotic convergence of the likelihood ratio statistic to find a bound for the number of particles N :

$$N > \frac{1}{2\epsilon} \chi_{k-1, 1-\delta}^2 \quad (3)$$

where ϵ is the upper bound for the error given by the KL-divergence, $(1 - \delta)$ is the quantile of the χ^2 distribution with $k - 1$ degrees of freedom, and k is given by the number of bins with support.

The problem with KLD-Sampling is the derivation of the bound using the empirical distribution, which has the implicit assumption that the samples comes from the true distribution. This is not the case for particle filters where the samples come from an importance function. Furthermore, the quality of the match between this function and the true distribution is one of the main elements that determines the accuracy of the filter, hence the suitable number of particles. The bound given by KLD-Sampling only uses information about the complexity of the true posterior, but it ignores any mismatch between the true and the proposal distribution.

To fix the problem of KLD-Sampling we need a way to quantify the degradation in the estimation using samples from the importance function. The goal is to find the equivalent number of samples from the importance and the true densities that capture the same amount of information about the latter.

In the context of Monte Carlo (MC) integration, [Geweke, 1989] introduced the concept of relative numerical efficiency (RNE), which provides an index to quantify the influence of sampling from an importance function. The idea behind RNE is to compare the relative accuracy of solving an integral using samples coming from both the true and the proposal density. Accuracy is measured according to the variance of the estimator of the integral.

If we use MC integration to estimate the mean value of the state ($E_{MC}(x)$), the variance of the estimator is given by [Doucet *et al.*, 2001]²:

$$Var[E_{MC}^N(x)] = Var_p(x)/N \quad (4)$$

where N is the number of samples coming from the true distribution $p(x)$, and the subscript p expresses that the variance involved is computed using the target distribution.

When the samples come from an importance function $q(x)$, the variance of the estimator corresponds to the variance of Importance Sampling (IS), which is given by [Geweke, 1989]:

$$Var[E_{IS}^N(x)] = E_q((x - E_p(x))^2 w(x)^2)/N_{IS} = \sigma_{IS}^2/N_{IS}, \quad (5)$$

where $w(x)$ corresponds to $p(x)/q(x)$ the weights of IS and N_{IS} is the number of samples coming from the importance function.

¹Here complexity is understood in terms of the amount of information needed to code the distribution.

²In the rest of the section, we concentrate on one iteration of the filter and we drop the subscript t .

To achieve similar levels of accuracy, the variance of both estimators should be equal. This allow us to find a relation that quantifies the equivalence between samples from the true and the proposal density,

$$N = \frac{N_{IS} \text{Var}_p(x)}{\sigma_{IS}^2} \quad (6)$$

Replacing (6) in (3) allows us to correct the bound given by KLD-Sampling when the samples do not come from the true distribution but from an importance function:

$$N_{IS} > \frac{\sigma_{IS}^2}{\text{Var}_p(x)} \frac{1}{2\epsilon} \chi_{k-1, 1-\delta}^2. \quad (7)$$

Using MC integration, $\text{Var}_p(x)$ and σ_{IS}^2 can be estimated by:

$$\text{Var}_p(x) = E_p(x^2) - E_p(x)^2 \approx \frac{\sum_{i=1}^N x_i^2 w_i}{\sum_{i=1}^N w_i} - E_p(x)^2$$

and

$$\sigma_{IS}^2 \approx \frac{\sum_{i=1}^N x_i^2 w_i^2}{\sum_{i=1}^N w_i} - \frac{2 \sum_{i=1}^N x_i w_i^2 E_p(x)}{\sum_{i=1}^N w_i} + \frac{\sum_{i=1}^N w_i^2 E_p(x)^2}{\sum_{i=1}^N w_i} \quad (8)$$

with $E_p(x) = \sum_{i=1}^n x_i w_i / \sum_{i=1}^n w_i$. Equation (8) shows that using appropriate accumulators, it is possible to calculate the bound incrementally keeping the $O(N)$ complexity of the filter.

3.2 Asymptotic Normal Approximation

Usually, the particle filter keeps track of the posterior density with the goal of estimating the mean or higher order moments of the state. This suggests an alternative error metric to determine the number of particles. Instead of checking the accuracy of the estimation of the posterior, it is possible to check the accuracy of a particle filter in the estimation of a moment of this density.

Under weak assumptions and using the strong law of large numbers, it is possible to show that at each iteration the estimation of the mean given by the particle filter is asymptotically unbiased [DeGroot, 1989]. Furthermore, if the variance of this estimator is finite, the central limit theorem justifies an asymptotic normal approximation for it [DeGroot, 1989], which is given by:

$$E_{PF}(x) \sim \mathcal{N}(E_p(x), \sigma_{IS}^2/N_{IS}) \quad (9)$$

where $\mathcal{N}(\mu, \sigma^2)$ denotes the normal distribution with mean μ and standard deviation σ .

Using this approximation, it is possible to build a one sided confidence interval for the number of particles that limits the error in the estimation of the mean:

$$P(|\frac{E_{PF}(x) - E_p(x)}{E_p(x)}| \leq \epsilon) \geq (1 - \alpha) \quad (10)$$

where $|\cdot|$ denotes absolute value; $E_p(x)$ is the true mean value of the state; ϵ corresponds to the desired error; and $(1 - \alpha)$ corresponds to the confidence level.

Following the usual derivation of confidence intervals, Equation (10) produces the following bound for the number of particles:

$$N_{IS} \geq \frac{Z_{1-\alpha/2}^2 \sigma_{IS}^2}{\epsilon^2 E_p(x_t)^2} \quad (11)$$

3.3 Testing the Bounds

Figure 1 shows the distributions used to test the bounds. The true distribution corresponds to $p(x) = 0.5\mathcal{N}(3, 2) + 0.5\mathcal{N}(10, 2)$ and the importance function to $q(x) = 0.5\mathcal{N}(2, 4) + 0.5\mathcal{N}(7, 4)$. In all the trials, the desired error was set to 0.01 and the confidence level to 95%.

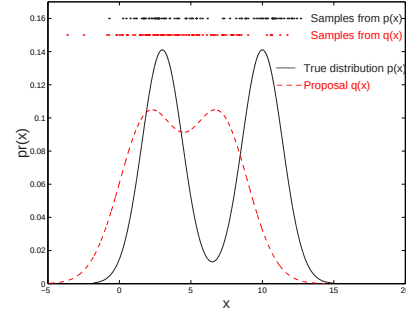


Figure 1: Inefficient allocation of samples due to a mismatch between $p(x)$ and $q(x)$.

Figure 2a shows the number of particles predicted by the different bounds. The predicted number of particles is highly consistent. Also, the revised versions of KLD-Sampling consistently require a larger number of particles than the original algorithm. This is expected because of the clear mismatch between the proposal and the true densities.

Figure 2b shows the resulting KL-divergence for each case. It is interesting to note how the practical results match closely the theoretical ones. Using the original version of KLD-Sampling, the error in the estimation is significantly greater than the one specified. However, when the same predicted number of particles is sampled from the true distribution (solid-line), the resulting error matches closely the one specified. This shows clearly that constraining the sampling to the right assumption, the original bound predicted by KLD-Sampling is correct. In the case of the revised versions of KLD-Sampling the resulting error using Equation (7) matches closely the one specified. In the same way, the error provided by the bound in Equation (11) also matches closely the level specified.

4 Adaptive Propagation of the Particles

The regular implementation of the particle filter propagates the particles using the dynamic prior $p(x_t/\vec{y}_{t-1})$. This strategy has the limitation of propagating the samples without considering the most recently available observation, y_t . Importance sampling suggests that one can use alternative propagation functions that can provide a better allocation of the samples, for example, a suitable functional of y_t . Unfortunately, the use of an arbitrary importance function significantly increases the computational load of the particle filter. In this case, as opposed to the standard particle filter, the estimation of each weight requires the evaluation of the dynamic prior. This section shows a method to build an importance function that takes into account the most recent observation y_t without increasing the computational complexity of the filter.

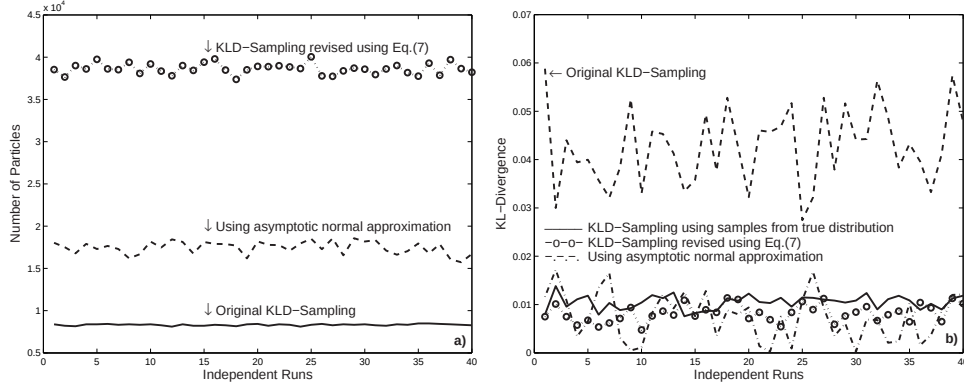


Figure 2: a) Number of particles given by each bound. b) Resulting KL-divergence.

4.1 Previous Work

In the literature about particle filters and importance sampling, it is possible to find several techniques that help to allocate the samples in areas of high likelihood under the target distribution. The most basic technique is rejection sampling [Tanner, 1996]. The idea of rejection sampling is to accept only the samples with an importance weight above a suitable value. The drawback is efficiency: there is a high rejection rate in cases where the proposal density does not match closely the target distribution. In [West, 1993], West presents a kernel-based approximation to build a suitable importance function. However, the computational complexity of the method is unaffordable. In the context of mobile robot localization, Thrun et al. [Thrun et al., 2000] propose to sample directly from the likelihood function, but in many applications this is not feasible or prohibitive.

Pitt and Shephard propose the auxiliary particle filter [Pitt and Shephard, 1999]. They augment the state representation with an auxiliary variable. To sample from the resulting joint density, they describe a generic scheme with computational complexity of $O(N)$. The disadvantage is the additional complexity of finding a convenient importance function. Pitt and Shephard provide just general intuitions about the form of a this function. In this paper we improve on this point by presenting a method to find a suitable importance function.

4.2 Adaptive Propagation of the Samples

Sampling from the dynamic prior in Equation (2) is equivalent to sample from the following mixture distribution:

$$p(x_t/\vec{y}_{t-1}) \approx \sum_{k=1}^n \beta_k p(x_t/x_{t-1}^k) \quad (12)$$

where the mixture coefficients β_k are proportional to $p(x_{t-1}/\vec{y}_{t-1})$. The key observation is that under this scheme the selection of each propagation density depends on the mixture coefficients β_k 's, which do not incorporate the most recent observation y_t . From an MC perspective, it is possible to achieve a more efficient allocation of the samples by including y_t in the generation of the coefficients. The intuition is that the incorporation of y_t increases the number of samples

drawn from mixture components $p(x_t/x_{t-1}^k)$ associated with areas of high probability under the likelihood function.

Under the importance sampling approach, it is possible to generate a new set of coefficients β_k^* that takes into account y_t by sampling from the importance function $p(x_{t-1}/\vec{y}_t)$. In this way, the set of samples x_t^i from the dynamic prior $p(x_t/\vec{y}_{t-1})$ is generated by sampling from the mixture,

$$\sum_{k=1}^n \beta_k^* p(x_t/x_{t-1}^k) \quad (13)$$

and then adding to each particle x_t^i a compensatory weight given by,

$$w_t^i = \frac{p(x_{t-1}^k/\vec{y}_{t-1})}{p(x_{t-1}^k/\vec{y}_t)}, \quad \text{with } x_t^i \sim p(x_t/x_{t-1}^k) \quad (14)$$

The resulting set of weighted samples $\{x_t^i, w_t^i\}_{i=1}^n$ still comes from the dynamic prior, so the computational complexity of the resulting filter is still $O(N)$. The extra complexity of this operation comes from the need to evaluate and to draw samples from the importance function $p(x_{t-1}^i/\vec{y}_t)$. Fortunately, the calculation of this function can be obtained directly from the operation of the regular particle filter. To see this clearly, consider the following:

$$\begin{aligned} p(x_t, x_{t-1}/\vec{y}_t) &\propto p(y_t/x_t, x_{t-1}, \vec{y}_{t-1}) p(x_t, x_{t-1}/\vec{y}_{t-1}) \\ &\propto p(y_t/x_t) p(x_t/x_{t-1}, \vec{y}_{t-1}) p(x_{t-1}/\vec{y}_{t-1}) \\ &\propto p(y_t/x_t) p(x_t/x_{t-1}) p(x_{t-1}/\vec{y}_{t-1}) \end{aligned} \quad (15)$$

Equation (15) shows that, indeed, the regular steps of the particle filter generate an approximation of the joint density $p(x_t, x_{t-1}/\vec{y}_t)$. After re-sampling from $p(x_{t-1}/\vec{y}_{t-1})$, propagating these samples with $p(x_t/x_{t-1})$, and calculating the weights $p(y_t/x_t)$, the set of resulting sample pairs (x_t^i, x_{t-1}^i) with correcting weights $p(y_t/x_t^i)$ forms a valid set of samples from the joint density $p(x_t, x_{t-1}/\vec{y}_t)$. Considering that $p(x_{t-1}/\vec{y}_t)$ is just a marginal of this joint distribution, the set of weighted-samples x_{t-1}^i are valid samples from it.

The previous description provides an adaptive algorithm that allows the particle filter to use y_t in the allocation of the samples. First, N particles are used to generate the importance function $p(x_{t-1}/\vec{y}_t)$. Then, starting from this importance function, another N particles are used to generate

the desired posterior $p(x_t/\bar{y}_t)$. The relevant compensatory weights are calculated according to Equation (14) and the likelihood term $P(y_t/x_t)$. The resulting filter has a computational complexity of $O(2N)$.

In the previous algorithm the overlapping between a regular iteration of the regular particle filter and the process of generating the importance function provides a convenient way to perform an online evaluation of the benefits of updating the dynamic prior with information from the last observation. While in cases of a poor match between the dynamic prior and the posterior distribution the updating of the dynamic prior can be beneficial, in cases where these distributions agree, the updating does not offer a real advantage, and the extra processing should be avoided. To our current knowledge, this issue has not been addressed before.

The basic idea is to quantify at each iteration of the particle filter the trade-off between continuing drawing samples from a known but potentially inefficient importance function $p(x_{t-1}/\bar{y}_{t-1})$ versus incurring in the cost of building a new importance function $p(x_{t-1}/\bar{y}_t)$ that provides a better allocation of the samples under the likelihood function. The important observation is that, once the regular particle filter reaches an adequate estimate, it can be used to estimate both the posterior distribution $p(x_t/\bar{y}_t)$ and the updated importance function $p(x_{t-1}/\bar{y}_t)$.

The last step of the algorithm is to find a metric that provides a way to quantify the efficiency in the allocation of the samples. Considering that the efficiency in the allocation of the samples depends on how well the dynamic prior resembles the posterior distribution, an estimation of the distance between these two distributions is a suitable index to quantify the effectiveness of the propagation step. We found a convenient way to estimate the Kullback-Leibler divergence (KL-divergence) between these distributions, and in general between a target distribution $p(x)$ and an importance function $q(x)$:

$$KL(p(x), q(x)) \approx \log(N) - H(\hat{w}_i). \quad (16)$$

Equation (16) states that for a large number of particles, the KL-divergence between the dynamic prior and the posterior distribution can be estimated by calculating how far the entropy of the distribution of the weights, $H(\hat{w}_i)$, is from the entropy of a uniform distribution ($\log(N)$). This is an intuitive result because in the ideal case of importance sampling, where $p(x) = q(x)$, all the weights are equal. In consequence the entropy of the weights is a suitable value to quantify the efficiency in the allocation of the samples.

5 Application

To illustrate the advantages of the self adaptive particle filter, we use a set of frames of a video sequence consisting of two children playing with a ball. The goal is to keep track of the positions of the ball and the left side child. Each hypothesis about the position of a target is given by a bounding box defined by its height, width, and the coordinates of its center. The motion model used for the implementation of the particle filter corresponds to a Gaussian function of zero mean and diagonal covariance matrix with standard deviations of 20 for the center of each hypothesis and 0.5 for the width and height.

Figure 3 shows the results of tracking the targets using the self adaptive particle filter. The bounding boxes correspond to the most probable hypotheses in the sample set used to estimate the posterior distributions of the states. In the estimation of the number of particles, we just consider the x and y coordinates of the center of the bounding boxes, assuming independence to facilitate the use of Equation (7). We set the desired error to 0.01 and the confidence level to 95%. A minimum number of 1000 samples is always used to ensure that convergence has been achieved. In the adaptation of the propagation function we set the threshold for the entropy of the weights in 2.

Figure 4-left shows the number of particles needed to estimate the posterior distribution of the ball at each frame without adapting the propagation function. Figure 4-right shows the number of particles in the case of adapting the importance function. The tracking engine decides to adapt the importance function at all the frames where the ball travels from one child to the other (Frames 3-7).

In the case of tracking the child, the result shows that there is not a major difference between the self adaptive particle filter and the regular filter. The self adaptive filter needs a roughly constant number of particles during the entire sequence without needing to adapt the importance function. This is expected because the child has only a small and slow motion around a center position during the entire sequence. Therefore the stationary Gaussian motion model is highly accurate and there is not a real advantage of adapting the number of particles or the propagation function.

In the case of the ball, the situation is different. During the period that the ball travels from one child to the other (Frames 3 to 7), it has a large and fast motion, therefore the Gaussian motion model is a poor approximation of the real motion. As a consequence there is a large mismatch between the dynamic prior and the posterior distribution. This produces an inefficient allocation of the samples and the estimate without adapting the importance function needs a larger set of samples to populate the relevant parts of the posterior. In contrast, when adapting the importance function during Frames 3 to 7 it is possible to observe a significant reduction in the number of samples due to a better allocation of them.

6 Conclusions

In this paper we present a self adaptive version of the particle filter that uses statistical techniques to estimate a suitable number of particles and to improve the propagation function. In terms of the estimation of the number of particles, the validation of the bounds using a synthetic example shows that the empirical results match closely the theoretical predictions. In particular, the results indicate that by considering the complexity of the true density and how closely the proposal density mimics the true density, the new bounds show a clear improvement over previous techniques such as KLD-Sampling.

The mechanisms used by the self adaptive filter to adapt the importance function and to identify when the adaptation of the importance function may be beneficial proved to be highly relevant. Using these mechanisms to track targets in a real video sequence, the self adaptive filter was able to ef-

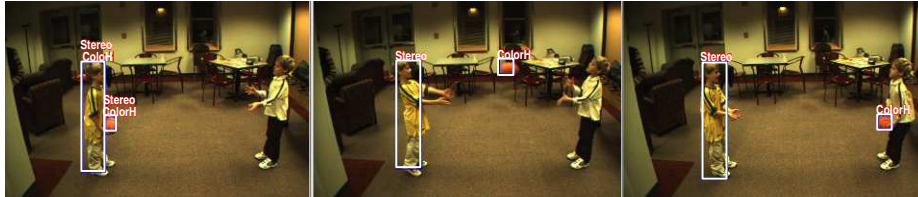


Figure 3: Tracking results for the ball and the left side child for frame 1, 5, and 14. The bounding boxes correspond to the most probable hypotheses in the sample set used to estimate the posterior distributions.

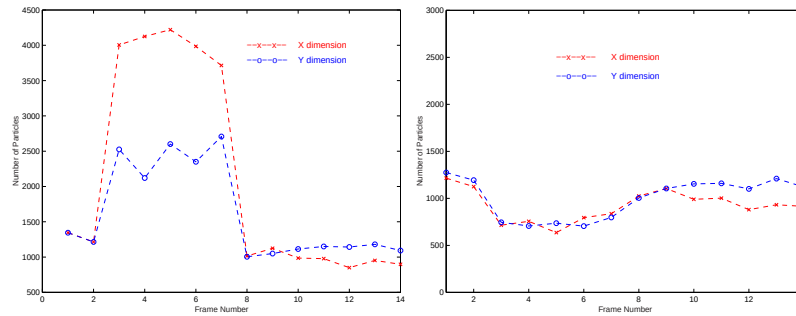


Figure 4: Number of particles used at each iteration to track the ball. Left: without adapting the importance function. Right: Adapting the importance function.

ficiently track targets with different motions using a general Gaussian motion model. Furthermore, by avoiding the need of overestimating the number of particles and by allocating these particles in areas of high likelihood, the self adaptive filter proved to operate with a similar computational complexity to the regular particle filter.

References

- [Boers, 1999] Y. Boers. On the number of samples to be drawn in particle filtering. In *IEEE Colloquium on Target Tracking: Algorithms and Applications*, pages 5/1–5/6, 1999.
- [DeGroot, 1989] M. DeGroot. *Probability and Statistics*. Addison Wesley, 2nd edition, 1989.
- [Doucet et al., 2001] A. Doucet, N. de Freitas, and N. Gordon. An introduction to sequential Monte Carlo methods. In *Sequential Monte Carlo Methods in Practice*, pages 3–14. Springer, 2001.
- [Fox et al., 1999] D. Fox, W. Burgard, F. Dellaert, and S. Thrun. Monte Carlo localization: Efficient position estimation for mobile robots. In *Proc. of the nat. conf. on AI (AAAI)*, 1999.
- [Fox, 2001] D. Fox. KLD-Sampling: Adaptive particle filters. In *Advances in Neural Information Processing Systems 14 (NIPS)*, 2001.
- [Geweke, 1989] J. Geweke. Bayesian inference in econometric models using Monte Carlo integration. *Econometrica*, 57:1317–1339, 1989.
- [Koeller and Fratkina, 1998] D. Koeller and R. Fratkina. Using learning for approximation in stochastic processes. In *Proc. of the international conf. on machine learning (ICML)*, 1998.
- [Pitt and Shephard, 1999] M. Pitt and N. Shephard. Filtering via simulation: Auxiliary particle filters. *Journal of the American Statistical Association*, 94(446):590–599, June 1999.
- [Tanner, 1996] M. Tanner. *Tools for Statistical Inference*. Springer-Verlag, 3rd edition, 1996.
- [Thrun et al., 2000] S. Thrun, D. Fox, and W. Burgard. Monte Carlo localization with mixture proposal distribution. In *AAAI National Conf. on Artificial Intelligence*, pages 859–865, Austin, Tx, 2000.
- [West, 1993] M. West. Approximating posterior distributions by mixtures. *Journal of the Royal Statistical Society, Serie B*, 55(2):409–422, 1993.

USER INTERFACE AND MODELING

Dynamically Constructed Bayes Nets for Multi-Domain Sketch Understanding

Christine Alvarado* and Randall Davis†

MIT CSAIL

Cambridge, MA 02139

{calvarad,davis}@csail.mit.edu

Abstract

This paper presents a novel form of dynamically constructed Bayes net, developed for multi-domain sketch recognition. Our sketch recognition engine integrates shape information and domain knowledge to improve recognition accuracy across a variety of domains using an extendible, hierarchical approach. Our Bayes net framework integrates the influence of stroke data and domain-specific context in recognition, enabling our recognition engine to handle noisy input. We illustrate this behavior with qualitative and quantitative results in two domains: hand-drawn family trees and circuits.

1 Introduction

There is a gap between how people naturally express ideas and the ability of computers to use that information. For example, while sketching provides a natural way to record design ideas in many domains, sketches are, unavoidably, static pictures. Computer aided design tools, on the other hand, offer powerful capabilities, but require designers to interact through buttons and menus. The hardware to draw on the computer exists; the missing element is the computer's ability to interpret hand-drawn symbols in a domain. To address this problem, we are constructing a general sketch recognition architecture applicable to a number of domains, and capable of parsing freely-drawn strokes in real time and interpreting them as depicting objects in the domain of interest.

Sketch recognition involves two related subproblems: stroke segmentation and symbol recognition. *Segmentation* determines which strokes should be grouped to form a single symbol. *Symbol recognition* determines what symbol a given set of strokes represents.

The difficulty of doing segmentation and recognition simultaneously has led previous approaches to place constraints on the user's drawing style, or focus on tasks where assumptions can greatly reduce segmentation complexity. For example, the multimodal approach in [Wu *et al.*, 1999] assumes that each symbol will be drawn independently, and that

the user will often say the name of the symbol when drawing it. The approach in [Kara and Stahovich, 2004] assumes that the diagram (a feedback control system) consists of shapes linked by arrows, which is not true in other domains.

While these systems have proven useful for their respective tasks, we aimed to create a more general system, independent of drawing assumptions in any one domain. Our system is designed, instead, to be applied to a number of symbolic domains by giving it descriptions of shapes and commonly occurring combinations of shapes. We use these descriptions in a three-stage, constraint-based approach to recognition. Our system first relies on a rough processing of the user's strokes to generate a number of interpretation hypotheses. In the second stage, the system uses a novel form of dynamically constructed Bayes net to determine how well each hypothesis fits the data. In the third stage, it uses this evaluation to guide further hypothesis generation. This paper focuses specifically on the second stage, exploring hypothesis evaluation; the other two stages are described in [Alvarado, 2004].

Using Bayes nets for hypothesis evaluation offers two advantages over previous constraint-based recognition approaches (e.g., [Futrelle and Nikolakis, 1995; Hammond and Davis, 2004]). First, our system can interpret drawings as they develop, identifying shapes before they are complete. Second, the system's belief in a hypothesis can be influenced by both the strokes and the context in which the shape appears, allowing the system to cope with noise in the drawing.

We constructed a sketch recognition engine and used it in two domains: family tree diagrams and circuit diagrams. We show that the Bayes net successfully allows domain-specific context to help the system overcome noise in the stroke data, reducing interpretation errors compared to a baseline system.

2 Dynamically Constructed Graphical Models

While time-based graphical models (e.g., Dynamic Bayes Nets) have been widely used, they are not suitable for sketch understanding, for two reasons. First, we must model shapes based on two-dimensional constraints (e.g., touches) rather than on temporal constraints (i.e., follows). Second, our models cannot simply unroll in time as data arrives: we cannot necessarily predict the order in which the user will draw the strokes, and things drawn previously can be changed.

While it is not difficult to use Bayes nets to model spatial relationships, static Bayes nets are not suitable for our task

*Currently at Harvey Mudd College, Claremont, CA, 91711

†This work was funded by the MIT iCampus project supported by Microsoft and MIT Project Oxygen.

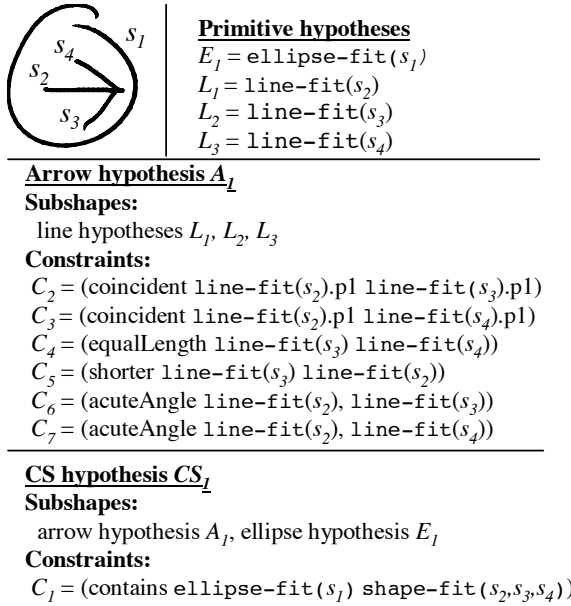


Figure 1: A single current source hypothesis (CS_1) and associated lower-level hypotheses.

because we cannot predict *a priori* the number of strokes or symbols the user will draw. For sketch recognition, as in related dynamic tasks, models to reason about specific problem instances (e.g., a particular sketch) must be dynamically constructed in response to the input. This problem is known as the task of *knowledge-based model construction* (KBMC).

Early approaches to KBMC focused on generating Bayes nets from probabilistic knowledge bases [Glessner and Koller, 1995; Haddawy, 1994]. A recently proposed representation uses generic template knowledge directly as Bayes net fragments that can be instantiated and linked together at run-time [Laskey and Mahoney, 1997]. Finally, Koller *et al.* have developed a number of object-oriented frameworks [Koller and Pfeffer, 1997; Pfeffer *et al.*, 1999; Getoor *et al.*, 1999]. These models represent knowledge in terms of relationships among objects and can be instantiated dynamically in response to the number of objects in a particular situation.

Although these frameworks are powerful, they are not directly suitable for sketch recognition. First, because of the size of the networks we encounter, it is sometimes desirable to generate only part of a complete network, or to prune nodes from the network. In reasoning about nodes in the network, we must account for the fact that the network may not be fully generated or relevant information may have been pruned from it (see [Alvarado, 2004] for details). Second, these previous models have been optimized for responding to specific queries about a single node in the network. In contrast, our model must provide probabilities for a full set of possible interpretations of the user's strokes.

3 Network Structure

Our Bayes net model is built around hierarchical descriptions of shapes in a domain, described in a language called LADDER [Hammond and Davis, 2003]. The basic unit in this

language is a *shape*, which we use to mean a pattern recognizable in a given domain. Shapes may be *compound*, i.e., composed of *subshapes* fit together according to *constraints*. These subshapes also may be compound, but all shapes must be non-recursive. A shape that cannot be decomposed into subshapes (e.g., a line), is called a *primitive shape*. Primitive shapes may have named *subcomponents* that can be used when describing other shapes, e.g., the endpoints of a line, "p1" and "p2", used in Figure 1.

We refer to each shape description as a *template* with one *slot* for each subpart. A *shape hypothesis* is a template with an associated mapping between slots and strokes, generated during the recognition process. Similarly, a *constraint hypothesis* is a proposed constraint on one or more of the user's strokes. A *partial hypothesis* is a hypothesis in which one or more slots are not bound to strokes.

To introduce our Bayes net model, we begin by considering how to model a current source hypothesis (CS_1) for the strokes in Figure 1. A current source is a compound shape, so CS_1 involves two lower-level shape hypotheses— E_1 , an ellipse hypothesis for stroke s_1 ; and A_1 , an arrow hypothesis involving strokes s_2 , s_3 and s_4 —and one constraint hypothesis— C_1 , indicating that an ellipse fit for stroke s_1 contains strokes s_2 , s_3 , and s_4 . A_1 is also compound and is further broken down into three line hypotheses (L_1 , L_2 and L_3) and six constraint hypotheses ($C_2, \dots, C_7$) according to the description of an arrow. Thus, determining the strength of hypothesis CS_1 can be transformed into the problem of determining the strength of a number of lower-level shape and constraint hypotheses.

The Bayes net used to evaluate CS_1 is shown in Figure 2. There is one node in the network for each hypothesis; each node represents a boolean random variable that is true if the corresponding hypothesis is correct. The probability of each hypothesis is influenced both through its children by stroke data and through its parents by the context in which it appears, allowing the system to handle noise in the drawing.

The nodes labeled $O_1, \dots, O_{11}$ represent measurements of the stroke data that correspond to the constraint or shape to which they are linked. The variables corresponding to these nodes have positive real numbered values that we discretize in our implementation¹. For example, the variable O_2 is a measurement of the squared error between the stroke s_1 and the best fit ellipse to that stroke. Its raw value (later discretized) ranges from 0 to the maximum possible error between any stroke and an ellipse fit to that stroke. The boxes labeled $s_1, \dots, s_4$ are not part of the Bayes net but serve to indicate the stroke or strokes from which each measurement, O_i , is taken (e.g., O_2 is measured from s_1). $P(CS_1 = t|ev)$ (or simply $P(CS_1|ev)$)², where ev is the evidence observed from the user's strokes, represents the probability that the hypothesis CS_1 is correct.

The direction of the links may seem counterintuitive, but there are two important reasons why the links are directed from higher-level shapes to lower-level shapes instead of the opposite direction. First, whether a higher-level hypothesis is true directly influences whether a lower-level hypothesis

¹The BN software we used did not support continuous variables.

²Throughout this paper, t means true, and f means false.

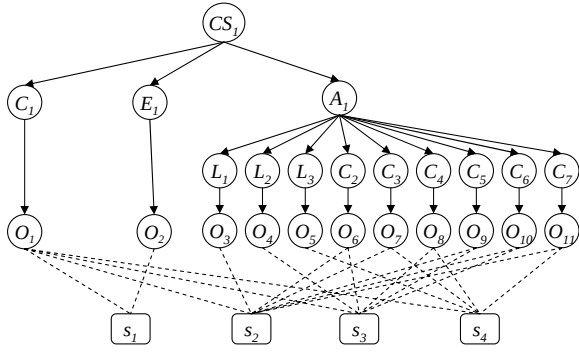


Figure 2: A Bayes net to verify a single current source hypothesis. Labels come from Figure 1.

esis is true. For example, if the arrow hypothesis A_1 is true, then it is extremely likely that all three line hypotheses, L_1, L_2, L_3 , are also true. Second, this representation allows us to model lower-level hypothesis as conditionally independent given their parents, which reduces the complexity of the data needed to construct the network.

Each shape description constrains its subshapes only relative to one another. For example, an arrow may be made from *any* three lines that satisfy the necessary constraints. Based on this observation, our representation models a symbol's subshapes separately from the constraints between them. For example, node L_2 represents the hypothesis that stroke s_3 is a line. Its value will be true if the user intended for s_3 to be a line, *any* line regardless of its position, size or orientation. C_4 separately represents the hypothesis that the line fit to s_3 and the line fit to s_4 are the same length.

The conditional independence between shapes and constraints might seem a bit strange at first. For example, whether or not two lines are the same length seems to depend on the fact that they are lines. However, observation nodes for constraints are calculated in such a way that their value is not dependent on the true interpretation for a stroke. For example, when calculating whether or not two lines are the same length, we first fit lines to the strokes (regardless of whether or not they actually look like lines), then measure their length. How well these lines fit the original strokes is not considered in this calculation.

The fact that there is no edge between the constraint nodes and the shapes they constrain has an important implication for using this model to perform recognition: There is no guarantee in this Bayes net that the constraints will be measured between the correct subshapes because the model allows subshapes and constraints to be detected independently. For example, we want C_4 in Figure 2 to indicate that L_2 and L_3 (the two lines in the head of an arrow) are the same length, not simply that *any* two lines are the same length. To satisfy this requirement, the system must ensure that O_8 is measured from the same strokes that O_4 and O_5 were measured from. We use a separate mechanism to ensure that only legal bindings are created between strokes and observation nodes.

The way we model shape and constraint information has two important advantages for recognition. First, this Bayes

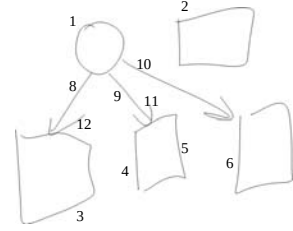


Figure 3: A partial sketch of a family tree. Quadrilaterals (Q) represent males (M); ellipses (E) represent females (F), and arrows (A) indicate a parent-child relationship.

net model can be applied to recognize a shape in any size, position and orientation. CS_1 represents the hypothesis that $s_1, \dots, s_4$ form a current source symbol, but the exact position, orientation and size of that symbol is determined directly from the stroke data³.

Second, the system can model competing higher-level interpretations for lower-level shapes. For example, the system may consider a stroke to be a line that is in turn part of either an arrow or a quadrilateral. Because the line hypothesis does not include any higher-level shape-specific constraint information, both an arrow hypothesis node and a quadrilateral hypothesis node can point to this same line hypothesis node. These two hypotheses then become alternate, competing explanations for the line hypothesis. We further discuss how hypotheses are combined below.

4 Recognizing a Complete Sketch

To recognize a complete sketch, we create a Bayes net similar to the one above for each shape. We call each of these Bayes nets a *shape fragment* because they can be combined to create a complete Bayes net for evaluating the whole sketch.

Given a set of hypotheses for the user's strokes (hypothesis generation is described in [Alvarado and Davis, 2004] and [Alvarado, 2004]), the system instantiates the corresponding shape fragments and links them together to form a complete Bayes net, called the *interpretation network*. To illustrate this process, consider a piece of a network generated in response to Strokes 6 and 7 in the example of Figure 3. Figure 4 shows the part of the Bayes net representing the hypotheses that the system generated for these strokes. Low level processing recognizes Strokes 6 and 7 as L-shaped polylines and breaks each into two individual lines ($L_1 \dots L_4$) that meet.

4.1 Linking Shape Fragments

When Bayes net fragments are linked during recognition, each node H_n may have several parents, $S_1 \dots S_m$, where each parent represents a possible higher-level interpretation for H_n . We use a noisy-OR function to combine the influences of all the parents of H_n to produce the complete conditional probability table (CPT) for $P(H_n | S_1, \dots, S_m)$. The noisy-OR function models the assumption that each parent can independently cause the child to be observed. For example, a single

³Orientation-dependent symbols (e.g., a downward-facing arrow) can be recognized by including orientation-dependent constraints in their descriptions (e.g., vertical, below).

stroke might be part of a quadrilateral or an arrow, but both interpretations would favor that interpretation of the stroke as a line. We set $P(H_n = f | S_j = t) = 0$ for all parents S_j in which H_n is a required subshape or constraint, and we set $P(H_n = f | S_k = t) = 0.5$ for all parents S_k in which H_n is an optional subshape or constraint. A consequence of these values is that $S_j = t \Rightarrow P(H_n | S_1, \dots, S_m) = 1$ for any S_j in which H_n is required, which is exactly what we intended.

We experimented with a noisy-XOR construct, implemented using a “gate node” similar to that described in [Boutilier *et al.*, 1996], but found that noisy-OR semantics were simpler and in fact produced better results. In effect, a noisy-OR node in a Bayes net with low prior probabilities behaves as a non-aggressive XOR. The fact that one parent is true does not actively prohibit the other parents from being true, but it causes their probabilities to tend back to their prior values because of the “explaining away” phenomenon.

4.2 Signal Level Noise

The bottom layer of the network deals with signal level noise by modeling the differences between the user’s intentions and the strokes that she draws. For example, even if the user intends to draw L_1 , her stroke likely will not match L_1 exactly, so the model must account for this variation. Consider $P(O_1 | L_1 = t)$ (recall that O_1 is a discretized continuous valued variable). If the user always drew perfect lines, this distribution would be 1 when $O_1 = 0$ (i.e., the error is 0), and 0 otherwise. However, most people do not draw perfect lines (due to inaccurate pen and muscle movements), and this distribution allows for this error. It should be high when O_1 is close to zero, and fall off as O_1 gets larger. The wider the distribution, the more error the system will tolerate, but the less information a perfect line will provide.

The other distribution needed is $P(O_1 | L_1 = f)$ which is the probability distribution over line error given that the user did not intend to draw an line. This distribution should be close to uniform, with a dip around 0, indicating that if the user specifically does not intend to draw an line, she might draw any other shape, but probably won’t draw anything that resembles an line. Details about how we determined the conditional probability distributions between primitive shapes and constraints and their corresponding observation nodes are given elsewhere [Alvarado, 2004].

4.3 Missing Strokes

A_1 is a partial hypothesis—it represents the hypothesis that L_1 and L_2 (from Stroke 6) are part of an arrow whose other line has not yet been drawn. Line nodes representing lines that have not been drawn (e.g., L_5) are not linked to observation nodes because there is no stroke from which to measure these observations. We refer to these nodes (and their corresponding hypotheses) as *virtual*.

The fact that partial hypotheses have probabilities allows the system to assess the likelihood of incomplete interpretations based on the evidence it has seen so far. In fact, even virtual nodes have probabilities, corresponding to the probability that the user (eventually) intends to draw these shapes but either has not yet drawn this part of the diagram or the

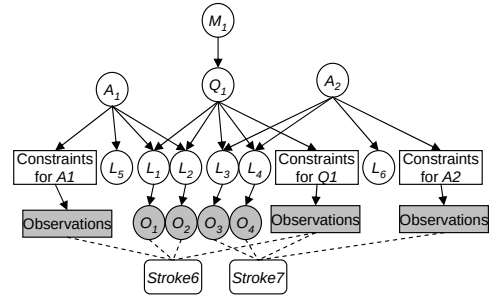


Figure 4: A portion of the interpretation network generated while recognizing the sketch in Figure 3.

correct low-level hypotheses have not yet been proposed (because of low-level recognition errors). A partial hypothesis with a high probability cues the system to examine the sketch for possible missed low-level interpretations during the hypothesis generation step.

5 Implementation and Bayesian Inference

Our system updates the structure of the Bayes net in response to each stroke the user draws, using an off-the-shelf, open source Bayes net package for Java called BNJ [bnj, 2004].

Generating and modifying the BNJ networks can be time consuming due to the exponential size of the CPTs between the nodes. We use two techniques to improve performance. First, networks are generated only when the system needs to evaluate the likelihood of a hypothesis. This on-demand construction is more efficient than continuously updating the network, because batch construction of the CPTs is often more efficient than incremental construction. Second, the system modifies only the portion of the network that has changed between strokes, rather than creating it from scratch every time.

We experimented with several inference methods and found that loopy belief propagation (loopy BP) [Weiss, 1997] was the most successful⁴. On our data, loopy BP almost always converged (messages initialized to 1, run until node values stable within 0.001). We further limited the algorithm’s running time in two ways. First, we terminated the algorithm after 60 seconds if it had not yet converged. Second, we allowed each node to have no more than 8 parents (i.e., only 8 higher-level hypotheses could be considered for a single hypothesis), ensuring a limit on the complexity of the graphs produced. These restrictions had little impact on recognition performance in the family tree domain, but for complex domains such as circuit diagrams, more efficient inference algorithms or graph simplification techniques are needed to improve recognition results.

6 Application and Results

We applied our implemented system to two non-trivial domains—family trees and circuits—and found that it is capable of recognizing sketches in both domains without reprogramming. Qualitative and quantitative evaluation of our

⁴Junction Tree [Jensen *et al.*, 1990] was often too slow and Gibb’s Sampling did not produce meaningful results.

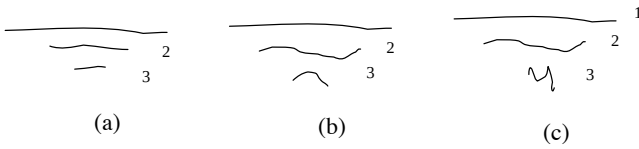


Figure 5: Part of three possible ground symbols.

system illustrates its strengths over previous approaches and suggests some extensions.

Applying our system to a particular domain involves two steps: specifying the structural descriptions for the shapes in the domain, and specifying the prior probabilities for top-level shapes and domain patterns, i.e., those that will not have parents in the Bayes net. For each domain, we wrote a description for each shape and pattern in that domain. We hand-estimated priors for each domain pattern and top level shape based on our intuition about the relative prevalence of each shape. For example, in family-tree diagrams, we estimated that marriages were much more likely than partnerships, and set $P[Mar] = 0.1$ and $P[Part] = 0.001$. These priors represent the probability that a group of strokes chosen at random from the page will represent the given shape and, in most cases, should be relatively low. Although setting priors by hand can be tedious, we found through experimentation that the system’s recognition performance was relatively insensitive to the exact values of these priors. For example, in the circuit diagrams, increasing all the priors by an order of magnitude did not affect recognition performance; what matters instead is the relative values of the prior probabilities.

Our system is capable of recognizing simple sketches nearly perfectly in both the family-tree and circuit domains. We also tested its performance on more complex, real-world data. As there is no standard test corpus for sketch recognition, we collected our own sketches and have made them available online to encourage others to compare their results with those presented here [Oltmans *et al.*, 2004]. In total, we tested our system on 10 family tree sketches and 80 circuit diagram sketches, with between 23 and 110 strokes each.

6.1 Qualitative Results

Qualitative analysis reveals that the Bayes net mechanism successfully aggregates information from stroke data and context, resolves inherent ambiguities in the drawing, and updates its weighting of the various existing hypotheses as new strokes are drawn. We show through an example that the Bayes net scores hypotheses correctly in several respects: it prefers to group subshapes into the fewest number of interpretations, it allows competing interpretations to influence one another, it updates interpretation strengths in response to new stroke data, and it allows both stroke data and context to influence interpretation strength.

To illustrate the points above, we consider in detail how the system responds as the user draws the three sets of strokes in Figure 5 (a ground symbol). To simplify the example, we consider a reduced circuit domain in which users draw

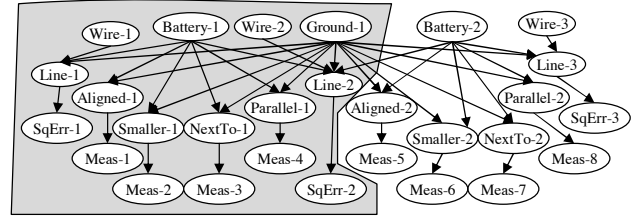


Figure 6: The Bayes net produced in response to the strokes in each ground symbol in Figure 5. The shaded area shows the network produced in response to the first two strokes.

Name	$P[Shape stroke\ data]$					
	after 2 strokes			after 3 strokes		
	(a)	(b)	(c)	(a)	(b)	(c)
Wire-1	0.4	0.41	0.41	0.4	0.4	0.42
Wire-2	0.4	0.38	0.38	0.4	0.4	0.38
Wire-3	N/A	N/A	N/A	0.4	0.4	0.1
Battery-1	0.51	0.51	0.51	0.09	0.1	0.91
Battery-2	N/A	N/A	N/A	0.09	0.1	0.0
Ground-1	0.51	0.51	0.51	0.95	0.94	0.03

Table 1: Posterior probabilities for part of the network in Figure 6 for each sketch in Figure 5.

only wires, resistors, ground symbols and batteries. Figure 6 shows the Bayes nets produced in response to the user’s first two and first three strokes. After the first two strokes, the network contains only the nodes in the shaded area; after three strokes it contains all nodes shown. Continuous variables corresponding to observation nodes were discretized into three values: 0 (low error: stroke data strongly supports the shape or constraint), 1 (medium error: stroke data weakly supports the shape or constraint), and 2 (high error: stroke data does not support the shape or constraint).

Posterior probabilities are given in Table 1. For the relatively clean ground symbol (Figure 5(a)), the stroke data strongly supports all the shapes and constraints, so all shaded nodes in Figure 6 have value 0. After the first two strokes, the battery symbol and the ground symbol have equal weight, which may seem counterintuitive: after two strokes the user has drawn what appears to be a complete battery, but has drawn only a piece of the ground symbol. Recall, however, that the Bayes net models not what has been drawn, but what the user *intends to draw*. After two strokes, it is equally likely that the user is in the process of drawing a ground symbol. After the third stroke, the ground symbol’s probability increases because there is more evidence to support it, while the battery’s probability decreases. The fact that the ground symbol is preferred over the battery illustrates that the system prefers interpretations that result in fewer symbols (Okham’s razor). Interpretations that involve more of the user’s strokes have more support and get a higher score in the Bayes net. The fact that the battery symbol gets weaker as the ground symbol gets stronger results from the “explaining away” behavior in this Bayes net configuration: each of the low-level components (the lines and the constraints) are effectively “explained” by the existence of the ground symbol, so the battery is no longer

needed to explain their presence.

Another useful property of our approach is that a small amount of noise in the data is counteracted by the context provided by higher-level shapes. The slightly noisy second and third strokes in Figure 5(b) cause the values of SqErr-2 and SqErr-3 to be 1 instead of 0 (all other shaded node values remain 0). Despite this noise, after three strokes the posterior probability of the ground symbol interpretation is still high (0.94), because Line-1 and all of the constraints are still strongly supported by the data. In addition, the probabilities for Line-2 and Line-3 are high (both 1.0, not shown in Table 1), indicating that the context provided by the ground symbol provides support for the line interpretations even when the evidence from the data is not as strong.

On the other hand, if the data is too messy, it causes the probability of the higher-level interpretations to decrease. The sketch in Figure 5(c) causes the value of SqErr-2 to be 1, and SqErr-3 to be 2, and results in a low posterior probability for Ground-1 (0.03) and a high posterior for Battery-1 (0.91), because the hypothesis Line-3 is contradicted by the user's third stroke. For now, the third stroke remains uninterpreted.

6.2 Quantitative Results

We ran our system on all of the sketches we collected and compared its performance to a baseline constraint-based recognition system that used a fixed threshold for detecting shapes and constraints and did not reinterpret low-level shapes. We measured recognition performance by determining the number of objects identified correctly in each sketch. Our system significantly outperformed the baseline system ($p \ll 0.001$), correctly recognizing 77% ($F=0.83$) of the shapes in the family tree diagrams and 62% ($F=0.65$) of the shapes in the circuit diagrams, while the baseline system correctly recognized 50% ($F=0.63$) and 54% ($F=0.57$).

While not yet real time, in general our system's processing time scaled well as the number of strokes increased. However, it occasionally ran for a long period. The system had particular trouble with areas of the sketch that involved many strokes drawn close together in time and space and with domains that involve more complicated or overlapping symbols. This increase in processing time was due almost entirely to increase in Bayes net complexity.

We believe we can speed up loopy BP based on the observation that it repeatedly sends messages between the nodes until each node has reached a stable value. When a stroke is added, our system resets all messages to 1, essentially erasing the work done the last time inference was performed, even though most of the graph is unchanged. The algorithm should instead begin with the messages that remain at the end of the previous inference step.

7 Conclusion

We have described a model for dynamically constructing Bayes nets to represent varying hypotheses for the user's strokes. Our model, specifically developed for the task of recognition, allows both stroke data and contextual data to influence the probability of an interpretation for the user's strokes. Using noisy-OR, multiple potential higher-level

interpretations mutually influence each other's probabilities within the Bayes net. The net result is a sketch recognition approach that brings us a significant step closer to sketching as a natural and powerful interface.

References

- [Alvarado and Davis, 2004] C. Alvarado and R. Davis. SketchREAD: A multi-domain sketch recognition engine. In *Proc. of UIST-04*, 2004.
- [Alvarado, 2004] C. Alvarado. *Multi-Domain Sketch Understanding*. PhD thesis, MIT, 2004.
- [bnj, 2004] Bayesian network tools in java (bnj), 2004. <http://bnj.sourceforge.net>.
- [Boutilier *et al.*, 1996] C. Boutilier, N. Friedman, M. Goldszmidt, and D. Koller. Context-specific independence in Bayesian networks. In *Proc. of UAI '96*, 1996.
- [Futrelle and Nikolakis, 1995] R. P. Futrelle and N. Nikolakis. Efficient analysis of complex diagrams using constraint-based parsing. In *ICDAR-95*, pages 782–790, Montreal, Canada, 1995.
- [Getoor *et al.*, 1999] L. Getoor, N. Friedman, D. Koller, and A. Pfeffer. Learning probabilistic relational models. In *Proc. of IJCAI '99*, pages 1300–1309, 1999.
- [Glessner and Koller, 1995] S. Glessner and D. Koller. Constructing flexible dynamic belief networks from first-order probabilistic knowledge bases. In *Synthetic and Quantitative Approaches to Reasoning and Uncertainty*, pages 217–226, 1995.
- [Haddawy, 1994] P. Haddawy. Generating bayesian networks from probability logic knowledge bases. In *Proc. of UAI '94*, 1994.
- [Hammond and Davis, 2003] T. Hammond and R. Davis. LADDER: A language to describe drawing, display, and editing in sketch recognition. *Proc. of IJCAI '03*, 2003.
- [Hammond and Davis, 2004] T. Hammond and R. Davis. Automatically transforming symbolic shape descriptions for use in sketch recognition. *Proc. of AAAI '04*, 2004.
- [Jensen *et al.*, 1990] F. V. Jensen, S. L. Lauritzen, and K. G. Olesen. Bayesian updating in causal probabilistic networks by local computations. *Computational Statistics Quarterly*, 4, 1990.
- [Kara and Stahovich, 2004] L. B. Kara and T. F. Stahovich. Hierarchical parsing and recognition of hand-sketched diagrams. In *Proc. of UIST '04*, 2004.
- [Koller and Pfeffer, 1997] D. Koller and A. Pfeffer. Object-oriented bayesian networks. In *Proc. of UAI '97*, 1997.
- [Laskey and Mahoney, 1997] K. B. Laskey and S. M. Mahoney. Network fragments: Representing knowledge for constructing probabilistic models. In *Proc. of UAI '97*, 1997.
- [Oltmans *et al.*, 2004] M. Oltmans, C. Alvarado, and R. Davis. Etcha sketches: Lessons learned from collecting sketch data. In *Making Pen-Based Interaction Intelligent and Natural*. AAAI Fall Symposium, 2004.
- [Pfeffer *et al.*, 1999] A. Pfeffer, D. Koller, B. Milch, and K. Takusagawa. SPOOK: A system for probabilistic object-oriented knowledge representation. In *Proc. of UAI '99*, 1999.
- [Weiss, 1997] Y. Weiss. Belief propagation and revision in networks with loops. Technical report, MIT, November 1997.
- [Wu *et al.*, 1999] L. Wu, S. L. Oviatt, and P. R. Cohen. Multimodal integration—a statistical view. *IEEE Transaction on Multimedia*, 1(4):334–341, December 1999.

You Are Wrong!—Automatic Detection of Interaction Errors from Brain Waves

Pierre W. Ferrez, José del R. Millán

IDIAP Research Institute

Rue du Simplon 4, 1920 Martigny, Switzerland

{pierre.ferrez, jose.millan}@idiap.ch

Abstract

Brain-computer interfaces, as any other interaction modality based on physiological signals and body channels (e.g., muscular activity, speech and gestures), are prone to errors in the recognition of subject's intent. In this paper we exploit a unique feature of the “brain channel”, namely that it carries information about cognitive states that are crucial for a purposeful interaction. One of these states is the awareness of erroneous responses. Different physiological studies have shown the presence of error-related potentials (ErrP) in the EEG recorded right after people get aware they have made an error. However, for human-computer interaction, the central question is whether ErrP are also elicited when the error is made by the interface during the recognition of the subject's intent and no longer by errors of the subject himself. In this paper we report experimental results with three volunteer subjects during a simple human-robot interaction (i.e., bringing the robot to either the left or right side of a room) that seem to reveal a new kind of ErrP, which is satisfactorily recognized in single trials. These recognition rates significantly improve the performance of the brain interface.

1 Introduction

A brain-computer interface (BCI) is a system that translates the intention of a subject, as represented by his brain waves (typically, electroencephalogram signals), into a control signal without using activity of any muscles or peripheral nerves [Millán, 2002; Wolpaw *et al.*, 2002]. Electroencephalogram (EEG) is recorded non-invasively by means of electrodes placed on the scalp. Then some features are extracted from the EEG and sent to a classifier, whose response is translated into some action thus making possible mental control of different devices. For instance, recent studies have shown that after a few days of training, subjects are able to control a miniature robot in an indoor environment with several rooms, corridors and doorways only using the signals derived from an EEG-based BCI [Millán *et al.*, 2003]. This same system could be used to drive a wheelchair so that a

subject suffering from severe motor disabilities, but with intact cognitive capabilities, could greatly gain in autonomy. Other applications, such a virtual keyboard, could more generally provide an alternative way of communication with the outside world [Birbaumer *et al.*, 1999; Wolpaw *et al.*, 2003; Millán *et al.*, 2004].

Nevertheless, BCIs—as any other interaction modality based on physiological signals and body channels (e.g., muscular activity, speech and gestures)—are prone to errors in the recognition of subject's intent, and those errors can be frequent. Indeed, even well-trained subjects rarely reach 100% of success. A possible way to reduce errors consists in a verification procedure whereby each output consists of two opposite trials, and success is required on both to validate the outcome [Wolpaw *et al.*, 1998]. Even if this method greatly reduces the errors, it requires much more mental effort from the subject and reduces the communication rate.

In contrast to other interaction modalities, a unique feature of the “brain channel” is that it conveys both information from which we can derive mental control commands to operate a brain-actuated device as well as information about cognitive states that are crucial for a purposeful interaction, all this on the millisecond range. One of these states is the awareness of erroneous responses, which a number of groups have recently started to explore as a way to improve the performance of BCIs [Schalk *et al.*, 2000; Blankertz *et al.*, 2003; Parra *et al.*, 2003]. Since the late 1980s, different physiological studies have shown the presence of error-related potentials (ErrP) in the EEG recorded right after people get aware they have made an error [Carter *et al.*, 1998; Falkenstein *et al.*, 2000; Holroyd and Coles, 2002]. Nevertheless, most of these studies show the presence of ErrP in typical choice reaction tasks [Carter *et al.*, 1998; Falkenstein *et al.*, 2000; Blankertz *et al.*, 2003; Parra *et al.*, 2003]. In this kind of tasks, the subject is asked to respond as quickly as possible to a stimulus and ErrP (sometimes referred to as “response ErrP”) arise following errors due to the subject's incorrect motor action (e.g., the subject pressed a key with the left hand when he should have responded with the right hand). More recently, other studies have also shown the presence of ErrP in typical reinforcement learning tasks where the subject is asked to make a choice and ErrP (sometimes referred to as “feedback ErrP”) arise following the presentation of a stimulus that indicate incorrect performance [Holroyd and Coles, 2002]. In

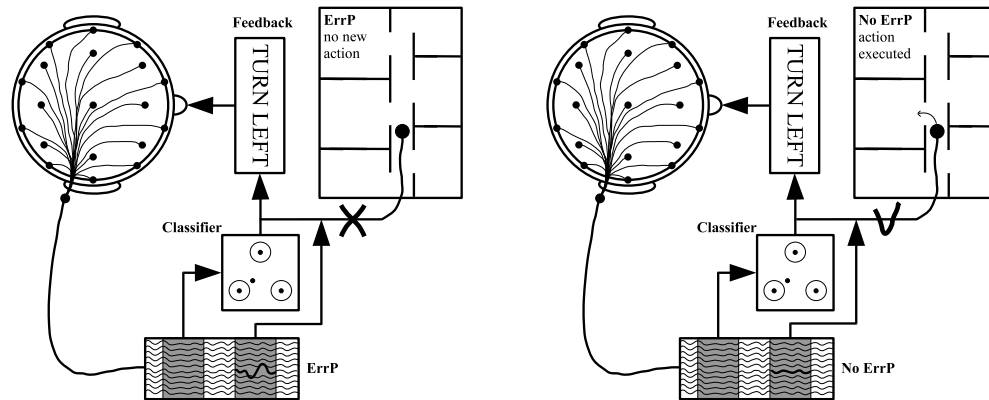


Figure 1: Exploiting error-related potentials (ErrP) in a brain-controlled mobile robot. The subject receives visual feedback indicating the output of the classifier before the actual execution of the associated command (e.g., “TURN LEFT”). If the feedback generates an ErrP (left), this command is simply ignored and the robot will stay executing the previous command. Otherwise, the command is sent to the robot (right).

both cases, response ErrP and feedback ErrP, it has been observed a negative potential in the EEG with a fronto-central scalp distribution (see Figure 3) whenever subjects make errors. This negative potential is most probably generated in a brain area called anterior cingulate cortex, which is crucial for regulating emotional responses.

An important aspect of the two described ErrP is that they always follow an error made by the subject himself. First the subject make a selection, and then ErrP arise either simply after the occurrence of an error (choice reaction task) or after a feedback indicating the error (reinforcement learning task). However, in the context of a BCI, or human-computer interaction in general, the central question is:

“Are ErrP also elicited when the error is made by the interface during the recognition of the subject’s intent?”

In order to consider the full implications of this question, let’s imagine that the subject’s intent is to make a robot reach a target to the left. What would happen if the interface fails to recognize the intended command and the robot starts turning in the wrong direction? Are ErrP still present even though the subject did not make any error but only perceive that the interface is performing wrongly?

The objective of this paper is to investigate how ErrP could be used to improve the performance of a BCI. Thus, we will first explore whether or not ErrP also follow a feedback indicating incorrect responses of the interface and no longer errors of the subject himself. If ErrP are also elicited in this case, then we could integrate them in a BCI in the following way as shown in Figure 1: after translating the subject’s intention into a control command, the BCI provides a feedback of that command, which will be actually executed only if no ErrP follows the feedback. This should greatly increase the reliability of the BCI as we will see later. Of course, this new interaction protocol depends on the ability to detect ErrP no longer in averages of a large number of trials [Schalk *et al.*, 2000], but in each single trial using a short window following the feedback that shows the response of the classifier embedded in the BCI.

In this paper we report experimental results with three volunteer subjects during a simple human-robot interaction (i.e., bringing the robot to either the left or right side of a room)

that seem to reveal a new kind of ErrP, which is satisfactorily recognized in single trials. These recognition rates significantly improve the performance of the interface.

2 Experimental Setup

To test the presence of ErrP after a feedback indicating errors made by the interface in the recognition of the subject’s intent, we have simulated a real interaction with a robot, where the subject wishes to bring the robot to one side of a room (left or right) by delivering repetitive commands until the robot reaches the target. This virtual interaction is implemented by means of two horizontal progress bars made of 10 steps each. One of the bars goes from the center of the screen to the left side (left bar), and the other bar progresses to the right side (right bar). Figure 2 shows the left and right horizontal progress bars used as feedback.

To isolate the issue of the recognition of ErrP out of the more difficult and general problem of a whole BCI where erroneous feedback can be due to non-optimal performance of both the interface (i.e., the classifier embedded into the interface) and the user himself, in the following experiments the subject delivers commands manually and not mentally. That is, he simply presses a left or right key with the left or right hand. In this way, any error feedback is only due to a wrong recognition of the interface of which is the subject’s intention.

Three volunteer healthy subjects participated in these experiments. The subject sends a command after a stimulus delivered by the system (the word “GO” appears in the screen). The system filled the bars with an error rate of 20%; i.e., at each step, there was a 20% probability that the robot made a movement in the wrong direction. Subjects performed 10 series of 5 progress bars, the delay between 2 consecutive steps (2 consecutive “GO” from the system) was of 3-4 seconds (random delay to prevent habituation). Duration of each interaction experiment (i.e., filling a progress bar) was about 40 seconds, with breaks of 5-10 minutes between two series but no break between interaction experiments of the same series.

EEG potentials were acquired with a portable system (Biosemi ActiveTwo) by means of a cap with 32 integrated electrodes covering the whole scalp and located according to



Figure 2: Left and right horizontal progress bars. The goal of an interaction experiment is to fill one of the bars, which simulates a real interaction with a robot that needs to reach one side of a room (left or right). The system fills the bars with an error rate of 20%; i.e., at each step, there was a 20% probability that the robot made a movement in the wrong direction.

the standard 10/20 International system. The sampling rate was 512 Hz and signals were measured at full DC. Raw EEG potentials were first spatially filtered by subtracting from each electrode the average potential (over the 32 channels) at each time step. The aim of this re-referencing procedure is to suppress the average brain activity, which can be seen as underlying background activity, so as to keep the information coming from local sources below each electrodes. Then, we applied a 1-10 Hz bandpass filter as ErrP are known to be a relatively slow cortical potential. Finally, EEG signals were subsampled from 512 Hz to 128 Hz (i.e., we took one point out of 4) before classification, which was entirely based on temporal features. Indeed the actual input vector for the statistical classifier described below is a 1/2-second window starting 150 ms after the feedback and ending 650 ms after the feedback for channels Cz and Fz. The choice of these channels follows the fact that ErrP are characterized by a fronto-central distribution along the midline (see Figure 3). Thus the dimensionality of the input vector is 128; i.e., concatenation of two windows of 64 points (EEG potentials) each.

No artifact rejection algorithm (for removing or filtering out eye or muscular movements) was applied and all trials were kept for analysis. It is worth noting, however, that after a visual a-posteriori check of the trials we found no evidence of muscular artifacts that could have contaminated one condition differently from the other.

3 Statistical Classifier

The different classes are recognized by a Gaussian classifier trained to classify single trials as “correct” or “error”. The output of this statistical classifier is an estimation of the posterior class probability distribution for a single trial; i.e., the probability that a given single trial belongs to class “correct” or class “error”. The present classifier is quite similar to that proposed by Millán *et al.* [2004] in the framework of EEG-based BCIs, the main difference being the update rules.

In this statistical classifier, every Gaussian unit represents a prototype of one of the classes to be recognized. We use several prototypes per mental task. We assume that the class-conditional probability density function of class C_k is a superposition of N_k Gaussians (or prototypes) and that classes have equal prior probabilities. In our case, all the classes have the same number of prototypes N_p . In addition, we assume that all prototypes have an equal weight of $1/N_p$. Then, dropping constant terms, the activity a_k^i of the i^{th} prototype of the class C_k for the input vector, or sample, x derived from a trial as described above is

$$a_k^i(x) = |\Sigma_k|^{-1/2} \exp \left(-1/2 (x - \mu_k^i)^T \Sigma_k^{-1} (x - \mu_k^i) \right) \quad (1)$$

where μ_k^i is the center of the i^{th} prototype of the class C_k , Σ_k is the covariance matrix of the class C_k , and $|\Sigma_k|$ is the determinant of that matrix. Usually, each prototype has its own covariance matrix Σ_k^i . In order to reduce the number of parameters, we restrict our model to a diagonal covariance matrix Σ_k that is common to all the prototypes of the class C_k . Now, the posterior probability y_k of the class C_k is

$$y_k(x) = p(x|C_k) = \frac{a_k(x)}{A(x)} = \frac{\sum_{i=1}^{N_p} a_k^i(x)}{\sum_{k=1}^2 \sum_{i=1}^{N_p} a_k^i(x)} \quad (2)$$

where a_k is the activity of class C_k and A is the total activity of the network. The response of the classifier for the input vector x is simply the class with the highest probability.

To initialize the center of the prototypes, μ_k^i , of the class C_k we run a clustering algorithm—typically, self-organizing maps [Kohonen, 1997]. We then initialize the diagonal covariance matrix Σ_k of the class C_k by setting

$$\Sigma_k = \frac{1}{|S_k|} \sum_{x \in S_k} (x - \mu_k^{i*}) (x - \mu_k^{i*})^T \quad (3)$$

where S_k is the set of the training samples belonging to the class C_k , $|S_k|$ is the cardinality of this set, and i^* is the nearest prototype of this class to the sample x .

During learning we improve these initial estimations iteratively by stochastic gradient descent so as to minimize the mean square error $E = 1/2 \sum_k (y_k - t_k)^2$, where t_k is the k^{th} component of the target vector in the form 1-of-c; e.g., the target vector for class “error” is coded as (0, 1). Taking the gradient of the error function yields

$$\Delta \mu_k^i(x) = \alpha \frac{\partial E}{\partial \mu_k^i}(x) = \alpha \frac{a_k^i(x)}{A(x)} \frac{(x - \mu_k^i)}{\Sigma_k} e_k(x) \quad (4)$$

and

$$\Delta \Sigma_k^i(x) = \beta \frac{\partial E}{\partial \Sigma_k^i}(x) = \beta \frac{a_k^i(x)}{A(x)} \frac{(x - \mu_k^i)^2}{(\Sigma_k^i)^3} e_k(x) \quad (5)$$

where α and β are the learning rates and $e_k(x) = (t_k(x) - y_k(x)) - \sum_j y_j(x) (t_j(x) - y_j(x))$. After updating μ_k^i and Σ_k^i for each training sample, the covariance matrices of all the prototypes of the same class are averaged to obtain the common class covariance matrix Σ_k . This simple operation leads to better performance than if separate covariance matrices are kept for each individual prototype. It is also worth

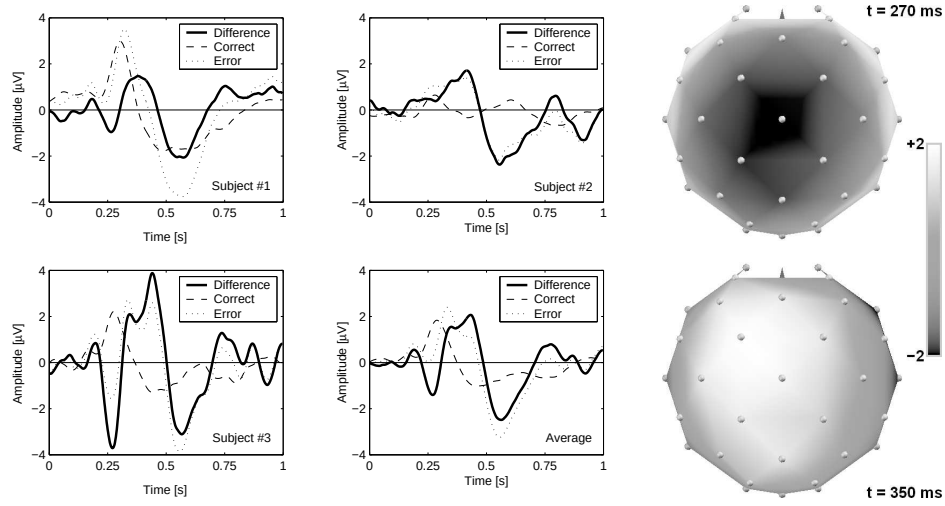


Figure 3: *Left.* Average EEG for error, correct and difference error-minus-correct at channel Cz for the three subjects plus the grand average of them. Feedback is delivered at time 0 seconds. The negative (Ne) and positive (Pe) peaks show up about 270 ms and between 350 and 450 ms after the feedback, respectively. *Right.* Scalp potentials topographies, for the grand average EEG of the three subjects, at the occurrence of the Ne and the Pe. Small filled circles indicate positions of the electrodes (frontal on top), Cz being in the middle of the scalp.

noting that, due to the small number of training samples for the “error” class, it is sometimes preferable to have a single covariance matrix common to all classes, which is obtained by averaging all the individual Σ_k .

4 Experimental Results

With this protocol, it is first necessary to investigate whether or not ErrP are present not more in reaction to an error made by the subject himself, but in reaction to an erroneous response made the interface as indicated by the feedback visualizing the recognized subject’s intention. Figure 3 shows the different curves (error, correct and difference error-minus-correct) for channel Cz for the three subjects plus the grand average of the three subjects. A first sharp negative peak (Ne) can be clearly seen 270 ms after the feedback (except for subject 2). A later positive peak (Pe) appears between 350 and 450 ms after the feedback. Finally an additional negative peak occurs ~ 550 ms after the feedback. Figure 3 also shows the scalp potentials topographies, for the grand average EEG of the three subjects, at the occurrence of the maximum of the Ne and the Pe: a first central negativity appears after 270 ms, followed by a fronto-central positivity at 350 ms.

These experiments seem to reveal a new kind of error-related potentials that, for convenience, we call “interaction ErrP”. The general shape of this ErrP is quite similar to the shape of the response ErrP in a choice reaction task, whereas the timing is similar to the feedback ErrP of reinforcement learning tasks¹. As in the case of response ErrP, the interaction ErrP exhibits a first sharp negative peak followed by a broader positive peak. This is quite different from the shape of feedback ErrP that are only characterized by a small negative deflection. On the other hand, the time course of the interaction ErrP bears some similarities to that of the feedback ErrP: in both cases the first distinctive feature (negative

peak and negative deflection, respectively) appears ~ 250 ms after feedback. The time course of response ErrP is definitely different as the peaks show up much faster.

4.1 Single Trial Classification

To explore the feasibility of detecting single-trial erroneous responses, we have done a 10-fold cross-validation study where the testing set consists of one of the recorded sessions. In this way, testing is always done on a different recording session to those used for training the model. This yields a better estimation of the actual generalization capabilities because brain activity naturally changes over time, as Figure 4 illustrates. Top panels show the average EEG of two consecutive series (5 interaction experiments each) at channel Cz for subject 3, while the bottom panels show samples of single trials of those series. Their variability is rather evident.

Table 1: Percentages of correctly recognized error trials and correct trials for the 3 subjects and the average of them.

Subjects	Error [%]	Correct [%]
#1	87.3 $\pm$ 11.3	82.8 $\pm$ 7.2
#2	74.4 $\pm$ 12.4	75.3 $\pm$ 10.0
#3	78.1 $\pm$ 14.8	89.2 $\pm$ 4.9
Average	79.9 $\pm$ 6.6	82.4 $\pm$ 7.0

Table 1 reports the recognition rates (mean and standard deviations) for the three subjects plus the average of them. The different hyper-parameters—i.e., the learning rates of the centers and diagonal covariance matrices, number of prototypes, and common/single covariance matrices for each class—were chosen by model selection in the training sets. Regarding the learning rates, usual values were 10^{-4} to 10^{-6} for the centers and 10^{-6} to 10^{-8} for the variances, while the usual number of prototypes was rather small (from 2 to 4). These results show that single-trial recognition of erroneous responses is almost 80% on average, while the recognition rate of correct responses is slightly better (82.4%). Quite im-

¹Due to space limitations, experiments showing response and feedback ErrP cannot be reported.

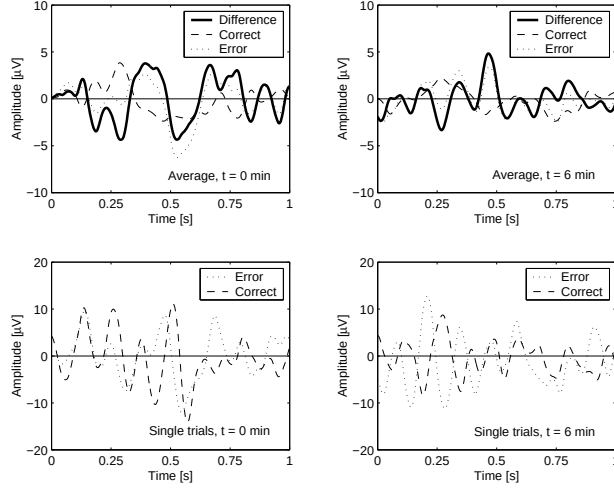


Figure 4: *Top*. Average EEG (error, correct and difference error-minus-correct) of two consecutive experimental series at channel Cz for subject 3. *Bottom*. Samples of single trials (error and correct) of those two series.

portantly, even for the subject with the worse detection rates, they are around 75%. But the key point for the exploitation of the automatic recognition of interaction errors is that they translate into an actual improvement of the performance of the BCI, which we can measure in terms of the bit rate.

4.2 Bit Rate Improvement

A traditional measure of the performance of a system is the bit rate, the amount of information communicated per unit time. The bit rate is usually expressed in bits per trial (bits per selection). If a single trial has N_c possible outcomes, if the probability p that this outcome is correct (accuracy of the BCI), and if finally each of the other outcomes has the same probability of selection (i.e., $(1-p)/(N_c-1)$), then the information transfer rate in bits per trial (BpT) is

$$BpT = \log_2(N_c) + p \log_2(p) + (1-p) \log_2\left(\frac{1-p}{N_c-1}\right) \quad (6)$$

Let's consider now how the performance of the BCI changes after introducing ErrP and that the system detects a proportion e of erroneous trials and a proportion c of correct trials. In the general case, after detecting an erroneous trial the outcome of the interface is simply stopped and not sent to the brain-actuated device. The new accuracy p' of the BCI becomes $p' = p \cdot c / p_t$, where $p_t = p \cdot c + (1-p) \cdot (1-e)$. Now the new information transfer rate in bits per trial is

$$BpT = p_t \left(\log_2(N_c) + p' \log_2(p') + (1-p') \log_2\left(\frac{1-p'}{N_c-1}\right) \right) \quad (7)$$

In the case of a 2-class BCI ($N_c = 2$), after detecting an erroneous trial, it could be possible to replace the “wrong” outcome by the opposite one, what yields an accuracy $p'' = p \cdot c + (1-p) \cdot e$. The information transfer rate in this case is calculated by replacing p by p'' in equation 6, because now there is no stopped outcome.

Table 2 reports the performances of a BCI that integrates ErrP for the 3 subjects and the average of them, where we have assumed an accuracy of 80% in the recognition of the

subject's intent. These figures are to be compared to the performance of a standard BCI (i.e., without integrating ErrP). We have also reported the performances in the case of $N_c = 3$, as the mind-controlled robot described by Millán *et al.* [2003]. In the case of standard 2-class and 3-class BCI, their performances are 0.28 and 0.66 bits per trial, respectively. Results indicate that there is a significant improvement in performance in the case of stopping outcomes, which is above 70% on average and higher than 90% for one of the subjects. Surprisingly, replacing leads to smaller improvements and, in the case of subject 2, even to a significant degradation.

Table 2: Performances of the BCI integrating ErrP for the 3 subjects and the average of them.

Subjects	$N_c = 3$		$N_c = 2$		$N_c = 2$	
	BpT	Gain	BpT	Gain	BpT	Gain
#1	0.91	37%	0.53	91%	0.36	29%
#2	0.73	10%	0.40	42%	0.19	-32%
#3	0.92	38%	0.52	86%	0.44	59%
Average	0.85	28%	0.48	72%	0.32	14%

5 Discussion

In this paper we have reported first results on the detection of the neural correlate of error awareness for improving the performance and reliability of brain-computer interfaces. In particular, we have found what seems to be a new kind of error-related potential elicited in reaction to an erroneous recognition of the subject's intention. More importantly, we have shown the feasibility of detecting single-trial erroneous responses of the interface that leads to significant improvements of the information transfer rate of a BCI.

Given the promising results obtained in a simulated human-robot interaction, we are currently working in the actual integration of ErrP detection into our BCI system. In parallel, we are exploring how to increase the recognition rate of single-trial erroneous and correct responses. A basic issue here is to find what kind of feedback elicits the

strongest “interaction ErrP”. The feedback can be of very different nature—visual, auditory, somatosensory, or even a mix of these different types. More importantly, we will need to focus on alternative methods to exploit at best the current “interaction ErrP”. In this respect, Grave *et al.* [2004] have recently developed a technique that estimates the so-called local field potentials (i.e., the synchronous activity of a small neuronal population) in the whole human brain from scalp EEG. Furthermore, recent results show significant improvements in the classification of bimanual motor tasks using estimated local field potentials (LFP) with respect to scalp EEG [Grave *et al.*, 2005]. Consequently, we plan to utilize the ELECTRA method to best discriminate erroneous and correct responses of the interface. As a matter of fact, a key issue for the success in the above-mentioned study was the selection of those relevant voxels inside the brain whose estimated LFP were more discriminant. It turns out that the sources of the ErrP seem to be very well localized into the anterior cingulate cortex and thus we may well expect a significant improvement in recognition rates by focusing on the LFP estimated in this specific brain area.

More generally, the work described in this paper suggests that it could be possible to recognize in real time high-level cognitive and emotional states from EEG (as opposed, and in addition, to motor commands) such as fatigue, stress, frustration, confusion, or attention that are crucial for an effective and purposeful interaction. Indeed, the rapid recognition of these states will lead to truly adaptive interfaces that customize dynamically in response to changes of the cognitive and emotional/affective states of the user.

Acknowledgments

This work is supported by the Swiss National Science Foundation NCCR “IM2” and by the European IST Programme FET Project FP6-003758. This paper only reflects the authors’ views and funding agencies are not liable for any use that may be made of the information contained herein.

References

- [Birbaumer *et al.*, 1999] N. Birbaumer, N. Ghanayim, T. Hinterberger, I. Iversen, B. Kotchoubey, A. Kubler, J. Perelmouter, E. Taub, and H. Flor. A spelling device for the paralysed. *Nature*, 398:297–298, 1999.
- [Blankertz *et al.*, 2003] B. Blankertz, G. Dornhege, C. Schäfer, R. Krepi, J. Kohlmorgen, K.-R. Müller, V. Kunzmann, F. Losch, and G. Curio. Boosting bit rates and error detection for the classification of fast-paced motor commands based on single-trial EEG analysis. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 11:127–131, 2003.
- [Carter *et al.*, 1998] C.S. Carter, T.S. Braver, D.M. Barch, M.M. Botvinick, D. Noll, and J.D. Cohen. Anterior cingulate cortex, error detection, and the online monitoring of performance. *Science*, 280:747–749, 1998.
- [Falkenstein *et al.*, 2000] M. Falkenstein, J. Hoormann, S. Christ, and J. Hohnsbein. ERP components on reaction errors and their functional significance: A tutorial. *Biological Psychology*, 51:87–107, 2000.
- [Grave *et al.*, 2004] R. Grave de Peralta, M.M. Murray, C.M. Michel, R. Martuzzi, and S.L. Gonzalez Andino. Electrical neuroimaging based on biophysical constraints. *NeuroImage*, 21:527–539, 2004.
- [Grave *et al.*, 2005] R. Grave de Peralta, S.L. Gonzalez Andino, L. Perez, P.W. Ferrez, and J. del R. Millán. Non-invasive estimation of local field potentials for neuroprosthesis control. *Cognitive Processing*, 6, 2005.
- [Holroyd and Coles, 2002] C.B. Holroyd and M.G.H. Coles. The neural basis of human error processing: Reinforcement learning, dopamine, and the error-related negativity. *Psychological Review*, 109:679–709, 2002.
- [Kohonen, 1997] T. Kohonen. *Self-Organizing Maps*, 2nd ed. Springer-Verlag, Berlin, 1997.
- [Millán *et al.*, 2003] J. del R. Millán, F. Renkens, J. Mouriño, and W. Gerstner. Non-invasive brain-actuated control of a mobile robot. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence*, pages 1121–1126, Acapulco, Mexico, 2003. Morgan Kaufmann.
- [Millán *et al.*, 2004] J. del R. Millán, F. Renkens, J. Mouriño, and W. Gerstner. Brain-actuated interaction. *Artificial Intelligence*, 159:241–259, 2004.
- [Millán, 2002] J. del R. Millán. Brain-computer interfaces. In M.A. Arbib, editor, *Handbook of Brain Theory and Neural Networks*, pages 178–181. The MIT Press, Cambridge, Massachusetts, 2002.
- [Parra *et al.*, 2003] L.C. Parra, C.D. Spence, A.D. Gerson, and P. Sajda. Response error correction—A demonstration of improved human-machine performance using real-time EEG monitoring. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 11:173–177, 2003.
- [Schalk *et al.*, 2000] G. Schalk, J.R. Wolpaw, D.J. McFarland, and G. Pfurtscheller. EEG-based communication: Presence of an error potential. *Clinical Neurophysiology*, 111:2138–2144, 2000.
- [Wolpaw *et al.*, 1998] J.R. Wolpaw, H. Ramoser, D.J. McFarland, and G. Pfurtscheller. EEG-based communication: Improved accuracy by response verification. *IEEE Transactions on Rehabilitation Engineering*, 6:326–333, 1998.
- [Wolpaw *et al.*, 2002] J.R. Wolpaw, N. Birbaumer, D.J. McFarland, G. Pfurtscheller, and T.M. Vaughan. Brain-computer interfaces for communication and control. *Clinical Neurophysiology*, 113:767–791, 2002.
- [Wolpaw *et al.*, 2003] J.R. Wolpaw, D.J. McFarland, T.M. Vaughan, and G. Schalk. The Wadsworth Center brain-computer interface (BCI) research and development program. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 11:204–207, 2003.

A Live-User Evaluation of Collaborative Web Search*

Barry Smyth, Evelyn Balfe, Oisín Boydell, Keith Bradley, Peter Briggs, Maurice Coyle, Jill Freyne

Smart Media Institute, Department of Computer Science,
University College Dublin, Ireland
{firstname.surname}@ucd.ie

Abstract

Collaborative Web search exploits repetition and regularity within the query-space of a community of like-minded individuals in order to improve the quality of search results. In short, search results that have been judged to be relevant for past queries are promoted in response to similar queries that occur in the future. In this paper we present the results of a large-scale evaluation of this approach, in a corporate Web search scenario, which shows that significant benefits are available to its users.

1 Introduction

Collection size, document diversity, and limited searcher expertise all combine to make the Web a very challenging information retrieval environment. In 2000 the entire World-Wide Web consisted of just 21 terabytes of information; now it grows by 3 times this figure every single day [Roush, 2004; Lyman and Varian, 2003]. Moreover, the average search query contains only about 2 query terms [Lawrence and Giles, 1998] and the terms used are often poorly chosen [Bollmann-Sdorra and Raghavan, 1993; Furnas *et al.*, 1987]. These problems have led to rapid developments in the term-based matching approaches at the heart of modern search engines. For the most part this has meant looking for new sources of knowledge with which to guide search. For example, Brin & Page [Brin and Page, 1998] and Kleinberg [Kleinberg, 1999] have argued for the need to consider factors such as link-connectivity information, while others have sought to exploit context as a way to disambiguate vague queries (see [Lawrence, 2000]). Still others have begun to consider the structure of the query-space as a new source of search knowledge. For example, [Fitzpatrick and Dent, 1997; Glance, 2001; Raghavan and Sever, 1995; Wen, 2002] have all demonstrated how query logs can be mined to identify useful past queries that may help the current searcher.

In [Freyne *et al.*, 2004; Smyth *et al.*, 2003; In Press], a novel approach to Web search—*collaborative Web search*—was introduced. It combined techniques for exploiting knowledge of the query-space with ideas from social networking

to develop a Web search platform capable of adapting to the needs of (ad-hoc) communities of users. In brief, the queries submitted and the results selected by a community of users are recorded and reused in order to influence the results of future searches for similar queries. Results that have been reliably selected for similar queries in the past are promoted. For example, users of an AI-related Web site might have a tendency to select case-based reasoning results in response to vague queries such as ‘CBR’, while largely ignoring alternatives such as Google’s higher-ranking ‘*Central Bank of Russia*’ or ‘*Comic Book Resources*’ results. In this instance collaborative search will gradually adapt its result-lists to emphasise case-based reasoning results, for searches that originate from such a site, perhaps through a search-box on the site.

While intuitively appealing, the collaborative Web search approach has never been fully evaluated under realistic conditions. Previous evaluations have been limited to the use of artificial users [Freyne *et al.*, 2004] or closed-world search scenarios [Smyth *et al.*, 2003; In Press]. In our work we have implemented the collaborative Web search technique as a robust and scalable meta search engine architecture and the central contribution of this paper is to evaluate its deployment in a realistic, real-world Web search setting involving the employees of a local software company over an extended period of time. While the results indicate that there is indeed a significant benefit accruing from collaborative Web search, they also serve to highlight certain issues, in relation to the manner in which promotions are made, that are likely to lead to critical problems over time. We conclude by discussing how these problems have been overcome in our implementation.

2 Regularity & Repetition in Web Search

Collaborative Web search is motivated by regularity and repetition that is assumed to be inherent in Web search, especially among the searches of communities of like-minded individuals. It proposes to exploit these regularities when responding to new queries by reusing the result selections from similar past queries. But how commonplace is community-based search? And how regular and repetitive is its query-space?

2.1 The Case for Community-Based Web Search

While most searches are conducted through generic search engines, servicing the needs of individuals, many are never-

*The Support of the Enterprise Ireland Informatics Initiative and Science Foundation Ireland is gratefully acknowledged.

theless examples of community-based searches. For instance, the use of a Google search box on a specialised Web site (e.g. a motoring enthusiast's site) means that its searches are likely to be initiated by users with some common (motoring) interest. Alternatively, searches originating from a computer laboratory assigned to 2nd year students are likely to share certain characteristics related to their studies (courses, projects etc.) and social lives (college societies, local gigs etc.)

Of course, more formalised examples of community-based search are also possible. The advent of blogging services and social networking services such as Friendster and Orkut pave the way for a growing number of community-based search applications. While the precise nature of a community's shared interests may not be easy to characterise, they are nevertheless likely to be encoded within the search patterns (queries and result selections) of the community's members.

2.2 How Much Repetition?

If many searches can be traced back to ad-hoc communities of searchers, what degree of regularity can be observed? We can begin to answer this question by profiling the degree of term overlap between queries from different communities of searchers. One way to measure query similarity is by the degree of overlap between query terms as in Equation 1; for example, $Sim('jaguar pictures', 'jaguar photos')=0.33$.

$$Sim(q, q') = \frac{|q \cap q'|}{|q \cup q'|} \quad (1)$$

Previous analyses of a variety of search engine logs have shown that query repetition is prevalent in specialised search scenarios that are likely to attract communities of like-minded searchers. For example [Smyth *et al.*, In Press] report how it is common to find that up to 70% of search queries may share at least 50% of their query terms with other queries'; this drops to 30% for more general search scenarios. Later in this paper we describe a major evaluation of collaborative Web search involving the employees of a local software company. Prior to this evaluation, we performed a similar query analysis over 9 weeks worth of search sessions extracted from the company's Internet access logs. Our working hypothesis at the time was that these employees would behave as a community of like-minded searchers and that their search queries would exhibit a high degree of similarity, thus motivating collaborative Web search.

The results are presented in Figure 1 as the percentage of queries at set similarity thresholds and the average number of similar queries for these different thresholds. The results show that the group of searchers do appear to behave as a community of like-minded users as high degrees of repetition are noted for many similarity thresholds. For example, we see that nearly 60% of queries share at least 50% of their query terms with other queries and that on average each of these queries shares 50% of its terms with about 6 other queries.

3 A Review of Collaborative Web Search

The collaborative Web search technique is conceived of as a form of meta-search; see Figure 2 for the summary architecture and refer to [Freyne *et al.*, 2004; Smyth *et al.*, In Press]

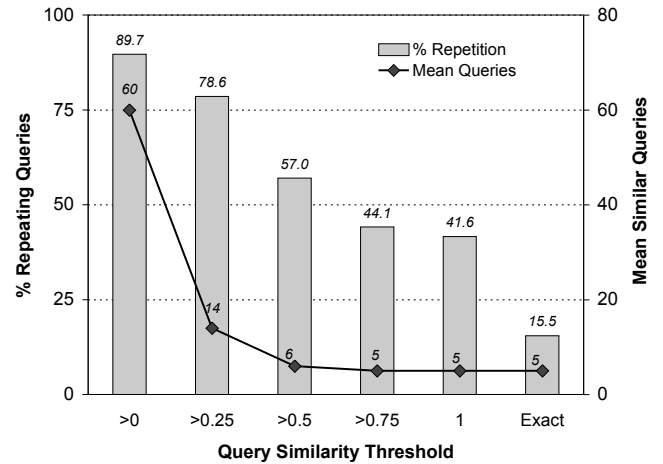


Figure 1: Query repetition in the search logs of the community used as part of the evaluation in Section 4.

for a more comprehensive technical description. Briefly, each new user query, q_T , is submitted to a set of underlying search engines and their results are combined to form a meta-search result-list, R_M . The novelty of collaborative Web search stems from the way that this result-list is processed to produce a new result-list, R_T , that reflects the learned preferences of a community of like-minded searchers. It achieves this by recording the selections of searchers. In other words, collaborative search records the fact that a result s^i has been selected for query q_T , and then reuses this information for similar queries in the future, by promoting results that were reliably selected in the past.

3.1 Profiling Community Preferences

The *hit-matrix*, H , is a key data structure for collaborative Web search. It is a record of the results selected in past search sessions by a specific community of users, and multiple hit matrices can be readily maintained to reflect the separate preferences of many different communities. Each time a searcher (from a specific community) selects a result page, p_j , that was retrieved for query, q_T , the value of H_{Tj} is incremented. Thus, H_{Tj} represents the number of times that p_j has been selected as a result for q_T . The row of H that corresponds to q_T provides a complete account of the number of all page selections for this query over all search sessions that have used this query. Note that no record is maintained of which user selected which result, so in effect the hit matrix serves as an anonymous account of community preferences.

3.2 Reusing Similar Queries

The similarity between a new query, q_T , and a search record (row) in a hit-matrix can be estimated by the term overlap between the new query and the query of the past search record (Equation 1); see [Balfe and Smyth, 2005] for a number of alternative query similarity models. Collaborative Web search selects those rows from the hit matrix whose corresponding query has a similarity to q_T that is above some specified threshold (typically 0.5). The pages associated with these

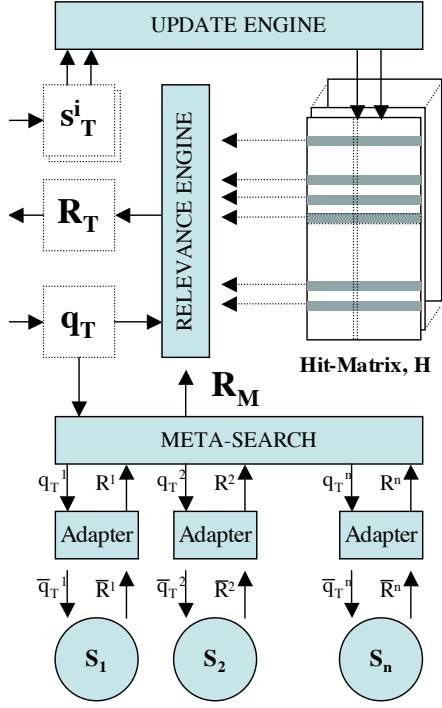


Figure 2: The collaborative web search architecture.

records (rows) are called *promotion candidates* and they are assumed to be potentially relevant to the new target query because they have been relevant for similar queries, and for the same community of searchers, in the past.

3.3 Result Relevancy & Ranking

Consider a page, p_j , that is associated with query, q_i . The relevance of p_j to q_i is estimated by the relative number of times that p_j has been selected for q_i ; see Equation 2. And the relevance of p_j to q_T is a combination of $Relevance(p_j, q_i)$ for all q_i 's ($q_1, \dots, q_n$) deemed similar to q_T , as shown in Equation 3. Each $Relevance(p_j, q_i)$ is weighted by $Sim(q_i, q_T)$ to discount the relevance of results from less similar queries; $Exists(p_j, q_i) = 1$ if $H_{ij} \neq 0$ and 0 otherwise.

$$Relevance(p_j, q_i) = \frac{H_{ij}}{\sum_{\forall j} H_{ij}} \quad (2)$$

$$WRel(p_j, q_T, q_1, \dots, q_n) = \frac{\sum_{i=1 \dots n} Relevance(p_j, q_i) \bullet Sim(q_T, q_i)}{\sum_{i=1 \dots n} Exists(p_j, q_i) \bullet Sim(q_T, q_i)} \quad (3)$$

This weighted relevance metric is used to rank-order the promotion candidates. These ranked pages are then listed ahead of the remaining meta-search results, which are themselves ranked (according to a standard meta-search scoring metric), to give R_T . Of course, alternative promotion models can also be envisaged but are omitted here for space reasons.

3.4 Communities and Collaboration

Obviously this approach assumes that the contents of a given hit-matrix reflect some relatively uniform domain of inter-

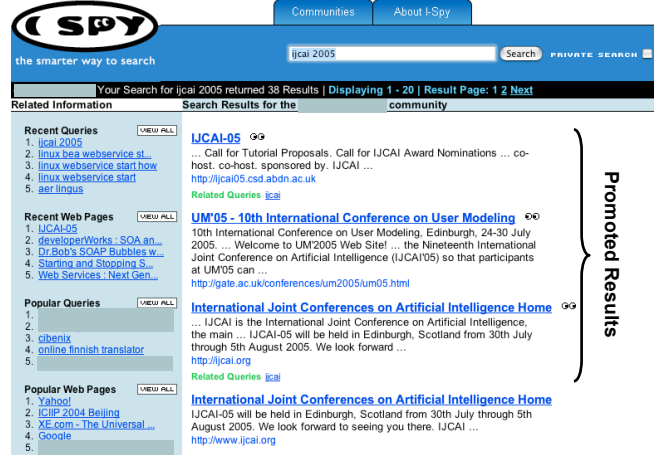


Figure 3: The I-SPY result page for the 'ijcai 2005' query.

ests, for a community of like-minded searchers. Collaborative Web search contemplates the creation of multiple hit-matrices to enable different communities of users to access a search service that is adapted for their query-space and their preferred pages. For example, a large Web portal might create different hit-matrices for different portal sections (e.g. News, Sports, Entertainment, Business sections) on the grounds that searchers are more likely to submit queries that are related to the content that is found within this portal section.

3.5 An Example Session

Collaborative Web search has been implemented in the form of the I-SPY search engine (<http://ispy.ucd.ie>). I-SPY can be configured to use a range of different search engines as its base-level search engines, including Google, Teoma, HotBot etc., and it allows users to use existing search communities or to create new ones via a simple form-based interface.

Figure 3 shows the results of a typical search for the query 'ijcai 2005' by a particular I-SPY community. The result-list is presented in the main panel, flanked by recent and popular queries and web pages lists; certain sensitive information items have been blanked out in the figure. In this case the top 4 results are shown and the first 3 of these are result promotions; indicated by the 'I-SPY eyes' icon next to the promoted result titles. This means that these results have been previously selected for this query or for similar queries. In fact we can see from the 'related queries' lists after the first and third results that these have been previously selected for the similar query 'ijcai'. The results shown are obviously relevant to the target query. The top result is for the main *IJCAI 2005* home page and the third result corresponds to the main *IJCAI Conferences* page, for example. However it is also worth noting that the second result is for the forthcoming user modeling conference, *UM 2005*. This page has been promoted because it has been selected in the past, for the current query, by members of the current community—these community members have a specific business interest in user modeling technology—but ordinarily this result would not be expected to appear so high in the result list for 'ijcai 2005'.

This result is, however, relevant to this query given the community context, especially since *UM 2005* takes place directly before *IJCAI 2005* and in the same city. This type of promotion speaks to the potential power of I-SPY to promote results that are uniquely relevant to the specific needs of a community of like-minded searchers—results that would ordinarily be lost among the competing results of traditional, generic search engines.

4 Live-User Evaluation

Past evaluations of collaborative Web search have included a mixture of artificial-user and live-user studies [Smyth *et al.*, 2003; Freyne *et al.*, 2004; Smyth *et al.*, In Press]. However these studies have been limited; for example, the live-user evaluation studied a narrowly focused, single-shot, question-answering search task which did not allow for more realistic open-ended search scenarios over an extended period of time. In this section we describe the results of a more realistic trial, which took place over a 4-week period among the 50 staff members of a local software company.

4.1 Preliminaries

The trial began on Monday, November 8, 2004 and the results presented in this paper account for the 4 working-weeks (Monday to Friday) up to and including December 3. During this time employees were asked to use I-SPY as their primary search engine; prior to the trial 90% of search sessions used Google. I-SPY was configured to draw on Google and HotBot as a source of search results and a new community was created for participants with a hit-matrix trained from search log data for the 9 weeks prior to the start of the trial. I-SPY's query-similarity threshold was set at 50%, so that only those past sessions that shared more than 50% of their query terms with the current target query would be considered to be similar for the purposes of result promotion (see Section 3). Participants were introduced to I-SPY via a short explanatory email and encouraged to use it as they would a normal search engine. Over the 4 weeks more than 1500 queries were submitted and more than 1800 result URLs were selected.

Figure 4 presents a histogram of the number of search sessions with different numbers of promotions. It shows that 46% of search sessions contained at least 1 promoted result; on average these sessions contained 3.7 promotions. This speaks to the potential for I-SPY's result-promotion technique to usefully influence a significant percentage of searches. The results are in broad agreement with the pre-trial query-overlap analysis described in Section 2, which suggested that we could expect up to 57% of new queries to have 50%-similar queries from the past to draw on as a source of promotions (see Figure 1).

4.2 Successful Sessions

While the above figures indicate that I-SPY is making promotions in roughly half of the search sessions, the real test is whether these promotions turn out to be relevant for the searcher, and whether they are *more* relevant than non-promoted results. Evaluating the relevance of search results in a trial such as this is difficult to do, at least in a direct fashion. Standard search interfaces do not provide a facility to

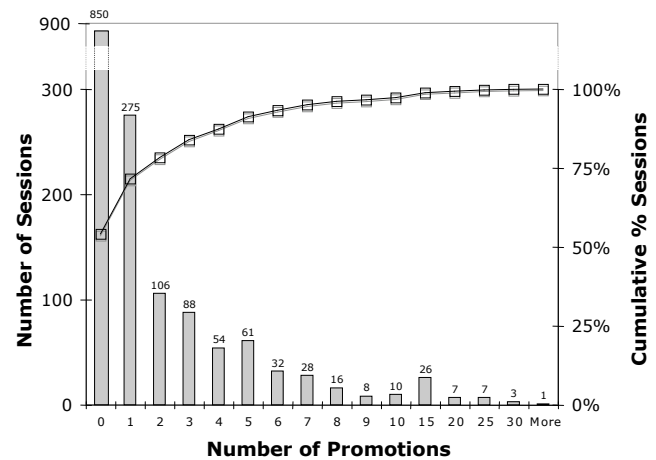


Figure 4: A histogram of the number of search sessions with different numbers of result promotions.

allow users to indicate how well their information needs have been answered by search results, and while it would be possible to add such a facility to I-SPY for the purpose of measuring relevance in this trial, many users indicated that they would find this to be a nuisance. For this reason we examine a less direct measure of relevance.

We propose that the selection of at least one result in a given search session acts as a crude, but nevertheless useful indicator of result-list relevance. We refer to a search session, where at least one result has been selected, as a *successful session*. If no results are selected (a *failed session*) then we can be relatively confident that the search engine has not retrieved a result that is *obviously* relevant to the searcher. Note that we do not distinguish here between sessions with different numbers of selected results, mainly because it is not possible to conclude much from the frequency of result selections. For example, one might be tempted to conclude that users selecting more results is a sign of increasing result relevance, except that a similar argument can be made in support of decreasing result relevance, on the basis that the initial selections must not have satisfied the users.

To analyse the ability of collaborative search to deliver successful sessions, we split the search sessions into those that contained promotions (*promoted sessions*) and those that did not (*standard sessions*). The former correspond to sessions where collaborative search has the potential to influence relevance, whereas the latter serve as a pure meta-search benchmark against which to judge this influence. Incidentally, there appears to be no difference between the queries for the promoted sessions when compared to those for standard sessions and both sets of queries have almost identical distributions; for example, an average of 2.4 terms per query for the promoted sessions compared to 2.5 for the standard sessions was measured. Indeed, given enough time it is likely that many of the standard queries would eventually be paired with new similar queries and so participate in future promoted sessions.

Figure 5(a) presents the average percentage of successful sessions among the promoted and standard sessions and

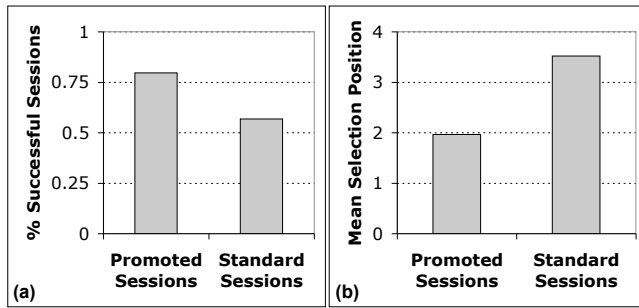


Figure 5: Promoted vs. Standard Sessions: (a) the percentage of successful sessions; and (b) the mean position of selected results among successful sessions.

demonstrates a clear advantage for the promoted sessions. On average, 80% of the promoted sessions were successful, compared to 56% for the standard sessions, a difference that is significant at the 99% confidence level. In other words, the collaborative search result-promotion mechanism leads to a 40% relative improvement in the chances that a given search will translate into a successful search session.

4.3 Selection Positions

As a complementary measure of result-relevance, it is also interesting to compare the promoted and standard sessions in terms of the average position of selected results within successful sessions; that is, those sessions in which selections have been made. We would like to see relevant results appearing higher up in result-lists. Moreover, assuming that users are likely to select results that at least appear to be more relevant than those that do not, then we would like to minimise the mean position of a selected result.

Figure 5(b) presents the mean position of the selected results among the successful sessions of the promoted and standard sessions. This once again shows a clear advantage for the former. On average, the mean position of a selected result among the successful promoted sessions is 1.96, compared to 3.51 for the successful standard sessions. This difference is statistically significant at the 99% confidence level and corresponds to a 44% reduction in the position of relevant results for promoted sessions compared to standard sessions.

It is worth commenting on the importance of this observed difference in the selection positions. While there is an advantage due to the promoted sessions, one might ask whether the observed reduction of one or two places is likely to be important. We believe that it is, for a number of reasons, not the least of which is that results should be ordered by their expected relevance as a matter of course. In addition, users have a tendency to focus their attention on the top-ranked results. The fact that promoted sessions have a higher success rate than the standard sessions is likely due to this difference in the position of apparently relevant results, because for the most part I-SPY promotes results from lower-down in the standard result-lists (returned by Google and HotBot) to higher positions.

This observed difference may become even more important

in other search scenarios, such as mobile Web search, where screen-space is so restricted as to severely limit the number of results that may be presented on a single screen. For instance, on many mobile devices (eg. WAP phones), screen-space is so restricted that only 3 results can be presented per screen. The positional advantage enjoyed by I-SPY results suggests that it has the potential to ensure that relevant results will appear on the first page of such mobile-search results. In fact 99% of the result selections that occur in the promoted sessions are for results in the top 3 of a result-list, compared to only 79% of the standard session selections. Moreover, 93% of promoted session selections are for the top result, compared to only 63% of standard session selections.

5 Discussion

The results so far indicate that I-SPY's collaborative Web search has the potential to significantly improve search performance. Result promotions are made frequently, and when they are they translate into more successful search sessions and a better ranking for relevant results. In this section we briefly consider the number of promotions made during a session and the likely success of this session. In our analysis to date we have noticed that promoted sessions can contain up to 10 or more promoted results; this may be a problem because too many promotions may *swamp* result-lists to the detriment of search performance. In addition, sessions with many promoted results are likely to be caused by the reuse of large numbers of past search sessions, some of which may be the result of less reliable query overlaps, which in turn are more likely to contribute results of limited relevance to the target query.

One solution that we have adopted recently is to provide the searcher with a facility to adjust the level of community personalization that is offered, by manipulating a slider-bar to increase the number of promoted results that are displayed (see Figure 6). We are also considering different result-integration strategies to allow for a more flexible combination of I-SPY relevance and meta-search result scores.

We are concerned about issues relating to fairness, reliability and security. For example, it should be clear that as it stands, older results will tend to be preferred over newer results; the former will have had a greater opportunity to attract selections. This may cause problems when it comes to the promotion of very recent results. We are currently looking at ways to address this issue, for example by using a suitable decay function to gradually erode the selections of older results. We are also investigating ways to detect false selections by unreliable searchers as a way to defend against the fraudulent activities of self-interested parties; see also [O'Mahony *et al.*, 2002; Smyth *et al.*, In Press].

6 Conclusions

Collaborative Web search is an approach to Web search that exploits the natural regularity that exists within the search behaviours of ad-hoc communities of users. It espouses the reuse of search sessions for past queries that are similar to the current target query, resulting in the active promotion of those results that have been preferred by the community in

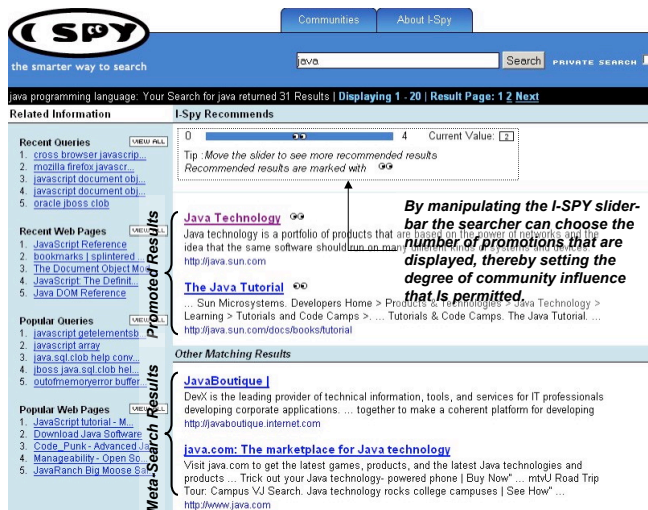


Figure 6: I-SPY's promotion slider-bar allows the searcher to manipulate the degree of community personalization.

the past. In effect, this approach offers a form of anonymous personalization, thus protecting the privacy of individuals.

In this paper we have presented the results of a significant evaluation of the collaborative Web search approach in a corporate search environment. The results show the clear benefits that are available: the promotion of results translates into more successful search sessions and relevant results are ranked more highly. The evaluation has also helped to clarify the potential pitfalls of swamping result-lists with too many promotions. We have found that the best benefits arise from between 5-8 result promotions indicating the need for a mechanism to limit the number of promotions on a session by session basis. In response, we have described how I-SPY's interface has been adapted to include a means for searchers to interactively control the number of promotions at search time.

References

- [Balfe and Smyth, 2005] Evelyn Balfe and Barry Smyth. An Analysis of Query Similarity in Collaborative Web Search. In *Proceedings of the European Conference on Information Retrieval*, pages 330–344. Springer-Verlag, 2005.
- [Bollmann-Sdorra and Raghavan, 1993] Peter Bollmann-Sdorra and Vijay V. Raghavan. On the Delusiveness of Adopting a Common Space for Modeling IR objects: Are Queries Documents? *Journal of the American Society for Information Science*, 44(10):579–587, 1993.
- [Brin and Page, 1998] Sergey Brin and Lawrence Page. The Anatomy of a Large-Scale Hypertextual Web Search Engine. *Computer Networks and ISDN Systems*, 30(1-7):107–117, 1998.
- [Fitzpatrick and Dent, 1997] Larry Fitzpatrick and Mei Dent. Automatic Feedback using Past Queries: Social Searching? In *Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 306–313. ACM Press, 1997.
- [Freyne et al., 2004] Jill Freyne, Barry Smyth, Maurice Coyle, Evelyn Balfe, and Peter Briggs. Further Experiments on Collaborative Ranking in Community-Based Web Search. *Artificial Intelligence Review*, 21(3-4):229–252, 2004.
- [Furnas et al., 1987] George W. Furnas, Thomas K. Landauer, Louis M. Gomez, and Susan T. Dumais. The Vocabulary Problem in Human-System Communication. *Communications of the ACM*, 30(11):964–971, 1987.
- [Glance, 2001] Natalie S. Glance. Community Search Assistant. In *Proceedings of the International Conference on Intelligent User Interfaces*, pages 91–96. ACM Press, 2001.
- [Kleinberg, 1999] Jon M. Kleinberg. Authoritative Sources in a Hyperlinked Environment. *Journal of the ACM*, 46(5):604–632, 1999.
- [Lawrence and Giles, 1998] Steve Lawrence and C. Lee Giles. Context and Page Analysis for Improved Web Search. *IEEE Internet Computing*, July-August:38–46, 1998.
- [Lawrence, 2000] Steve Lawrence. Context in Web Search. *IEEE Data Engineering Bulletin*, 23(3):25–32, 2000.
- [Lyman and Varian, 2003] Peter Lyman and Hal R. Varian. How Much Information. Retrieved from <http://www.sims.berkeley.edu/how-much-info-2003> on January 14th, 2005, 2003.
- [O'Mahony et al., 2002] Michael O'Mahony, Neil Hurley, and Guenole C. M. Silvestre. An Attack on Collaborative Filtering. In *Proceedings of the 13th International Conference on Database and Expert Systems Applications*, pages 494–503. Springer-Verlag, 2002.
- [Raghavan and Sever, 1995] Vijay V. Raghavan and Hayri Sever. On the Reuse of Past Optimal Queries. In *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 344–350. ACM Press, 1995.
- [Roush, 2004] Wade Roush. Search Beyond Google. *MIT Technology Review*, pages 34–45, 2004.
- [Smyth et al., 2003] Barry Smyth, Evelyn Balfe, Peter Briggs, Maurice Coyle, and Jill Freyne. Collaborative Web Search. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence, IJCAI-03*, pages 1417–1419. Morgan Kaufmann, 2003. Acapulco, Mexico.
- [Smyth et al., In Press] Barry Smyth, Evelyn Balfe, Jill Freyne, Peter Briggs, Maurice Coyle, and Oisín Boydell. Exploiting Query Repetition & Regularity in an Adaptive Community-based Web Search Engine. *User Modeling and User-Adapted Interaction: The Journal of Personalization Research*, (In Press).
- [Wen, 2002] Query clustering using user logs. *ACM Transactions on Information Systems*, 20(1):59–81, 2002.

Clinical-Reasoning Skill Acquisition through Intelligent Group Tutoring

Siriwan Suebnukarn and Peter Haddawy

Asian Institute of Technology

Computer Science and Information Management Program

Pathumthani, Thailand, 12120

{Siriwan.Suebnukarn, haddawy}@ait.ac.th

Abstract

This paper describes COMET, a collaborative intelligent tutoring system for medical problem-based learning. COMET uses Bayesian networks to model individual student knowledge and activity, as well as that of the group. Generic domain-independent tutoring algorithms use the models to generate tutoring hints. We present an overview of the system and then the results of two evaluation studies. The validity of the modeling approach is evaluated in the areas of head injury, stroke and heart attack. Receiver operating characteristic (ROC) curve analysis indicates that, the models are accurate in predicting individual student actions. Comparison of learning outcomes shows that student clinical reasoning gains from our system are significantly higher than those obtained from human tutored sessions (Mann-Whitney, $p = 0.011$).

1 Introduction

The transformation from medical student to physician is a gradual one, requiring the assimilation of vast amount of knowledge as well as the development of clinical-reasoning skills. Clinical reasoning is the cognitive process by which the information contained in a clinical case is synthesized, integrated with the physician's knowledge and experience, and used to diagnose or manage the patient's problem [Newble *et al.*, 1994]. Problem-based learning (PBL) has been introduced as an alternative to traditional didactic medical education to teach clinical-reasoning skills at the early stages of medical education. PBL is designed to challenge learners to build up their knowledge and develop effective clinical-reasoning skills around practical patient problems. PBL instructional models vary but the general approach is student-centered, small group, collaborative problem solving activities [Barrows, 1986]. The main arguments for using collaborative problem solving in medical PBL include the wider range of ideas generated and the higher quality of discussion that ensues. In addition, students obtain training in the skill of consultation and group clinical problem solving, which are important for the successful practice of clinical medicine. But effectively implementing PBL in the clinical curriculum is difficult due to

the lack of standards for PBL tutoring [Das *et al.*, 2002] and a lack of properly trained tutors. In addition, effective PBL requires the tutor to provide a high degree of personal attention to the students. In the current academic environment where resources are becoming increasingly scarce and costs must be reduced, providing such attention becomes increasingly difficult. This is exacerbated by the fact that medical school faculty, in particular, often have limited time to devote to teaching. As a consequence, medical students often do not get as much facilitated PBL training as they might need or want.

There has been increasing interest in application of intelligent technologies to medical training to provide rich environments for maximizing learning while minimizing risks to patients, until sufficient competency is established. The majority of the work in intelligent medical training system has focused on particular domains, such as Radiology [Sharpley *et al.*, 2000] or Pathology [Crowley and Medvedeva, 2003] for training students in feature perception and disease classification. Little or no work has addressed providing a general domain-independent framework for intelligent medical tutoring and no work has addressed intelligent medical tutoring in group settings.

We have developed a collaborative intelligent tutoring system for medical problem-based learning called COMET. COMET uses Bayesian networks to model individual student knowledge and activity, as well as that of the group. It uses generic tutoring algorithms applied to the models to generate tutorial hints to guide problem solving activity. In previous work [Suebnukarn and Haddawy, 2004] we presented a basic Bayes net student model and details of the tutoring algorithms. We also presented results of a study showing that the hints generated by COMET agree with those of a majority of human tutors. In this paper we present a new, more expressive Bayesian network student model, along with an ROC analysis evaluating of the model's accuracy. We also evaluate the overall effectiveness of COMET in imparting clinical reasoning skills to medical students by comparing clinical reasoning exam scores of COMET tutored students to those of human tutored students.

2 Medical Problem-Based Learning

Problem-based learning (PBL) can be described as “the learning that results from the process of working toward the understanding or resolution of a problem” [Barrows, 1986]. PBL is typically carried out in three phases: (1) *Problem analysis*: In group discussion the students evaluate the patient problem presented to them exactly as they would a real patient, attempting to determine the possible underlying anatomical, physiological, or biochemical dysfunctions and to enumerate all possible causal paths (hypotheses and their causal relations) that would explain the progression of the patient’s problems. (2) *Self-directed study*: In this phase, students work outside the tutorial session, using any relevant learning resources, e.g. literature, laboratories, specialists, to address any open issues identified in the first phase. (3) *Synthesis and application of newly acquired information*: The students analyze data and wrap up the problem collaboratively after they return from their self-study period.

One of the main issues in PBL is the role of the tutor. Like a good coach, a tutor needs enough command of whatever the learners are working on to recognize when and where they most need help [Das *et al.*, 2002]. The ideal tutor should be an expert in both learning content and learning process, which is rare to find among human tutors. The tutor intervenes to as small an extent as possible, giving hints only when the group appears to be getting stuck or off track. In this way, the tutor avoids making the students dependent on him for their learning.

3 A Collaborative Medical Tutor

3.1 Conceptual Framework

COMET is designed to provide an experience that emulates that of live human-tutored medical PBL sessions as much as possible while at the same time permitting the students to participate collaboratively from disparate locations. COMET incorporates a multi-modal interface that integrates text and graphics so as to provide a rich communication channel between the students and the system, as well as among students in the group (Fig. 1). Students collaboratively create the problem solution on the hypothesis board, shown at the bottom of Fig. 1. Typically each student works from a separate computer. COMET can currently support PBL problem analysis in the domains of Head injury, Stroke and Heart attack. Note that these three domains are quite different since the knowledge used to reason about head injury is primarily anatomical, while that used to reason about stroke and heart attack is primarily physiological. Furthermore, the patho-physiology of the latter two diseases is more dynamic.

Generating appropriate tutorial actions in COMET requires a model of the students’ clinical reasoning for the problem domain. This modeling task is necessarily wrought with uncertainty since we have only a limited number of observations from which to infer each student’s level of understanding. Thus we have chosen to use Bayesian networks as our modeling technique.

The system is implemented as a Java client/server combination, which can be used over the Internet or local area networks and supports any number of simultaneous PBL groups.

The system implementation is modular and the tutoring algorithms are generic so that adding a new scenario requires only adding the appropriate model representing how to solve a particular case (*domain clinical reasoning model*). The *student clinical reasoning model*, which is a probabilistic overlay of the domain clinical reasoning model, is then constructed during runtime by instantiating the nodes that represent the knowledge and activity of an individual student. The architecture of COMET differs from that of most ITS’s in that the domain model and student model are embodied in one representation. The domain model is contained in the part of the structure of the network that represents the hypotheses and the cause-effect relations among them. The student model is contained in the part of the network that represents how the hypotheses are derived and in the network’s probabilities. The probabilities do not represent the likelihood of occurrence of the hypotheses but rather the likelihood that a student will be able to create the hypotheses.

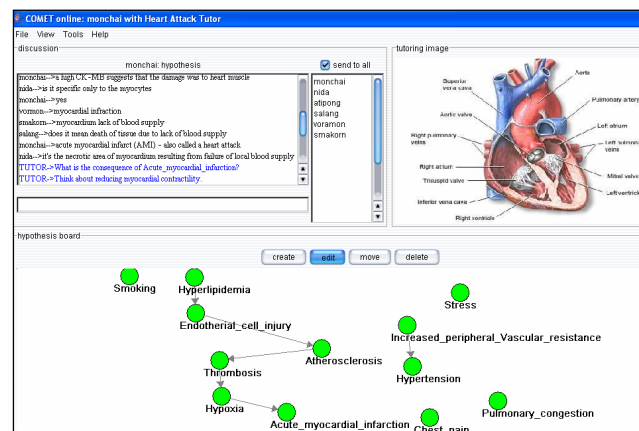


Figure 1. COMET student interface.

3.2 Domain and Student Clinical Reasoning Model

We built the domain clinical reasoning model based on the process of hypothesis generation in problem-based learning. Consider, for example, the heart attack scenario taken from a PBL session at Thammasat University Medical School.

“Mr. C, a 56-year-old who was diagnosed as having essential hypertension four years ago, is complaining of chest pain which feels like indigestion. You have noticed that he is mildly obese, pale, clammy and sweating profusely...”

Here students must enumerate possible hypotheses to explain why the patient is experiencing chest pain. Figure 2 is a photograph of the white board of the group PBL session, showing a directed acyclic graph representing cause-effect relationships among hypotheses. This graph represents the problem solution developed by the students. Since we would like to reason about the state of knowledge of each student concerning the solution, this graph is our starting point for the student model. The hypothesis graph can be conveniently represented as a Bayesian network since

Bayesian networks are also directed acyclic graphs. In addition, Bayesian networks can represent our uncertainty about the state of knowledge of the students.

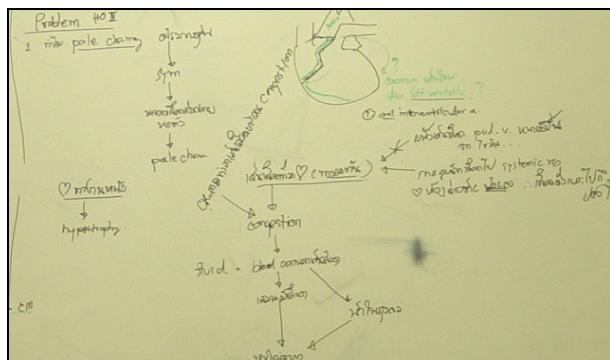


Figure 2. A photograph of the white board after a PBL session at Thammasat University Medical School. The graph shows hypotheses with arrows indicating cause-effect relations among them. (Note: Some hypotheses are written in Thai.)

The BN structure contains two types of information: (1) the hypotheses and the causal links of the problem solution (Fig. 3, right half) and (2) how students derive the hypotheses (Fig. 3, left half). We represent the hypothesis structure following the model of Felty and Barrows [1984], which defines three categories of illness features: enabling conditions, faults, and consequences. Enabling conditions are illness features associated with the acquisition of illness (e.g., compromised host factors, unusual travel, or hereditary factors). Faults are the major real malfunctions in illness (e.g., direct trauma, invasion of tissue by pathogenic organisms, or inadequate blood supply). Consequences are the secondary consequences of faults within the organism, and generally comprise different types of signs and symptoms, e.g., chest pain, breathlessness, or tachycardia. In Figure 3 (right half), we have five possible faults associated with the single consequence chest pain: *Myocardial infarction*, *Angina*, *Musculoskeletal injury*, *Gastrointestinal disorder*, and *Stress*. *Atherosclerosis* is the enabling condition of *Myocardial infarction* and *Angina*. The remaining hypothesis nodes are consequences of *Myocardial infarction*. Each hypothesis node has parent nodes, which have a direct casual impact on it. For example, *Right heart failure* has parents *Pulmonary congestion* and *Myocardial infarction*. All hypothesis nodes have two states, indicating whether or not the student knows that the hypothesis is a valid hypothesis for the case.

In the PBL sessions, the students create the hypotheses as well as the causal links between them (Fig. 2). We would like to be able to reason about the probability that students know the correct causal links. But in a Bayes net, random variables are represented with nodes. So we use *link nodes* to represent the causal links between hypotheses. For every hypothesis that is a direct cause of another hypothesis (e.g. *Atherosclerosis* and *Myocardial infarction*), we have a node (e.g. *Link 14*) representing the causal link between them. The two hypothesis nodes (*Atherosclerosis*, *Myocardial infarction*) are the parents of the link node. The intuition is that the link can-

not be created unless both hypotheses are created first. Each link node has two states, indicating whether or not the student creates a causal link between two hypotheses.

The derivation of hypotheses (Fig. 3, left half) is represented in terms of three kinds of nodes: goals, general medical knowledge, and apply actions. Every hypothesis node has a unique *Apply* node as one of its parents. The *Apply* node represents the application of a medical concept to a goal in order to derive the hypothesis. For example the *Apply 13* node indicates that the student is able to use knowledge of the *Vessel Lumina occlusion* medical concept to infer that *Myocardial infarction* is a consequence of *Atherosclerosis*. Each hypothesis node thus has a conditional probability table specifying the probability of the hypothesis being known conditioned on whether the parent hypotheses are known and whether the student is able to apply the appropriate piece of knowledge to determine the cause-effect relationship. The conditional probability tables for the *Apply* nodes are simple AND gates.

Our BN student model is similar to the student model used by Conati, et al [2002]. Their model includes five types of nodes: Context-Rule, Rule-Application, Fact, Goal, and Strategy. The correspondence between their node types and ours is: Context-Rule = Concept, Rule-Application = Apply, Fact = Hypothesis, and Goal = Goal. Strategy nodes, which represent different correct solutions to a problem, are implicitly encoded in our model by the fact that students can enumerate the causal hypothesis structure in any order. Our model contains causal links among hypotheses, which are not present in their model. The reason for this is that in our medical domains a problem solution is represented by the hypotheses and causal links among them, while in their physics domains a problem solution is represented by a sequence of rule applications and the derived facts.

For each problem scenario, we consulted medical textbooks and expert PBL tutors to obtain the hypotheses, the causal relations among them, the goals, and the medical concepts used to derive the hypotheses. The conditional probability tables for each resulting network were obtained by learning from data obtained from transcripts of PBL sessions. The data for this study consisted of tape recordings and photographs of tutorial sessions for the head injury, stroke and heart attack scenarios at Thammasat University Medical School. A total of 15 groups of third year medical students were involved in this study. Each group, consisting of eight students with different backgrounds, was presented with the head injury, stroke and heart attack cases and asked to construct possible hypotheses for the case, under the guidance of a tutor. After the sessions the tape and the results on the whiteboard were analyzed to determine whether or not each goal, concept, hypothesis, and link was mentioned. We used the EM learning algorithm provided by the HUGIN Researcher software to learn the conditional probabilities of each node [Lauritzen 1995].

3.3 Individual and Collaborative Student Clinical Reasoning Modeling

The domain clinical reasoning model is instantiated for each student by entering that student's medical background

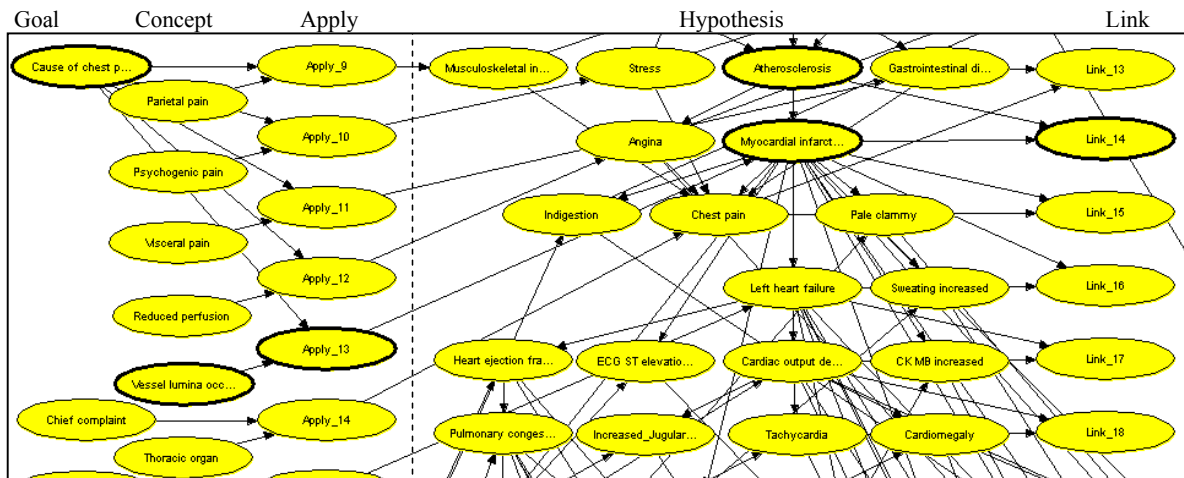


Figure 3. Part of the Bayesian Network clinical reasoning model of the heart attack scenario. The complete network contains 194 nodes. The model contains five types of nodes: goal, concept, apply, hypothesis, and link.

knowledge as evidence. For example, if a student has a background in Thoracic anatomy, we would instantiate the *Thoracic organ* node. Since all students have *basic* knowledge in Anatomy, Physiology and Pathology before they encounter the PBL tutorial sessions, we make the assumption that once a hypothesis in the domain model is created by one student in the group, every student knows that hypothesis. So as hypotheses are created, they are instantiated in each student model.

Following commonly accepted practice in medical PBL, we assume that students should and generally do enumerate the possible hypotheses by focusing sequentially on the various causal paths in the domain, linking enabling conditions with faults and consequences. So for each student, we must determine what causal path he is reasoning along, which we do by identifying the path of highest probability in that student's model. This is computed as the joint probability of the nodes along the path. The most likely current reasoning path for each student is path that gives the maximum joint probability. Since the students work in a group, it is also necessary to identify a causal path that can be used to focus group discussion, particularly when the discussion seems to be diverging in different directions. We would like to identify a path that has much of the attention of much of the group and has at least one member whose attention is focused on that path. We identify a set of candidate paths by taking the most likely path for each student. This guarantees that each candidate path has at least one student currently focused on it. We then compute the sum of the probabilities of each candidate path over all students and select the path with the highest sum. This gives us the candidate path with the highest average attention over all students.

3.4 Generating Tutorial Hints

Our automated tutor takes on the role of guiding the tutorial group to construct possible hypotheses for the case by the use of open-ended questions. From our study of the tutoring session transcripts, we identified and implemented seven hint strategies commonly used by experienced human tutors:

- (1) *Focus group discussion*: Members of the group may suggest various valid hypotheses without focusing on any given causal path. When such lack of focus becomes apparent, COMET intervenes by directing the students to focus on one of the hypotheses in the group path.
- (2) *Promote open discussion*: If a student proposes a hypothesis that is not on the current group reasoning path, COMET provides positive feedback by encouraging the student to relate the hypothesis to the current focus of discussion.
- (3) *Deflect uneducated guessing*: If a student creates an incorrect causal link, COMET points this out and encourages the student to correct the error.
- (4) *Avoid jumping critical steps*: If a student creates a link that jumps directly from one hypothesis to a down-stream consequence, leaving out intermediate hypotheses, COMET asks the student for the more direct consequences.
- (5) *Address incomplete information*: Once the students have completed elaborating all hypotheses on the group path, COMET identifies another path for them to work on.
- (6) *Refer to experts in the group*: If after COMET provides a general and then a more specific hint, the students still do not respond correctly, COMET determines the student most likely to know the answer and refers directly to him.
- (7) *Promote collaborative discussion*: If one student dominates the discussion, COMET asks for input from the other students. If a student does not contribute after a certain number of hypotheses have been mentioned, COMET solicits input from that student.

All strategies except strategies 6 and 7 have general and specific versions. COMET first gives a general hint using the parent goal node of the hypothesis that it has determined the students should focus on, and if there is no student response or an incorrect response, the more specific parent medical concept node is used. If the students can still not come up with the hypothesis of interest, COMET refers directly to the student in the group most likely to know the answer. If this doesn't work, COMET identifies this as a learning objective for study outside the session.

We developed algorithms to generate each of these types of hints, using as input the interaction log and the Bayesian

network student models. All strategies except strategy 7 use both the structure and the probabilities of the Bayes net models. Strategy 7 uses only a count of the number of inputs from each student. Strategies 1, 2, 5 make use of the group reasoning path discussed in the previous section. The following transcript shows the interaction with the system after the students read the heart attack problem scenario, described in Section 3.2. The system selects hint strategies and content based on the student input on the hypothesis board.

Students: Gastrointestinal disorder, Smoking, Hypertension, Angina, Myocardial infarction, Chest pain (*Students in the group gradually create six hypotheses on the board, while analyzing the problem.*)

Tutor: What is the consequence of Myocardial infarction? (*Strategy 1: COMET focuses group discussion by identifying which causal path the group should focus on, finding the highest probability non-mentioned node along the path (Left heart failure), and providing a hint using its parent goal node (Consequence of Myocardial infarction).*)

Student: Cardiac output decreased (*Cardiac output decreased is a node along the group reasoning path but not the node that COMET wants the group to focus on.*)

Tutor: Think about decrease in myocardial contractility. (*Strategy 1: COMET gives the next hint using the medical concept parent node of Left Heart Failure.*)

Student: Right heart failure (*Right heart failure is a node along the group reasoning path but not the node that COMET wants the group to focus on.*)

Tutor: It seems we have a problem. Nida, can you help the group? (*Strategy 7: Nida is called on since she has the highest probability of knowing the Left heart failure node among the students.*)

4 Evaluation

4.1 Evaluation of the Student Model

In order to determine the accuracy of the model, we compared the probabilities of hypotheses and causal links from the student model with actual student actions considered as a “gold standard”. We recruited 15 second-year medical students from Thammasat University Medical School. That is, they had not yet had PBL experience in Head injury, Stroke, or Heart attack. Stratified random sampling was applied to divide the students into 3 groups based on their background knowledge. Students were asked to answer pre-test questions to determine their background knowledge. This information was used to instantiate the general student model for each individual student. Students participated individually in the problem solving sessions on head injury, stroke, and heart attack with COMET. Each student was asked to enumerate hypotheses and the causal links without any help from the COMET tutor. The student actions of creating hypotheses and their links served as a gold standard for comparison with the predicted probabilities from the Bayesian network student model.

Results

To determine whether our student models are accurate in predicting student actions, we evaluated them by means of receiver operating characteristic (ROC) curve analysis. The area under the curve (AUC) represents an overall measurement of performance of the student model, with 1.0 a perfect test and 0.5 representing a model with no discriminating capacity.

Table 1 shows the ROC curve analysis of the student models for the Head injury, Stroke, and Heart attack scenarios. There were more false positive cases in the Stroke scenario than in the others, since the student model for the Stroke scenario was built from students who had already studied Cerebrovascular knowledge from the Head injury scenario, while in this study, we recruited students who had not yet studied any of the three scenarios. Averaging results over all scenarios shows high accuracy in predicting both hypotheses and causal links.

Table 1. ROC analysis showing AUC for three scenarios

Scenarios	Prediction	
	Hypotheses	Causal links
Head injury	0.909	0.898
Stroke	0.765	0.838
Heart attack	0.868	0.905
All	0.832	0.900

4.2 Evaluation of Student Clinical Reasoning Gains

To evaluate the overall impact of the system on student learning, we designed a study to test the hypothesis that a COMET tutorial will result in similar student clinical reasoning gains to those obtained from a session with an experienced human PBL tutor.

We compared three groups of students tutored by COMET with three other groups of students tutored by experienced human tutors. The study had a pre/post test control group design. All students were assessed on their clinical reasoning before and after the PBL tutorial session on heart attack and stroke to determine the reasoning gains for each individual student.

We used the Clinical Reasoning Problem (CRP) approach for clinical reasoning assessment [Groves *et al.*, 2002]. Each CRP consisted of a clinical scenario that was vetted for clinical accuracy and realism by a specialist physician. Four cases in the pre-test set measured each student’s initial ability to solve the problems. Four other post-test cases measured their ability to generalize the clinical reasoning acquired from tutorial session to the new related cases. Participants were asked to nominate the two diagnoses they considered most likely, to list the features of the case that they regarded as important in formulating their diagnoses, and to indicate whether these features were positively or negatively predictive. To establish reference scores, ten volunteer general practitioners (GPs) were asked to complete both sets of CRPs.

Results

There were no statistically significant differences between pre- and post-test scores obtained from the GPs, indicating

that the pre- and post-tests were of approximately equal difficulty (Table 2). The GPs' scores varied from 88.20 to 91.50 indicating that the questions were not trivial. Reliability, the measure of the reproducibility of a test, was measured using Cronbach's alpha. Cronbach's alpha for pre-and post-test student scores were 0.901 and 0.921 respectively. A reliability coefficient of 0.80 or higher is commonly considered as acceptable.

Table 2. Mean score for all CRPs (CRPs 1.1, 1.2, 2.1, 2.2 are chest pain cases. CRPs 1.3, 1.4, 2.3, 2.4 are stroke cases.)

	CRPs	GP's score (SD)	Student's score (SD)	
			COMET	Human tutor
Pre-test	1.1	88.70 (2.45)	34.67 (4.51)	34.00 (2.70)
	1.2	91.50 (2.46)	34.00 (3.27)	34.78 (2.73)
	1.3	88.20 (1.69)	37.72 (2.21)	38.61 (3.39)
	1.4	89.80 (3.49)	39.17 (2.18)	38.22 (3.54)
Post-test	2.1	89.50 (3.37)	62.28 (2.11)	58.11 (1.94)
	2.2	87.70 (4.42)	63.94 (1.95)	58.67 (2.40)
	2.3	90.60 (2.63)	64.06 (1.94)	65.00 (2.74)
	2.4	89.50 (3.27)	65.56 (1.98)	64.05 (2.39)

Table 3 shows that there were no statistically significant differences between pre-test mean scores of the COMET and human tutored groups. The post-test mean scores were significantly higher than the pre-test mean scores in both COMET and human tutored groups (Wilcoxon, $p = 0.000$), indicating that significant learning occurred. But the average post-test score for the COMET groups (64.96) was significant higher than that obtained for the human tutored groups (60.46) (Mann-Whitney, $p = 0.011$), indicating that students were learning more in the COMET sessions than in the human tutored sessions.

Table 3. Mean CRP score for each cohort

Cohort	Mean score (SD)	
	Pre-test	Post-test
COMET (1)	36.38 (3.45)	66.12 (3.38)
COMET (2)	37.00 (4.11)	64.33 (2.78)
COMET (3)	35.54 (4.24)	65.42 (3.10)
COMET (all)	36.31 (3.90)	64.96 (3.08)
Human tutor (1)	36.42 (2.95)	60.96 (2.49)
Human tutor (2)	37.42 (2.37)	62.63 (1.99)
Human tutor (3)	35.38 (3.42)	58.79 (2.68)
Human tutor (all)	36.40 (3.68)	60.46 (2.40)

5 Discussion

The results showing that clinical reasoning gains for COMET tutored students are higher than those for human tutored students were unexpected. This is particularly true in light of our earlier study showing that on average 74% of human tutors used the same hint strategy and content as COMET [Suebnuakarn and Haddawy, 2004]. We believe the explanation lies primarily in the 26% disagreement. Human tutors often give up after providing a general hint, jumping right to identifying the hypothesis as a learning objective. In contrast, COMET is more relentless in pushing the students,

always following the sequence of general hint, specific hint, referring to expert, and finally identifying as a learning objective. It is generally agreed that students should generate as many hypotheses as possible in a PBL session, leaving only the truly difficult issues as learning objectives.

Acknowledgments

We thank Hugin Expert for providing us the use of the Hugin Researcher software. Thanks to Thammasat University Medical School for their participation in the data collection and system evaluation, and to Dr. Kesorn Jongjarern for her helpful suggestions in designing the student clinical reasoning model.

References

- [Barrows, 1986] H. Barrows. A taxonomy of problem-based learning methods. *Medical Education*, 20: 481-486, 1986.
- [Crowley and Medvedeva, 2003] R. Crowley. and O. Medvedeva. A General Architecture for Intelligent Tutoring of Diagnostic Classification Problem Solving. In *Proceedings of AMLA Symposium*, pages 185-189, Washington DC, November 2003.
- [Conati et al., 2002] C. Conati, A. Gertner, and K. VanLehn.: Using Bayesian Networks to Manage Uncertainty in Student Modeling, *Journal of User Modeling and User-Adapted Interaction*. 12:371-417, 2002.
- [Das et al., 2002] M. Das, D. Mpofu, M. Hasan, and T. Stewart. Student perceptions of tutor skills in problem-based learning tutorials. *Medical Education*, 36:272-278, 2002.
- [Feltovich and Barrows, 1984] P. Feltovich and H. Barrows. Issues of generality in medical problem solving. *Tutorials in problem-based learning: A new direction in teaching the health professions*. Van Gorcum, The Netherlands, 1984.
- [Groves et al., 2002] M. Groves, I. Schott, and H. Alexander. Assessing Clinical Reasoning: A Method to Monitor its Development in PBL Curriculum. *Medical Teacher*. 5: 507-515, 2002.
- [Lauritzen 1995] S. Lauritzen. The EM-algorithm for graphical association models with missing data. *Computational Statistics and Data Analysis*, 1: 191-201, 1995.
- [Newble et al., 2000] D. Newble, G. Norman, and C. Vleuten. Assessing clinical reasoning, in: J. H. M. Jones (Ed.) *Clinical Reasoning in the Health Professions*, 2nd ed, pp. 156-168, Oxford, Butterworth-Heinemann, 2000.
- [Sharples et al., 2000] M. Sharples, N. Jeffery, B. du Boulay, B. Teather, D. Teather, and G. du Boulay. Structured computer-based training in the interpretation of neuroradiological images. *International Journal of Medical Informatics*, 60: 263-280, 2000.
- [Suebnuakarn and Haddawy, 2004] S. Suebnuakarn and P. Haddawy. A collaborative intelligent tutoring system for medical problem-based learning. In *Proceedings of the 9th International Conference on Intelligent User Interfaces*, pages 14-21, Madeira, Portugal, January 2004.

VISION AND ROBOTICS

Learning Partially Observable Deterministic Action Models

Eyal Amir

Computer Science Department
University of Illinois, Urbana-Champaign
Urbana, IL 61801, USA
eyal@cs.uiuc.edu

Abstract

We present the first tractable, *exact* solution for the problem of identifying actions' effects in partially observable STRIPS domains. Our algorithms resemble Version Spaces and Logical Filtering, and they identify all the models that are consistent with observations. They apply in other deterministic domains (e.g., with conditional effects), but are inexact (may return false positives) or inefficient (we could not bound the representation size). Our experiments verify the theoretical guarantees, and show that we learn STRIPS actions efficiently, with time that is significantly better than approaches for HMMs and Reinforcement Learning (which are inexact). Our results are especially surprising because of the inherent intractability of the general deterministic case. These results have been applied to an autonomous agent in a virtual world, facilitating decision making, diagnosis, and exploration.

1 Introduction

Autonomous agents have limited prior knowledge of their actions' preconditions and effects when they explore new domains. They can act intelligently if they learn how actions affect the world and use this knowledge to respond to their goals. This is important when their goals change because then they can reason about their actions instead of trying them in the world.

Learning actions' effects and preconditions is difficult in partially observable domains. The world state is not known completely, actions' effects mix with each other, and it is hard to associate change in one feature with a specific action or situation. Thus, it is not surprising that work so far has been limited to fully observable domains (e.g., [Wang, 1995; Pasula *et al.*, 2004]) and to hill-climbing (EM) approaches that have unbounded error in deterministic domains (e.g., [Ghahramani, 2001; Boyen *et al.*, 1999]).

This paper presents an approach called *SLAF* (Simultaneous Learning and Filtering) for exact learning of action's effects and preconditions in partially observable deterministic domains. This approach determines a set of possible transition relations, given a sequence of actions and partial observations. It is *online*, and updates a logical formula that models the possible transition relations and world states with every time step. It finds exactly those transition relations when actions are STRIPS (i.e., no conditional effects) or they map states 1:1.

Our algorithms take polynomial time in the number of features, n , and the number of time steps, T , for many cases. Their exact complexity varies with properties of the domain. For example, the overall time for learning STRIPS actions' effects is $O(T \cdot n)$. For other cases the update per time step takes linear time in the representation size. We can bound this size by $O(n^k)$ if we approximate the representation with a k -CNF formula, yielding an overall time of $O(T \cdot n^k)$ for the entire algorithm.

Our experiments verify these theoretical results and show that our algorithms are faster and better qualitatively than related approaches. For example, we can learn STRIPS actions' effects in domains of > 100 features *exactly*. In contrast, work on learning in Dynamic Bayesian Networks (e.g., [Boyen *et al.*, 1999]), reinforcement learning in POMDPs (e.g., [Littman, 1996]), and Inductive Logic Programming (ILP) (e.g., [Wang, 1995]) either approximate the solution with unbounded error for deterministic domains, or take time $\Omega(2^{2^n})$ (thus, are inapplicable in domains larger than 10 features). Section 7 provides a comparison with these and other works.

Our technical advance for deterministic domains is important for many applications such as automatic software interfaces, internet agents, virtual worlds, and games. Other applications, such as robotics, human-computer interfaces, and program and machine diagnosis can use deterministic models as approximations. Finally, understanding the deterministic case better can help us develop better results for stochastic domains.

In the following, Section 2 defines SLAF precisely,

Section 3 provides a deduction-based exact SLAF algorithm, Section 4 presents tractable model-update algorithms, Section 5 gives sufficient conditions and algorithms for keeping the model representation compact (thus, overall polynomial time), and Section 6 presents experimental results.

2 SLAF Semantics

We define our SLAF problem with the following formal tools, borrowing intuitions from work on Bayesian learning of Hidden Markov Models (HMMs) [Ghahramani, 2001] and Logical Filtering [Amir and Russell, 2003].

A *transition system* is a tuple $\langle \mathcal{P}, \mathcal{S}, \mathcal{A}, R \rangle$, where

- $\mathcal{P}$ is a finite set of propositional fluents;
- $\mathcal{S} \subseteq \text{Pow}(\mathcal{P})$ is the set of world states;
- $\mathcal{A}$ is a finite set of actions;
- $R \subseteq \mathcal{S} \times \mathcal{A} \times \mathcal{S}$ is the transition relation.

Thus, a *world state*, $s \in \mathcal{S}$, is a subset of $\mathcal{P}$ that contains propositions true in this state, and $R(s, a, s')$ means that state s' is a possible result of action a in state s . Our goal in this paper is to find R , given known $\mathcal{P}, \mathcal{S}, \mathcal{A}$, and a sequence of actions and partial observations (logical sentences on any subset of $\mathcal{P}$).

A *transition belief state* is a set of tuples $\langle s, R \rangle$ where s is a state and R a transition relation. Let $\mathfrak{R} = \text{Pow}(\mathcal{S} \times \mathcal{A} \times \mathcal{S})$ be the set of all possible transition relations on $\mathcal{S}, \mathcal{A}$. Let $\mathfrak{S} = \mathcal{S} \times \mathfrak{R}$. When we hold a transition belief state $\rho \subseteq \mathfrak{S}$ we consider every tuple $\langle s, R \rangle \in \rho$ possible.

For example, consider the situation presented in Figure 1. There, we have two rooms, a light bulb, a switch, an action of flipping the switch, and an observation, E (we are in the east room). The real states of the world, s_2, s_2' (shown in the top part), are unknown to us.

The bottom part of Figure 1 demonstrates how our knowledge evolves after performing the action sw-on . ρ_1, ρ_2 are our respective transition belief states. In ρ_1 we know that the possible world states are s_2, s_1 , or s_3 (s_1, s_3 are arbitrary world states), with R_2, R_1 , and R_3 , their respective transition relations. ρ_2 is the resulting transition belief state after action sw-on . Action sw-on takes state s_2 to s_2' according to transition relation R_2 , so $\langle s_2', R_2 \rangle$ is a pair ρ_2 . Similarly, it takes state s_1 to one of s_1', s_1'' according to transition relation R_1 , and it takes s_3 to s_3' according to R_3 . Thus, $\langle s_1', R_1 \rangle, \langle s_1'', R_1 \rangle, \langle s_3', R_3 \rangle$ are in pair ρ_2 . Finally, observing E eliminates the pair $\langle s_3', R_3 \rangle$ from ρ_2 , if E is false in s_3' .

Definition 2.1 (SLAF Semantics) Let $\rho \subseteq \mathfrak{S}$ be a transition belief state. The SLAF of ρ with actions and observations $\langle a_j, o_j \rangle_{1 \leq j \leq t}$ is defined by

1. $\text{SLAF}[a](\rho) = \{ \langle s', R \rangle \mid \langle s, a, s' \rangle \in R, \langle s, R \rangle \in \rho \};$
2. $\text{SLAF}[o](\rho) = \{ \langle s, R \rangle \in \rho \mid o \text{ is true in } s \};$
3. $\text{SLAF}[\langle a_j, o_j \rangle_{i < j \leq t}](\rho) = \text{SLAF}[\langle a_j, o_j \rangle_{i < j \leq t}](\text{SLAF}[o_i](\text{SLAF}[a_i](\rho)))$.

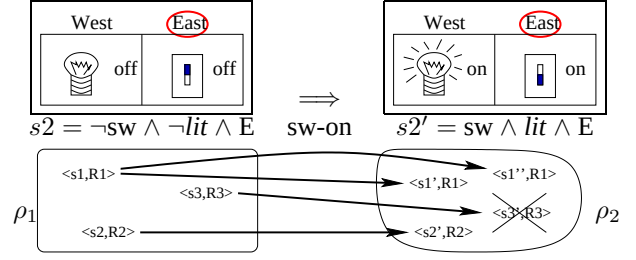


Figure 1: *Top*: Two rooms and flipping the light switch. *Bottom*: SLAF semantics; progressing an action (the arrows map state-transition pairs) and then filtering with an observation (crossing out some pairs).

Step 1 is progression with a , and Step 2 filtering with o .

We assume that observations (and observation model relating observations to state fluents) are given to us as logical sentences over fluents after performing an action. They are denoted with o . When actions can be inexecutable or may fail, it is useful to assume that $\mathcal{P}$ contains a special boolean fluent, OK , whose value is the success of the last action attempted.

Transition belief state ρ generalizes version spaces (e.g., [Wang, 1995]) as follows: If the current state, s , is known, then the version space's lattice contains the set of transition relations $\rho^s = \{R \mid \langle s, R \rangle \in \rho\}$. It also generalizes belief states: If the transition relation, R , is known, then the belief state (set of possible states) is $\rho^R = \{s \mid \langle s, R \rangle \in \rho\}$ (read ρ restricted to R), and Logical Filtering [Amir and Russell, 2003] of belief state σ and action a is equal to (thus, we define it as)

$$\text{Filter}[a](\sigma) = (\text{SLAF}[a](\{\langle s, R \rangle \mid s \in \sigma\}))^R.$$

3 SLAF via Logical Inference

Learning transition models using Definition 2.1 directly is intractable because it takes space $\Omega(2^{2^{|\mathcal{P}|}})$ in many cases. Instead, in this section we represent transition belief states more compactly (sometimes) using logic, and compute SLAF using general purpose logical inference.

We represent every *deterministic* transition relation, R , with a language, L , of propositions a_G^F whose meaning is “If G holds, then F will hold after executing a ”. We have $a_G^F \in L$ for every action $a \in \mathcal{A}$, F a literal (possibly negated proposition) from $\mathcal{P}$, and G a complete term over $\mathcal{P}$ (a conjunction of literals from $\mathcal{P}$ such that every fluent appears exactly once). We call $\mathcal{F}, \mathcal{G}$ the sets of effects, F , and preconditions, G , respectively. Thus, we have $2^{|\mathcal{P}|} \cdot 2^{|\mathcal{P}|} \cdot |\mathcal{A}|$ new propositional variables (in Section 5 we decrease this number).

Define φ_R , the logical theory that represents R , by $\varphi_R^0 = \{a_G^F \in L \mid \forall \langle s, a, s' \rangle \in R, s \models G \Rightarrow s' \models F\}$ and $\varphi_R = \varphi_R^0 \cup \{\neg a_G^F \mid a_G^F \in L \setminus \varphi_R^0\}$.

Some intuitive properties hold for this representation. φ_R is a complete theory for every R (a sentence or its negation is always implied from φ_R). Also, $\varphi_R \models \neg a_G^F \Leftrightarrow a_G^{\neg F}$, for every $F \in \mathcal{F}, G \in \mathcal{G}$.

Thus, for every transition belief state ρ we can define a theory in $\mathcal{L}(L \cup \mathcal{P})$ that corresponds to it: $\varphi_\rho = \bigvee_{\langle s, R \rangle \in \rho} (s \wedge \varphi_R)$. Similarly, for every theory φ in $\mathcal{L}(L \cup \mathcal{P})$ we define a transition belief state $\rho_\varphi = \{\langle s, R \rangle \mid s \in \mathcal{S}, s \wedge \varphi_R \models \varphi\}$, i.e., all the state-transition pairs that satisfy φ . We say that theory φ is a *transition belief formula*, if $\varphi_{\rho_\varphi} \equiv \varphi$ (note: $\rho_{\varphi_\rho} = \rho$ always holds).

For a deterministic (possibly conditional) action, a , define the effect model of a for time t to be

$$T_{\text{eff}}(a, t) = \bigwedge_{l \in \mathcal{F}, G \in \mathcal{G}} ((a_t \wedge a_G^l \wedge G_t) \Rightarrow l_{t+1}) \wedge \bigwedge_{l \in \mathcal{F}} (l_{t+1} \wedge a_t \Rightarrow (\bigvee_{G \in \mathcal{G}} (a_G^l \wedge G_t))) \quad (1)$$

where a_t asserts that action a occurred at time t , and we use the convention that φ_t is the result of adding subscript t to every fluent symbol of φ . The first part of (1) says that if a executes at time t , and it causes l when G holds, and G holds at time t , then l holds at time $t + 1$. The second part says that if l holds after a 's execution, then it must be that a_G^l holds, with G corresponding to the current state. These two parts correspond to *effect axioms* and *explanation closure axioms* used in Situation Calculus.

Now, we are ready to describe our zeroth-level algorithm ($SLAF_0$) for SLAF of a transition belief formula. Denote by $Cn^V(\varphi)$ the set of consequences of φ that are in vocabulary V , and let $L_{t+1} = \mathcal{P}_{t+1} \cup L$ the vocabulary that includes only fluents of time $t + 1$ and effect propositions from L . At time t we apply progression for the given action a and current transition belief formula, φ_t , and then apply filtering with the current observations:

1. $SLAF_0[a](\varphi_t) = Cn^{L_{t+1}}(\varphi_t \wedge a_t \wedge T_{\text{eff}}(a, t))$
2. $SLAF_0[o](\varphi_t) = \varphi_t \wedge o_t$

We can implement $Cn^V(\varphi)$ using consequence finding algorithms, such as resolution and some of its variants (e.g., [Simon and del Val, 2001]). The following theorem shows that this formula-SLAF algorithm is correct and exact.

Theorem 3.1 For φ transition belief formula, a action,

$$SLAF[a](\{\langle s, R \rangle \in \mathcal{S} \mid \langle s, R \rangle \text{ satisfies } \varphi\}) = \{\langle s, R \rangle \in \mathcal{S} \mid \langle s, R \rangle \text{ satisfies } SLAF_0[a](\varphi)\}$$

Our zeroth-level algorithm may enable more compact representation, but it does not guarantee it, nor does it guarantee tractable computation. In fact, no algorithm can maintain compact representation or tractable computation in general. SLAF is NP-hard because Logical Filtering is NP-hard [Eiter and Gottlob, 1992] even for deterministic actions. Also, any representation of transition belief states that uses $\text{poly}(|\mathcal{P}|)$ propositions grows exponentially (in the number of time steps and $|\mathcal{P}|$) for some starting transition belief states and action sequences.

4 Efficient Model Update

Learning world models is easier when SLAF distributes over logical connectives. The computation becomes tractable, with the bottleneck being the time to compute SLAF for each part separately. In this section we examine when such distribution is possible, and present a linear-time algorithm that gives an exact solution in those cases (and a weaker transition belief formula otherwise).

Distribution properties that always hold for SLAF follow from set theoretical considerations and Theorem 3.1:

Corollary 4.1 For φ, ψ transition belief formulae, a action,

$$SLAF[a](\varphi \vee \psi) \equiv SLAF[a](\varphi) \vee SLAF[a](\psi) \\ \models SLAF[a](\varphi \wedge \psi) \Rightarrow SLAF[a](\varphi) \wedge SLAF[a](\psi)$$

Stronger distribution properties hold for SLAF whenever they hold for Logical Filtering.

Theorem 4.2 Let ρ_1, ρ_2 be transition belief states. $SLAF[a](\rho_1 \cap \rho_2) = SLAF[a](\rho_1 \cap \rho_2)$ iff for every R $Filter[a](\rho_1^R \cap \rho_2^R) = Filter[a](\rho_1^R) \cap Filter[a](\rho_2^R)$.

We conclude the following corollary from Theorems 3.1, 4.2 and theorems in [Amir and Russell, 2003].

Corollary 4.3 For φ, ψ transition belief formulae, a action, $SLAF[a](\varphi \wedge \psi) \equiv SLAF[a](\varphi) \wedge SLAF[a](\psi)$ if for every relation R in ρ_φ, ρ_ψ one of the following holds:

1. a in R maps states 1:1
2. a has no conditional effects (and we know if it fails), and $\varphi \wedge \psi$ includes all its prime implicates.
3. The state is known for R : for at most one s , $\langle s, R \rangle \in \rho_\varphi \cup \rho_\psi$.

Figure 2 presents Procedure Factored-SLAF, which computes SLAF exactly when the conditions of Corollary 4.3 hold. Consequently, Factored-SLAF returns an exact solution whenever our actions are known to be 1:1. If our actions have no conditional effects and their success/failure is observed, then a modified Factored-SLAF can solve this problem exactly too (see Section 5).

If we pre-compute (and cache) the $2n$ possible responses of Literal-SLAF, then every time step t in this procedure requires linear time in the representation size of φ_t , our transition belief formula. This is a significant improvement over the (super exponential) time taken by a straightforward algorithm, and over the (potentially exponential) time taken by general-purpose consequence finding used in our zeroth-level SLAF procedure above.

Theorem 4.4 Step-SLAF(a, o, φ) returns a formula φ' such that $SLAF[a, o](\varphi) \models \varphi'$. If every run of Literal-SLAF takes time c , then Step-SLAF takes time $O(|\varphi|c)$. Finally, if we assume one of the assumptions of Corollary 4.3, then $\varphi' \equiv SLAF[a, o](\varphi)$.

We can give a closed-form solution for the SLAF of a belief-state formula (a transition belief formulae that has no effect propositions a_G^F). This makes procedure Literal-SLAF tractable, and also allows us to examine the structure of belief state formulae in more detail.

PROCEDURE Factored-SLAF ($\langle a_i, o_i \rangle_{0 \leq i \leq t}, \varphi$) $\forall i, a_i$ action, o_i observation, φ transition belief formula. 1. For i from 1 to t do, (a) Set $\varphi \leftarrow \text{Step-SLAF}(o_i, a_i, \varphi)$. (b) Eliminate subsumed clauses in φ . 2. Return φ .
PROCEDURE Step-SLAF (o, a, φ) o an observation formula in $\mathcal{L}(\mathcal{P})$, a an action, φ a transition belief formula. 1. If φ is a literal, then return $o \wedge \text{Literal-SLAF}(a, \varphi)$. 2. If $\varphi = \varphi_1 \wedge \varphi_2$, return $\text{Step-SLAF}(o, a, \varphi_1) \wedge \text{Step-SLAF}(o, a, \varphi_2)$. 3. If $\varphi = \varphi_1 \vee \varphi_2$, return $\text{Step-SLAF}(o, a, \varphi_1) \vee \text{Step-SLAF}(o, a, \varphi_2)$.
PROCEDURE Literal-SLAF (a, φ) a an action, φ a proposition in L_t or its negation. 1. Return $Cn^{L_{t+1}}(\varphi_t \wedge a_t \wedge T_{\text{eff}}(a, t))$.

Figure 2: SLAF using distribution over $\wedge, \vee$

Theorem 4.5 For belief-state formula $\varphi \in \mathcal{L}(\mathcal{P})$, time t , action a , $C_a = \bigwedge_{G \in \mathcal{G}, l \in \mathcal{F}} (a_G^l \vee a_G^{-l})$, and $G_1, \dots, G_m \in \mathcal{G}$ all the terms in $\mathcal{G}$ such that $G_i \models \varphi$,

$$\text{SLAF}[a](\varphi_t) \equiv \bigwedge_{l_1, \dots, l_m \in \mathcal{F}} \bigvee_{i=1}^m (l_i^{t+1} \vee a_{G_i}^{-l_i}) \wedge C_a$$

PROOF SKETCH We follow the characterization offered by Theorem 3.1 and Formula (1). We take $\varphi_t \wedge a_t \wedge T_{\text{eff}}(a, t)$ and resolve out the literals of time t . This resolution is guaranteed to generate a set of consequences that is equivalent to $Cn^{L_{t+1}}(\varphi_t \wedge a_t \wedge T_{\text{eff}}(a, t))$.

Assuming $\varphi_t \wedge a_t$, $T_{\text{eff}}(a, t)$ is logically equivalent to $T_{\text{eff}}(a, t)|_{\varphi} = \bigwedge_{l \in \mathcal{F}, G \in \mathcal{G}, G \models \varphi} ((a_G^l \wedge G_t) \Rightarrow l_{t+1}) \wedge \bigwedge_{l \in \mathcal{F}, G \in \mathcal{G}, G \models \varphi} (l_{t+1} \Rightarrow (G_t \Rightarrow a_G^l))$. This follows from two observations. First, notice that φ_t implies that for any $G \in \mathcal{G}$ such that $G \not\models \varphi$ we get $G_t \Rightarrow a_G^l$ and $(a_G^l \wedge G_t) \Rightarrow l_{t+1}$ (the antecedent does not hold, so the formula is true). Second, notice that, in the second part of the original $T_{\text{eff}}(a, t)$, $(\bigvee_{G \in \mathcal{G}, G \models \varphi} (a_G^l \wedge G_t))$ is equivalent (assuming φ) to $(\bigwedge_{G \in \mathcal{G}, G \models \varphi} (G_t \Rightarrow a_G^l))$.

Now, resolving out the literals of time t from $\varphi_t \wedge a_t \wedge T_{\text{eff}}(a, t)|_{\varphi}$ should consider all the resolutions of clauses (G_t is a term) of the form $a_G^l \wedge G_t \Rightarrow l_{t+1}$ and all the clauses of the form $l_{t+1} \Rightarrow (G_t \Rightarrow a_G^l)$ with each other. This yields the equivalent to

$$\bigwedge_{\substack{G_1, \dots, G_m \in \mathcal{G} \\ \varphi \models \bigvee_{i=1}^m G_i \\ l_1, \dots, l_m \in \mathcal{F}}} \left[\bigvee_{i=1}^m l_i^{t+1} \vee \bigvee_{i=1}^m (a_{G_i}^{-l_i} \wedge \neg a_{G_i}^{l_i}) \right]$$

because to eliminate all the literals of time t we have to resolve together sets of clauses with matching G_i 's such that $\varphi \models \bigvee_{i=1}^m G_i$. The formula above encodes all the

resulting clauses for a chosen set of $G_1, \dots, G_m$ and a chosen set of literals $l_1, \dots, l_m$. The reason for including $(a_{G_i}^{l_i} \wedge \neg a_{G_i}^{l_i})$ is that we can always choose a clause with G_i, l_i of a specific type (either one that includes $\neg a_{G_i}^l$ or one that produces $a_{G_i}^l$).

Finally, we get the formula in our theorem because $a_G^F \equiv \neg a_G^{-F}$ (G characterizes exactly one state in S), and the fact that there is one set of $G_1, \dots, G_m$ that is stronger than all the rest (it entails all the rest) because $G_1, \dots, G_m$ are complete terms. This set is the complete fluent assignments G_i that satisfy φ . ■

A consequence of Theorem 4.5 is that we can implement Procedure Literal-SLAF using the equivalence

$$\text{SLAF}[a](l) \equiv \begin{cases} \text{Theorem 4.5} & L(l) \subseteq \mathcal{P} \\ l \wedge \text{SLAF}[a](\text{TRUE}) & \text{otherwise} \end{cases}$$

Notice that the computation in Theorem 4.5 for $\varphi = l$ a literal is simple because $G_1, \dots, G_m$ are all the complete terms in $\mathcal{L}(\mathcal{P})$ that include l .

5 Compact Model Representation

In Theorem 4.5, m could be as high as $2^{|\mathcal{P}|}$, the number of complete terms in $\mathcal{G}$. Consequently, clauses may have exponential length (in $n = |\mathcal{P}|$) and there may be a super-exponential number of clauses in this result.

In this section we restrict our attention to STRIPS actions [Fikes *et al.*, 1972], and provide an overall polynomial bound on the growth of our representation, its size after many steps, and the time taken to compute the resulting model. STRIPS is a class of actions that has been at the focus of interest for the area of AI planning and execution for many years. It includes deterministic, unconditional (but sometimes not executable) actions.

5.1 Actions of Limited Effect

In many domains we can assume that every action a affects at most k fluents, for some small $k > 0$. It is also common to assume that our actions are STRIPS, and that they may fail without us knowing, leaving the world unchanged. Those assumptions together allow us to progress SLAF with a limited (polynomial factor) growth in the formula size.

We use a language that is similar to the one in Section 3, but which uses only action propositions a_G^l with G being a fluent term of size k (instead of a fluent term of size n in $\mathcal{G}$). Semantically, $a_{l_1 \wedge \dots \wedge l_k}^l \equiv \bigwedge_{l_{k+1}, \dots, l_n} a_{l_1 \wedge \dots \wedge l_n}^l$.

Theorem 5.1 Let $\varphi \in \mathcal{L}(\mathcal{P})$ be a belief-state formula, t time, and a a STRIPS action with $\leq k$ fluents affected or in the precondition term. Let $\mathcal{G}^k$ be the set of all terms of k fluents in $\mathcal{L}(\mathcal{P})$ that are consistent with φ . Then,

$$\text{SLAF}[a](\varphi_t) \equiv \bigwedge_{\substack{G_1, \dots, G_k \in \mathcal{G}^k \\ G_1 \wedge \dots \wedge G_k \models \varphi \\ l_1, \dots, l_k \in \mathcal{F}}} \bigvee_{i=1}^k (l_i^{t+1} \vee a_{G_i}^{-l_i}) \wedge C_a$$

The proof (omitted) uses two insights. First, if a has only one case in which change occurs, then every clause in Theorem 4.5 is subsumed by a clause that is entailed by $SLAF[a](\varphi_t)$, has at most one $a_{G_i}^{l_i}$ per literal l_i (i.e., $l_i \neq l_j$ for $i \neq j$) and G_i is a fluent term (has no disjunctions). Second, every a_G^l with G term is equivalent to a formula on $a_{G_i}^{l_i}$ with G_i terms of length k , if a affects only k fluents.

Thus, we can encode all of the clauses in the conjunction using a subset of the (extended) action effect propositions, a_G^l , with G being a term of size k . There are $O(n^k)$ such terms, and $O(n^{k+1})$ such propositions. Every clause is of length $2k$, with the identity of the clause determined by the first half (the set of action effect propositions). Consequently, $SLAF[a](\varphi_t)$ takes $O(n^{k^2+k} \cdot k^2)$ space to represent using $O(n^{k^2+k})$ clauses of length $\leq 2k$.

5.2 Always-Executable Actions

Many times we can assume that our actions are always executable. For example, our sequence of actions does not include action failures when this sequence is chosen by a knowledgeable agent (e.g., a human) that has better observations than we do.

We use a language that is similar to the one in Section 3, but which uses only action propositions a^l (l always holds after executing a) and $a^{l\circ}$ (l is not affected by a). Semantically, $a^l \equiv \bigwedge_{l_1, \dots, l_n} a_{l_1 \wedge \dots \wedge l_n}^{l_i}$, and $a^{l\circ} \equiv \bigwedge_{l_1, \dots, l_{i-1}, l_{i+1}, \dots, l_n} a_{l_1 \wedge \dots \wedge l_n}^{l_i}$.

We assume that transition-belief formula φ is *fluent-factored*, i.e., it is the conjunction of φ_f such that each φ_f concerns only one fluent, f , and actions' effects on it. Also, for every f , $\varphi_f \equiv (\neg f \vee \text{expl}_f) \wedge (f \vee \text{expl}_{\neg f}) \wedge A_f$, with $\text{expl}_f, \text{expl}_{\neg f}, A_f$ formulae over action propositions a^f, a^{-f} , and $a^{f\circ}$ (possibly multiple different actions). We call this format *fluent-factored*. The intuition here is that expl_f and $\text{expl}_{\neg f}$ are all the possible explanations for f being true and false, respectively. Also, A_f holds knowledge about actions' effects on f , knowledge that does not depend on f 's current value. For example, if we know nothing about actions that affect f (e.g., when we start our exploration), then $\varphi_f = (\neg f \vee \text{TRUE}) \wedge (f \vee \text{TRUE}) \wedge \text{TRUE}$.

With this representation, $SLAF[a](\varphi_f)$ is

$$(\neg f \vee (a^f \vee (a^{f\circ} \wedge \text{expl}_f))) \wedge (f \vee (a^{-f} \vee (a^{f\circ} \wedge \text{expl}_{\neg f}))) \wedge A_f$$

A similar formula holds for observations.

This contributes to a much simpler algorithm, AE-STRIPS-SLAF, that we present in Figure 3. It computes SLAF of a fluent-factored transition belief state in this form with a sequence of actions and observations.

Theorem 5.2 *Let φ be a fluent-factored transition belief formula, and assume $a_1, \dots, a_T$ are always-executable STRIPS actions, and $o_1, \dots, o_T$ are observation terms.*

PROCEDURE AE-STRIPS-SLAF($\langle a_i, o_i \rangle_{0 \leq i \leq T}, \varphi$)
 a_i actions, o_i observations, $\varphi = \bigwedge_{f \in \mathcal{P}} \varphi_f$ fluent-factored
 with $\varphi_f = (\neg f \vee \text{expl}_f) \wedge (f \vee \text{expl}_{\neg f}) \wedge A_f$.

1. For every $f \in \mathcal{P}$ do: For i from 1 to T do,
 - (a) Set $\varphi_f \leftarrow (\neg f \vee (a^f \vee (a^{f\circ} \wedge \text{expl}_f))) \wedge (f \vee (a^{-f} \vee (a^{f\circ} \wedge \text{expl}_{\neg f}))) \wedge A_f$.
 - (b) If $o_i \models f$ (we observed f), then set $\varphi_f \leftarrow (\neg f \vee \text{TRUE}) \wedge (f \vee \text{FALSE}) \wedge A_f \wedge \text{expl}_f$.
 - (c) If $o_i \models \neg f$ (we observed $\neg f$), then set $\varphi_f \leftarrow (\neg f \vee \text{FALSE}) \wedge (f \vee \text{TRUE}) \wedge A_f \wedge \text{expl}_{\neg f}$.
2. Eliminate subsumed clauses in $\varphi = \bigwedge_{f \in \mathcal{P}} \varphi_f$.
3. Return $\varphi_T = \varphi$.

Figure 3: SLAF with always-executable STRIPS.

Then, Procedure AE-STRIPS-SLAF($\langle a_i, o_i \rangle_{0 \leq i \leq T}, \varphi$) returns $\varphi^T \equiv SLAF[\langle a_i, o_i \rangle_{0 \leq i \leq T}](\varphi_0)$ in time $O(T \cdot |\mathcal{P}|)$, and $|\varphi^T| \leq O(T \cdot |\mathcal{P}|)$.

The reason for the space bound on the end formula $|\varphi^T|$ is that at every time step we increase the size of the formula by adding to it at most 4 new elements per fluent.

We can further improve the implementation to take time $O(T \cdot |o|)$, for $|o|$ the largest number of fluents observed at once. Also, the bound on the space taken by φ_T can be improved in two ways. First, $O(|\mathcal{P}| \cdot 2^{|\mathcal{A}| \log |\mathcal{A}|})$, for $\mathcal{A}$ our set of actions, is a bound on the CNF representation of φ^T because every part φ_f^T of it is similar to $(f \Rightarrow (a_1^f \vee (a_1^{f\circ} \wedge (a_2^f \vee (a_2^{f\circ} \wedge \dots))))$. Thus, if $|\mathcal{A}|$ is small, then this bound is manageable. Second, if every fluent f is observed at least once every k actions (or an action which affects f occurs, and we know its effect), then the CNF representation of φ_T takes space $O(|\mathcal{P}| \cdot |\mathcal{A}|^k)$. Again, if k is small, this bound is useful.

5.3 Action Preconditions

Unfortunately, a simple procedure such as AE-STRIPS-SLAF (Figure 3) does not provide an exact solution in general. The reason is that action failure makes fluents co-dependent (each of them could have caused this failure). For instance, consider Figure 1, and assume that we add the fluent OK which indicates that the last action succeeds. If we know only $\neg on$ (the light is off), then $SLAF[sw-on](\neg on) \wedge \neg OK$ (i.e., $sw-on$ failed, e.g., because we are in the West room) implies $sw \vee E \vee a_{\neg E \wedge \neg sw \wedge \neg lit}^{\neg OK} \vee a_{\neg E \wedge sw \wedge \neg lit}^{sw} \vee a_{E \wedge \neg sw \wedge \neg lit}^{\neg E} \vee a_{E \wedge sw \wedge \neg lit}^{OK}$, by Theorem 4.5.

Instead, we compute the effect for every literal using Theorem 5.1, (assuming exactly k fluents are changed by action a) and then approximate by dropping all clauses with more than one fluent (other than OK). Thus, we can augment step 1(a) in AE-STRIPS-SLAF with a condition that a succeeded ($o_i \models OK$), and add that if a fails, then we use the following formula instead: Set $\varphi_f \leftarrow (\neg f \vee (\text{expl}_f \wedge \text{expl}_{\neg OK, f})) \wedge (f \vee (\text{expl}_{\neg f} \wedge \text{expl}_{\neg OK, \neg f})) \wedge A_f$,

$$\text{for } \text{expl}_{-OK,l} = \bigwedge_{\substack{G_1, G_2 \in \mathcal{G}^k \\ G_1 \wedge G_2 \models l}} (a_{G_1}^l \vee a_{G_2}^{-OK}).$$

The size of the resulting formula is larger by an additive $O(|\text{expl}_{-OK,l}|) = O(n^{2k})$ per fluent. Thus, the result after T steps is a formula φ_T with $|\varphi_T| \leq O(T \cdot |\mathcal{P}|^{2k})$, which is polynomial and feasible for small k 's.

5.4 Using the Model and Adding Bias

Our algorithms compute a solution to SLAF as a logical formula. We can use a SAT solver to answer queries over this formula, such as checking if it entails a^f , for action a and fluent f . The number of variables in the result formula is *always independent of T* , and is linear in $|\mathcal{P}|$ for some of our algorithms, so we can use current SAT solvers to treat domains of 1000 features and more.

Many times it is also desirable to prefer some models over others. We can represent such bias using a preference model (e.g., [McCarthy, 1986]) or a probabilistic prior over transition relations. We add this bias on top of our SLAF result, φ_T , and get an exact solution if we can run inference on the combined model. Preferential bias is well studied and fits easily with logical formula φ_T (e.g., we can use implementations of Circumscription for inference with such bias).

Also, algorithms for inference with probabilistic bias and logical sentences are now emerging and can be used here too [Hajishirzi and Amir, 2005]. We can use these algorithms to apply probabilistic bias to the resulting logical formula. For example, given a probabilistic graphical model (e.g., Bayesian Networks) and a set of propositional logical sentences, we can consider the logical sentences as observations. With this approach, a logical sentence φ gives rise to a characteristic function $\delta_\varphi(\vec{x})$ which is 1 when $\vec{x}$ satisfies φ and 0 otherwise. For a conjunction of clauses we get a set of such functions (one per clause). Thus, inference in the combined probabilistic-logical system is a probabilistic inference (such as variable elimination (e.g., [Dechter, 1999])) in which there are additional potential functions.

6 Experimental Evaluation

We tested our AE-STRIPS-SLAF algorithm in partially observable blocks-world domains. We generated random action-observation sequences of 4000 steps from a description of the domain in PDDL. Our SLAF algorithm receives no domain information besides the sequence of actions and observations that is given by the generator.

We ran the algorithm with blocks-world domains of increasing sizes (31 to 157 features). We collected the time and space taken for each time step in the execution, starting with zero knowledge. The results are shown in Figure 4. Each of the graphs belongs to a different domain size. They show that the time per step approaches a (very small, order of < 9 milliseconds) limit as the execution proceeds. The time that is taken to perform SLAF of different domains grows linearly with domain size.

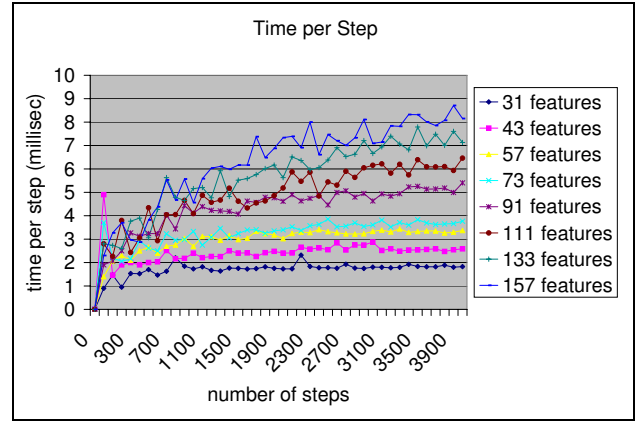


Figure 4: AE-STRIPS-SLAF in the blocks world.

Also, [Hlubocky and Amir, 2005] has included a modified version of this algorithm in their architecture and tested it on a suite of adventure-game-like virtual environments that are generated at random. These include arbitrary numbers of places, objects of various kinds, and configurations and settings of those. There, an agent's task is to exit a house, starting with no knowledge about the state space, available actions and their effects, or characteristics of objects. Their experiments show that the agent learns the effects of its actions very efficiently. This agent makes decisions using the learned knowledge, and inference with the resulting representation is fast (a fraction of a second per SAT problem in domains including more than 30 object, modes, and locations).

7 Comparison with Related Work

HMMs [Boyen and Koller, 1999; Boyen *et al.*, 1999; Murphy, 2002; Ghahramani, 2001] can be used to estimate a stochastic transition model from observations. Unfortunately, HMMs require an explicit representation of the state space, and our smallest domain (31 features) requires a transition matrix of $(2^{31})^2$ entries. This prevents initializing HMMs procedures on any current computer.

Structure learning approaches in Dynamic Bayes Nets (DBNs and also fHMMs) use EM and additional approximations (e.g., using factoring, variation, or sampling), and are more tractable. However, they are still limited to small domains (e.g., 10 features [Ghahramani and Jordan, 1997; Boyen *et al.*, 1999]), and also have unbounded errors in discrete deterministic domains, so are not usable in our settings.

Reinforcement Learning (RL) approaches compute a mapping between world states and preferred actions. They are highly intractable in partially observable domains [Kaelbling *et al.*, 1998], and approximation (e.g., [Kearns *et al.*, 2000]) is practical only for small domains (e.g., 10-20 features) with small horizon time T .

In contrast to HMMs, DBNs, and RL, our algorithms are exact and tractable in large domains (> 300 features). We take advantages of properties common to discrete domains, such as determinism, limited effects of actions, and observed failure.

Previous work on learning action models assumes *fully observable* deterministic domains. These learn parametrized STRIPS actions using, e.g., version spaces [Gil, 1994; Wang, 1995], general classifiers [Oates and Cohen, 1996], or hill-climbing ILP [Benson, 1995]. Recently, [Pasula *et al.*, 2004] gave an algorithm that learns stochastic actions with no conditional effects. [Schmill *et al.*, 2000] approximates partial observability by assuming that the world is fully observable. We can apply these to partially observable learning problems (sometimes) by using the space of belief states instead of world states, but this increases the problem size to exponentially, so this is not practical for our problem.

8 Conclusions

We presented a framework for learning the effects and preconditions of deterministic actions, and showed that in several common situations this can be done exactly in time that is polynomial (sometime linear) in the number of time steps and features. We can add bias and compute an exact solution for large domains (hundreds of features), in many cases.

The results that we presented are promising for many applications, including reinforcement learning, agents in virtual domains, and HMMs. Already, this work is being applied to autonomous agents in adventure games, and exploration is guided by the transition belief state that we compute and information gain criteria. In the future we plan to extend these results to stochastic domains, and domains with continuous features.

9 Acknowledgements

I wish to thank Megan Nance for providing the code and samples for the generator. I also wish to acknowledge support from DAF Air Force Research Laboratory Award FA8750-04-2-0222 (DARPA REAL program).

References

- [Amir and Russell, 2003] Eyal Amir and Stuart Russell. Logical filtering. In *IJCAI '03*, pages 75–82. MK, 2003.
- [Benson, 1995] Scott Benson. Inductive learning of reactive action models. In *Proc. ICML-94*, 1995.
- [Boyen and Koller, 1999] Xavier Boyen and Daphne Koller. Approximate learning of dynamic models. In Michael S. Kearns, Sara A. Solla, and David A. Kohn, editors, *NIPS 1998*, pages 396–402. Cambridge: MIT Press, 1999. Available at <http://www.cs.stanford.edu/xb/nips98/>.
- [Boyen *et al.*, 1999] Xavier Boyen, Nir Friedman, and Daphne Koller. Discovering the hidden structure of complex dynamic systems. In *Proc. UAI '99*, pages 91–100. MK, 1999.
- [Dechter, 1999] Rina Dechter. Bucket elimination: A unifying framework for reasoning. *Artificial Intelligence*, 113(1–2):41–85, 1999.
- [Eiter and Gottlob, 1992] T. Eiter and G. Gottlob. On the complexity of propositional knowledge base revision, updates, and counterfactuals. *AIJ*, 57(2-3):227–270, 1992.
- [Fikes *et al.*, 1972] R. Fikes, P. Hart, and N. Nilsson. Learning and executing generalized robot plans. *AIJ*, 3, 1972.
- [Ghahramani and Jordan, 1997] Z. Ghahramani and M. I. Jordan. Factorial Hidden Markov Models. *Machine Learning*, 29:245–275, 1997.
- [Ghahramani, 2001] Zoubin Ghahramani. An introduction to Hidden Markov Models and Bayesian networks. *Intl J. of Pattern Recog. and AI*, 15(1):9–42, 2001.
- [Gil, 1994] Yolanda Gil. Learning by experimentation: Incremental refinement of incomplete planning domains. In *Proc. ICML-94*, pages 10–13, 1994.
- [Hajishirzi and Amir, 2005] Hannaneh Hajishirzi and Eyal Amir. Tractable exact inference in some dynamic Bayesian networks. In *submitted for publication*, 2005.
- [Hlubocky and Amir, 2005] Brian Hlubocky and Eyal Amir. Logic-based approach to decision making in expanding worlds. In *submitted for publication*, 2005.
- [Kaelbling *et al.*, 1998] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra. Planning and acting in partially observable stochastic domains. *AIJ*, 101:99–134, 1998.
- [Kearns *et al.*, 2000] M. Kearns, Y. Mansour, and A. Y. Ng. Approximate planning in large pomdps via reusable trajectories. In *Proc. NIPS'99*, pages 1001–1007, 2000.
- [Littman, 1996] M. L. Littman. *Algorithms for sequential decision making*. PhD thesis, Brown U., 1996.
- [McCarthy, 1986] John McCarthy. Applications of Circumscription to Formalizing Common Sense Knowledge. *Artificial Intelligence*, 28:89–116, 1986.
- [Murphy, 2002] K. Murphy. *Dynamic Bayesian Networks: Representation, Inference and Learning*. PhD thesis, UC Berkeley, 2002.
- [Oates and Cohen, 1996] T. Oates and P. R. Cohen. Searching for planning operators with context-dependent and probabilistic effects. In *Proc. AAAI '96*, 1996.
- [Pasula *et al.*, 2004] H. M. Pasula, L. S. Zettlemoyer, and L. P. Kaelbling. Learning probabilistic relational planning rules. In *Proc. ICAPS'04*, 2004.
- [Schmill *et al.*, 2000] M. D. Schmill, T. Oates, and P. R. Cohen. Learning planning operators in real-world, partially observable environments. In *Proc. AIPS'00*, 2000.
- [Simon and del Val, 2001] Laurent Simon and Alvaro del Val. Efficient consequence-finding. In *IJCAI '01*, 2001.
- [Wang, 1995] Xuemei Wang. Learning by observation and practice: an incremental approach for planning operator acquisition. In *Proc. ICML-95*, pages 549–557. MK, 1995.

Learning Forward Models for Robots

Anthony Dearden, Yiannis Demiris

Department of Electrical and Electronic Engineering

Imperial College London

Exhibition Road, London, SW7 2BT

E-mail: {anthony.dearden, y.demiris}@imperial.ac.uk

Abstract

Forward models enable a robot to predict the effects of its actions on its own motor system and its environment. This is a vital aspect of intelligent behaviour, as the robot can use predictions to decide the best set of actions to achieve a goal. The ability to *learn* forward models enables robots to be more adaptable and autonomous; this paper describes a system whereby they can be learnt and represented as a Bayesian network. The robot's motor system is controlled and explored using 'motor babbling'. Feedback about its motor system comes from computer vision techniques requiring no prior information to perform tracking. The learnt forward model can be used by the robot to imitate human movement.

1 Introduction

The purpose of any robot is to interact with its environment. There is, however a separation between the effects a robot can produce in the environment and its mechanisms for producing them, its motor commands. To overcome this, inverse and forward models can be used [Jordan and Rumelhart, 1992]. In robotics, forward models enable a robot to predict the consequences of its actions: given the current state of the robot they predict how a set of motor commands will influence the state of the robot or its environment, or how the observations will be perceived by the robot's sensors, as shown in figure 1. Conversely, inverse models are used to output the necessary motor commands to achieve or maintain a desired goal. Both these *internal models* have been hypothesised to be used in human motor control [Wolpert *et al.*, 1998]. The ability to predict the consequence of actions has several uses in robotics, such as allowing mental rehearsal of possible actions or imitating the actions of humans or other robots [Demiris, 2002].

Practically any environment a robot works in will change, or have properties which cannot be modelled beforehand. Even if the environment is assumed to be completely predictable, endowing the robot with this knowledge may be beyond the abilities or desires of its programmer. A truly autonomous robot, therefore, needs to be able to learn and adapt

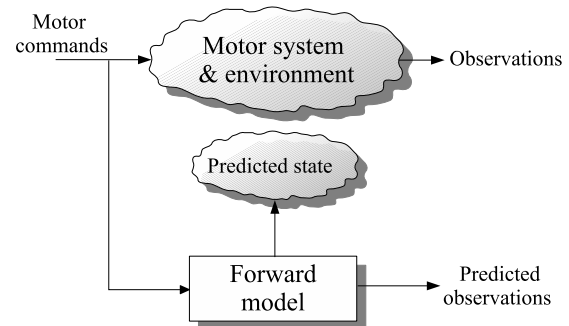


Figure 1: A forward model

its own forward models. Forward models of dynamic and uncertain environments, and how they can be learnt, is the topic which is investigated in this paper. Although forward models are used in a different context, there are similarities here to work on calibrating action-sensor models in robotics e.g. [Roy and Thrun, 1999]. A robot learning about itself and its environment is in the position of being an active part of the system it is trying to learn about; this situation draws interesting parallels with learning in human infants, with mechanisms such as babbling (issuing random motor commands) used by infants to learn how to control their own motor system [Meltzoff and Moore, 1997].

A system is presented here that enables a robot to autonomously learn a forward model with no prior knowledge of its motor system or the external environment. Information about the effects of the robot's actions are captured via a vision system that clusters low-level image features to automatically find and track moving objects in a scene. The robot sends out random motor commands to its motor system and receives information back from the vision system. This set of evidence is used to learn the structure and parameters of a Bayesian network, which represents the forward model. This model can then be used to enable the robot to predict the effects of its actions. By observing the gestures of a human with its own vision system, the robot can imitate the human gesture using the forward model it has learnt of its own motor system.

2 Representing forward models with Bayesian networks

Bayesian networks [Pearl, 1988] are an ideal method for representing forward models that can be learnt. They offer a way of representing the causal nature of a robot's control system within a rigorous probabilistic framework. The system here is aimed at learning and representing the causal associations between the robot's motor commands, the robot's state, and the observations of the robot's state that are received from the vision system. The motor commands and state of the robot are represented as random variables in the Bayesian network, and the causal relationships between them are shown with arcs. The Bayesian network represents a learnt probability distribution across the previous N motor commands, $M_{1:N}[t-d]$, the P current states $S_{1:P}[t]$, and the P observations of the state, $O_{1:P}[t]$. The variable d represents the delay between a motor command being issued and robot's state changing; in real robotic systems it cannot be assumed that the effect of a motor command will occur after just one time-step, so this is a parameter that must be modelled and learnt. Figure 2 shows this structure.

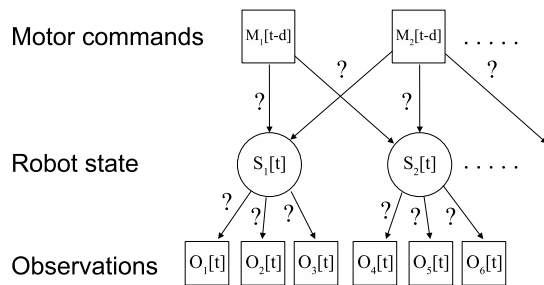


Figure 2: The 'template' Bayesian network of a forward model. The question marks represent the part of the structure which needs to be learnt: the causal association between the motor commands and the state of the robot, and which observations are associated with each object in the scene. Rectangular nodes correspond to observed nodes.

The N motor random variables represent specific low-level motor commands of the robot. They could be discrete, e.g. 'open' or 'close' robot's gripper, or continuous, e.g. to set the height of the gripper. When the robot is issuing these commands there is no uncertainty as to their value, so they can be modelled as observed nodes. Alternatively, when the robot is observing the actions of another robot, or a human, whilst using its own forward model, they will be hidden nodes because the Bayesian network will receive no direct evidence of their value.

The P observations of the state-space $O[t]$ are represented with square nodes descended from $S[t]$ on figure 2. The model treats $O[t]$ as noisy observations of the actual state space so that noise from the vision system and other uncertainty can be modelled. Evidence about the actual state of the robot therefore enters the system here. How the state space, $S[t]$, of the robot will be represented depends on the information received from the computer vision system. The state

variables are all hidden nodes, because the robot's knowledge of itself only comes from noisy observations of the state, $O[t]$.

3 How a robot can learn and use forward models

The structure for a forward model described in section 2 require several aspects to be learnt. Firstly, a representation of the state of the robot or the environment is needed. The richest source of this information comes from the robot's own vision system. However, this is also the most challenging source: information comes in the low-level format of pixels. Furthermore, if the robot is seeking to learn its forward models it has no prior information about what to track, because the environment is unknown. To extract information from a scene, a method is needed of automatically initialising and tracking the objects in it. The system presented here works by clustering tracked low-level features in the image according to their position and dynamics. This allows the position of objects in the scene to be tracked without any knowledge of the objects having to be supplied.

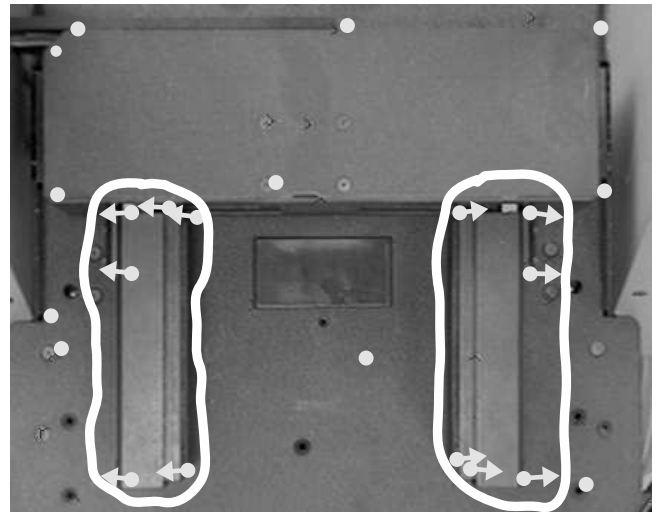


Figure 3: The vision system finds moving objects in the scene by clustering together optical flow points.

The low-level point features used are tracked using the Lucas-Kanade optical flow algorithm [Lucas and Kanade, 1981]. The tracked points are clustered using the k -means algorithm according to their position and velocity. One drawback of this method of clustering is that the number of clusters must be known beforehand. This is overcome by first excluding stationary points and extracting increasing numbers of clusters. The error of a particular number of clusters is then taken to be the average squared distance of the optical flow points from the centre of the cluster they are part of. The cluster number used is the one at which the curvature of this error with respect to the number of clusters is minimised [Hoppner *et al.*, 1999]. The positions, velocities and sizes of each of these clusters can now be used as the observations, $O[t]$, of objects for the forward model learning system. The

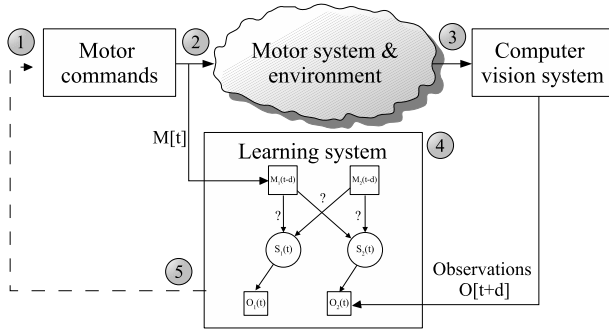


Figure 4: How the robot can learn about its environment. The stages 1-5 involved in the experiment are outlined in the text.

actual state of the motor system, $S[t]$, is taken to be a hidden discrete node for each object in the scene found by the vision system, and this will be connected to a continuous observation.

With all the possible nodes in the Bayesian network specified, the final task is to learn how the motor commands interact with the state, and train the parameters of the Bayesian network so that its predictions are as accurate as possible. To train a Bayesian network, a set of *evidence* is required: a data set containing actual observed values of the observed nodes in the network. This is a set of N motor commands executed at each time step, $M_{1:N}[t-d]$, together with the P observations of the state $O_{1:P}[t]$, the size, position or velocity of an object. The problem now is how to use this data to learn the structure and parameters of the Bayesian network. An inspiration for solving the problem of how to learn this Bayesian network can be taken from developmental psychology. Gopnik and Meltzoff compare the mechanism an infant uses to learn to that of a scientist: scientists form theories, make observations, perform experiments, make generalisations and revise existing theories [Gopnik and Meltzoff, 1998]. The notion of forming theories about the world is already dealt with here: a forward model can be seen as a theory of the robot's own world. The process a robot uses to obtain its data and infer a forward model from it is its *experiment*. An outline of what a robot's experiment involves is as follows, and is highlighted in figure 4:

1. A particular motor command, $M_1[t]$ is chosen;
2. A value for this motor command is sent to the robot's motor system;
3. The vision system returns a set of observations of the scene, $O_{1:P}[t+d]$, for example, moving forward $v \text{ ms}^{-1}$ in direction x ;
4. This training set of data, $M_1[t]$ and $O_{1:P}[t+d]$, is used to train the Bayesian network to develop a forward model;
5. The whole process is repeated, either with a different value for the current motor command, or a different motor command altogether.

The open issues here relate to the last two stages. Firstly, if the structure of the Bayesian network is not completely

known, how should the data be used to learn the structure and parameters of the network? Most methods for learning structure from data are based on performing a search through the space of possible structures, with the goal of finding the one that maximises the log-likelihood of the model given the data [Friedman, 1997; Neapolitan, 2004]; the higher the log-likelihood, the more accurate a particular network is at predicting the data. This is subject to a constraint on the complexity of the model, as the structure with the highest likelihood will be the one with connections between every node. In this paper the network structure is learnt as part of the experiment by performing a search through the set of possible structures and choosing the one which maximises the log-likelihood. This search is performed online by training and evaluating simultaneously a set of candidate models. The search space is the set of models with different delays in motor commands and different observation nodes for each possible object.

The second issue is how particular motor commands and their values should be chosen so as to learn the most accurate forward model. Unlike most machine learning situations, with a forward model the robot is an active part of the Bayesian network structure it is attempting to learn. This situation, referred to as *active learning*, has received relatively little attention in Bayesian networks literature. Active learning for Bayesian networks has been discussed for unknown parameters and unknown structure [Tong and Koller, 2000; 2001]. The method of choosing motor commands in the experiment in this paper was inspired by the idea of babbling in infant motor skill learning [Meltzoff and Moore, 1997]; the motor commands were chosen from a random walk through a Markov model.

4 Experiments to learn a forward model

An experiment was carried out to show how forward models can be autonomously learnt and used. The robot used was an ActivMedia PeopleBot, as shown in figure 5. The task was to learn the forward model for the movement of the robot's grippers; the robot needed to learn to predict the effects of the motor commands. No prior information about the appearance of the grippers or the nature of the motor commands was specified. The robot was to perform an experiment: it babbled with its motor commands, observed what happened, and then learnt the relationship between the motor command and the observation using a Bayesian network; this process was repeated until an accurate model had been learnt.

The motor command in this case was limited to just one degree of freedom: the robot's grippers, which could be either opened, closed or halted. The motor commands were decided at random using a random walk through the Markov model shown in figure 6. Motor commands were sent according to the current state of the Markov model, and the state would change each time-step according to the transition probabilities shown on the figure. The parameters of the Markov model were chosen so that the motor system stays in one babbling state just long enough to get enough information to train its forward model before moving to the next state.

When the motor commands were sent, the grippers moved; the vision system correctly calculated and tracked the posi-

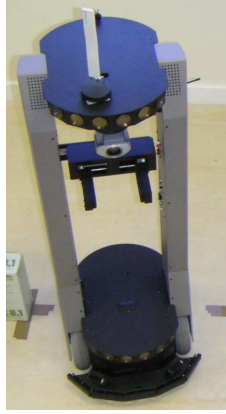


Figure 5: An ActivMedia PeopleBot.

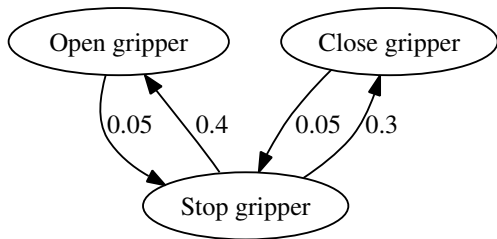


Figure 6: Markov model of gripper babbling. The values on the arcs represent the transition probabilities of going from one state to the next; self transitions are not shown.

tions of the moving grippers in the scene: figure 7 shows the two grippers being located and tracked. The tracking system worked well in this situation: black objects are automatically being tracked even against the black background of the robot's base.

The vision system provided the random variables for the states $S[t]$ and observations $O[t]$ in the Bayesian network for the forward model. In this situation the two objects, the grippers, were automatically added to the Bayesian network. Each gripper was represented as a discrete node for the state, and a set of continuous nodes for the observation, both of which were represented as a Gaussian distribution. The set of possible observations, $O[t]$, was the velocity, $VEL[t]$, the position, $POS[t]$, or size, $SIZE[t]$ of the tracked objects. The basic template for the structure of the forward models to be learnt is shown by the Bayesian network in figure 8.

Even though the forward model of the robot's gripper was relatively simple, this example highlighted how much the robot needed to learn: all the parameters and two important properties of the structure needed to be learnt. Firstly, it is not known how long the delay is between the motor command being issued and the gripper changing state. This is shown in figure 8 as the multiple possible motor commands connected to each state; only the four previous commands are shown to keep the diagram simple. Secondly, it was not known which observation from the vision system is the best

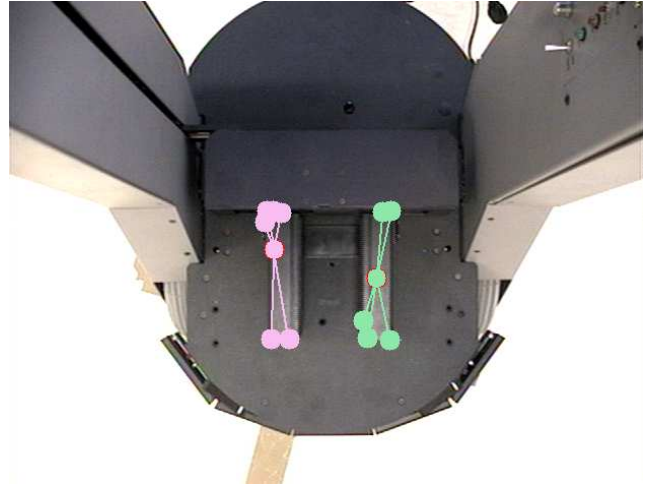


Figure 7: The tracking system working on the grippers. The system has correctly identified two moving objects, which are identified as grouped primitives shown with the same colour.

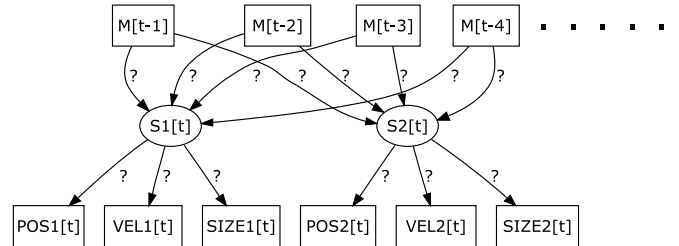


Figure 8: The template Bayesian network for the gripper forward model. The vision system has supplied the information that two objects can be interacted with: the two grippers. The robot still has to learn what the delay, d , is from action to effect and which of the observations the change of state of each object represents, a translation or a change in velocity, and all the parameters that specify the Bayesian network.

one to use as part of the forward model: the velocity, position or size. The task of the structure learning algorithm here is therefore to maximise the log-likelihood of the data given the model, which in this case is $\log(P(M, O | G))$, where M is the set of motor commands over the experiment, O the set of corresponding observations and G is the directed acyclic graph representing the structure of the model, together with the parameters of the model. This value is calculated as part of the parameter learning process, as a particular model is trained to the data.

For each model, the parameters of the Bayesian network were learnt using the expectation maximisation (EM) algorithm, with the inference stage being performed with the junction-tree algorithm [Pearl, 1988]. In this experiment, the search space is small enough for the robot to consider each possible candidate structure simultaneously. If the previous 20 motor commands are considered and 6 possible combinations of observation variables ($VEL[t]$, $POS[t]$ and

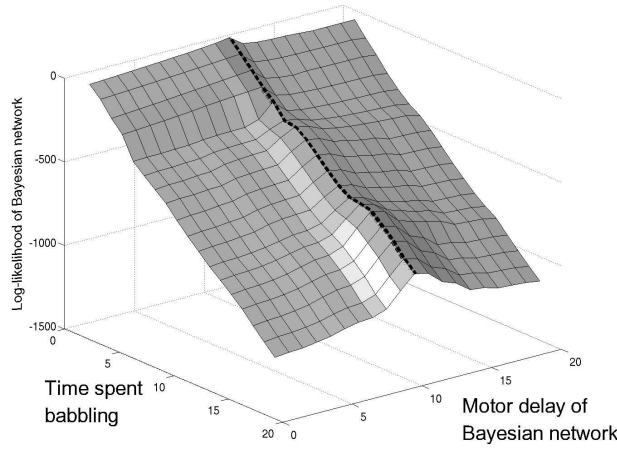


Figure 9: How the log-likelihoods of the Bayesian networks using $VEL[t]$ as the observation node evolve over the course of a typical experiment. The maximum log-likelihood is for the Bayesian network with a motor delay of 11 time steps, or 550 ms

SIZE[t], for each gripper), the search space contains 120 possible models. Figure 9 shows how the log-likelihoods for the Bayesian networks with varying motor delays and the velocity observation change over time. As the figure shows, the motor delay which maximise the log-likelihood, d , was 11 time-steps, which is about 550 ms. Following the search, the learnt forward model for the grippers is shown in figure 10. The motor command was learnt to be $M[t-11]$, and the best observation was learnt to be the velocity of the grippers. This was correct since the motor commands do control the grippers' velocities.

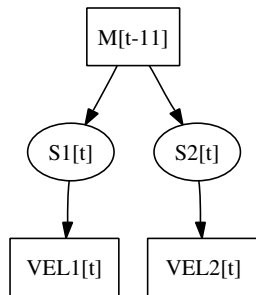


Figure 10: The learnt forward model of the grippers.

5 Using the Bayesian network as a forward model

Once the Bayesian network in figure 10 has been learnt, it can be used as a forward model to give a prediction of the consequences of the motor command. For the forward models of the grippers, the predicted output is the velocity of the grippers, calculated from the conditional probability distribution: $P(VEL[t]|M[t-4]=m)$. Because the observations here are modelled as Gaussians, the most likely values will be the

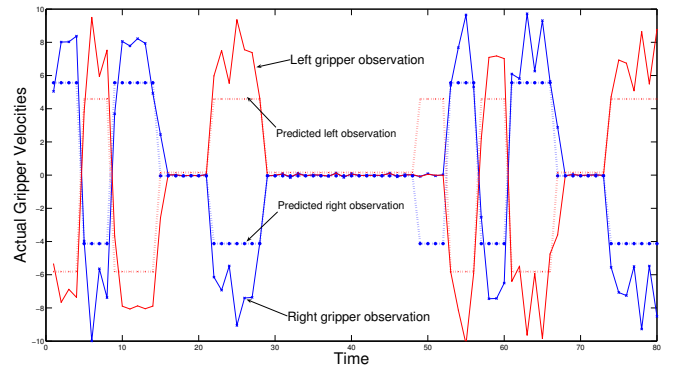


Figure 11: Evaluating the predictive abilities of a learnt forward model. The predicted gripper movements (dashed) are close to the actual ones (solid). The differences are principally because the variation in gripper velocities for each of the three motor commands are modelled as noise in the forward model.

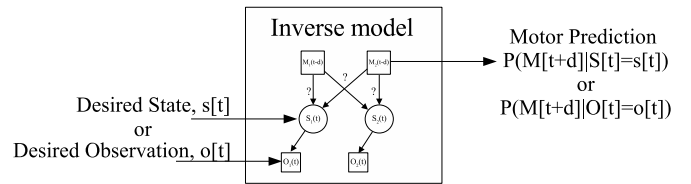


Figure 12: Using the learnt Bayesian network as an inverse model. Evidence is supplied to the observations or the state, and the task is to infer the probability distribution of motor commands.

means of the observations, $VEL1[t]$ and $VEL2[t]$. The inference algorithm used to calculate this marginal distribution was the junction-tree algorithm [Pearl, 1988]. To evaluate the quality of the learnt forward model the grippers were babbled in a similar way as before, using a different Markov model. Figure 11 shows how the most likely predicted observation compares with the actual observation. Only the x-coordinate predictions are plotted because the grippers' movements are predominantly in this plane. The prediction is shown to be very accurate.

The learnt Bayesian network can also be used as an inverse model by calculating the most likely motor command given the current velocity observation for each time-step, as shown in figure 12. One use of this inverse model is for imitation: by replacing the robot's observations of its own movement with its observations of a human's hand movements in the inverse model, the robot is able to reproduce the motor commands that would be most likely to recreate that human movement in its own gripper system. The same principle is used by Demiris to switch from imitation to recognition, albeit in a multiple paired inverse and forward model system [Demiris, 2002]. Figure 13 shows the results of the imitation experiment: the robot is able to imitate a simple hand waving motion. The human's hand movements are tracked and

recognised using the vision system, which is able to correctly track the two hands. The observations are used as evidence for the Bayesian network already learnt by the robot. This network is then used as an inverse model to predict the most likely motor commands that would produce this observation. These motor commands are sent to the robot's motor system enabling it to perform the action that best replicates the observed movement.

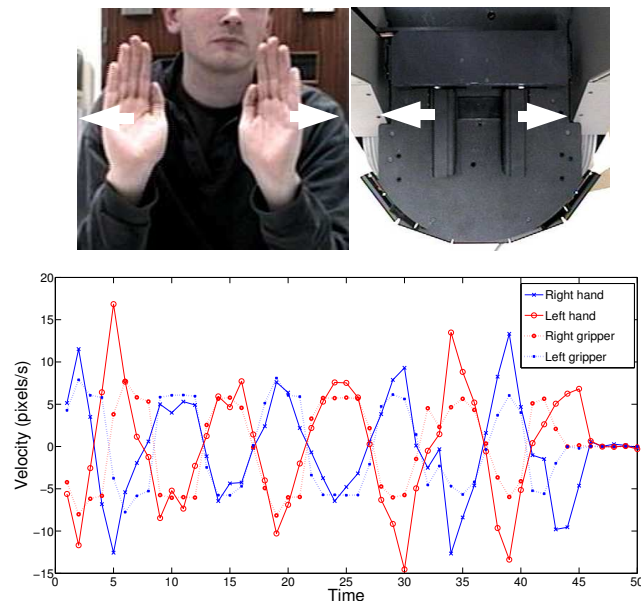


Figure 13: Imitation using a single inverse model. The top images are corresponding frames from the demonstration (left) and imitating (right) sequences. The graphs show the trajectory of the demonstrating hands, and the corresponding imitating trajectory of the grippers.

6 Conclusion

Forward models relate a robot's motor commands to their effect on its motor system and the robot's environment. The system presented here allows a robot to autonomously learn forward models of its motor system, using feedback from its vision system, by babbling with its motor commands. By representing the forward model with a Bayesian network, the uncertainty in its prediction can also be represented. An experiment was implemented showing how a robot can learn a forward model of its gripper system. The forward model was subsequently used to allow the robot to imitate a simple 'waving hand' gesture of a human. This imitation occurred without any prior information having to be supplied by a human: the robot learns how to control its motor system, then uses this information to imitate a human gesture. Future work will involve extending the system to learn forward models with more degrees of freedom and interaction with objects.

7 Acknowledgements

This work has been supported through a Doctoral Training Award from the UK's Engineering and Physical Sciences Research Council (EPSRC), and through a bursary from the BBC Research & Development Department. The authors wish to thank Gavin Simmons, Bassam Khadhour, Matthew Johnson and Paschalis Veskis of the Biologically inspired Autonomous Robots Team (BioART) at Imperial College.

References

- [Demiris, 2002] Y. Demiris. Imitation, mirror neurons, and the learning of movement sequences. In *Proceedings of the International Conference on Neural Information Processing (ICONIP-2002)*, pages 111–115. IEEE Press, 2002.
- [Friedman, 1997] N. Friedman. Learning belief networks in the presence of missing values and hidden variables. In *Proc. 14th International Conference on Machine Learning*, pages 125–133, 1997.
- [Gopnik and Meltzoff, 1998] A. Gopnik and A. N. Meltzoff. *Words, Thoughts, Theories*. MIT Press, 1 edition, 1998.
- [Hoppner et al., 1999] Frank Hoppner, Frank Klawonn, Rudolf Kruse, and Thomas Runkler. *Fuzzy Cluster Analysis: Methods for Classification, Data Analysis and Image Recognition*. John Wiley and Sons Ltd, 1999.
- [Jordan and Rumelhart, 1992] M. Jordan and D. E. Rumelhart. Forward models: Supervised learning with a distal teacher. *Cognitive Science*, (16):307–354, 1992.
- [Lucas and Kanade, 1981] B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proc. of 7th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 674–679, 1981.
- [Meltzoff and Moore, 1997] A. N. Meltzoff and M. K. Moore. Explaining facial imitation: A theoretical model. *Early Development and Parenting*, (6):179–192, 6 1997.
- [Neapolitan, 2004] Richard E. Neapolitan. *Bayesian Structure Learning*. Prentice Hall, 2004.
- [Pearl, 1988] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann, 1988.
- [Roy and Thrun, 1999] Nicholas Roy and Sebastian Thrun. Online self-calibration for mobile robots. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 1999.
- [Tong and Koller, 2000] S. Tong and D. Koller. Active learning for parameter estimation in bayesian networks. *Advances in Neural Information Processing Systems (NIPS)*, 12 2000.
- [Tong and Koller, 2001] S. Tong and D. Koller. Active learning for structure in bayesian networks. *Seventeenth International Joint Conference on Artificial Intelligence*, pages 863–869, 2001.
- [Wolpert et al., 1998] D. M. Wolpert, R. C. Miall, and M. Kawato. Internal models in the cerebellum. *Trends in Cognitive Sciences*, 2(9):338–347, 1998.

Inferring Image Templates from Classification Decisions

Arnab Dhua, Florin Cutzu
Computer Science Department
Indiana University
Bloomington, IN 47405
adhua,florin@cs.indiana.edu

Abstract

Assuming human image classification decisions are based on estimating the degree of match between a small number of stored internal templates and certain regions of the input images, we present an algorithm which infers observers classification templates from their classification decisions on a set of test images. The problem is formulated as learning prototypes from labeled data under an adjustable, prototype-specific elliptical metric. The matrix of the elliptical metric indicates the pixels that the template responds to. The model was applied to human psychophysical data collected in a simple image classification experiment.

1 Introduction

Consider a psychophysical experiment in which subjects classify a set of images into two classes. The image classes are designed such that correct responses require that the subjects detect the presence of certain features in certain regions of the image. Given the images, and the subjects' classification decisions, is it possible to infer the feature templates the subjects have developed and employed?

This represents an *inverse problem* in perception, in which the goal is to invert a "direct" model for the subject's responses with the goal of deriving the perceptual features used by the subject. The idea of employing inverse problem techniques is certainly not new in vision research, although it has been initially applied to low-level vision phenomena [Bertero *et al.*, 1988]. The psychological literature on the subject of inverting perceptual similarity data is dominated by research on multidimensional scaling methods (MDS). Metrical MDS [Shepard, 1980] is probably the oldest inversion method allowing the derivation of features from similarities. A neural net-based inversion method for extracting features from judgments of similarity is presented in [Lee, 1998]. More recent work in this direction has addressed the derivation of features involved in higher-level perceptual phenomena [Cutzu and Tarr, 1999; Cutzu, 2000].

The model for the subjects' decisions in the image classification task considered in this work is a generalized nearest-neighbor (1-NN) classifier, presented in detail in Section 2. Each of the two image classes is represented by a small set

of *image templates*, or *prototypes*. Given an input image that must be classified, its degree of match to all templates from all classes is estimated, and its class is determined by the class of the best matching template. Our goal is to infer the prototypes of each class from a set of images labeled by human subjects. Therefore, we are seeking to invert the nearest neighbor problem. This particular inverse problem has received considerable attention in the machine learning community. A comprehensive, recent review of the field is given in [Toussaint, 2002]. One large class of algorithms (Hart's algorithm [Hart, 1968] and its numerous subsequent improvements) selects prototypes from among the labeled data points, a restriction that cannot be applied to our problem due to the presence of pixel noise in the test images.

Our algorithm belongs to the so-called prototype generation methods, which creates prototypes at new locations in feature space. The first such algorithm ([Chang, 1974]) repeatedly merged nearest neighboring points from the same class as long as the classification error rate did not increase. [Bezdek *et al.*, 2001] and later [Mollineda *et al.*, 2002] refined [Chang, 1974] by recursively merging clusters based on geometrical criteria.

The nearest neighbor rule has been generalized by the use of an adaptable metric, a technique also employed in this paper, albeit differently. In [Hastie and Tibshirani, 1996] the metric is locally adapted to the training point neighborhood surrounding the query point at the time of classification. In [Friedman, 1994] also a flexible metric is used for distance calculation in a neighborhood around the query point. However the ideas of recursive partitioning are used in determining the neighborhood for the query point. In [Ricci and Avesani, 1999] a reduced set of prototypes is selected from the training examples and a different metric is associated with each of the selected prototypes. This varies from our approach in two ways. Firstly, the authors in [Ricci and Avesani, 1999] select the prototypes essentially in a random manner. Secondly, we select prototypes that in general do not coincide with the training data points;

A work whose spirit is similar to the present paper is [Xing *et al.*, 2003]: the authors discuss the problem of learning a global (that is, valid throughout feature space) elliptical metric for 1-NN classification from classification data. The metric is fitted to the subject data by solving a convex optimization problem.

2 A model for prototype-based image classification

Perceptual considerations required the use of a prototype-based image classifier in which the various prototypes (the image templates) “specialize” in various regions of the image, i.e., a template pays more “attention” to some pixels than to others. This was modeled as follows. Let $\mathbf{x} \in \mathbb{R}^d$ be an input image and $\mathbf{c}_k \in \mathbb{R}^d$ one of the templates. This prototype is characterized by a positive (semi) definite matrix $\mathbf{Q}_k$ which modulates the distance (degree of match) between the template and the image:

$$d(\mathbf{c}_k, \mathbf{x})^2 = (\mathbf{x} - \mathbf{c}_k)^T \mathbf{Q}_k (\mathbf{x} - \mathbf{c}_k). \quad (1)$$

The elliptical metric matrix $\mathbf{Q}_k$ specifies the combination of image pixels the template $\mathbf{c}_k$ “specializes” in. In the special case where $\mathbf{Q}_k$ is diagonal, the distance above reduces to:

$$d(\mathbf{c}_k, \mathbf{x})^2 = \sum_{i=1}^d \mathbf{Q}_k(i, i) [\mathbf{x}(i) - \mathbf{c}_k(i)]^2 \quad (2)$$

where $\mathbf{Q}_k(i, i) \geq 0$. The pixels i for which $\mathbf{Q}_k(i, i) = 0$ are ignored by the template $\mathbf{c}_k$. This distance measure is termed elliptical. Under the elliptic norm, the set of points (each point in pixel space represents an input image) that lie at equal distance to the prototype is no longer a hypersphere, as for the Euclidean metric, but a hyper-ellipsoid of arbitrary shape and orientation. Each template is characterized by a different ellipsoid. Interestingly, by endowing each prototype with its own, elliptical metric, the resulting Voronoi cells are not necessarily convex or even connected.

An equivalent formulation is derived by using similarity (rather than distance) to prototype: a point is assigned to the most similar prototype. We define the similarity to prototype as the probability of the point “belonging” to the prototype and we associate a Gaussian *pdf* with each prototype. The shape of the Gaussian varies from prototype to prototype, playing the role of the metric matrix $\mathbf{Q}_k$. The similarity of point i to prototype k is, therefore: $s_{ik} = \exp [-(\mathbf{x}_i - \mathbf{c}_k)^T \mathbf{Q}_k (\mathbf{x}_i - \mathbf{c}_k)]$. Or, expressed in Gaussian *pdf* form:

$$s(\mathbf{c}_k, \mathbf{x}_i) = \exp [-(\mathbf{x}_i - \mathbf{c}_k)^T \mathbf{W}_k^{-1} (\mathbf{x}_i - \mathbf{c}_k)] \quad (3)$$

where the covariance $\mathbf{W}_k = \mathbf{Q}_k^{-1}$. Therefore, under the 1-NN rule, point $\mathbf{x}_i$ is assigned to the prototype whose Gaussian is the largest at $\mathbf{x}_i$.

3 Finding prototypes for elliptical-metric 1-NN classifiers

Each of the N points $\mathbf{x}_i \in \mathbb{R}^d$ (the images) is labeled class 1 or class 2. We seek class-1 and class-2 prototypes which correctly label the N points under the 1-NN rule using the elliptical-norm-based similarity measure in Equation (3). Computing a prototype entails determining its location $\mathbf{c}_k$ as well as its metric matrix $\mathbf{Q}_k$.

Our **prototype finding** algorithm is as follows:

1. Start with two prototypes, derived by applying **Gaussian clusters** (algorithm explained below) with two centers to the full data set. Go to step 2.

2. At iteration step t there are K_t prototypes. Consider the points assigned to prototype k ($k = 1, \dots, K_t$) by **Gaussian clusters**. There are n_{k1} class 1 and n_{k2} class 2 such points, with $n_k = n_{k1} + n_{k2}$. If $n_{k1} > n_{k2}$, then the prototype is labeled 1, otherwise it is labeled 2. The points of the minority class are, therefore, misclassified by this prototype. The “impurity” of the prototype was defined as the misclassification rate: $r_k = 1 - \max \left[\frac{n_{k1}}{n_k}, \frac{n_{k2}}{n_k} \right]$. If the impurity of prototype k is greater than some user-defined threshold g , $r_k > g$, then prototype k is considered for being split into two prototypes (just like a node in a classification tree is split when too impure). The splitting is performed only if the splitting reduces the impurities in the two resulting clusters. If a node does get split, we obtain a class 1 and a class 2 prototype. These “child” prototypes are placed at the centers of mass of the n_{k1} , respective n_{k2} points. Due to splitting, at the end of this step there are $K_{t+1} \geq K_t$ prototypes.

If no prototypes are split, go to Step 4, else go to Step 3.

3. Re-apply **Gaussian clusters** to the data set, ignoring point labels. **Gaussian clusters** is initialized with the K_{t+1} prototypes determined at Step 2. Go to Step 2.
4. The algorithm also returns a global classification error rate which measures the errors caused by using the prototypes for classification. This error rate is defined by the total number of minority class points at all prototypes. The prototypes are labeled according to the majority class at the prototype. Thus, for each prototype a center ($\mathbf{c}_k$), a covariance matrix ($\mathbf{W}_k$), and a class label are returned.

Observations: i) The algorithm has as sole adjustable parameter the impurity threshold g . The number of prototypes of either class is determined automatically. ii) The class labels of the points are solely used in splitting decisions, and are irrelevant to the Gaussian k-means algorithm. iii) Note the similarity to classification and regression trees (CART).

The **Gaussian clusters** algorithm fits K Gaussian clusters to input data set. The **Gaussian clusters** algorithm is basically like k-means clustering except that each center also has a covariance matrix associated with it. So for every iteration of the **Gaussian clusters** algorithm when the new centers are being estimated we also estimate the new covariance matrices. This covariance matrix distorts the attraction of each center along the different dimensions.

4 Results: Artificial data

We applied our prototype finding algorithm to several synthetic data sets, illustrated in Figures 1-3. These synthetic data sets were generated in two dimensions (two pixel images), so that the structure of the point clouds are clearly visible.

In experiment 1 (Figure 1) there were 320 points of class 1 (blue) and 380 points of class 2 (red). The class 1 points were arranged in a single vertical band and the class 2 points were shared by 1 vertical and 1 horizontal band. As expected three

centers have been assigned to the three distinct point clouds based on the desired purity tolerance.

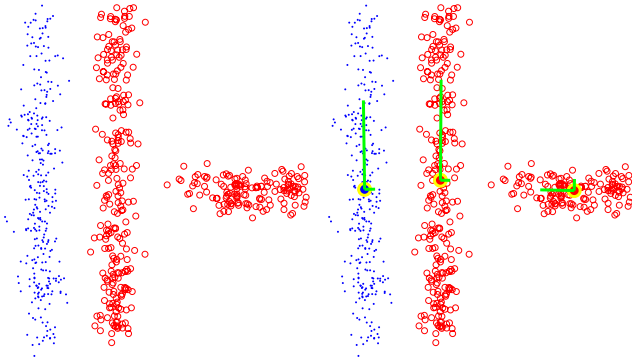


Figure 1: *Left*: Training set: class 1 points are blue (dots) and class 2 points, red (circles). *Right*: The algorithm has found three prototypes, whose centers are indicated by yellow circles. The green lines indicate the axes of the Gaussians. The cyan/magenta points (if present) are misclassified.

In experiment 2 (Figure 2) there were 340 points of class 1 (blue) and 340 points of class 2 (red). The points were arranged into two non-overlapping, non-concentric circles. As seen in the figure two prototypes were assigned, one to each circle. Note that the prototype centers are at locations where there were no training examples.

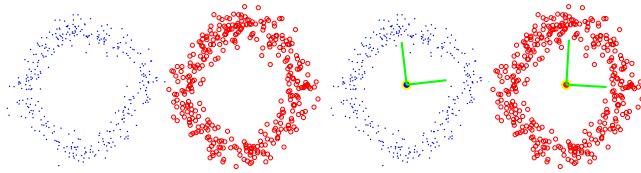


Figure 2: *Left*: Training set: class 1 points are blue (dots) and class 2 points, red (circles). *Right*: The algorithm has found two prototypes, whose centers are indicated by yellow circles. The green lines indicate the axes of the Gaussians. The cyan/magenta points (if present) are misclassified.

In experiment 3 (Figure 3), there were 1849 points of class 1 (blue) and 1849 points of class 2 (red). Both the classes were drawn from a gaussian distribution, with different means and standard deviations. Notice the overlap of the two point clouds. In the results obtained from the algorithm, we can see that the points assigned to the class 2 prototype surround the class 1 cluster: this non-convexity is an effect of the elliptical norm in determining nearest-neighbors.

5 Results: Psychophysical experiment data

The prototype finding algorithm was applied to simulated data and also to human image classification data collected in a psychophysical experiment in which the subjects had to discriminate between two classes of visual patterns. The visual pattern discrimination task is illustrated in Figure 4. In this task, the observer is shown a single white square corrupted

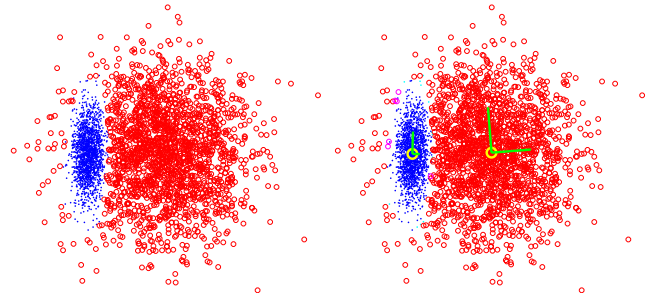


Figure 3: *Left*: Training set: class 1 points are blue (dots) and class 2 points, red (circles). *Right*: The algorithm has found two prototypes, whose centers are indicated by yellow circles. The green lines indicate the axes of the Gaussians. The cyan/magenta points (if present) are misclassified.

by additive Gaussian pixel noise at one of four possible locations. The location of the square is randomly chosen on each trial. Two of the possible locations are above the fixation point and two are below the fixation point, with each location being equidistant from the central fixation point. Noise is added to each pixel in each of the four locations, with each location divided into a 4×4 grid of pixels. No noise is added to the regions in between the four square locations, and the fixation point remains on the screen throughout the experiment. The observers task is to indicate whether the white square signal appeared above or below fixation. The contrast of the white square is manipulated across trials according to a 2-down 1-up adaptive staircase procedure to place it at a level where performance is at approximately 71% correct. Accuracy feedback is given in the format of a high or low beep.

Before collecting human subject data two simulations were performed using the above experimental setup. The two models simulated were the “exemplar” model and the “prototype” model. In an “exemplar” model the observers are assumed to use multiple noisy “templates” to form a representation for each category. Each of these templates are compared to the input to reach a classification decision. The templates used in the “exemplar” model simulation were the ideal templates (the four signals shown in Figure 4). In a “prototype” model observers are assumed to use a single summary representation for each category. Thus the templates used in the prototype model simulation were combined versions of the two signals within each category. This resulted in a ‘top’ prototype template composed of the two top squares and a ‘bottom’ prototype template composed of the two bottom squares. For each of these two models a simulation with 16000 trials was performed.

Four human subjects were tested. For each subject, 4000 experimental trials were conducted. The staircase procedure was used to keep the performance rate for each subject pegged at around 71% correct.

Since each white square was 4×4 pixels, the classification images viewed by the subjects were 8×8 pixels (the center of the display was a constant gray-level). Therefore, the algorithm learned prototypes in 64-dimensional space. The simplifying assumption was made that there were no percep-

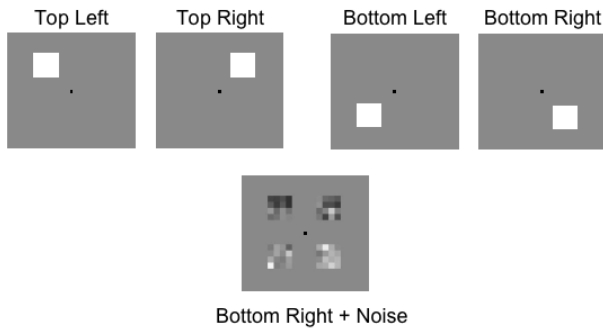


Figure 4: *Top*: The four types of stimuli used in the experiments, presented without noise. Top left and top right represent the class “above fixation”, and the bottom left and right represent the class “below fixation”. *Bottom*: An actual test image used during the experiments. The correct response is ‘bottom’, since the bottom right square is (slightly) brighter. Note the large noise level.

tual interactions between the pixels, and therefore the covariance matrices associated with the prototypes were restricted to being diagonal. A diagonal covariance matrix has one entry per pixel, and can be interpreted as an image-sized *mask* that indicates which pixels the corresponding prototype pays attention (responds) to.

The algorithm yielded the expected templates when applied to the simulation data. Figure 5¹ shows the templates obtained on running the algorithm on the “exemplar” model simulation. As expected the centers represent the four different stimuli. The masks shown alongside each center show the relative importance of each pixel for that template. The masks are actually the inverses of the diagonal values of the diagonal covariance matrices. A whiter pixel in the mask indicates that the template is more sensitive to that particular pixel. The masks are quite noisy but one can see that the pixels of the image that include the stimuli are considered more important by the mask. To show the relative importance of each of the four quadrants we also displayed the masks averaged over each of the four quadrants (Figure 6). In this figure the higher importance of the quadrant containing the stimulus is clearly visible.

Figure 7 shows the templates obtained on running the algorithm on the “prototype” model simulation. As expected the two centers represent the combination of the stimuli that form each class. The masks shown alongside each center show the relative importance of each pixel for that template. The masks are again quite noisy but one can see that the pixels of the image that do not include the stimuli are considered more important by the mask. To show the relative importance of each of the four quadrants we also displayed the masks averaged over each of the four quadrants (Figure 8). In this figure the higher importance of the quadrants not containing the stimu-

¹The results (Figures 5 - 12) are displayed in a more concise format than the format (example Figure 4) used during experiments. In particular, the “fixation point” is not shown and the four, 4×4 pixel squares are shown adjacent to each other in the results.

lus is clearly visible.

The difference in the behavior of the mask across the “exemplar” and “prototype” simulations is interesting. In the case of the “prototype” simulation it seems that the classification into top (class 1) is made by the presence of darker pixels in both the bottom quadrants and vice versa.

The algorithm yielded interesting results on the human subjects. Three subjects (subjects 1,3 and 4) appeared to have used four centers, one for each corner square, and thus two per class, as can be seen in Figures 9, 11, 12. The masks obtained for the subject data show that the pixels considered most important do not seem to follow any particular pattern. Also the difference in importance across pixels is very small. This seems to lead to the conclusion that the human subjects based their decision on the entire image rather than on a particular quadrant.

Interestingly, subject two appears to have used two centers, one per class (Figure 10). The two centers correspond to the two top and bottom corners taken together. Note that in no actual stimulus both corner squares are “on” simultaneously. This would correspond, in the language of the psychology of categorization, to a so-called “prototype model”, while the other three subjects operated using an “exemplar model”. The masks in this case are more distinct and like in the “prototype” simulation show a stronger response for the pixels in which the stimulus is absent. This can intuitively be explained by noting that for example in class ‘top’ no input image actually contains a stimulus in both the ‘top left’ and the ‘top right’ quadrant. However for class ‘top’ both the bottom quadrants can be expected to have a lower contrast value, because the stimulus is absent in the lower two quadrants. A similar intuition can be applied to the ‘bottom’ class.

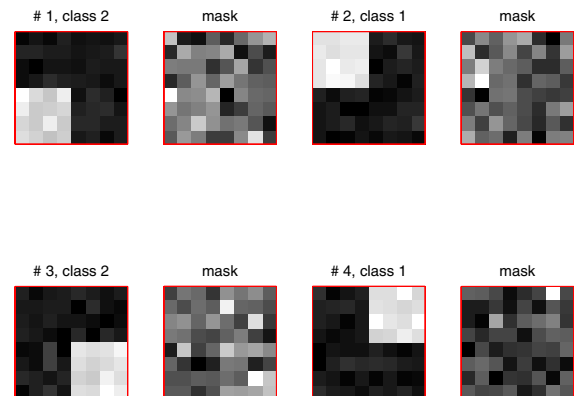


Figure 5: Templates obtained: The 4 “exemplar” centers and corresponding masks obtained by running our algorithm on the data generated by running an “exemplar” model simulation.

6 Discussion

We introduced a prototype generating method for nearest-neighbor classification. Each prototype is endowed with its own elliptical metric, and our method is therefore related to adaptive metric methods such as [Hastie and Tibshirani,

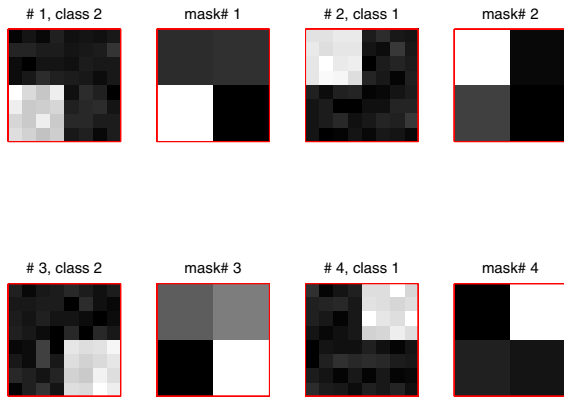


Figure 6: Templates obtained: The 4 “exemplar” centers and corresponding masks obtained by running our algorithm on the data generated by running an “exemplar” model simulation. The masks are averaged over each of the four quadrants to give an idea of the overall importance of each quadrant in the classification.

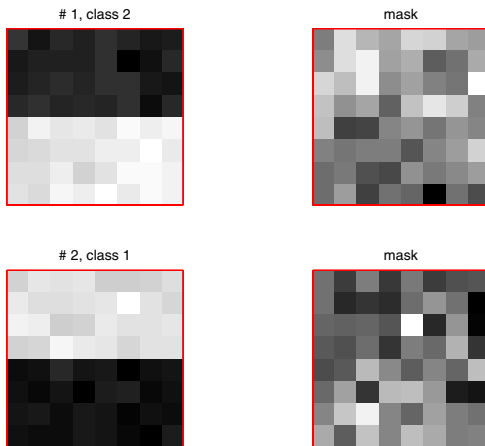


Figure 7: Templates obtained: The 2 “prototype” centers and corresponding masks obtained by running our algorithm on the data generated by running a “prototype” model simulation.

1996] and especially [Xing *et al.*, 2003]. The prototypes are not a subset of the data points, and therefore our method belongs in the prototype generation category. However, as opposed to Chang’s method and its later refinements ([Bezdek *et al.*, 2001], [Mollineda *et al.*, 2002]) we find new prototypes by recursively splitting clusters, rather than by aggregating them.

The main contribution of this paper lies in the application of the prototype learning algorithm to understanding human image classification. If an image classification experiment is designed such that correct classification decisions require the detection of one of several class-specific features in certain regions of the image, then nearest-neighbor is an appropriate classification model. In the experiment described in this paper, class 1 was defined by the presence of feature A (Top

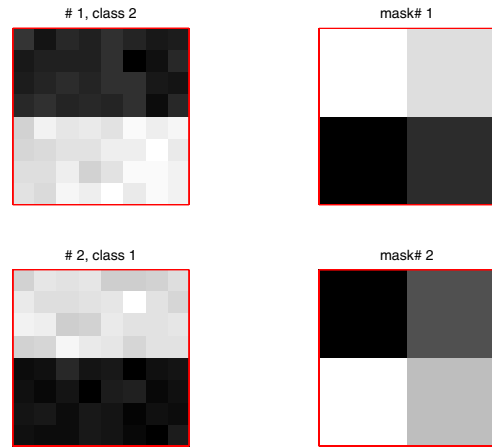


Figure 8: Templates obtained: The 2 “prototype” centers and corresponding masks obtained by running our algorithm on the data generated by running a “prototype” model simulation. The masks are averaged over each of the four quadrants to give an idea of the overall importance of each quadrant in the classification.

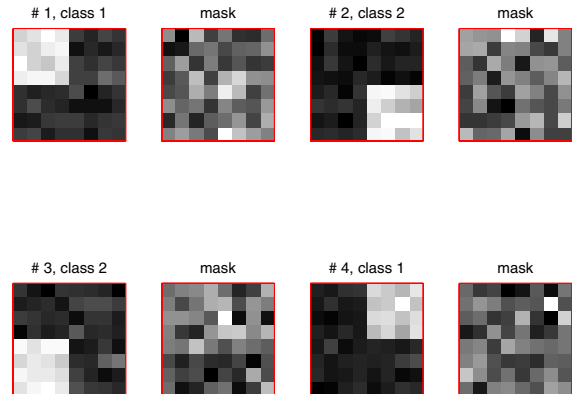


Figure 9: Templates obtained: The 4 “exemplar” centers and corresponding masks obtained for subject 1 by running our algorithm on the subjects classification data.

Left) or feature B (Top Right) and class 2 by the presence of feature C (Bottom Left) or feature D (Bottom Right). Any image contains *only* one of the four features, and, moreover, each feature occupies a specific image region. The algorithm correctly identified the regions of the image each prototype “specializes” in.

In this situation, there are two equally effective decision strategies available to a nearest-neighbor classifier: 1. store two prototypes per class, one for each of the features A, B, C, D; or, 2. store one prototype per class, one for the composite feature (A or B) and one for the composite feature (C or D). The first strategy corresponds, in the terminology of the psychology of categorization, to categorization by exemplar, and the second, to categorization by prototype. Most interestingly, our algorithm revealed that one of the subjects employed the second strategy, and the other three the first. It

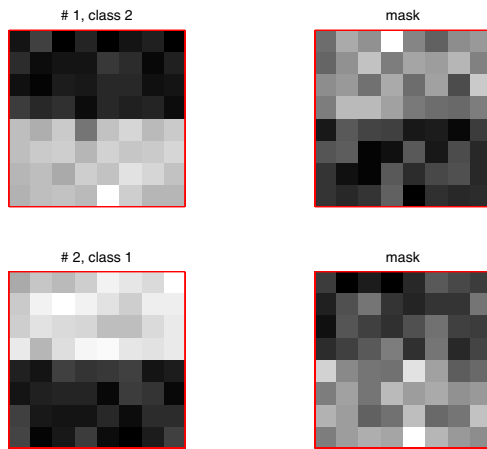


Figure 10: Templates obtained: The 2 “prototype” centers and corresponding masks obtained for subject 2 by running our algorithm on the subjects classification data.

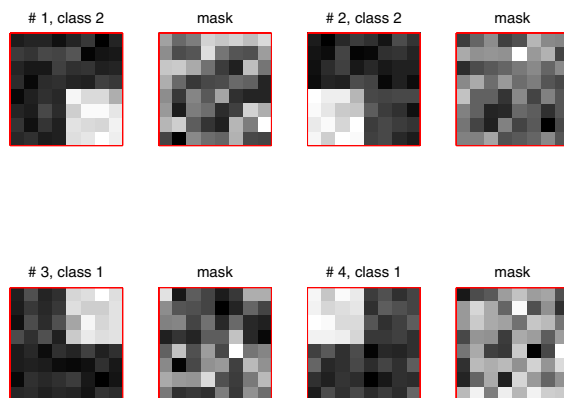


Figure 11: Templates obtained: The 4 “exemplar” centers and corresponding masks obtained for subject 3 by running our algorithm on the subjects classification data.

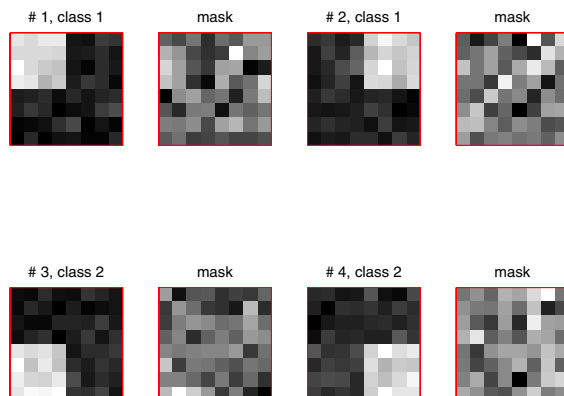


Figure 12: Templates obtained: The 4 “exemplar” centers and corresponding masks obtained for subject 4 by running our algorithm on the subjects classification data.

is important to note that a conventional analysis of error rates would not be able to distinguish between the two strategies.

References

- [Bertero *et al.*, 1988] M. Bertero, T. Poggio, and V. Torre. Ill-posed problems in early vision. In *Proceedings of the IEEE*, volume 76, pages 869–889, 1988.
- [Bezdek *et al.*, 2001] J. C. Bezdek, T. R. Reichherzer, G. S. Lim, and Y. Atikiouzel. Multiple-prototype classifier design. *IEEE Trans. Systems, Man and Cybernetics – Part B*, 31:408–413, 2001.
- [Chang, 1974] C. L. Chang. Finding prototypes for nearest neighbor classifiers. *IEEE Transactions on Computers*, 23:1179–1184, 1974.
- [Cutzu and Tarr, 1999] F. Cutzu and M. Tarr. Inferring perceptual saliency fields from viewpoint-dependent recognition data. *Neural Computation*, 11:1331–1348, 1999.
- [Cutzu, 2000] Florin Cutzu. Computing 3D Object Parts from Similarities among Object Views. In *Computer Vision and Pattern Recognition*, volume 2, pages 2095–2100, 2000.
- [Friedman, 1994] J.H. Friedman. Flexible metric nearest neighbor classification. Technical report, Stanford University, Stanford, CA., Nov 1994.
- [Hart, 1968] Peter E. Hart. The condensed nearest neighbor rule. *IEEE Transactions on Information Theory*, 14:515–516, 1968.
- [Hastie and Tibshirani, 1996] T. Hastie and R. Tibshirani. Discriminant Adaptive Nearest Neighbor Classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18:607–616, 1996.
- [Lee, 1998] M. D. Lee. Neural Feature Abstraction from Judgments of Similarity. *Neural Computation*, 10:1815–1830, 1998.
- [Mollineda *et al.*, 2002] R. A. Mollineda, F. J. Ferri, and E. Vidal. An efficient prototype merging strategy for the condensed 1-NN rule through class-conditional hierarchical clustering. *Pattern Recognition*, 35:2771–2782, 2002.
- [Ricci and Avesani, 1999] F. Ricci and P. Avesani. Data compression and local metrics for nearest neighbor classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1999.
- [Shepard, 1980] R. N. Shepard. Multidimensional scaling, tree-fitting, and clustering. *Science*, 210:390–397, 1980.
- [Toussaint, 2002] Godfried Toussaint. Proximity Graphs for Nearest Neighbor Decision Rules: Recent Progress. In *Proceedings of INTERFACE-2002, 34th Symposium on Computing and Statistics*, Montreal, Canada, 2002.
- [Xing *et al.*, 2003] Eric P. Xing, Andrew Y. Ng, Michael I. Jordan, and Stuart Russell. Distance Metric Learning with Application to Clustering with Side-Information. In S. Thrun S. Becker and K. Obermayer, editors, *Advances in NIPS 15*, pages 505–512. 2003.

Combining Structural Descriptions and Image-based Representations for Image, Object, and Scene Recognition*

Nicolas Do Huu[†], Williams Paquier and Raja Chatila

LAAS-CNRS

7, avenue du Colonel Roche 31077 Toulouse Cedex 4 - France
nicolas.dohuu@laas.fr, william.paquier@laas.fr, raja.chatila@laas.fr

Abstract

Object and scene learning and recognition is a major issue in computer vision, in robotics and in cognitive sciences. This paper presents the principles and results of an approach which extracts structured view-based representations for multi-purpose recognition. The structures are hierarchical and distributed and provide for generalization and categorization. A tracking process enables to bind views over time and to link consecutive views. Scenes can also be recognized using objects as components. Illustrative results are presented.

1 Introduction

Object and scene learning and recognition is a major issue in computer vision, in robotics, in neuroscience, and in cognitive sciences as well. And one of the main questions to this respect is how to extract knowledge from 2D patterns of light in the camera or the retina.

For the recognition of objects, two models were proposed in the literature. In the 'image-based' or 'view-based' model, an object is represented as a collection of view-specific local features [Poggio and Edelman, 1990; Ullman, 1998; Tarr and Bülthoff, 1998; Riesenhuber and Poggio, 1999]. Representations are organized in trees in which a set of view-tuned units constitutes the weighted inputs to a higher level object-invariant unit. Each unit measures the similarity between the input image and its stored view, and the higher level unit computes the weighted sum from its incoming connections. If the resulting value reaches a given threshold, then the learned object is recognized. Riesenhuber's HMAX model of object recognition in the ventral visual stream of primates also proposes a similar grouping method where higher level cells compute the maximum response of view-tuned cells [Riesenhuber and Poggio, 1999].

The second model is based on structural descriptions. One of the most important approaches is the "Recognition-By-Components" (RBC) model [Biederman, 1987] in which each

object is represented as a collection of volume parts. Thus, there is no need of multiple view-tuned representations since 3D models can be virtually rotated and compared to the input image. Moreover, the use of such a model can make the recognition invariant to illumination and color.

We can identify certain properties of the modeling process for efficient learning and recognition. Firstly, the training and thus the construction of new representations must be sufficiently fast. Lack of speed is the principal cutoff of models based on structural descriptions because building 3D models from images remains a non trivial task. Secondly, an efficient modeling process must organize knowledge in a structured way. A first means of organizing the data is to build categorical representations. Categorical structures make it possible to obtain capacities of class generalization at low cost. Indeed, to effectively exploit the extracted data, those must be describable at various levels of specificity: a bottle can be described as a container, a plastic object for recycling, etc. Such a structure can also accelerate recognition that deals with large collections of stored objects by reducing the number of candidate object models. Another mode to structure meanings is the decomposition of a representation as the set of its parts. Thanks to structural descriptions, RBC offers more robustness, in particular with respect to noise and occlusion. Moreover, it is interesting to share components among several structural descriptions to save memory: one could use the representation of a wheel to describe either a car, a truck or a bike. Thirdly, a learning system for knowledge extraction and structuring must allow the addition of new representations at non-prohibitive memory cost and without explosion of complexity. This open-endedness property can be approached by using the two kinds of data structures referred to above, in which a representation can be factorized in categories or shared as components.

While considering object recognition one must mention important results achieved by template-based approaches for object classification and recognition [LeCun *et al.*, 2004]. Nearest Neighbor methods, Support Vector Machines and Convolutional Networks provide efficient solutions but the needs of structured knowledge, reusability, incremental and autonomous learning are several points addressed here which are not, to the best of the authors' knowledge, well dealt with by the pre-cited methods.

This paper presents an efficient approach [Paquier, 2004]

*The work described in this paper was partially conducted within the EU Integrated Project COGNIRON ("The Cognitive Companion") funded by the European Commission Division FP6-IST Future and Emerging Technologies under Contract FP6-002020.

[†]supported by the European Social Fund.

(which is partially implemented) for building structured representations without using 3D primitives and exhibiting the properties mentioned above. The paper is organized as follows. Section 2 presents the model we choose to both extract and organize representations. Each essential property is presented and illustrated by an example produced by the implemented system. Section 3 introduces the view-binding algorithm and the incremental building of object-invariant detectors. The use of objects as landmarks for building structured representation of scenes is presented in section 4. We conclude in section 5.

2 Architecture for View-based Recognition

We will first describe the *map* structure which constitutes the basic building block of our model, and discuss its inherent shift-invariant property. In the learning process, the maps will specialize in certain features, and we will present the way we associate them in order to prevent redundancy in this specialization and structure extracted features.

2.1 Maps and Inter-maps Connectivity

The Map: a Set of Local Classifiers

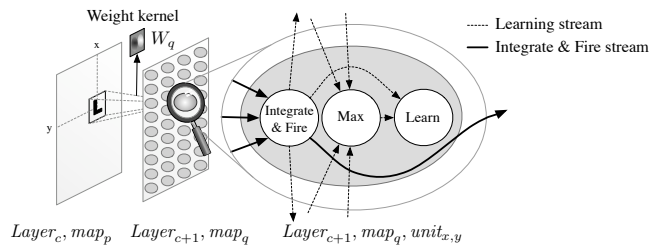


Figure 1: A *map* is a collection of *units* sharing the same set of weights (*kernel*). Each unit is composed as a three stage calculation pipeline. It receives signals incoming from previous layers for integration and lateral signals from *maps* situated on the same layer for competition.

In order to satisfy real-time constraints, the learning and recognition system must be able to update its representations at a frequency that is compatible with the modifications of the environment, and process images at a rather high frequency (e.g., 15 fps). With this end in view we chose to implement an integrate-and-fire model in local classifiers built as calculation pipelines (figure 1).

We call a *map* a collection of local classifiers called *units*, organized in a retinotopic¹ way. Each *map* is associated with one or more afferent *maps* which are the locations of its *units*' input domains (or their receptive fields). Receptive fields of each *unit* are static and each *unit* is thus associated to a set of afferent *units* in each afferent *map*. We denote Ω_i the set of afferent *units* of a given unit U_i .

At time t and for a given unit U_i , information flows from its receptive fields through a set of learning weights $W_i(t)$. Learning weights are organized in *kernels*, and there are as

¹A retinotopic mapping means that stimuli that are adjacent to each other in the visual world are processed by adjacent sets of *units*.

many weight kernels as afferent *maps*. All the *units* of a *map* are sharing the same set of weights, thus they can detect and learn a pattern which is at different positions in the input *maps*. As shown in figure 1, the coordinates of a unit in a given map and its receptive field in the afferent *map* are the same. Therefore the position of an active unit corresponds to the position of the detected pattern, which provides for a shift-invariance property.

This type of mapping has been previously introduced by Fukushima's *Neocognitron* and was successfully applied to handwritten digit recognition [Fukushima, 2003]. Figure 2 illustrates this intrinsic *map* feature. In this experiment, we produce an image containing a set of four randomly distributed letters (k, p, s, u). After less than a minute, the system has learned autonomously its own distributed representation of the input image. Each resulting *map* can extract its learned feature at any location in the input image. To obtain this result we must also provide our system with an inter-*maps* communication channel, so that *maps* don't learn the same feature several times. This procedure is called "local competition" and is introduced next.

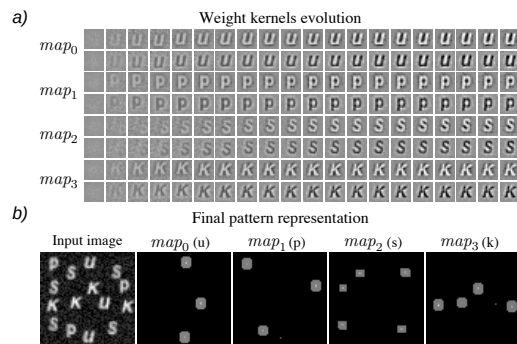


Figure 2: Letter extraction from noisy image input (20% random noise, 20% scale and angle variation, 70 steps, 2 images per second). a) weight kernels evolution across time. b) input image and associated *maps* activations.

Preventing Redundancy with Competition

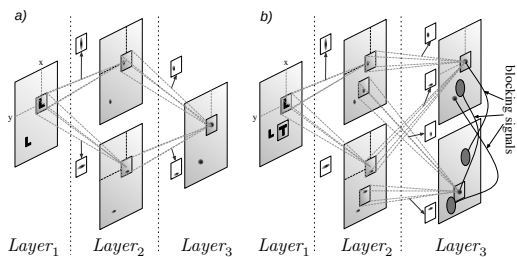


Figure 3: Hierarchical *maps* connectivity and local competition. a) The letter "L" is detected as a relative positioning of horizontal and vertical bars. b) input image containing the letters "L" and "T": inter-*maps* competition with local inhibition leads to distinct specialisations.

One previously mentioned requirement concerns the struc-

tural decomposition into several components. To achieve such a decomposition, different maps observing the same inputs must "decide" to specialize into distinct component detectors. The undergoing process is illustrated in figures 3.a and 3.b. In this example, we want to train two *maps* to detect the letters "L" and "T" in the input image. A first step of extraction is composed of a layer of two *maps* which respectively detect vertical and horizontal bars in their inputs. When the letter "L" is present in the image (figure 3.a), the pattern is decomposed as the positions of these two local orientations. The relative positioning of the two patterns is then detectable and can be learned by a dedicated *map* on the next layer (*Layer*₃).

From this point, if we add another letter ("T") to the input image and a second learning *map* to the third layer to learn it (figure 3.b), then we must prevent this new *map* from learning the "L" pattern one more time. This differentiation is achieved by using a local competition in which we compare *map* detection values and choose the best fitted (See section 2.2 for the computation of the detection value).

Right after calculating its detection value, every unit of each *map* broadcasts it to all other *units* on *maps* of the same layer and at the same coordinates. Thus, every *unit* whose coordinates correspond to the position of the letter receives incoming values at the 'Max' stage of its pipeline architecture (see figure 1). The 'Max' stage, as its name indicates, computes the maximum incoming value and sends it to its own 'Learn' stage. The learning process is then able to compare the local *unit* activation incoming from the 'Integrate & Fire' stage to distant activations. By allowing learning only to the best fitted *unit*, we ensure that no *map* could learn one pattern if another is already specialized to detect it. We illustrate this inter-*maps* competition on Figure 3.b with the dark discs which represent the blocking signal.

Connecting Maps to Build Distributed and Hierarchical Representations

We adopted a layered hierarchical architecture for several reasons. Firstly we need this architecture for *categorization*. As our units achieve linear separation in their input set, complex features cannot be extracted with a single layer. This is a known limitation of the single layer perceptron that cannot simulate an exclusive disjunction (logical XOR). The second reason is *reusability*. A huge number of extracted features are encountered in different shapes: oriented segments, arcs of circles or color blobs are building blocks for more complex images. Since extracted information is shared in the network, we avoid redundancy and the resulting computational overhead. Following the same idea, similar meanings should be encoded using shared sets of units. For example, the internal representations of a truck and of a car should intersect. This intersection representing the shared meaning could contain the internal representation of the four wheels, the steering wheel, etc. Thanks to this distributed representation architecture, it is easier to pool similar objects into cross categories (e.g., wheeled vehicles).

This capability of building hierarchical and shared representations is shown in figure 4. In this experiment we trained a network to recognize faces using a set of ten different im-

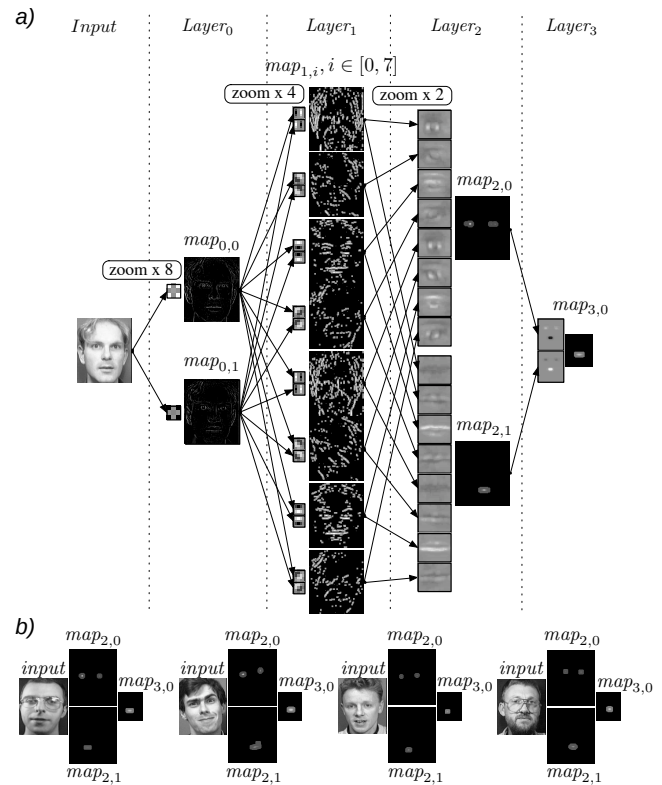


Figure 4: An example of a hierarchical representation of faces. a) network connectivity showing weight kernels and *maps* activities. *Layer*₀, *Layer*₁, *Layer*₂, *Layer*₃ respectively extract contrasts, oriented segments, eyes and mouth, and face. b) recognition robustness for different subjects and variable postures.

ages of each of 40 distinct subjects². In the same way as in the previous example of letter extraction, *Layer*₂ learns to detect the position of each eye in one *map* and the position of the mouth in the other one, sharing the same segments extraction in the previous layer (*Layer*₁). In *Layer*₃, we train another *map* which receives as inputs the detection values of the mouth and eyes specialized *maps*. As a result, this last *map* learns the representation of the face with relative positioning of eyes and mouth. We also see in this example that the resulting face recognition is robust to high variations in subject posture and morphology. Nevertheless, this kind of recognition only applies to images with limited face orientations. In section 3, we will propose additional methods to recognize 3D objects based on the pooling of overlapping views.

2.2 Integration, Firing and Learning

Now that we have presented the global mechanisms involving the computing *units*, we present the internal workings of the system more precisely and the computation of the detection values mentioned in section 2.1.

²We used the faces database of the AT&T Laboratories, Cambridge <http://www.uk.research.att.com/facedatabase.html>

Integration

Given a *unit* U_i , we denote $\beta_i(t) \in [0, 1]$ its calculated output burst at time t . The burst is the detection value mentioned before. Let $w_{ij}(t) \in W_i(t)$ be the learning weight associated to the connection between *units* U_i and U_j , and $\Omega_i^+(t)$ a subset of afferent Ω_i defined as:

$$W_i(t) = \{w_{ij}(t), U_j \in \Omega_i\} \quad (1)$$

$$\Omega_i^+(t) = \{U_j \in \Omega_i, \beta_j(t) > 0\} \quad (2)$$

We also define three weight sums as follows :

$$S_i^*(t) = \sum_{w_{ij}(t) \in W_i(t)} |w_{ij}(t)| \quad (3)$$

$$S_i^{\beta^+}(t) = \sum_{U_j \in \Omega_i^+(t)} w_{ij}(t) \quad (4)$$

$$S_i^+(t) = \sum_{w_{ij}(t) \in W_i^+(t)} w_{ij}(t) \quad (5)$$

where $W_i^+(t) = \{w_{ij}(t), U_j \in \Omega_i \text{ and } w_{ij}(t) > 0\}$. Then the integrated potential P_i at time $t+1$, which expresses a level of similarity between the learnt pattern and the inputs, is given by:

$$P_i(t+1) = \frac{\alpha_P P_i(t) + (1 - \alpha_P) S_i^{\beta^+}(t)}{S_i^+(t)} \quad (6)$$

Where $\alpha_P \in [0, 1]$ is a fixed potential leak.

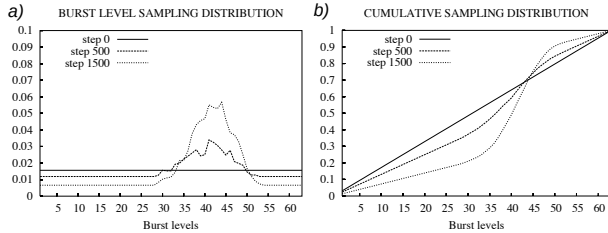


Figure 5: Burst sampling distribution and cumulative distribution evolution. a) burst sampling distributions computed at steps 0, 500 and 1500. Burst values are discretized within 64 levels. b) corresponding cumulative distribution used as a transfer function for integration.

Firing

Output bursts β_i generated by a unit U_i are produced both for integration processes situated on the next layer and for its own "Learn" stage (see figure 1). In the first case, burst levels are not taken into account since all positive bursts participate to integration (see equation 2). In the second case, we previously explained that we need to compare *units* activation in order to find the best fitted *map* in a competition process. The burst level is thus computed as follows:

$$\beta_i(t) = \begin{cases} 0 & \text{if } P_i(t) < T_i, \\ F_i(t, P_i(t)) & \text{otherwise.} \end{cases} \quad (7)$$

Where T_i is a fixed threshold and F_i is an adaptive transfer function whose variations across time are illustrated by figure 5.b. The underlying idea of this function is that, as units

are learning, their output function must converge to a binary-like response corresponding to a more strict linear separation. At each step we compute a sample distribution of the input burst by discretizing the output burst value in 64 levels. The transfer function F_i is in fact the associative cumulative distribution at time t :

$$F_i(t, x) = \sum_{0 < b \leq x} f_i(t, b) \quad (8)$$

Where function f_i is a sampling distribution of burst levels updated at each step (figure 5.a). We thus obtain a sigmoidal transfer function which permits a binary-like classification behavior and a more effective competition process. Until now, object detection is internally represented as the activation of a limited set of computing *units*. Due to our connectivity scheme, recognitions are indeed only reflected at positions located near the centre of the learned pattern. However patterns presence are more than just their center since the whole surface they cover is relevant. So we increase the pool of activated units to cover a wider surface. As a first approach, we chose a simple disc shape whose surface is given by $S_i^*(t)$.

Learning

We use in our model a hebbian-like learning rule in which *units* tend to learn patterns responsible for their activation. In other words, a learning process occurs when one *unit* has both fired and won the competition in its layer. Moreover, we compute a stochastic standard deviation σ_{ij} for each weight w_{ij} which is computed and used as coefficient in the learning calculation as follows:

$$\begin{aligned} \mu_{ij}(t+1) &= (1 - \alpha_W) \mu_{ij}(t) + \alpha_W |\gamma_{ij}(t)| \\ \sigma_{ij}(t+1)^2 &= (1 - \alpha_W) \sigma_{ij}(t) + \alpha_W [\mu_{ij}(t+1) - \gamma_{ij}(t+1)]^2 \\ w_{ij}(t+1) &= (1 - \alpha_W) w_{ij}(t) + \alpha_W [1 - 2\sigma_{ij}(t+1)] \gamma_{ij}(t+1) \end{aligned}$$

Where μ_{ij} is a stochastic mean and α_W the learning rate. Hence we ensure that weights corresponding to noisy inputs (with high standard deviation) will tend to 0 and then won't take part in the representation. γ_{ij} is a function of input burst of *unit* i which takes its values in $\{-1, 0, 1\}$. *Maps* among the same layer are in competition as shown in figure 3. If at time t , an input burst from *unit* j is received $\gamma_{ij}(t) = 1$, else if a burst is received from any other afferent *unit* then $\gamma_{ij}(t) = -1$. Finally if no burst was present $\gamma_{ij}(t) = 0$.

3 From Views to Objects

In this section, we consider the 3D-object recognition problem. Let us consider a camera rotating around an object. For a given orientation we can train a dedicated *map* and obtain enough robustness to detect this view, say within about a 45 degrees angle interval centered on the learnt prototype. Wider intervals can be obtained with "simple" objects, such as balls or paper cups, whose views are relatively invariant during view-point modification, while turning around them. Our approach is to pool overlapping view detectors. Then, a view-invariant (or object-tuned) detector can be obtained simply by computing the maximum view-tuned *map* response [Riesenhuber and Poggio, 1999]. This will provide for continuous object recognition.

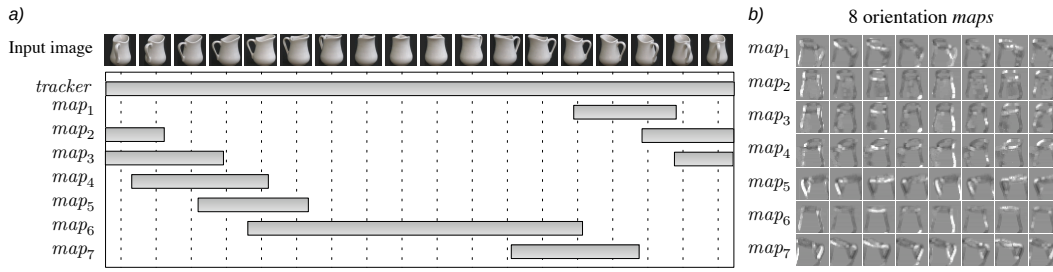


Figure 6: Object invariant detection with view-tuned *maps*. a) *maps* activities overlap to produce continuous detection. b) resulting learning weights. These are composed of 8 kernels for each *map* which correspond to 8 afferent oriented segment detectors situated on previous layer (see figure 4).

Assuming the support of an external module producing the initial object detection (this could be simply a user's selection on the input image, but this module could be based on saliency, motion, etc...), we must resolve several remaining problems. Firstly, the object-tuned *map* must keep track of the object during point of view modifications (temporal binding). Secondly, as we have no initial information about the object's complexity and, therefore cannot anticipate the exact number of required view-tuned *maps*, we must find a method which incrementally adds new *maps* to the network and associate them together.

Map-tracker : Short-term Memory for Temporal Binding

The different views of the same object tend to occur close together in time and space [Tarr and Bülthoff, 1998]. Several authors proposed to take advantage of this property [Stinger and Rolls, 2002; Wallis, 1996]. We particularly want to mention Stinger and Rolls's hypothesis about the functional architecture and the operation of the ventral visual system tested in their model called VisNet. In this model, each activation of a view-invariant neuron is maintained during a short period of time. The pooling is then achieved by using an associative memory to link temporal memory trace and current view detection. Although this method seems biologically plausible, it seems difficult to apply to real-time robotics. As the view-to-object membership is not *a priori* known, there is no criteria to decide whether or not a newly specialized *map* corresponds indeed to the view of an object. If no preselection is applied, the size of the associative memory could exceed computable capacities. So, it is crucial to restrict *map* specialization to relevant views. A solution comes from a tracking approach. In this domain, temporal coherence is also used but in another way. The aim of tracking is not so much extracting knowledge but rather following the position of a collection of pixels. Temporal coherence can here be exploited by considering that persistent informations are preserved between two frames and can thus be extracted. We propose to use the *map* architecture for pixel tracking in order to follow the position of the object and use this information as a first step for an incremental view-tuned *map* creation.

We can obtain the desired tracking capability by increasing the learning rate α_W of a map (see section 2.2), which we will call *tracker-map*, to a value near 1. The *tracker-map* is trained at the time the first object position is produced by the external

module. We call this process "one-shot learning" because of the use of a high learning rate. In the next frame, thanks to the robustness of our model, the *tracker-map* can detect the object even if it has started to move. According to the hebbian rule, this detection is followed by a learning phase. Then, another learning occurs which almost completely renews the stored prototype due to the high learning rate. From now the process can restart allowing continuous detection. *Tracker-maps* can be viewed as short-term memories that never reach stable representations.

Temporal Binding

Now that we can use temporal coherence to track the object position, we must resolve the remaining problem of building incrementally view-specific representations of the tracked object. To reach this goal we granted *tracker-maps* the capability of creating new maps during runtime. As we are in the context of moving objects and/or moving point of views, one-shot learning (learning with high learning rate will also be used to catch object's views during movement. The very specific resulting prototype will be refined as view-tuned maps compute standard learning rates. The binding algorithm develops as follows: if no view-tuned *map* recognizes the object at the position of the *tracker-map*'s detection during a short period of time (we use a 5 frames time-out), which is initially the case, then the *tracker* creates a new *map* and forces it to learn the unknown view with a one-shot-learning signal. If a view is detected, because a *map* has previously learnt it, then a standard learning process occurs in this *map* and the detection produces a one-shot learning signal for the *tracker-map* in order to focus tracking on this view. This process ensures that long-term memory of a specific view has a higher priority during object detection than the *tracker*. This algorithm has been used in the experiment of figure 6 in which the image of a rotating object has been used to train a three-layers network. In figure 6.a, we see that the *tracker* always detects the object, thanks to the feedback from long-term to short-term memory, and that activity of *map6* is about 3 times longer than the others. Indeed, this *map* learned to detect orientations that are rather invariant during rotation. Our model can thus adapt the number of required *maps* depending on the complexity of objects.

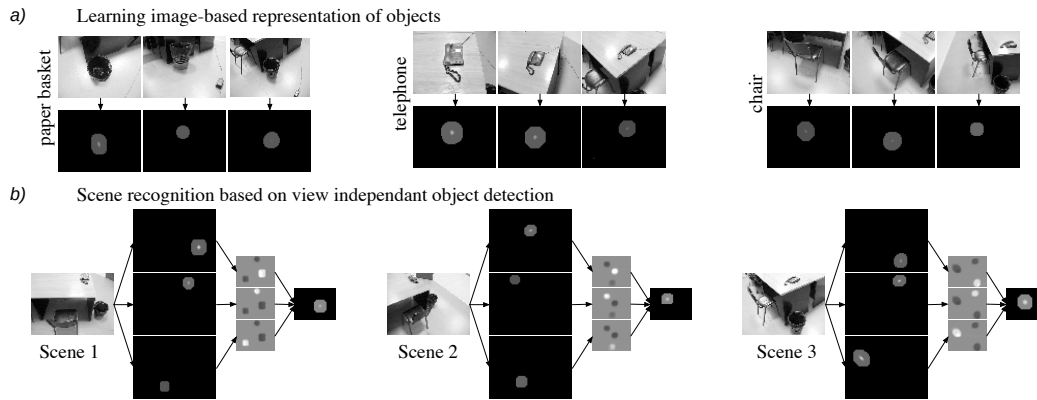


Figure 7: Representing scenes as arrangements of objects in an office environment. a) examples of object-invariant detections from different points of view learn. b) results of scene recognition by relative positioning of object-invariant detections.

4 Objects as landmarks for scene recognition

We present in this section experimental results of our system in the context of scene recognition. We can define a scene as a particular arrangement of objects. We demonstrated previously with face recognition that our architecture is able to construct hierarchical representations of an image. We can use this same technique in order to learn a scene as relative arrangement of objects. For this experiment we observed a standard office environment including 3 basic objects: a paper basket, a phone and a chair. In a first phase, object representations are learnt on the fly using temporal coherence by the means of tracker-maps (section 3). The resulting network then detects each object simultaneously in view-specific maps and in the corresponding object-invariant tracker-map. In a second phase, three higher level maps connected to all object-invariant detectors learn representations of the scene from three different points of view. Figure 7.a shows the outputs of the three object-invariant detectors built as presented in section 3 with natural images. Without needing any additional algorithm, our model extracts a view-point specific representation of the scene in a hierarchical structure (figure 7.b). From now, a location-invariant map can be trained thanks to tracker-maps for scene-invariant maps pooling as explained in section 3. These multi-layered architecture allowed by parallel-pipeline calculations is the key property which permits to combine view-based representations and structural descriptions.

5 Conclusion

In this paper we have introduced several methods and algorithms to extract knowledge from visual data. Our model is able to build representations by decomposition of image features into local components structured in a global hierarchy of concepts. Such decompositions are minimal requirements for both sharing representation in order to save memory and computational time, and providing generalization capabilities due to the fact that extracted components can be used as description blocks to recognize new elements. The main advantage of our model is that one does not need any prior knowledge or model to build robust dedicated detectors of simple to

complex features in a unified language. Thus, building heterogeneous libraries of meanings becomes feasible. These representations could be then provided as input to a symbolic reasoning level.

References

- [Biederman, 1987] I. Biederman. Recognition-by-components: a theory of human image understanding. *Psychological Review*, 94:115–147, 1987.
- [Fukushima, 2003] K. Fukushima. Neocognitron for handwritten digit recognition. *Neurocomputing*, 51:161–180, April 2003.
- [LeCun et al., 2004] Y. LeCun, F.-J. Huang, and L. Bottou. Learning methods for generic object recognition with invariance to pose and lighting. In *CVPR'04*, 2004.
- [Paquier, 2004] W. Paquier. *Apprentissage Ouvert de Représentations et de Fonctionnalités en Robotique : Analyse, Modèle et Implémentation*. PhD thesis, 2004.
- [Poggio and Edelman, 1990] T. Poggio and S. Edelman. A network that learns to recognize three dimensional objects. *Letters to Nature*, 343:263–266, 1990.
- [Riesenhuber and Poggio, 1999] M. Riesenhuber and T. Poggio. Hierarchical models of object recognition in cortex. *Nature Neuroscience*, 28:1019–1025, 1999.
- [Stinger and Rolls, 2002] S. M. Stinger and E. T. Rolls. Invariant object recognition in the visual system with novel views of 3d objects. *Neural Computation*, 14(11):2585–2596, November 2002.
- [Tarr and Bülthoff, 1998] M. Tarr and H. Bülthoff. Image-based object recognition in man, monkey and machine. *Cognition*, (67):1–20, 1998.
- [Ullman, 1998] S. Ullman. Three-dimensional object recognition based on the combination of views. *Cognition*, 67:21–44, 1998.
- [Wallis, 1996] G. Wallis. How neurons learn to associate 2d-views in invariant object recognition. Technical Report 37, Max-Planck-Institute für Biologische Kybernetik, Tübingen, Germany, 1996.

Compound Effects of Top-down and Bottom-up Influences on Visual Attention During Action Recognition

Bassam Khadhour and Yiannis Demiris

Department of Electrical and Electronic Engineering

Imperial College London

Exhibition Road, London SW7 2BT

Email: {bassam.khadhour, y.demiris}@imperial.ac.uk

Abstract

The limited visual and computational resources available during the perception of a human action makes a visual attention mechanism essential. In this paper we propose an attention mechanism that combines the saliency of top-down (or goal-directed) elements, based on multiple hypotheses about the demonstrated action, with the saliency of bottom-up (or stimulus-driven) components. Furthermore, we use the bottom-up part to initialise the top-down, hence resulting in a selection of the behaviours that rightly require the limited computational resources. This attention mechanism is then combined with an action understanding model and implemented on a robot, where we examine its performance during the observation of object-directed human actions.

1 Introduction

In an attempt to arrive at a definition for *attention*, Tsotsos, in his review paper [Tsotsos, 2001], arrives at the following proposal: “*Attention is a set of strategies that attempts to reduce the computational cost of the search processes inherent in visual perception*”. Our work aims at producing a model of *visual attention for dynamic scenes*, emphasising the importance of *top-down knowledge* in directing the attention during action recognition.

After introducing the two different (bottom-up and top-down) elements of attention, we will proceed to review a model of action understanding [Demiris and Johnson, 2003] that will make use of our attention model, to correctly allocate the limited resources available to it. Subsequently, we will proceed to describe how the multiple hypotheses generated by our model *while the human demonstration is unfolding* can feed top-down signals to the attention mechanism to direct attention to the important aspects of the demonstration.

Furthermore, we propose a method for *initialising* the top-down part, using the saliency of the bottom-up part in our visual attention mechanism. We have implemented our model on an ActivMedia Robot, running experiments observing a human acting upon various objects. These results will be presented and discussed, not only in terms of whether our attention model allocates the resources correctly, but also to see if

it results in a faster recognition of the correct behaviour being demonstrated in the action understanding model.

2 Background

Work in cognitive science suggests that the control inputs to the attention mechanism can be divided into two categories: stimulus-driven (or “bottom-up”) and goal-directed (or “top-down”) [Van Essen *et al.*, 1991]. A number of bottom-up attention models follow Treisman’s Feature Integration theory [Treisman and Gelade, 1980] by calculating the saliency for different low-level features of the object, e.g. colour, texture or movement. A winner-take-all approach is then used to decide on the most salient part of the scene (as in [Koch and Ullman, 1985], [Itti *et al.*, 1998], [Breazeal and Scasselati, 1999]). Top-down, on the other hand, covers the goal-directed factors, essentially the task-dependent part of the visual attention model. Wolfe [Wolfe and Gancarz, 1996] produced a biologically inspired Guided Search Model that controls the bottom-up features that are relevant to the current task by a top-down mechanism, *through varying the weighting of the feature maps*. However, it is not clear what the task relevant features are, particularly in the case of action recognition.

Our attention mechanism is inspired by Wolfe’s model, and integrates bottom-up elements with a top-down mechanism using *behaviours*¹ and *forward models* that can guide the robot’s attention according to the current task. A forward model is a function that, given the current state of the system and a control command to be applied on it as given by the behaviour, outputs the predicted next state. Our top-down part of the attention model, when observing a demonstration, will make a prediction of the next state for a number of different possible behaviours, producing a confidence value for each, based on the observed accuracy of the prediction. These confidence levels are important values that can be thought of as *saliencies* for the top-down mechanism of our attention model, hence producing a principled method of quantifying the top-down part of the attention mechanism.

In the experimental section, we will demonstrate that this attention mechanism improves performance on action understanding mechanisms that use multiple behaviours and for-

¹Also known as controllers or inverse models in the control literature

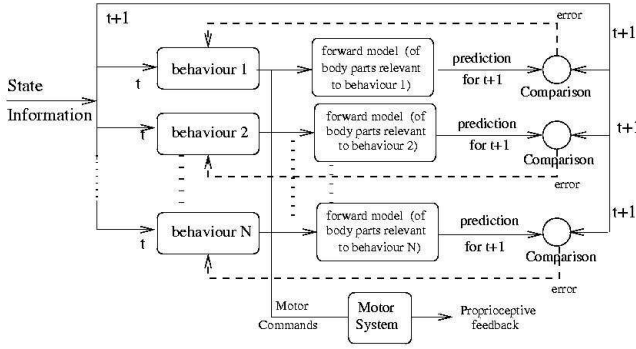


Figure 1: Action understanding model [Demiris and Johnson, 2003].

ward models such as [Demiris and Johnson, 2003]. This will be done in two ways: first by cutting down the number of computational cycles required for identifying the correct behaviour and by directing the limited computational resources to the relevant parts of the human demonstration, instead of the whole scene. Secondly, by using the saliency from the bottom-up part of our visual attention mechanism to initialise the top-down part, enabling it to select only the relevant behaviours to the demonstrated action, instead of activating and running all of them.

3 Action Understanding Model

Demiris’s action understanding model [Demiris and Johnson, 2003], shown in figure 1, identifies the correct behaviour that is being observed by using forward models for a number of behaviours. By predicting what will happen next, and comparing it with what actually does happen next (from the behaviour that is being demonstrated), confidence levels are generated for each of the predicted behaviours. From these, a winner is selected by picking the predicted behaviour with the highest confidence level.

The attention mechanism we propose in this paper can be used to cut computational costs on this action understanding model [Demiris and Johnson, 2003]. It would be far too computationally expensive to direct the attention of the observer towards all the parts of the demonstration to satisfy all the possible behaviours [Tsotsos, 1989]. Hence, the attention mechanism is used to restrict this, giving only one of the behaviours at a time the information it requires. Using this attention mechanism, we managed to cut down substantially on the computational costs, yet it was also achieved without affecting the quality of the system to the extent of producing wrong outcomes.

The success of our model will be demonstrated through comparisons of the new results with the results from the original action understanding model for a number of different behaviours. If, after having cut down on all the computational costs using our visual attention model, the final behaviour chosen in each situation with the previous model [Demiris and Johnson, 2003] remains the same, then our attention model will be deemed to have succeeded in its task.

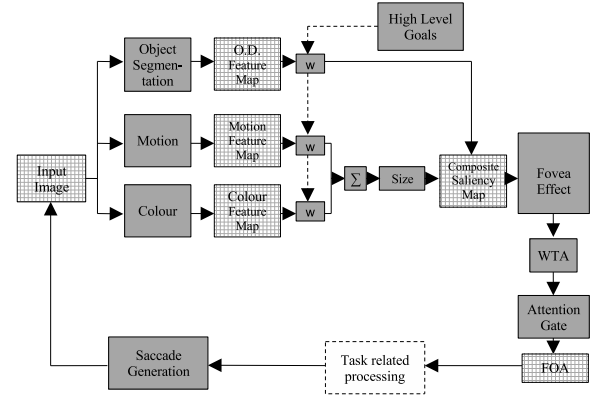


Figure 2: Our Bottom-up model of Visual Attention. The output is the Focus of Attention (FOA)

4 The Attention Mechanism

In this section we will describe the design of our attention mechanism, and the integration of both bottom-up and top-down elements.

4.1 Bottom-up

We have implemented a bottom-up model that is mainly based on Wolfe’s biologically inspired Guided Search 3.0 model of human visual attention and visual search [Wolfe and Gancarz, 1996]. This model uses Treisman’s Feature Integration theory [Treisman and Gelade, 1980] to construct a model of human visual attention. In this model, low-level filters are applied to various visual stimuli in order to produce individual feature maps in which high values indicate areas of interest.

All of the individual feature maps are weighted and then summed into a single activation map. Attention is guided to peaks in the activation map, because these represent the most salient areas in the scene. In our model, top-down task information can influence the bottom-up feature maps by changing the activation map through the modifying of the weights that are applied before the summation. Our model is shown in figure 2. There are certain features that can make objects salient. For example, brightly coloured objects are a most typical example, or if they are moving in a way that can attract attention, e.g a sudden, irregular and fast movement. Each of the bottom-up blocks in the model represents a certain feature that contributes towards the calculation of the saliency of an object. Our implementation focuses on three bottom-up blocks that are feature detectors. These are: *Motion*, *Colour* and the *Size* the object occupies in the image, which not only accounts for the actual size of an object, but also for the distance of the object from the camera, both of which are important in grabbing one’s attention.

The remaining blocks in the model are:

- Fovea-effect – this models the decrease in resolution away from the centre of the image, because our eyes’ resolution decreases dramatically with the distance from the fovea [Farid *et al.*, 2002].

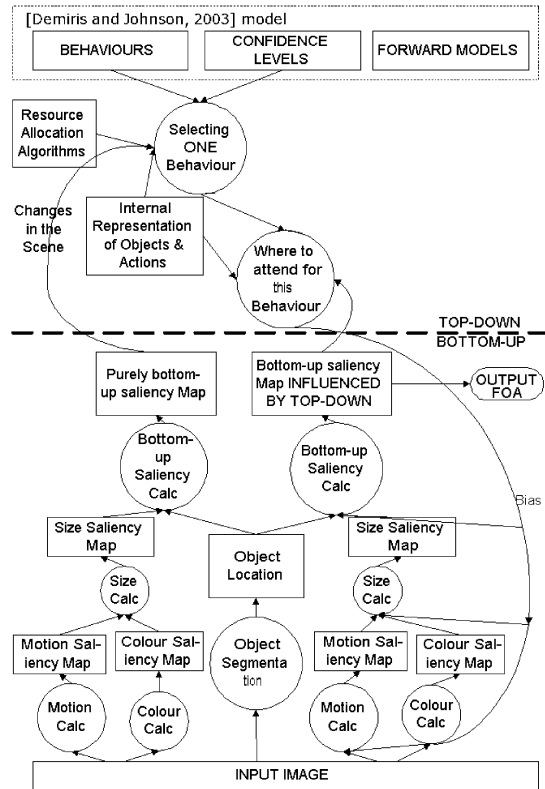


Figure 3: The architecture of the top-down part of the model and how it integrates with the bottom-up part

- Winner takes all (WTA) selection as in Itti's model [Itti *et al.*, 1998].
- An attention gate mechanism as in Wolfe's model [Wolfe and Gancarz, 1996] which keeps a record of the two most salient areas in the scene.
- A saccade generation system as in Wolfe's model [Wolfe and Gancarz, 1996].

4.2 Top-down

Figure 3 shows our complete visual attention model which includes the top-down part. This figure specifically illustrates how the top-down information could influence the bottom-up part of the model, and vice versa.

Our Top-down part of the model receives the list of behaviours from the action understanding model, described in the previous section, together with their confidence levels and their forward models. It must select only one behaviour out of the many behaviours to attend to at any given point in time. The block labelled "Selecting ONE Behaviour" in figure 3 has five inputs:

- Behaviours – A list of hypotheses (potential behaviours that explain the demonstration) is passed in, one of which must be selected.
- Confidence Levels – The current confidence levels for each behaviour.

- Resource Allocation Algorithms – we have performed experiments with different resource allocation algorithms [Stallings, 2000] that can be employed to decide on how to distribute resources between the behaviours.
- Internal Representations of objects and actions – this block gives information about objects and how they move, and interact with other objects etc.
- Purely bottom-up saliency map – this is the saliency map representing the most salient object in the scene.

From the above five inputs, one behaviour must be chosen. The block labelled "Where to attend for this Behaviour?" in figure 3 has three inputs:

- Output from "Selecting ONE Behaviour" – the winner behaviour that is selected from the previous stage is passed on.
- Internal Representations of objects and actions – this block gives information about objects and how they move, and interact with other objects etc.
- Bottom-Up Saliency map influenced by top-down – this is to give current information on where the attention of the model is.

The output of this block influences all the bottom-up calculation blocks in order to direct the attention of our model in such a way that it serves the current behaviour.

5 Experimental Setup

We implemented our model on an ActivMedia Peoplebot robot, equipped with a pan-tilt-zoom camera, as well as a two degrees of freedom gripper, sonar and infrared sensors. Only the camera was used in the following sets of experiments, and the saccade generation module was switched off as it was not needed.

For these experiments, three objects were chosen: A hand, a coke can, and an orange. Eight behaviours were then defined:

- Behaviour 1 - Pick coke can
- Behaviour 2 - Move coke can
- Behaviour 3 - Move hand away from coke can
- Behaviour 4 - Pick orange
- Behaviour 5 - Move orange
- Behaviour 6 - Move hand away from orange
- Behaviour 7 - Drop coke can
- Behaviour 8 - Drop orange

Each of these behaviours has a corresponding forward model. Figure 4 shows the arrangement for Behaviour 1.

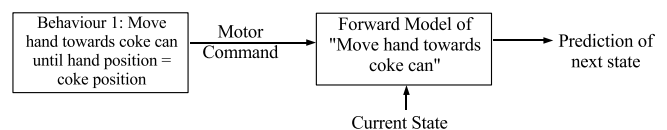


Figure 4: Behaviour1 - Picking a coke can

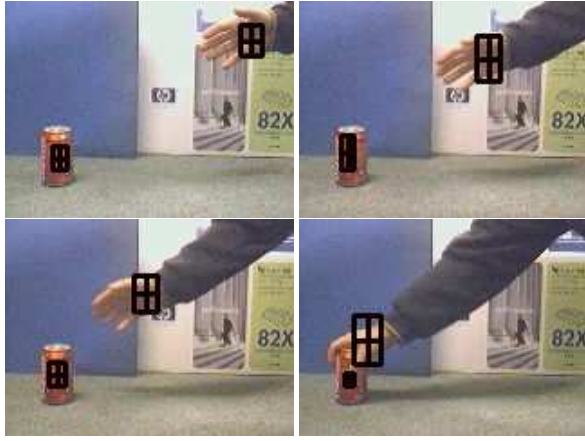


Figure 5: Images to show bottom-up block processing a scene of a hand picking a coke can

All the other behaviours are implemented in the same way. Forward models were hand coded using kinematic rules to output the prediction of the next state. The output from the forward model, which is a prediction of the actual next state, is compared with the next state. Based on this comparison, a confidence level is generated for this behaviour by either rewarding it with one confidence point if the prediction is successful, or otherwise punishing it by subtracting one confidence point.

Figure 5 shows an input example of what the robot sees when a behaviour is carried out (in this case, it is the demonstration of a hand picking a coke can). A background was chosen where these object's colours were minimally present. These are only snapshots of some frames.

The bottom-up block detects and tracks the presence of the coke can, the orange and/or the hand in the scene, depending on what the top-down part of the attention model requires. The output of this bottom-up block are the corresponding locations of where the hand, the coke can and/or the orange are in the scene. This information is then passed to the top-down part of the model for intelligent processing. The CAMShift algorithm [Bradski, 1998] was used to assist in doing this. We used a hue and saturation histogram back-projection of camera images taken at a pixel resolution of 160×120 and at 30 frames per second. The corresponding histograms of the three objects used in our experiments were pre-saved into the robot's memory, and used during the experiments as a simple method of object recognition.

Four different implementations were experimented with each of the eight behaviours:

- A pure implementation of the action understanding model without our attention model. Hence there was no layer of intelligence to cut down on computational costs, i.e. each behaviour gets to carry out all the computations it requires at *each* frame.
- Our Attention model is added to the action understanding model using a “round robin” scheduling algorithm (equal time sharing) [Stallings, 2000] to select between the behaviours. Therefore, each behaviour is processed

every eighth frame, since there are eight behaviours in total.

- Our Attention model is added to the action understanding model using the strategy of “highest confidence level always wins”, which means the behaviour with the previous highest confidence level gets the next computation.
- Our Attention model is added to the action understanding model using a combination of the “round robin” and the “highest confidence level always wins” strategies to select between the behaviours.

Finally, we also ran another set of experiments by adding to these implementations initialisation for the top-down part using bottom-up saliencies.

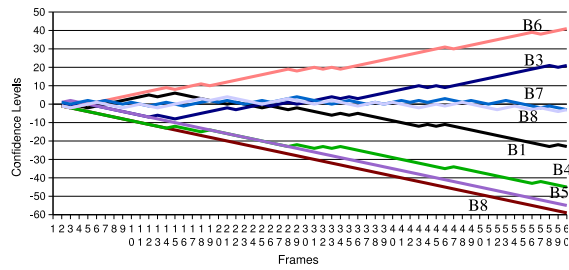
6 Experimental Results

We used 26 different videos, performing 130 experiments, using the different scheduling algorithms above, while varying some of the model's parameters. The results for behaviour 6 are shown in figure 6 as an example of a typical output to demonstrate how our system can work on top of the action understanding model, cutting down its computational costs by up to 87.5% (because every behaviour is now only being processed *once in every 8 frames*), and still producing the correct results to determine which behaviour is being demonstrated. But more importantly, it directs the limited computational resources to the relevant areas in the scene, instead of analysing the whole scene.

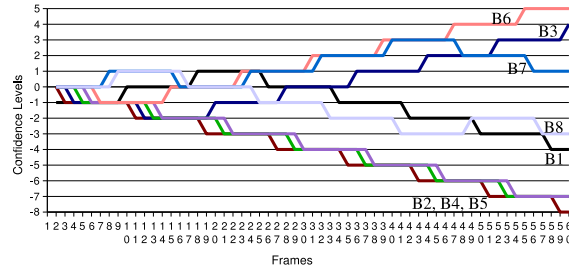
In addition to this, the above results were substantially improved by adding initialisation to the top-down part using the bottom-up saliencies to our attention model. Therefore, in the case of a scene where the orange does not exist, as shown in figure 5, our bottom-up block would detect the coke can and the hand as being salient objects. Using previously saved colour histograms, our system recognises that these salient objects are a coke can and a hand. This result is passed to the top-down part of our attention model, which in turn will only select the behaviours that are involved with these objects, as opposed to previously selecting every behaviour that exists in the database.

Results from behaviour 1, where no orange was present in the scene, are shown in figure 7 as an example for this initialisation process of only selecting the relevant behaviours using the bottom-up saliencies. The results are compared to the previous implementation without this initialisation. It can be seen that this initialisation process speeds up the correct recognition of the correct behaviour. Furthermore, it will also serve in allowing scalability to be added to our model.

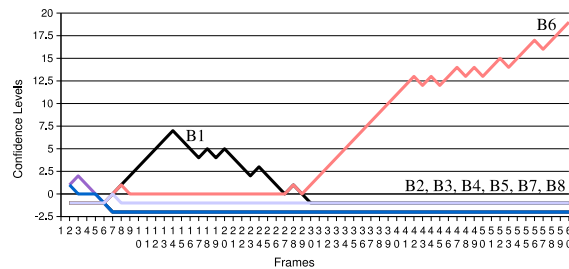
As can be seen from figure 6, our attention model not only still gives the correct results for the recognition of behaviour 6, but it does it with the saving of up to 87.5% of the total computational costs. Furthermore, it returns better results in recognising the correct behaviour by isolating it from the other wrong behaviours, due to the focusing on the correct areas of the scene only, instead of the entire image. These successful results were also seen for all of the other seven behaviours that were tested.



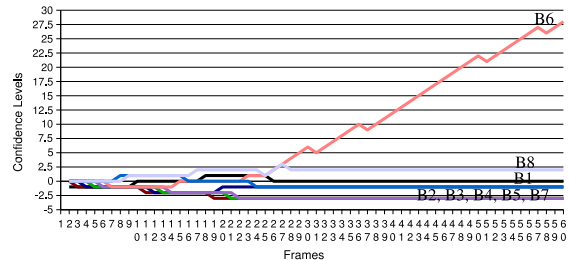
Each behaviour is processed every frame



Behaviours are processed according to Round Robin scheduling algorithm



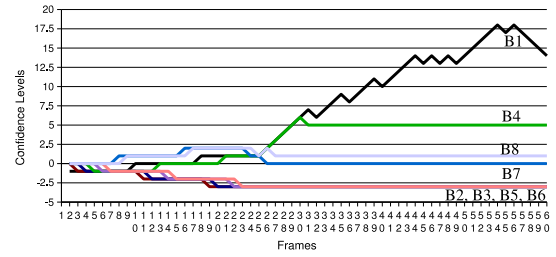
Behaviours are processed according to the highest confidence level



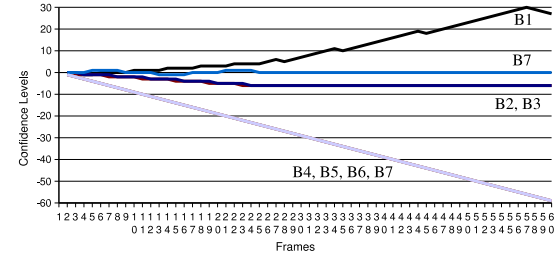
Behaviours are processed according to Round Robin at the start, and then according to the highest confidence level



Figure 6: Behaviour 6 - Move hand away from orange



Behaviours are processed *without* the bottom-up block initialising the top-down



Behaviours are processed *with* the bottom-up block initialising the top-down

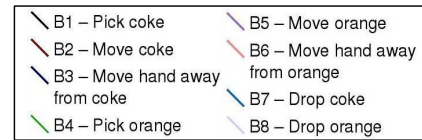


Figure 7: Behaviour 1 - Pick coke can

Behaviour 6 in figure 6 shows that when a “round robin” scheduling algorithm is applied, the correct behaviour is still picked, but it ends up with a lower confidence value that is quite close to the other resulting confidence values of the other behaviours. The reduction in computational costs using this algorithm has resulted in a decrease on the separability of the behaviours. This is because there are n frames in the demonstrated scene (in these experiments, n is 60), and each behaviour is only processed once every m frames, where m is the number of behaviours (in these experiments, m is 8) meaning a total number of n/m computations per behaviour (which is 7 in these experiments, hence, only a maximum score of 7 for the winning behaviour). Behaviour 6 in this case scores 5 out of 7, still enough to make it the winning behaviour, but much lower than the pure implementation of the action understanding model without the use of any attention mechanism.

When “highest confidence level always wins” is used, the opposite effect to “round robin” can be seen: attention acts like an accelerator to the winning behaviour once it recognises who is the winner, suppressing all the others. The problem with purely using the scheduling algorithm of “highest confidence level always wins”, is that it may not always initialise correctly, as can be seen in figure 6, hence taking some time before converging on the correct behaviour.

To alleviate this problem, we used the “round robin” scheduling algorithm as an initialisation step for the “highest confidence level always wins” which then acts as an ac-

celerator for the winning behaviour. Hence, “round robin” is applied only for half the length of the demonstration (30 frames in these experiments). Then the “highest confidence level always wins” is applied as an accelerator to the winning behaviour. As can be seen from figure 6, it returns accurate and fast results for selecting the winning behaviour. This process may be seen as being equivalent to “playing it safe” at the beginning by looking everywhere, because we are not yet sure what is happening. But once we start to understand the scene better, we are more confident to focus on the relevant parts and ignoring the rest.

Results from figure 7 show significant improvements when using the bottom-up saliencies to initialise the top-down selection of the relevant behaviours. Hence as a result, only the relevant behaviours are picked for processing, instead of picking all the behaviours in the robot’s database. In this experiment, the robot has 8 behaviours in its database, 4 of which involve a hand and a coke can, and the other 4 involve a hand and an orange. In the demonstrated video, the orange is not present, hence, all the 4 behaviours involving the orange, are immediately punished and not processed. The remaining 4 behaviours (instead of the total 8) are processed with the “round robin” scheduling algorithm at the beginning. Great improvement can be seen here in finding the correct behaviour sooner, at frame 10 instead of the previous frame 31. This can be thought of as enabling our model to recognise the correct behaviour being demonstrated easier and quicker in a less complicated scene with fewer objects.

7 Conclusion

Our attention model utilises both, resource scheduling algorithms, and initialisation of the top-down mechanism to cut down on the computational costs. We have found that a combination of the “round robin” and the “highest confidence level always wins” strategies, together with using the bottom-up saliencies for initialising the top-down selection of the relevant behaviours to the demonstrated scene, worked very well.

The computational costs needed to run our attention model are justified since the action-understanding model is aimed at having a large number of behaviours. As the number of behaviours increases therefore, the resultant search space makes the model indispensable. This is especially because the savings on the computational costs will also increase by $(n - 1)/n$ for n behaviours.

We are working towards further optimisation of our model by considering optimal path algorithms. For example, if the sequence of planned allocations involves looking at the head, feet, and arm of a human demonstrator, the algorithm will try to minimise the saccade required and rearrange the allocations to accommodate that.

Optimisation based on predicting future requests will further enhance the performance of our model, and will be our next step in our investigation between attention and action understanding, with the ultimate goal of having robots that will efficiently understand our human actions.

Acknowledgments

This research was funded by the Donal W Morphy Trust Scholarship and by the UK Engineering and Physical Sciences Research Council and The Royal Society. The authors would like to thank the BioART team for their support and assistance, especially Matthew Johnson who assisted in the implementation of the camshift tracker.

References

- [Bradski, 1998] G. R. Bradski. Computer vision face tracking for use in a perceptual user interface. *Intel Technology Journal*, Q2:1–15, 1998.
- [Breazeal and Scassellati, 1999] Cynthia Breazeal and Brian Scassellati. A context-dependent attention system for a social robot. In *IJCAI '99: Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence*, pages 1146–1153, San Francisco, CA, USA, 1999.
- [Demiris and Johnson, 2003] Y. Demiris and M. Johnson. Distributed, predictive perception of actions: a biologically inspired robotics architecture for imitation and learning. *Connection Science Journal*, 5(4):231–243, 2003.
- [Farid et al., 2002] M. M. Farid, F. Kurugollu, and F. D. Murtagh. Adaptive wavelet eye-gaze-based video compression [4877-32]. *Proceedings- Spie the International Society for Optical Engineering*, (4877):255–263, 2002.
- [Itti et al., 1998] L. Itti, C. Koch, and E. Niebur. A model of saliency-based visual attention for rapid scene analysis. *Ieee Transactions on Pattern Analysis and Machine Intelligence*, 20(11):1254–1259, 1998.
- [Koch and Ullman, 1985] C. Koch and S. Ullman. Shifts in selective visual attention: towards the underlying neural circuitry. *Human Neurobiology*, 4:219–227, 1985.
- [Stallings, 2000] William Stallings. *Operating systems : internals and design principles*. Prentice Hall, 4th ed edition, 2000.
- [Treisman and Gelade, 1980] A. Treisman and G. Gelade. A feature-integration theory of attention. *Cognitive Psychology*, 12(1):97–136, 1980.
- [Tsotsos, 1989] J. K. Tsotsos. Complexity of perceptual search tasks. In *Proceedings of the 11th Int. Joint Conf. on Artificial Intelligence*, pages 1571–1577, 1989.
- [Tsotsos, 2001] J. K. Tsotsos. Motion understanding: Task-directed attention and representations that link perception with action. *International Journal of Computer Vision*, 45(3):265–280, 2001.
- [Van Essen et al., 1991] D.C. Van Essen, B. Olshausen, C.H. Anderson, and J.L. Gallant. Pattern recognition, attention, and information bottlenecks in the primate visual system. *Proc. SPIE Conf. on Visual Information Processing: From Neurons to Chips*, 1991.
- [Wolfe and Gancarz, 1996] J. M. Wolfe and G. Gancarz. Guided search 3.0. In *Basic and Clinical Applications of Vision Science*, pages 189–192. Kluwer Academic Publishers, Dordrecht, 1996.

Collective AI: context awareness via communication

S. Kornienko, O. Kornienko, P. Levi

Institute of Parallel and Distributed Systems, University of Stuttgart,
Universitätsstr. 38, D-70569 Stuttgart, Germany

Abstract

Communication among participants (agents, robots) is central to an appearance of Collective AI. In this work we deal with the development of local communication mechanisms for real microrobotic swarms. We demonstrate that despite of very limited capabilities of the microrobot, the specific construction of communication hardware and software allows very extended collective capabilities of the whole swarm. We propose mechanisms providing information content and context for collective navigation, coordination and spatial perception in a group of microrobots.

1 Introduction

Communication is the central issue in collective systems such as collective/swarm robotics, multi-agent systems, sensor networks and so on. Communication provides and supports (among others) "awareness" about relevant events, collective decision making and coordination in a group, execution of cooperative activities, etc. Due to communication all robots/agents behave in a coordinated way like one "organism", they can even emerge new behavior types [Kornienko *et al.*, 2004]. There are many phenomena that appear in this way; in the vast literature they are denoted as collective/swarm intelligence [Bonabeau *et al.*, 1999].

In this paper we investigate the communication mechanisms for large microrobotic swarm [I-Swarm, 2003 2007]. Due to very small size and limited capabilities of microrobots, swarm robotics differs from other collective robotic systems in a couple of essential points [Sahin, 2004]. The most important are inaccessibility of global coordinates, global perception and global communication. Despite of the "very limited intelligence" of individual microrobots, the collective intelligence of the whole swarm does not change essentially. The swarm is still able for distributed spatial sensing, collective decision making, building spatial formation, coordinated acting and so on. This swarm intelligence primarily appears due to specific communication between robots. We are interested in the following question: *"Which communication mechanisms among microrobots do allow emerging collective properties of a swarm ?"*

Generally for AI and especially for "swarm intelligence" not only the transferred message is important ("I find something"), but also the context of the message ("where is it found") [Doyle and Dean, 1996]. One robot cannot provide this context, because e.g. it does not know its own position. Retrieving and providing this context represents also the collective task performed during communication. However not all communication approaches can provide the context of information and can generally be implemented in a swarm. In this paper we demonstrate that the well-known package-based routing is not useful for swarms and suggest instead an approach that "diffuses" information with its context.

Achieving collective capabilities in real microrobotic systems, we are very limited by hardware. Therefore not all approaches from AI domain are feasible here. The second question of this paper is *"Which collective capabilities a swarm are feasible by very limited communication hardware ?"* We demonstrate that the specific composition between multi-directional communication hardware and "diffusing" software protocols allows emerging some interesting spatial and functional collective capabilities of a swarm.

The rest of the paper is organized as follows. In Section 2 we investigate the local communication between robots. Sections 3, 4 and 5 are devoted for describing the hardware platform, logical protocols and communication context. Finally, in Section 6 we discuss some preliminary experiments and conclude our work.

2 Information diffusion, swarm density and communication radius

For collective systems a communication plays the role of nervous system in human body. Since microrobots in a swarm can communicate only locally with their neighbors, such a "swarm nervous system" can be produced only by a mechanism that propagates information through multiple robot-robot connections. Parameters of a global circulation of information (like global propagation speed or global propagation time) depend on characteristics of local communication (communication radius R_c , the number of robots within R_c). In this section we derive this relation, which is necessary for further development of the robot's communication hardware.

Since parameters of local communication between robots depend on their behavior, we differentiate three following be-

havioral cases:

1. Robots move only in small areas, so called clusters. In this case robots are situated more or less closely to each other, so that swarm peer-to-peer network (SPPN) is created "automatically". The main problem is a communication between such clusters.
2. Robots move in large areas (typical swarm scenario). Robots exchange information only when they meet each other. The inter-cluster communication belongs to this case.
3. Part of robotic swarm purposely creates and supports the SPPN. This is the most interesting case, that provide stable communication in swarm (see more in Section 6).

In the further calculation we consider the most hardest case of a large-area swarm. We can intuitively assume that the communication radius R_c , the swarm density D_{sw} , the robots motion velocity v and the time t are closely related in propagating the information. For deriving a relation between them, we take several analogies to molecular-kinetic theory of ideal gas, more exactly diffusion in ideal gas (by these analogies we denote also a "diffusion of information"). We introduce the following notions: the sensor radius R_s , where a collision-avoiding procedure is started; l_c the length of free path from the start of motion till the first communication contact; l_s the length of free path from the start of motion till the first collision-avoiding contact; n_c and n_s are correspondingly the number of communication and collision-avoiding contacts; S_c and S_s are the area of the "broken" rectangles built by a motion in some time interval t with R_c and R_s . In Figure 1(a) we sketch our consideration. Firstly, we are in-

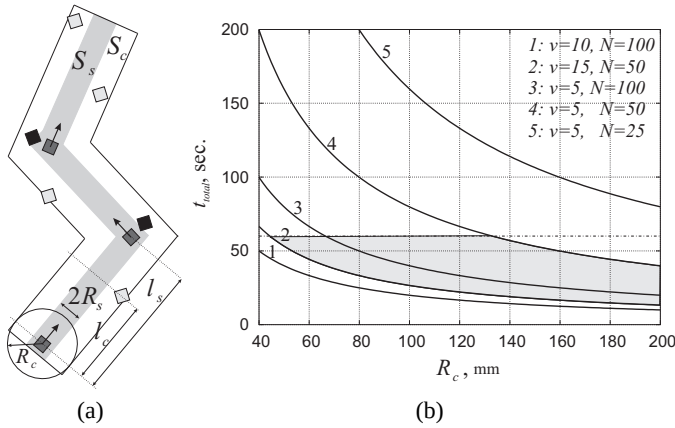


Figure 1: **(a)** Motion path of a robot with communication and collision-avoiding contacts; **(b)** Total propagation time t_{total} as a function of communication distance R_c with different values of velocity v and the number of robots N .

terested in the number of communication contacts n_c happen during the motion. This value is equal to the average number of robots in the area S_c , $n_c = S_c D_{sw}$, where D_{sw} is the swarm density. We assume that the collision avoiding radius and the robot's rotation radius are small so that we can neglect the area of fractures. In this case $S_c = 2R_c vt$. D_{sw} can be calculated as the number of robots N in swarm divided by

the area available for the whole swarm S_{sw} :

$$D_{sw} = \frac{N}{S_{sw}} \rightarrow n_c = \frac{2R_c vt N}{S_{sw}}. \quad (1)$$

In the relation (1) we assume only one robot moves whereas other are motionless. More exact relation, when all robots move, differs from (1) only by the numeric coefficient $\sqrt{2}$ (as proved by Maxwell for a diffusion in ideal gas). For the further calculation we use $n_c = \frac{2\sqrt{2}R_c vt N}{S_{sw}}$. Now we have to estimate how the information will be propagated after the first communication contact. This propagation dynamics is similar to "epidemic infection" dynamics, estimated as the series:

$$[n_c + 1] + n_c[n_c + 1] + n_c[n_c + 1 + n_c(n_c + 1)] + \dots \quad (2)$$

and written iteratively as

$$k_n = n_c k_{n-1} + k_{n-1} = k_{n-1}(n_c + 1), \quad k_0 = 1. \quad (3)$$

that is the "standard" exponential form $(n_c + 1)^n$. We are interested in the case when all robots are "infected" $(n_c + 1)^n \geq N$ or $n = \log_{(n_c + 1)} N$. From real experiments we know that for establishing a communication contact and transmitting messages, robots need some time p_t , that can be measured experimentally. The information transfer starts when the first robot "infects" one additional robot ($n_c = 1$); the time till the first infection t_{first} and the total time $t_{total} = n t_{first} + N p_t$ for infecting the whole swarm can be obtained as:

$$t_{first} = \frac{S_{sw}}{2\sqrt{2}R_c v N}, \quad t_{total} = N p_t + \frac{S_{sw}}{2\sqrt{2}R_c v N} \log_2(N). \quad (4)$$

In the performed simulations ($p_t = 0$), the swarm areal is $800 \times 650 \text{ pixels}^2$, $N = 50$ with $D_{sw} \approx 10 \text{ pix./sec.}$, $R_c = 40 \text{ pix.}$ Formula (4) gives us $t_{total} \approx 52 \text{ sec.}$ In many performed simulation cycles we observed t_{total} between 30 and 90 sec. Formula (4) is also useful in estimating the energy needed for each robot. For example, swarm during the running time has to propagate 100 different messages; it takes about 2 hours in the mentioned example. So the power supply should provide energy at least for 2 hours.

For developing a real microrobotic swarm we can take $S_{sw} = 1000 \times 1000 \text{ mm}^2$, $N = 50$ and assume first $p_t = 0$ (see Section 6 for the real p_t). In Figure 1(b) we plot t_{total} depending of R_c with different values of N and v . We see, that for the average propagation time 1 min, the R_c for $N = 50$ lies between 50 mm and 140 mm. Thus, for the targeted robots body of 23 mm, the communication radius R_c is of 4-5 times larger then the size of the robot.

At the end of this section we discuss such an important point as the critical swarm density D_{sw}^{crit} . The critical swarm density and the "coefficient of swarm efficiency" (the relation between the number of robots with useful/desired and useless/undesired activities) determine the minimal number of robots N_{min} in some areal S_{ws} required to perform some operation successfully. For the considered example with the given $S_{sw} = 1000 \times 1000 \text{ mm}^2$, $t_{total} = 30 \text{ sec.}$, $R_c = 100 \text{ mm}$ and $v = 20 \text{ mm/sec.}$ (related to the random motion), the minimal number $N_{min} \approx 29$ and the critical

swarm density $D_{sw}^{crit} = 28.46^{-6}$. This relation is not exact, because it does not involve the size of a robot into this calculation, however in the microrobotic case with $S_{robot} \ll S_{sw}$, it can serve as a good approximation.

3 Requirements and restrictions imposed on communication in real swarms

The requirements concern choosing the transmission equipment, the number of directional communication channels, communication radius and the hardware reduction of communication deadlocks. The communication equipment of a microrobot should consume as less energy as possible and be of small size (the robot's size is 23x23x28 mm). Finally, the communication equipment should include, as far as possible, other functions, like proximity or distance measurement. The communication radius $R_c = 50 - 140$ mm can be implemented in the radio-frequency (RF) and infrared (IR) way.

The RF provides two-way communication channel, the communication radius R_c is of several meters and modern one-chip RF modules, even 802.11b/802.11g modules, consume energy in mW area. However we have a serious objection against RF in microrobotic swarms. Firstly, a simultaneous transmission of many (80-150) microrobots leads to massive RF-interferences. Secondly, the RF-system (with the large R_c) transmits local information exchange between robots globally in a swarm. This local information does not have too much sense for all robots, so that we have high communication overhead in this case.

The IR communication is recently dominant in the so-called small-distance-domain, as e.g. for communication between laptops, hand-held devices, remote control and others. In IR domain we can choose between several different technologies, like IrDA¹, 34-38 Khz PCM-based devices and so on. Additional advantage of the IR-solution consists in combining communication equipment with sensors; we can think about proximity or distance sensors on the base of IR reflection. The IR-solution is not new in robotic domain (see e.g. [Kube, 1996], [Suzuki *et al.*, 1995]), however there are almost no solutions that combine perception, proximity sensing and communication.

The IR-equipment has also the problem of interferences. They appear, like in RF case, when several neighbor robots transmit simultaneously. The problem of IR-interferences can be avoided by restricting an opening angle of a pair IR-receiver-transmitter. For four communication channels, the opening angle of each channel is 90°. In this case we have 2- and 3-robots IR-interferences even in the "closest" radius (50 mm). Reducing the opening angle to 60° or to 40° allows avoiding IR-interferences in the "close" and "near" radius (100 mm) (Figure 2(a)). Since many microcontrollers have 8-channel ADC (one ADC input is used by the distance sensor), we choose 6-channel directional communication (Figure 2(b)).

Directional communication is extremely important in a swarm also from another reason. The point is that a robot

¹IrDA requires additional chips, and if we think about 4-6 channels communication, this solution is not really suitable for the implementation in microrobots.

has to know not only a message itself, but also the context of this message (e.g. the direction from which the message is received, intensity of signal, communicating neighbor and so on). Without directional communication hardware, we cannot implement algorithms providing a spatial context. From many software requirements the communication radius R_c and the number of directional communication channels are the most important ones. From this viewpoint, the IR is more suitable for robot-robot communication than the RF. The host-robot communication can also be implemented with IR (a sensor with PCM-filter for receiving global modulated signal). Such a signal can be thought as of a remote control or a global information exchange between robots and host.

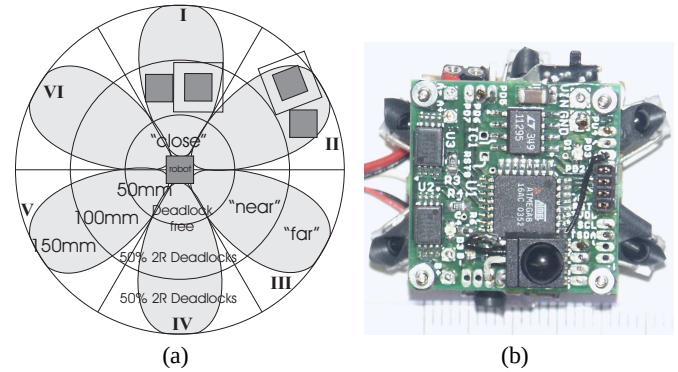


Figure 2: (a) Problem of IR-interferences in the "close", "near" and "far" communication zones; (b) The sensors board with Megabitty board that supports 6-x directional robot-robot and host-robot communication, proximity sensing and perception of surfaces geometry.

Speaking about IR communication we have to mention the problem of ambient light. It represents generally a very critical issue, because it can essentially distort or even completely break IR-communication/sensing. We performed experiments with luminescent lamps, filament lamps and daylight. Even for IR-receivers with daylight filter, swarm has to be protected against a light of filament lamps. As far as possible, the direct daylight should be also avoided. Use of modulated light can essentially improve communication against ambient light, however this solution is not always acceptable/feasible.

The filament lamps can be used as a global pheromone to control a swarm [Bonabeau *et al.*, 1999]. When it is emitted simultaneously with luminescent light, the robot reacts more intensively on filament light. This effect can be utilized in many purposes, like finding the food source, navigation or even a quick message about some global event. This communication way does not require any additional sensors, however should be used only as an exception, because it essentially distorts a regular communication.

In the following we briefly describe the developed hardware solution for the directional IR-communication and sensing. More details for hardware can be found in [Kornienko *et al.*, 2005]. In the hardware we do not use such popular sensors as IS471F or Sharp's GP2Dxxx with binary output, because they do not assume active control needed for com-

munication. We encountered that small integrated transistor-diode pairs like SFH9201, TCNT1000, TCRT 1000/1010, GP2D120, QRB1134 are not suitable as distance, proximity and communicating sensors for R_c of 130-150 mm. There are also several problems with spectral matching of some receiver-emitter pairs, despite they use the same wavelength. In the tested phototransistors with 60° angle, we choose TEFT4300 (60° , collector light current 3,2 mA, 875...1000 nm), TSKS5400-FSZ as IR-emitter for proximity measurement and communication (60° , 950 nm, 2-7 mW/sr) and GaAs/GaAlAs IR-emitter TSAL6100 (20° , 950 nm, >80 mW/sr) for distance measurement. This pair is very small (emitter $5 \times 5 \times 2.65$ mm and receiver $4,5 \times \phi 3$ mm) so that they can easily be integrated in the sensors board. In experiments the current I_F of IR-emitters was limited to 20 mA, that corresponds to I/O port of the microcontroller. For controlling we use the Megabitty board ($23 \times 23 \times 2$ mm) with Atmel AVR Mega 8 microcontroller with 8 kB ROM and 1 kB RAM [Megabitty, 2005]. The sensors board and Megabitty board are assembled in one chassis with accumulators and 2 DC motors as shown in Figure 3. The tested communication

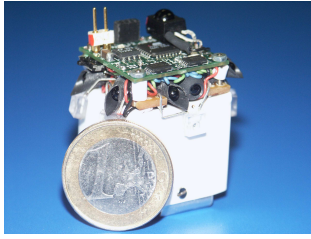


Figure 3: The prototype of the "Jasmine" microrobot.

speed is about 1000 bit/sec with low error rate, so that an application of error-correction approaches is not required.

4 Communication and Intelligence: logical protocols

After a description of the communication hardware of the microrobot, we return to our original question about Collective Intelligence. The question is "which degree of collective intelligence is still feasible in the group of microrobots?" and "How to implement it?" In Table 1 we collect some "swarm activities" that microrobots can collectively perform. These collective activities build a basis of swarm intelligence. We take the most simple example of spatial orientation. Let us assume, a robot has found a "food source" being relevant for the whole swarm. It sends the message "I, robot X, found Y, come to me". When this message is propagated through a swarm, each robot knows there is a resource Y at the robot X. However robots cannot find it because they do not know a coordinate of this "food source". The robot X cannot provide these coordinates because it does not know its own position. Therefore even for the simple collective feature of spatial orientation, a **local context of messages determines a global capability of the whole group**. Collective systems often have many different contexts, so that we have a context hierarchy.

Context	N	Swarm Capability
Spatial	1	Spatial orientation
	2	Building spatial structures
	3	Collective movement
Information	4	Building informational structures
	5	Collective decision making
	6	Collective information processing
	7	Collective perception/recognition
Functional	8	Building functional structures
	9	Collective task decomposition
	10	Collective planning
	11	Group-based specialization

Table 1: Some collective activities performed by the whole swarm.

The main point of this work is that the required context can be processed/provided by communication. However which level of communication can do it? After hardware level, there are four such levels: level of physical signal transmission, level of communication protocols and level of informational structures, that require communication. In swarm-based systems we have the additional level concerned to the robot's behavior for creating and supporting required communication.

1. On the level of physical transmission, the problem of communication is related to a choice of modulation/transmission approaches suitable for the IR-based signal transmission. On this level such properties of signals as strength, IR-interferences, directions can be extracted and incorporated into high-level protocols (it is closely related with the robot embodiment).

2. Level of communication protocols concerns the propagation of information in a swarm. Generally, there are only two main ways of such a propagation:

- each robot routes communication packages from other robots without any changes (package-based communication);
- each robot processes the information from other low-level packages and sends only its own messages further.

In the package-based communication each package consists of a header with IDs of sender and receiver, routing information and the package content. The package ID can be coded by 10 bits, IDs of sender/receiver by 12 bits (6 bit each), so the header is of 22 bits, the package content is only of 8 bits. For recording the package history each robot needs about 900 bytes RAM only for routing 300-600 packages within a few minutes ($N = 50$ robots, each sends max. 1-2 messages each 10 sec, propagation time of 1 min.). In order to use the (spatial) context of message (e.g. the spatial location of the sender), the robots can follow the propagation way by using ID-history. However since all robots are continuously moving, the propagation way does not exist a long time. In the simulation, when a particular robot tries to achieve the source of a message by following the propagation way, it fails in 80-90%! After many experiments we came at the conclusion, that pure routing is not really suitable for propagating information through a swarm (however package-based communication is used for local communication between neighbor robots). Thus, the second approach represents the main way of

incorporating the information context into communication.

3.4. Levels of informational structures and specific collective behavior belong to the high level of information processing in a swarm. These levels deal with optimal representation of information, a minimization of communication flow, availability of information and supporting multiple peer-to-peer connections for a large-distance information transmission. We demonstrate some ideas in Section 6, but generally, these levels overstep the framework of swarm-based communication.

5 Diffusion of messages context

As presented in the previous section, the context of message cannot be extracted from the message itself. This point has been discussed many time in collective AI community. In "AI world" there exist some approaches to retrieve the required context, however the microrobots are too limited to use them. Our proposal is that robots work with communication context during communication.

There are many different approaches to work with messages context. One of them is to incorporate the embodied information (signal intensity, direction, neighbors) into non-routed packages. The robots during "normal" communication process this context, so that it diffuses over a swarm. In this way a specific collective activity can be coordinated/created/controlled (see Section 6).

Another approach is a pheromone-based communication, well-known in natural [Bonabeau *et al.*, 1999] as well as in technical/robotic systems [Payton *et al.*, 2001]. Pheromone-based communication can be divided into two main groups: with pheromone leaved on immovable objects (ground, floor and so on) and pheromone leaved on moving objects like robots. Whereas the first type of pheromone assumes usually real (physical) pheromone, like chemical substances or electromagnetic marks, the second type of pheromone can also have some virtual nature. For instance, robots exchange the values of some variables, these variables are "located" on a robot and we can speak about "virtual pheromone".

Basic idea of pheromone-based communication is quite simple. Let assume that the information source, robot X, sends a message, say "I found Y". This message is binary, however the robot X represents it by some integer value. This value is maximal at the origin. Any other robot, when getting this value, subtracts some constant and sends it further. In this way, the far away from the source the value is propagated, the less is its intensity. Based on this gradient every robot can conclude about the source Y and its origin (Figure 4). In this way not only a content of information ("something is found") is propagated, but also a spatial context (spatial origin of this "something"). More generally, different temporary, spatial or functional context can be provided by this "field".

Independently of the implemented mechanism, the "diffusion field" can be of four different types: non-gradient (used simply for transmitting some signals), gradient (to provide spatial context of a message), oriented (some specific direction), functional (e.g. repelling or attracting). The values of this field can be calculated as a function of connectivity (the number of neighbors, see [Nembrini *et al.*, 2002]), time, spe-

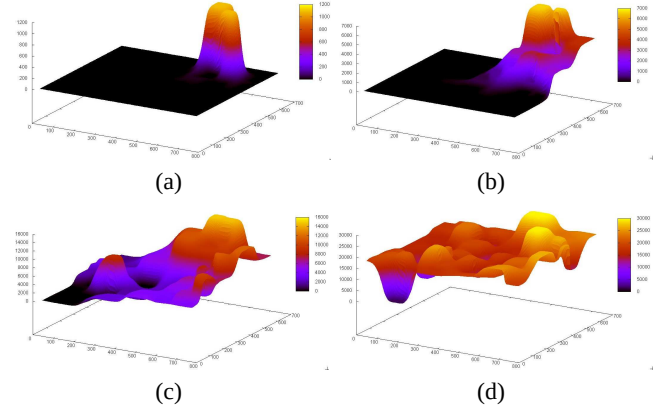


Figure 4: Propagation of the pheromone field from the initial to the final states. (a) Initial state of pheromone field; (b,c) Intermediate states where propagation of field is not finished; (d) Final state of pheromone field, where all robots get the message and know its spatial origin.

cific input (e.g. only robots that see something transmit a pheromone), embodied information. Diffusion field can consist of many different subfields, i.e. with hierarchical structure.

1. Diffusion of the size context. The diffusion field is a function of the connectivity degree: $\Phi_{n+1} = f(\sum_i \Phi_n^i)$ where i goes over all local neighbors. The more large is the group of robots, the more higher is the value, so that a context is the size of the whole group.

2. Diffusion of the spatial context. The source emits a constant value. All other robots subtract some constants C_i from this value (see Figure 4): $\Phi_{n+1}^{source} = C_1$, $\Phi_{n+1} = f(\sum_i \Phi_n^i) - C_2$ and transmit it further. Disadvantage of this relation is that robots can move in clusters, so that we can have local maximums of the diffusion field. Instead we can use $\Phi_{n+1} = f(\max(\Phi_n^1, \dots, \Phi_n^i, \dots)) - C_2$. In this case the clusterization effect is removed (however it cannot be completely removed from a swarm).

3. Diffusion of the directional context. As already mentioned, robots support directional communication. The source emits a specific signal only in one direction. All other robots transmit this signal also only in one direction, as opposite to a receiving direction (received on "north", send to "south"). In this way, "communication streets" appear, that can be used for e.g. navigation.

4. Diffusion of the temporal context. The diffusion field is a function of time: $\Phi_{n+1} = f(\max(\Phi_n^1, \dots, \Phi_n^i, \dots)) - f(t)$. This can be useful for coordinating some temporary event (activities) in a swarm.

5. Diffusion of the activity context. This kind of field transmits a stimulus for a specific activity. Since all robots are heterogenous, a robot can need an assistance of only specific robots (with some specific functionality). Field can have a gradient and non-gradient character.

6. Multiple diffusion. The context, especially spatial one, can be useful not only for information transfer, but also for many other spatial operations like navigation, localization

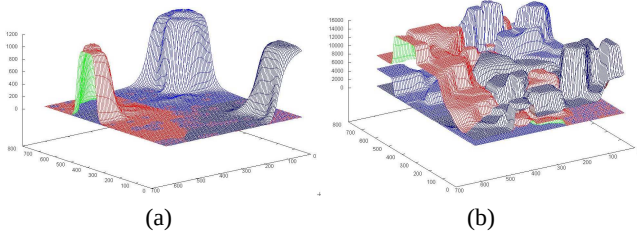


Figure 5: **(a)** Initial location of 3 different field sources; **(b)** Final distribution of 3-fields.

and so on. In the most simple form there are two or three field sources that are propagated in a swarm (Figure 5). Three fields are more preferable, because in this case robots can perform triangulation, like GPS.

6 Preliminary experiments and discussion

We performed two series of preliminary experiments with a small group of microrobots "Jasmine" and the goal of collective perception and spatial information processing (Figure 6). Some parameters and conditions of experiments and imple-

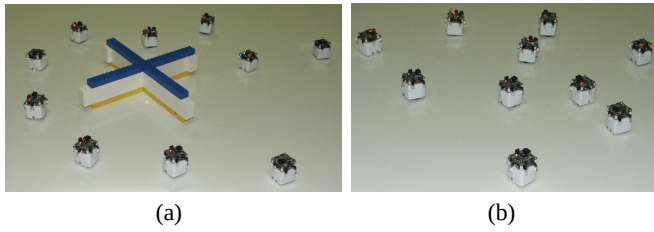


Figure 6: Preliminary experiments with 10 microrobots; **(a)** Collective perception; **(b)** Building a "communication street".

mented local protocols are collected in Table 2. The main

Parameter	Value
Swarm areal	$\sim 3.5m^2$, white plastic covering, only with a luminescent light
Signal transmission	PCM, 2 ms - logical "1" 1 ms - logical "0", 1 ms - pause
Logical protocols	with and without confirmation
Coding/decoding	Atmel's 16 MHZ Timer, divider 1024
Low-level packages	8 bit with one parity bit
Sending/receiving	sequential on all channels sending - 38 ms on one channel listening - 3 ms on one channel

Table 2: Some parameters and conditions of the experiments.

problem we encountered is a poor probability of bidirectional communication contact in the prototype's multi-channel communication system. The reasons for that are:

- **appearance of communication- dead- zones** (primarily corners of the chassis) and the problem of emitter-receiver optical isolation that additionally increases these zones (they are

different at emitters and receivers). We estimate that in average $\sim 10\% - 15\%$ of the 360° communication areal is lost;
- **nonlinear radiation patterns** (Figure 2(a)). For bidirectional communication contact, both radiation patterns have to match. Comparing to one-directional communication, the probability of bidirectional contact on any arbitrary channel is 0.5-0.25 (according to the communication distance);
- **the microrobot can send and receive only sequentially** by all channels. In order to send a message, sending and receiving channels have to be "synchronized" (the number of a "sending" channel has to correspond to the number of a "listening" channel). The probability that both channels "meet" is $1/6 \cdot 1/6 = 1/36$.

As shown in Table 2, sending on one channel continues $\sim 38ms$ for 8 bit package and is repeated each 10 – 100ms (depend on the currently executed activities). With the probability of 1/36, the communication contact will be established within $p_t = \sim 1s - 1,5s$ and a transmission of message (without confirmation) with 10 robots takes $Np_t = \sim 10 - 15s$. The transmission of messages with the confirmation protocol takes 20 – 60s. These data are confirmed by experiments [Pradier, 2005], [Fu, 2005], as shown in Figure 8(b) and Table 3. To improve matching of nonlinear radiation patterns, the robots can slowly rotate during "looking/listening phase". In Table 3 we demonstrate the results of these experiments for 6 robots (from [Fu, 2005]). Generally, we are

Receiver	Sender	Propagation Time
—	6° per cycle	not always stable
6° per cycle	—	stable, 7.36 s.
6° per cycle	6° per cycle	stable, 5.98 s.

Table 3: Rotation of robots during "looking/listening phase".

developing the second version of the sensors board, where at least a part of problems will be solved in hardware way.

The first series of experiments concerns collective perception. For that, all robots surround an object and scan the corresponding object's surfaces (Figure 7). The scan data provide information about surface's geometry and allow classifying the type of surface (flat, concave, convex; size of surface and so on). The classification data are exchanged be-

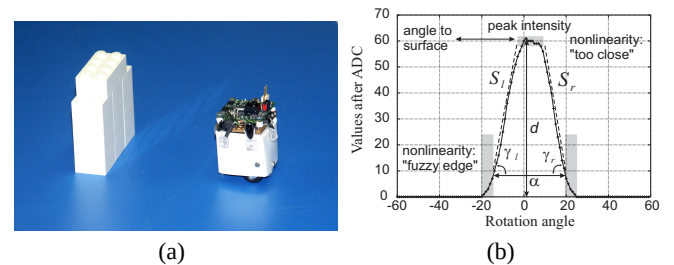


Figure 7: **(a)** Scanning of the finite-size surface; **(b)** The IR-diagram for finite-size surface and the used features of IR-diagrams relevant for identifying the surfaces.

tween robots and matched with the distributed model of the object. However for a particular robot is important not only

to recognize an object, but also to know its own position in relation to this object. This positional context cannot be obtained from the sensor data of individual robot (Figure 7(b)). The idea here is that during local communication, all robots know their neighbors, and this "embodied" information can be used for estimating a position. When sequences of connected particular observations are matched with models (e.g. the model A-B-C-A-C-D and the connected particular observation A-B), these connected observations can be located in the model. In this way the robots can collectively estimate their own spatial context (see more in [Pradier, 2005]). As

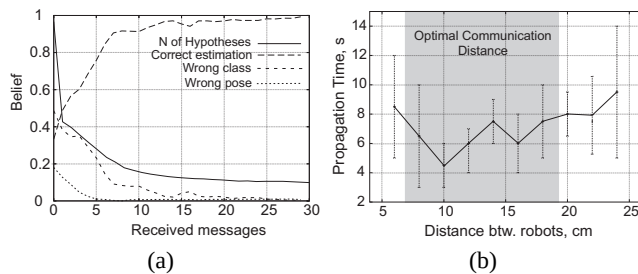


Figure 8: **(a)** Estimation of spatial context during collective perception; **(b)** Dependence between global information propagation time and the distances between robots (6 robots).

demonstrated by experiments, even for uncomplete observations, this approach allows deriving the positions, as shown in Figure 8(a).

The second series of experiments concerns the spatial information processing, like distance/area measurement over a swarm, collective calculation of the swarm's center of gravity and so on. To perform these operations, the robots have to create the SPPN so that a part of them is continuously contained within the communication radius of each other. In this way they build a collective information system, that we call a "communication street".

To maintain the desired distances in SPPN, the moving robots can periodically scan the neighborhood. However this consumes energy/time and essentially distorts the currently executed activities. The idea is to use the local communication context provided by the IR-signal strength, which nonlinearly decreases with the distance. During PCM encoding, the robots measure the amplitude of received from different directions PCM-signals and control the motion so that to maintain SPPN. In this way they not only support an optimal distance for global information propagation (Figure 8(b)), but also can collectively perform different spatial computation (distances, areas, centers of gravity and so on). The details of the experiments can be found in [Fu, 2005].

The performed experiments are too preliminary to be generalized, however the processing of context information during communication works, videos can be downloaded from the project home page (contact authors). During experiments we encountered several undesired effects, one of them consists in clusterization. This phenomenon appears when robots fall to groups so that any communication between groups is broken down. When the detached group is small, these robots usually lose "orientation" and are "lost" for swarm.

Generally, we have shown that despite the limited hardware capabilities of microrobots, the specific construction of hardware and software parts make feasible many collective properties. We also demonstrated that some features of collective AI can essentially be "improved" when using context-based communication. Context-awareness is closely related with physical processes (e.g. IR-signals transmission) and hardware/software components, i.e. with robotic embodiment. However the concept, or at least a systematic approach, that connects the embodiment, context-aware communication and collective intelligence still remains open and represents future works.

We are grateful to our students Mauricio Fernández Pradier and Zheng Fu for performing several experiments.

References

- [Bonabeau *et al.*, 1999] E. Bonabeau, M. Dorigo, and G. Theraulaz. *Swarm intelligence: from natural to artificial systems*. Oxford University Press, New York, 1999.
- [Doyle and Dean, 1996] J. Doyle and Th. Dean. Strategic directions in artificial intelligence. *ACM Comput. Surv.*, 28(4):653–670, 1996.
- [Fu, 2005] Zh. Fu. *Swarm-based computation and spatial decision making*. Master Thesis, University of Stuttgart, Germany, 2005.
- [I-Swarm, 2003 2007] I-Swarm. *I-Swarm: Intelligent Small World Autonomous Robots for Micro-manipulation*, 6th Framework Programme Project No FP6-2002-IST-1. European Communities, 2003-2007.
- [Kornienko *et al.*, 2004] S. Kornienko, O. Kornienko, and P. Levi. About nature of emergent behavior in micro-systems. In *Proc. of the Int. Conference on Informatics in Control, Automation and Robotics (ICINCO 2004)*, Setubal, Portugal, 2004.
- [Kornienko *et al.*, 2005] S. Kornienko, O. Kornienko, and P. Levi. Ir-based communication and perception in microrobotic swarms. In *submitted to IROS 2005, Canada*, 2005.
- [Kube, 1996] C.R. Kube. A minimal infrared obstacle detection scheme. *The Journal for Robot Builders*, 2(2):15–20, 1996.
- [Megabitty, 2005] Megabitty. see <http://groups.yahoo.com/group/megabitty/>. 2005.
- [Nembrini *et al.*, 2002] J. Nembrini, A. Winfield, and C. Melhuish. Minimalist coherent swarming of wireless connected autonomous mobile robots. In *Proc. Simulation of Artificial Behaviour*, Edinburgh, 2002.
- [Payton *et al.*, 2001] D.W. Payton, M. Daily, R. Estowski, M. Howard, and C. Lee. Pheromone robotic. *Auton. Robots*, 11(3):319–324, 2001.
- [Pradier, 2005] M. Pradier. *Collective Classification in a Swarm of Microrobots*. Master Thesis, University of Stuttgart, Germany, 2005.
- [Sahin, 2004] E. Sahin. *Swarm Robotics: From sources of inspiration to domains of application*. Springer-Verlag, Heidelberg, Germany, 2004.
- [Suzuki *et al.*, 1995] S. Suzuki, H. Asama, A. Uegaki, S. Kotosaka, T. Fujita, A. Matsumoto, H. Kaetsu, and I. Endo. An infra-red sensory system with local communication for cooperative multiple mobile robots. In *Proc. of International Conference on Intelligent Robots*, pages 220–225, 1995.

Relational Object Maps for Mobile Robots

Benson Limketkai and Lin Liao and Dieter Fox

Department of Computer Science and Engineering
University of Washington
Seattle, WA 98195

Abstract

Mobile robot map building is the task of generating a model of an environment from sensor data. Most existing approaches to mobile robot mapping either build topological representations or generate accurate, metric maps of an environment. In this paper we introduce relational object maps, a novel approach to building metric maps that represent individual objects such as doors or walls. We show how to extend relational Markov networks in order to reason about a hierarchy of objects and the spatial relationships between them. Markov chain Monte Carlo is used for efficient inference and to learn the parameters of the model. We show that the spatial constraints modeled by our mapping technique yield drastic improvements for labeling line segments extracted from laser range-finders.

1 Introduction

Building maps of indoor spaces is an extremely important task in mobile robotics. Over the last years, the SLAM (simultaneous localization and mapping) community has made tremendous progress in the development of efficient and highly accurate map building techniques. Most of these techniques focus either on capturing the metric layout of an environment with high accuracy [Eliazar and Parr, 2003; Konolige *et al.*, 2005] or on representing the topological structure of an environment [Kuipers, 2000]. While metric maps have the advantage of being well suited for robot navigation tasks, they are typically unstructured and contain no information about the different types of places or objects in an environment. Topological approaches, on the other hand, are more compact and expressive in that they describe significant places in an environment. However, these techniques often ignore valuable metric information and they still lack the ability to describe individual objects in an environment.

Our goal is to build on the progress made in metric mapping and to enhance metric maps with semantic information about different objects in an environment. Object-based representations have much higher expressive power in that they combine metric accuracy with a semantic description of an environment. In addition to allowing more natural interfaces between humans and robots (“take the third door on the right”), they enable robots to perform better reasoning, especially about dynamic objects such as doors.

Recently, Anguelov and colleagues made important steps toward building such maps by developing techniques for detecting and labeling objects in metric maps. In [Anguelov *et al.*, 2002], they showed how to learn models of non-stationary, free-standing objects such as rectangular or round boxes. To overcome the limitations of this technique, they then developed a generative Bayesian approach for detecting doors and walls in a hallway environment [Anguelov *et al.*, 2004]. While this approach yields good results for labeling line segments extracted from laser range-scans, it has limited capabilities in modeling context information. Even though the approach can express information such as all doors share the same width and color, it can not model more complex, spatial relationships between objects. As [Murphy *et al.*, 2003] showed for scene analysis in computer vision, such context information can be extremely useful for distinguishing between different types of objects. In mobile robot map building, for example, an object standing in the middle of a hallway cannot be a door, even if it has the same width and color as other doors in the hallway.

In this paper, we show how to reason about the appearance of and spatial relationships between objects so as to build metric maps with object representations. We denote the resulting mapping paradigm *Relational Object Maps*, RO-Maps for short. RO-Maps build on Relational Markov Networks (RMN) to represent hierarchical structure and the relationships between objects. RMNs are undirected graphical models that are learned discriminatively [Taskar *et al.*, 2002]. They provide an extremely flexible framework for describing and reasoning about relations. Our RO-Maps detect objects such as doors and walls by a combination of features, including appearance features such as the width of a door, neighborhood features describing what type of objects are next to each other, and spatial features such as the indentation of a door relative to the wall. In our experiments we show that the parameters of RO-Maps can be learned from a set of hallways and then be successfully applied to labeling the objects in a hallway of a different environment.

This paper is organized as follows. After providing background on RMNs, we will show how RO-Maps model object appearance and the spatial relationships between them. Inference and parameter learning will be discussed in Section 3. Experimental results are presented in Section 4, followed by conclusions and a discussion of future work.

2 Relational Map Building

This section describes RMNs and several extensions that make them applicable to the problem of mobile robot map building. We denote the resulting framework RO-Maps, short for relational object maps.

2.1 Relational Markov Networks

We will now briefly review the basic ideas of relational Markov networks, more detailed information can be found in [Taskar *et al.*, 2002]. RMNs are an extension of Conditional Random Fields (CRFs), which are undirected graphical models that were developed for labeling sequence data [Lafferty *et al.*, 2001]. CRFs are discriminative models that have been shown to out-perform generative approaches such as HMMs and Markov random fields in areas such as natural language processing [Lafferty *et al.*, 2001] and computer vision [Kumar and Hebert, 2003]. RMNs extend CRFs by providing a relational language for describing clique structures and enforcing parameter sharing at the template level. Thereby RMNs are an extremely flexible and concise framework for defining features and relationships that can be used in our map building context.

Specifically, an RMN consists of three parts: a *schema* $\mathcal{E}$ for the domain, a set of *relational clique templates* $\mathcal{C}$, and corresponding *potentials* Φ . The schema $\mathcal{E}$ specifies the set of *classes* (i.e., entity types) and the attributes in each class. In the map building context, the classes are types of objects, such as walls or doors, and the attributes describe their physical properties. An *instantiation* $\mathcal{I}$ of a schema specifies the set of entities for each class and the values of all attributes for each entity. In our case, $\mathcal{I}$ is an RO-Map consisting of line segments extracted from laser range-scans. A *relational clique template* $C \in \mathcal{C}$ is similar to a query in a relational database. It selects tuples from an instantiation $\mathcal{I}$; the query result is denoted as $C(\mathcal{I})$. In an RO-Map, for example, a clique template can select all wall or door objects on one side of a hallway.

Each clique template C is associated with a potential function $\phi_C(\mathbf{v}_C)$ that maps values $\mathbf{v}_C$ of variables in the clique to non-negative real numbers. Using a log-linear combination of feature functions to define these potentials, we get the following representation: $\phi_C(\mathbf{v}_C) = \exp\{\mathbf{w}_C^T \cdot \mathbf{f}_C(\mathbf{v}_C)\}$, where $\mathbf{f}_C()$ defines a feature vector for C and $\mathbf{w}_C^T$ is the transpose of the corresponding weight vector. For instance, a feature could be the indentation of a door, defined on a clique template selecting a door and wall objects on one side of a hallway.

For a specific instantiation $\mathcal{I}$, an RMN defines a conditional distribution $p(\mathbf{y}|\mathbf{x})$ over labels $\mathbf{y}$ given observations $\mathbf{x}$. To compute the conditional distribution of a label vector $\mathbf{y}$, the RMN generates an *unrolled* Markov network, in which the nodes correspond to the entities. The cliques of the unrolled network are built by applying each clique template $C \in \mathcal{C}$ to the instantiation, which can result in the generation of several cliques per template. All cliques that originate from the same template must share the same weights $\mathbf{w}_C$. The resulting cliques factorize the conditional distribution as

$$p(\mathbf{y} | \mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_{C \in \mathcal{C}} \prod_{\mathbf{v}_C \in C} \phi_C(\mathbf{v}_C) \quad (1)$$

$$= \frac{1}{Z(\mathbf{x})} \prod_{C \in \mathcal{C}} \prod_{\mathbf{v}_C \in C} \exp\{\mathbf{w}_C^T \cdot \mathbf{f}_C(\mathbf{v}_C)\} \quad (2)$$

$$= \frac{1}{Z(\mathbf{x})} \exp\{\mathbf{w}^T \cdot \mathbf{f}\}, \quad (3)$$

where the normalizing partition function $Z(\mathbf{x}) = \sum_{\mathbf{y}} \prod_{C \in \mathcal{C}} \prod_{\mathbf{v}_C \in C} \phi_C(\mathbf{v}_C)$. (3) follows by moving the products into the exponent and combining all summations into $\mathbf{w}$ and $\mathbf{f}$.

For efficient inference in RMNs, [Taskar *et al.*, 2002] suggest using (loopy) belief propagation. However, because of the extensions described below, we cannot apply belief propagation directly. Instead, we will show how to perform efficient inference and learning using MCMC.

2.2 Relational Object Maps

In the most general form, RO-Maps represent an environment by a hierarchical collection of objects and relations between them. RO-Maps estimate the types of objects under the assumption that their locations are known. This assumption is justified by the progress made by the SLAM community, which focuses on the problem of concurrently estimating the pose of a robot and the locations of sensor measurements. Such techniques generate well aligned laser range-scans even for large scale environments (e.g., [Eliazar and Parr, 2003; Konolige *et al.*, 2005]).

We introduce an extension of the relational framework of RMNs to reason about physical objects in RO-Maps. The RO-Map of a specific environment is modeled by an unrolled Markov network. Each node in such a network corresponds to an object or geometric primitive, and the links between nodes represent relations between these objects.

Objects: The basic building blocks of RO-Maps are geometric primitives such as line segments extracted from sensor data. These primitives typically describe objects such as doors or wall segments. More complex objects can be generated from existing ones by a process we call *physical aggregation*. This process first selects all objects with a certain property (just like relational clique templates do in RMNs), and then computes the parameters of the aggregated object. For example, physical aggregation generates a wall object by selecting all wall segments on one side of a hallway, followed by the computation of the line parameters and color distribution describing the wall. The left and right wall of a hallway can be aggregated into a hallway object, which has parameters such as orientation and width.

Relations between objects can be spatial, such as “distance from each other”, or appearance based, such as “similar color”. Relations can be defined on basic objects and objects generated by (multiple) physical aggregations. Relational clique templates of RMNs can be used to define RO-Map relations. For example, to consider the similarity between the widths of the doors in a hallway, RO-Maps use a clique template C that first selects all objects that are labeled as doors (such selections are defined via SQL queries on the objects [Taskar *et al.*, 2002;

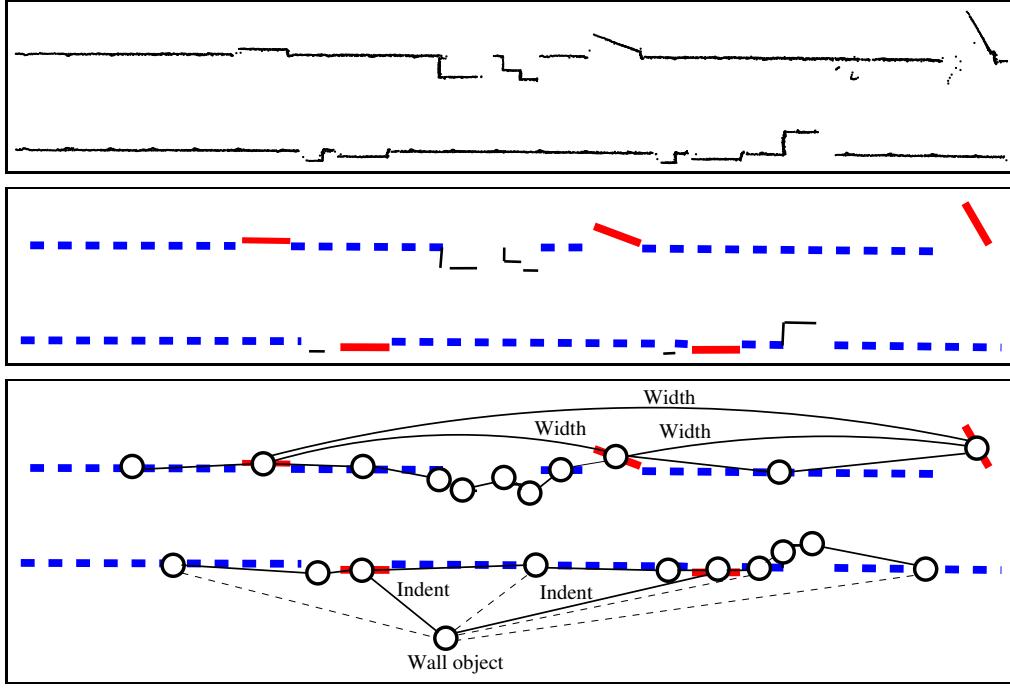


Figure 1: Upper graph: Endpoints of laser range-scans observed in a hallway. The points line up very accurately due to the position correction performed by a scan matching procedure. Middle graph: Line segments extracted from the scans. Wall segments are shown as dashed lines, doors as solid lines, others as thin black lines. Lower figure: Markov network with example cliques generated for this specific labeling. Thin solid lines define neighborhoods. The doors in the upper part of the hallway are linked by the door variance clique template. The wall object is generated by physical aggregation (dashed lines). It is linked to doors by the door indentation clique template (solid lines).

Liao *et al.*, 2005]). The widths of the selected doors define the value vector $\mathbf{v}_C$ of this clique, which is then fed into the feature function $\mathbf{f}_C()$ that computes the variance. The potential of the clique, $\phi_C(\mathbf{v}_C)$, is given by $\exp\{w_C \cdot \mathbf{f}_C(\mathbf{v}_C)\}$, where w_C is the feature weight.

RO-Maps are illustrated in Figure 1. The upper plot shows laser range-scans observed by a robot traveling down a hallway; the middle plot shows the labeled line segments extracted from these scans; and the lower plot shows the nodes in the corresponding RO-Map along with some of the undirected links generated by the clique templates for this labeling. Among others, the figure shows two cliques generated by the template that measures the indentation of doors. The template generates pairs of doors and the corresponding wall object. The potential of the clique is then based on the distance of the door from the line representing the wall. More details on features for detecting doors and walls will be given in the experimental results section.

3 Inference and Learning

3.1 Inference: Labeling objects in RO-Maps

The goal of inference is to estimate the labels (types) of the objects represented in an RO-Map. Inference in RO-Maps is more complicated than in regular RMNs. This is due to the fact that the clique structure of the Markov networks generated by RO-Maps depends on the unknown labels of objects, and thus changes during inference. Since it is not clear how (loopy) belief propagation can be applied to such *label-*

specific cliques, we use Gibbs sampling, an MCMC technique, for inference [Gilks *et al.*, 1996]. Whenever the label of an object is changed during sampling, we determine all (aggregated) objects and cliques that could be affected by this change and re-compute their parameters and potentials. To initialize a Markov chain, we randomly assign labels to the basic objects in the RO-Map. Based on these labels, we generate all aggregate objects and their parameters. Finally, the relational clique templates generate the cliques of the unrolled, fully instantiated Markov network.

At each MCMC step, we update the label of a randomly selected object by sampling from the conditional distribution

$$p(y_k | \mathbf{y}_{-k}, \mathbf{x}, \mathbf{w}) \propto \exp\{\mathbf{w}^T \cdot \mathbf{f}(\mathbf{x}, \mathbf{y}_{-k} \cup y_k)\} \quad (4)$$

where k is the index of the object, y_k is one of its possible labels, and $\mathbf{y}_{-k}$ are the labels for the other objects. To compute the probabilities of the different labels y_k , we update the cliques and the parameters of aggregated objects involving the object k . Even though this process can become inefficient for highly complex RO-Maps, inference on our current test hallways is very efficient, as we will describe in Section 4. We expect further efficiency gains by more elaborate sampling techniques.

3.2 Learning: Determining feature weights

The parameters of RO-Maps are the weights $\mathbf{w}$ of the features that define the clique potentials. The key idea of our technique is to learn these parameters from a set of labeled environments, and then apply the learned weights when the robot explores a new environment.

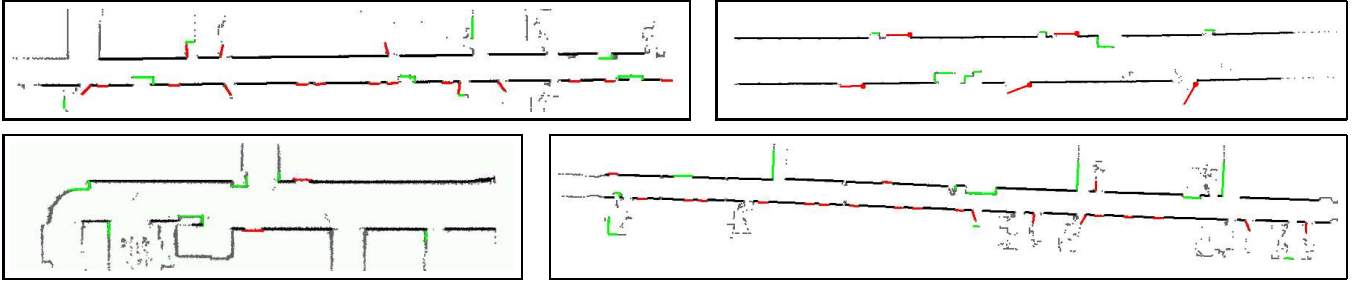


Figure 2: Examples of hallways used for learning and testing. Line segments were labeled manually as Door, Wall, or Other.

The details of our learning technique are beyond the scope of this paper, more information is given in [Liao *et al.*, 2005]. In a nutshell, the weights are learned by minimizing the *negative* log-likelihood of the labeled training data collected in N environments:

$$L(\mathbf{w}) \equiv \sum_{j=1}^N -\log p(\mathbf{y}_j | \mathbf{x}_j, \mathbf{w}) + \frac{\mathbf{w}^T \mathbf{w}}{2\sigma^2} \quad (5)$$

Here, j is the index of map m_j , and $\mathbf{y}_j$ are the labeled objects in the j -th map. The rightmost term avoids overfitting by imposing a zero-mean, Gaussian shrinkage prior on each component of the weight vector [Taskar *et al.*, 2002]. Since (5) is convex, we can determine the global minimum using a quasi-Newton method, where the value and gradient of $L(\mathbf{w})$ are determined by MCMC sampling [Liao *et al.*, 2005].

The outcome of the learning procedure is a set of feature weights that best represent the appearance and spatial constraints of the training maps by maximizing the classification rate. RO-Maps inherit the important benefit from RMNs of automatically enforcing parameter sharing between (aggregated) objects and relations of the same type. This “smoothing” over instances of classes provides a very powerful tool for dealing with sparse training data.

4 Implementational Details and Experiments

We evaluated RO-Maps on the task of labeling line segments extracted from laser range-finder scans collected by a robot traversing a hallway. The data sets were taken from the Radish robotics data set repository [Howard and Roy, 2003]. We took two to three hallways from five different environments each and manually labeled the extracted and aligned line segments into the three categories Door, Wall, and Other. Line segments were extracted using the technique described in [Gutmann and Konolige, 1999]. Some of the test hallways are shown in Figure 2. To perform the labeling, we relied on our experience in typical layouts of hallways. In several cases, we were not sure about the labeling or were only able to label doors based on the fact that we knew the real buildings represented by the maps.

4.1 Features

To label the line segments in the hallways, we implemented various features. The features were represented by the log-likelihood of the measured values under a generative model. This approach allows our technique to automatically take variability between different features into account.

Physical aggregation Wall objects are the only physically aggregated objects in our current system. Wall objects are lines that are generated from line segments that are labeled Walls and are on the same side of a hallway (see Figure 1). The Wall object parameters can be computed efficiently from the parameters of the line segments.

Local features describe appearance properties of objects. In RO-Maps, such features are modeled by generating a clique for each object and the corresponding attribute. We use the length of line segments as a feature. The corresponding feature functions return the log-likelihood of a specific length given a Gaussian representing the distribution over lengths, conditioned on the object type. The means and variances of the Gaussians for Doors, Walls, and Others were estimated from the labeled maps. In our framework it is straightforward to add other features, such as the color of objects [Anguelov *et al.*, 2004]. Unfortunately, the Radish data sets do not contain camera information.

Aggregated Wall objects have a feature that measures the alignment of the line segments that make up the wall object. Alignment is measured by the average distance of the line segments from the aggregated Wall line. This feature helps RO-Maps to label line segments as Walls only when they are well aligned with other Wall segments.

Neighborhood features describe which object types are located next to each other. For example, it is common to observe a Wall segment next to a Door, but uncommon to find many Doors in close proximity to one another. We compute a line segment’s neighborhood by determining all other segments with endpoints within 40 cm. Neighborhood features are then modeled by binary indicator functions on the possible label pairs (e.g., “WW”, “WD”, “DO”).

Spatial features describe the relative position of objects. The indentation of a Door object is computed relative to the aggregated Wall object on the same side of the hallway. Indentation is given by the minimum distance between the Door line endpoints and the Wall line, thereby estimating indentation at the hinge point of a door. As with line width, the indentation feature function returns the log-likelihood under a Gaussian indentation model estimated from the training maps.

Unless an object is a Wall segment, it is highly unlikely

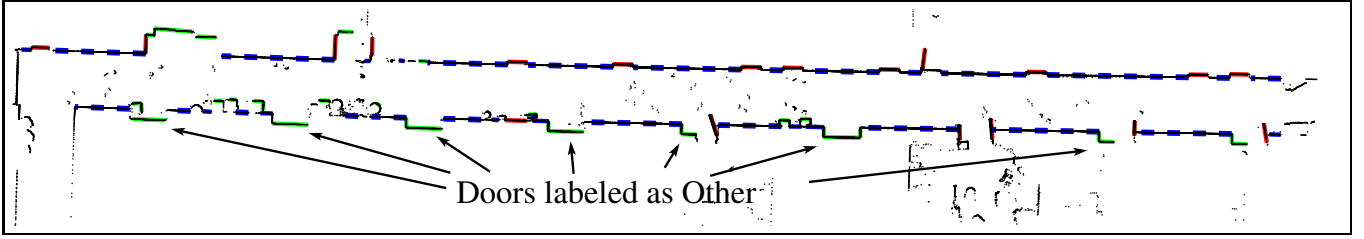


Figure 3: Hallway with the highest classification error. The doors in the lower half of the hallway are wide double doors with very large indentations. Since none of the other training environments contained such doors, our approach labeled them as Other. Some of the open double doors were detected as doors. All doors in the upper half and all walls (dashed lines) are labeled correctly.

to be both very close and well-aligned with a Wall object. RO-Maps detect Other objects by considering the distance and angle alignment of such objects relative to the closest Wall object. To compute the likelihood of specific distances and angles, we estimate discrete distributions from the training maps (these distributions are typically non-Gaussian).

Global features depend on multiple objects that are potentially far apart. In our experiments, we measure the similarity of the indentations of all doors in a hallway by computing their variance.

4.2 Experiments and Results

To test the performance of our learning method, we used 5-fold cross-validation on the five environments extracted from the Radish repository. To do so, the model parameters were trained on labeled hallways from four environments and then tested on the remaining one. Training on a set of 10 hallways took 2-3 hours. To generate the statistics needed for each iteration of the quasi-Newton optimization technique, we ran a Gibbs sampler for 1,000 iterations on each training map. The final 200 samples from each chain were used to compute differences between the sampled feature values and the feature values extracted from the labeled maps. These differences were then used to compute the relative, negative log-likelihood (5) and the gradient.

Testing, that is, labeling of the objects in an unknown map, was very efficient and took around one minute or less on average. For each environment, the labeling error rate was determined by comparing the most probable assignment to the manually generated label. Most probable assignments were extracted from the Markov chains by counting the different labels for each object. The label with the highest count was compared to the ground truth. To gauge the importance of different features for labeling the objects in these hallways, we varied the set of features we used.

Environm.	Lengths	Lengths + Neighbors	All
1	62.6%	88.5%	90.7%
2	58.7%	63.0%	93.5%
3	59.0%	79.2%	89.7%
4	51.8%	96.5%	97.7%
5	60.0%	68.5%	77.9%

Table 1: Average accuracy rates on maps using different features.

Table 1 summarizes the results. The numbers give the ac-

curacy rates of the labels, averaged over the two hallways of each test environment. The leftmost result column gives the accuracy rates achieved when using only the lengths of line segments to distinguish between the different object types. The next column summarizes results obtained when considering length and neighborhood features. The resulting models are similar to standard HMMs or MRFs in that they consider local appearance features (length) and connect neighboring objects. As can be seen, object neighborhood improves the results significantly. Furthermore, these results indicate that the labeling task is by far not trivial. The last column presents the accuracy rates achieved when using all features discussed in the previous section. The results show that the additional consideration of spatial constraints such as door indentation improves the classification performance by a large margin.

Figure 3 shows the hallway with the highest classification error (environment 5). As can be seen, most of the doors along the lower wall of the hallway are mislabeled as Other. However, this is not surprising given that the indentation of these doors is far larger than in the other training maps. Furthermore, most of these double doors are closed, which results in very long line segments. Thus, the labels are very reasonable, given the training data available to the approach.

Truth	Inferred labels		
	Wall	Door	Other
Wall	221	5	8
Door	1	122	21
Other	10	12	93

Table 2: Confusion matrix (counts).

Table 2 shows the confusion matrix summarized over all test environments using all features. The table shows that most errors are due to confusion between Others and Walls, and Others and Doors. Walls and Doors are rarely confused. The overall accuracy rate is 88.4%. This is a very encouraging result, especially since several maps contained objects for which even humans were uncertain about the correct labeling.

It should be noted that [Anguelov *et al.*, 2004] achieved relatively high accuracy using a hierarchical Bayesian approach without spatial constraints. This is due to the fact that they trained model parameters in the same environment and took visual color information into account. Adding such appearance information to RO-Maps is straightforward and we expect strong improvements from such additional information.

5 Conclusions and Future Work

We introduced relational object maps for mobile robot mapping. RO-Maps use relational, discriminative techniques to generate metric maps with semantic object representations. RO-Maps build upon and extend Relational Markov Networks so as to reason about the appearance of and the spatial constraints between objects in an environment. In order to distinguish between different types of objects, RO-Maps can incorporate a variety of features, ranging from global, such as the width of all doors in a hallway, to spatial, such as the indentation of a door. To handle the changing structure of the Markov networks instantiated by RO-Maps, we showed how to perform inference and parameter learning using MCMC. During learning, RO-Maps benefit from automatic parameter sharing provided by relational clique templates. This allows the models to be trained from relatively sparse training sets.

Experiments in five different environments demonstrate that RO-Maps are able to achieve high accuracy in labeling objects in previously unknown hallways. We show that the accuracy is improved drastically by taking spatial constraints between objects into account. These results are particularly encouraging given that objects are labeled based on laser range-data only.

RO-Maps provide an extremely flexible framework for combining metric accuracy with the representation of individual objects and structures in an environment. We believe that RO-Maps are an important step toward more expressive, semantic representations of environments. However, the work presented here is only the first step in this direction, laying the foundation for future research projects.

Our current implementation relies on line segments extracted from laser range-scans as the building blocks of RO-Maps. As shown by [Anguelov *et al.*, 2004], color information extracted from cameras strongly improves the performance of object classification. The flexibility of our framework allows us to integrate such additional appearance information without modifications to the underlying RO-Map concept. We observed significant improvements in initial experiments with a camera pointed toward the walls. We additionally plan to improve line segmentation by incorporating segmentation into the RO-Map inference process. Thereby, high-level information provided by the RO-Map can guide range-scan segmentation.

The RO-Maps presented here model objects in individual hallways only. A key next step will be the extension to more object classes and multiple, connected hallways, rooms, and open spaces. The RO-Map framework enables the definition of high-level structures such as hallways or rooms using the notion of physical aggregation. We are currently developing hierarchical representations and inference techniques that label both high-level structures and individual objects within these structures.

6 Acknowledgments

The authors would like to thank Jeff Bilmes for fruitful discussions during the early stages of this project. This research is sponsored in part by the National Science Foundation (CA-REER grant number 0093406).

References

- [Anguelov *et al.*, 2002] D. Anguelov, R. Biswas, D. Koller, B. Limketkai, S. Sanner, and S. Thrun. Learning hierarchical object maps of non-stationary environments with mobile robots. In *Proc. of the Conference on Uncertainty in Artificial Intelligence (UAI)*, 2002.
- [Anguelov *et al.*, 2004] D. Anguelov, D. Koller, E. Parker, and S. Thrun. Detecting and modeling doors with mobile robots. In *Proc. of the IEEE International Conference on Robotics & Automation (ICRA)*, 2004.
- [Eliazar and Parr, 2003] A. Eliazar and R. Parr. DP-SLAM: Fast, robust simultaneous localization and mapping without predetermined landmarks. In *Proc. of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2003.
- [Gilks *et al.*, 1996] W.R. Gilks, S. Richardson, and D.J. Spiegelhalter, editors. *Markov Chain Monte Carlo in Practice*. Chapman and Hall/CRC, 1996.
- [Gutmann and Konolige, 1999] J.S. Gutmann and K. Konolige. Incremental mapping of large cyclic environments. In *Proc. of the IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA)*, 1999.
- [Howard and Roy, 2003] A. Howard and N. Roy. The robotics data set repository (radish), 2003. radish.sourceforge.net.
- [Konolige *et al.*, 2005] K. Konolige, D. Fox, C. Ortiz, A. Agno, M. Eriksen, B. Limketkai, J. Ko, B. Morisset, D. Schulz, B. Stewart, and R. Vincent. Centibots: Very large scale distributed robotic teams. In M. Ang and O. Khatib, editors, *Experimental Robotics: The 9th International Symposium*, Springer Tracts in Advanced Robotics (STAR). Springer Verlag, 2005.
- [Kuipers, 2000] B. Kuipers. The spatial semantic hierarchy. *Artificial Intelligence*, 119:191–233, 2000.
- [Kumar and Hebert, 2003] S. Kumar and M. Hebert. Discriminative random fields: A discriminative framework for contextual interaction in classification. In *Proc. of the International Conference on Computer Vision (ICCV)*, 2003.
- [Lafferty *et al.*, 2001] J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. of the International Conference on Machine Learning (ICML)*, 2001.
- [Liao *et al.*, 2005] Lin Liao, Dieter Fox, and Henry Kautz. Location-based activity recognition using relational Markov networks. In *Proc. of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2005.
- [Murphy *et al.*, 2003] K. Murphy, A. Torralba, and W. Freeman. Using the forest to see the trees: A graphical model relating features, objects and scenes. In *Advances in Neural Information Processing Systems (NIPS)*, 2003.
- [Taskar *et al.*, 2002] Ben Taskar, Pieter Abbeel, and Daphne Koller. Discriminative probabilistic models for relational data. In *Proc. of the Conference on Uncertainty in Artificial Intelligence (UAI)*, 2002.

Maintaining Coherent Perceptual Information Using Anchoring

Amy Loutfi, Silvia Coradeschi and Alessandro Saffiotti

Center for Applied Autonomous Sensor Systems

Örebro University

Örebro, Sweden 701-82

www.aass.oru.se

Abstract

The purpose of this paper is to address the problem of maintaining coherent perceptual information in a mobile robotic system working over extended periods of time, interacting with a user and using multiple sensing modalities to gather information about the environment and specific objects. We present a system which is able to use spatial and olfactory sensors to patrol a corridor and execute user requested tasks. To cope with perceptual maintenance we present an extension of the anchoring framework capable of maintaining the correspondence between sensor data and the symbolic descriptions referring to objects. It is also capable of tracking and acquiring information from observations derived from sensor-data as well as information from a priori symbolic concepts. The general system is described and an experimental validation on a mobile robot is presented.

1 Introduction

Consider a scenario where a mobile robot is patrolling a corridor. Its task is to discover new objects and gather information about them using both traditional modalities such as vision and sonar and also non-traditional modalities such as an electronic nose. A user is able to monitor the robot, suggest actions to improve perceptual performance and request the robot to perform specific tasks. Tasks are requested from the user using a symbolic representation consisting of natural language concepts. While no requests by the user are given, the robot autonomously prioritises tasks alternating between patrolling the corridor and inspecting objects or simply waiting on stand-by.

An important facet to this scenario, and scenarios of a similar nature, is the ability to maintain coherent perceptual information. This means that the system should be able to maintain the correspondence between the symbolic representation of objects (requests from the user) and the perceptual data that refers to them. The system should collect information from different sensing modalities working concurrently and/or sequentially and correctly attribute that information to its internal representation of the object. The system should include

the preservation of object consistency, so that new information about previously seen objects is also correctly attributed. All the while, the insertion and/or removal of objects from the environment needs to be considered and accounted for. From a cognitive perspective, the maintenance of perceptual information is an integral part of the binding problem [Blackmore, 2003]. That is, how the conjunction of properties are represented, ranging from the binding of shape and colour in detecting blue triangles or red squares to the binding that must occur between and within the senses: “the way the smell and touch and sight of the sandwich in your hand all seem to belong to the same object” [Blackmore, 2003] (p.250).

For robotic systems, an important ingredient for maintaining perceptual information is an internal structure to store a representation of an object. Such an internal structure needs to satisfy a series of requirements. On one hand, perception management [Ronnie *et al.*, 2003], an extension of sensor management [Adrian, 1993], is required to integrate high-level information about the state of the object in order to make situation dependent decisions, direct control and select sensing actions. On the other hand, tracking and data association are also fundamental ingredients necessary to propagate information about coherent perceptions over time [D.Schulz *et al.*, 2003]. A third requirement, especially in the context of a cognitive robot, involves processes, which can create and maintain the link between high-level information (e.g. symbols) and low-level percepts.

In this paper, we show how the concept of anchoring can be used as a tool to confront some of the issues relating to perception management. The anchoring framework [Coradeschi and Saffiotti, 2000] aims at defining a theoretical basis for grounding symbols to percepts originating from physical objects. We present an extension of the framework that is able to cope with perception management considering multi-sensing resources and temporal factors. The pivot of this extension is the use of anchors as internal representations of objects that integrate symbolic and perceptual information across time and across sensing modalities. Our ultimate aim is to shed light on this important aspect of cognitive robotics as applications extend into realistic environments involving human-robot interactions and lifelong acquisition of knowledge.

We begin our discussions with a description and review of the anchoring framework in Section 2. We then present a

modification of the anchoring framework in Section 3 which is adapted for the task of perception maintenance. We proceed by detailing the robotic system architecture used in our experiments. Section 5 gives the implementation details and provides a performance example of the complete system. The paper concludes with a summary of the presented work.

2 The Anchoring Framework

To date, the anchoring framework presented by [Coradeschi and Saffiotti, 2000] has mainly been considered in the context of abstraction of perceptual information. This includes association of words to visually recognised objects [Knoblauch *et al.*, 2004], managing of dynamic object anchoring for high-level reasoning [Chella *et al.*, 2004] and preliminary works considering people tracking applications [Kleinehagenbrock *et al.*, 2002]. A good overview of the range of applications of the anchoring concept is found in 2004 Robotics and Autonomous Systems special issue on anchoring symbols to sensor data [Coradeschi and Saffiotti, 2003].

Before we present our extension of the framework, we summarise here the basic elements of the computational theory of anchoring. See ?? for a full account. The theory considers an autonomous system that includes a symbol system and a perceptual system, and it focuses on the problem of creating and maintaining a correspondence between symbols and percepts that refer to the same physical object. The main ingredients of anchoring are the following [Coradeschi and Saffiotti, 2000]:

- A *symbol system* including: a set $\mathcal{X} = \{x_1, x_2, \dots\}$ of individual symbols (variables and constants); a set $\mathcal{P} = \{p_1, p_2, \dots\}$ of predicate symbols; and an inference mechanism whose details are not relevant here.
- A *perceptual system* including: a set $\Pi = \{\pi_1, \pi_2, \dots\}$ of percepts; a set $\Phi = \{\phi_1, \phi_2, \dots\}$ of attributes; and perceptual routines whose details are not relevant here. A percept is a structured collection of measurements assumed to originate from the same physical object; an attribute ϕ_i is a measurable property of percepts, with values in the domain D_i .

The symbol system manipulates individual symbols, like 'cup-21', which are meant to denote physical objects and associates each individual symbol with a set of symbolic predicates, like 'red', that assert properties of the corresponding object. The perceptual system generates percepts and associates each percept with the observed values of a set of measurable attributes. The task of anchoring is to create, and maintain in time, the correspondence between individual symbols and percepts that refer to the same physical objects.

The symbol-percept correspondence is reified in an internal data structure α , called an *anchor*. Since new percepts are generated continuously within the perceptual system, this correspondence is indexed by time. It is important that the connections are dynamic, since the same symbol may be connected to new percepts every time a new observation of the corresponding object is acquired.

At every moment t , $\alpha(t)$ contains: a symbol, meant to denote an object; a percept, generated by observing that object;

and a signature, a collection of property values meant to provide the (best) estimate of the values of the observable properties of the object.

3 Anchoring for Perception Management

In this section, we present our extension to the anchoring framework. What makes this contribution unique is that we extend the applications of anchoring beyond its traditional use of data abstraction to also include percepts from different modalities that may be accessible at different times.

To effectively present our extension, we refer to the three abstract functionalities defined by [Coradeschi and Saffiotti, 2000]. used to manage anchors, namely, **Find**, **Reacquire**, and **Track**. The functionalities have been developed for the consideration of top-down approaches for information acquisition (i.e. imposed a priori symbolic concepts). In this paper, we revise these existing functionalities to include bottom-up approaches so that anchors can be created by perceptual observations derived from interactions with the environment. Bottom-up approaches have previously been considered on anchoring frameworks [Knoblauch *et al.*, 2004] but never in conjunction with top-down approaches. We advocate the presence of both approaches, in particular for robotic systems interacting with a human user. To accomplish this, an additional **Acquire** functionality is introduced.

3.1 Creation of Anchors

The creation of anchors can occur in both a top-down and bottom-up fashion. Bottom-up acquisition is driven by an event originating from a sensing resource (e.g. the recognition of a segmented region in an image) when perceptual information which cannot be associated to any existing anchor is perceived. Top-down acquisition occurs when a symbol needs to be anchored to a percept, such a call may originate from an external user or a top-level module (e.g. planner).

Acquire Initiates a new anchor whenever a percept is received which currently does not match any existing anchor. It takes a percept π , and return an anchor α defined at t and undefined elsewhere. To make this problem tractable, a priori information is given with regards to which percepts to consider. In bottom-up acquisition, a randomly generated symbol is attributed to the anchor. Furthermore, information about the object and its properties are included into the world model used by the planner, in this way the object can be reasoned about and acted upon.

Find Takes a symbol x and a symbolic description and returns an anchor α defined at t (and possibly undefined elsewhere). It checks if existing anchors that have already been created by the **Acquire** satisfy the symbolic description, and in that case, it selects one. Otherwise, it performs a similar check with existing percepts (in case, the description does not satisfy the constraint of percepts considered by the **Acquire**). If a matching percept is found an anchor is created. Matching of anchor or percept can be either partial or complete. It is partial if all the observed properties in the percept or anchor match

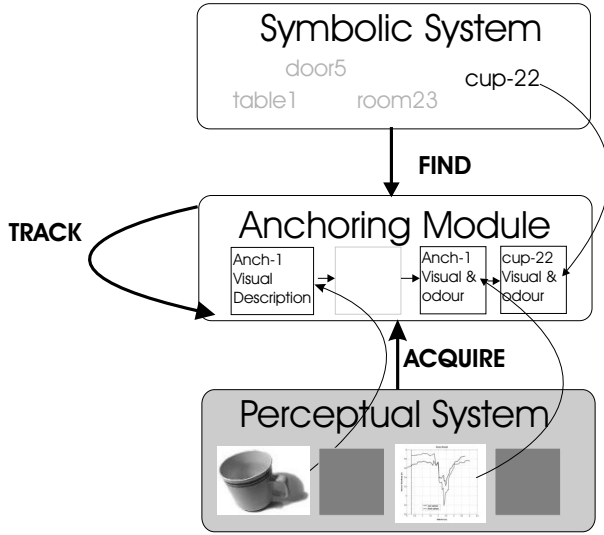


Figure 1: Graphical illustration of the extended anchoring functionalities where bottom-up and top-down information is possible and different sensing modalities are used.

the description, but there are some properties in the description that have not been observed.

3.2 Maintenance of Anchors

At each perceptual cycle, when new perceptual information is received, it is important to determine if the new perceptual information should be associated to existing anchors. The following functionality addresses the problem of tracking objects over time. In this extension, we include the previous **Reacquire** functionality as an integral part of the **Track** and make no special distinction for it.

Track The track functionality takes an anchor α defined for $t - k$ and extends its definition to t . The track assures that the percept pointed to by the anchor is the most recent and adequate perceptual representation of the object. We consider that the signatures can be updated as well as replaced but by preserving the anchor structure we affirm the persistence of the object so that it can be used even when the object is out of view. This facilitates the maintenance of information while the robot is moving as well as maintaining a longer term and stable representation of the world on a symbolic level without catering to perceptual glitches.

3.3 Deletion of Anchors

By having an anchor structure maintained over time, it is possible to preserve the perceptual information even if the object is not currently perceived (caused by the object being out of view and/or by the inaccuracy in the measurement of perceptual data). The challenge is to determine if the association of new percepts is justified or whether certain anchors should be removed. Mechanisms for destroying anchors when the corresponding object has been removed need to be in place. This is a difficult problem, because conceptually it is not clear

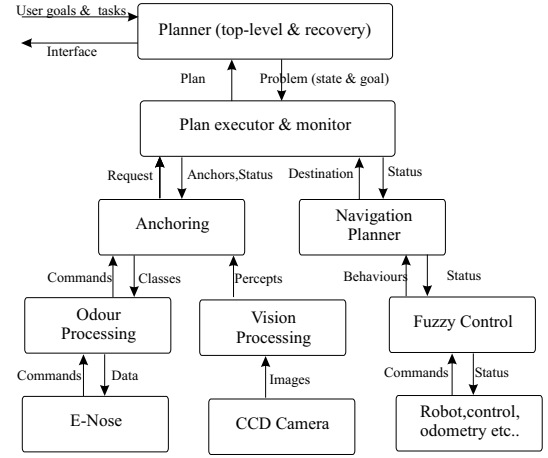


Figure 2: Overview of the robotic system which uses the anchoring module. Arrows indicate the flow of information.

when it is appropriate to remove anchors from the system. Anchors could be removed if they are not relevant for the current task, because the object to which it refers has been physically removed from the environment or the reliability of the perceptual information has expired. Anchors may also need to be removed if they have been associated to invalid perceptual data such as sensory glitches. We currently adopt simple solutions in which objects that are not perceived when expected decrease in a “life” value of the respective anchor. When the anchor has no remaining life, the anchor is removed. The converse could be implemented where anchors are created with initially lowlife values and persistent percepts increase its life value. The decreasing life of anchors is shown in Figure 4. A more adequate strategy to handle the maintenance of anchors may also be to include a “long term” memory where anchors may be stored for future use.

3.4 Integration of the Functionalities

The event-based functionalities are now restricted to the **Find** and **Acquire** while the **Track** functionality is regularly called. Figure 1 shows an overview and an example of the framework and its functionalities. In the example, anchors are created bottom-up from the visual percepts of a cup. Later, additional features of that object are required, for example, the olfactory property. These features are stored in the anchor. When a top-down request is sent to the anchor module to find a cup with matching properties denoted by the symbol “cup-22”, the **Find** functionality anchors the symbol to the perceptual data.

As seen in the figure, properties can be collected at different time points using different modalities. Even when certain perceptual properties are updated, such as the smell property, which may change over time, other perceptual properties are maintained. Conversely, if the visual percepts of an anchor is replaced, the smell property previously obtained is not lost. In this way, the anchor is used to compensate for any dynamically changing features of an object. Furthermore, the perceptual description of anchors can be accessed by the planner

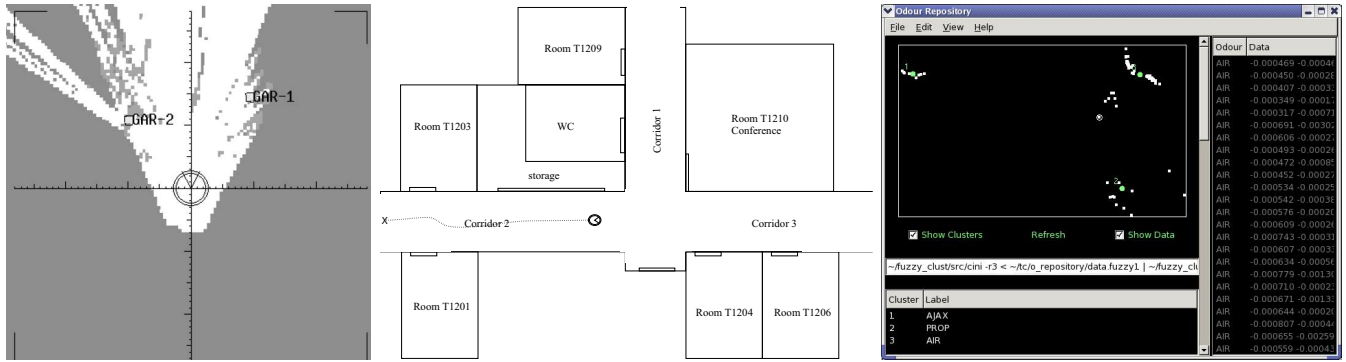


Figure 3: (Left) The local space shows the detected objects, the robot is located in the center of the space. Grey areas in the space are regions that are unexplored. Objects are represented by their visual percepts and are placed within the local space. In this screen, the robot identifies 2 garbage cans, denoted by the percepts *Gar-1* and *Gar-2*. (Middle) The a priori map of the office environment to be patrolled. The robot starting position is shown at point X and a path of the robot is denoted by the dotted line. (Right) The olfactory interface.

to reason about perceptual knowledge. In certain cases, this may result in specific calls to perceptual actions in order to disambiguate between similar objects.

4 The System Architecture

In this section, we present our own instantiation of the extended anchoring framework discussed. We begin our description at the sensor level and proceed to the higher levels which include a planner and user interfaces. An overview of the robotic system is given in Figure 2.

4.1 Sensing modalities and control

Our physical robot is a Magellan Pro compact robot and in the experiments we use its infrared sensors, sonars and tactile sensors. In addition, a CCD camera is mounted on the robot and the robot is able to recognise pre-defined signatures of objects using standard visual techniques. Perhaps the most novel of sensors on the robot is an electronic nose. The electronic nose consists of 32 conducting black polymer sensors, pattern recognition and classification components, and an electronic repository of odours stored in the on-board computer in the robot. The classification algorithm uses the odour repository as a training set for online recognition of new odours. To navigate the robot, a collection of basic behaviours is used. These behaviours are based on fuzzy control techniques and can be combined and reasoned about through use of a behaviour planner (B-plan) explained in [Saffiotti *et al.*, 1995].

4.2 Anchoring Module

The anchoring module, besides creating and maintaining the anchor data structure, also serves a secondary purpose to function as a flag between top-level tasks given by a planner and low-level sensor data. Although not immediately evident this function is important in order to co-ordinate perceptual processes such as a smelling action which may take up to several minutes. In such a case, specific calls to perceptual actions e.g. “Smell gar-22” are generated from the top-

level planner, which then translates into several behaviours being activated e.g. “Go near to Gar-22, Touch Gar-2” and calls to an odour server which activates the respective pumps on valves on the nose. The odour classification provides the smell description e.g. “gar-22 smells ethanol” and the perceptual information is updated. Here the anchoring module “polices” each event signalling to the respective modules when certain process need to be activated and when they have reached completion.

In our instance of the anchoring module, the track functionality is achieved by performing a fuzzy matching algorithm between newly created percepts and previously stored anchors in order to partly deal with sensor noise. However, the purpose of the anchoring framework allows different strategies for the track functionality. A part of the future aims to integrate more advanced solutions into the existing platform.

4.3 Planner

PTLplanner [Karlsson, 2001] is a planner for partially observable domains with uncertainty (probabilities). It searches in a space of epistemic states, or e-states for short, where an e-state represents the agent’s incomplete and uncertain knowledge about the world at some point in time. Although much of the planning component is standard, it is still worth emphasising that the planner can reason about perceptive actions, such as looking at or smelling an object. The consequence is that calls to perceptual actions may be made in order to gather more information about the environment.

4.4 Interfaces

In addition to the computations mentioned in the previous section for control, perception and autonomy, the system also has a number of processes for displaying the internal state of the robot as well as its current model of the external world. This model of the external world includes locally perceived objects and a gridmap of the environment built from sensor data. This is shown in Figure 3. In the left window, the local view of the robot shows the incoming percepts of the vision

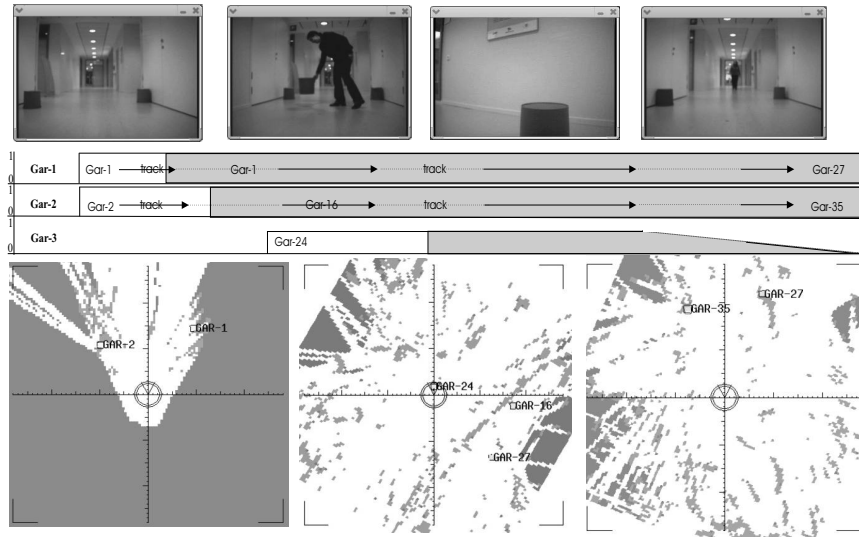


Figure 4: The top row shows the camera images at different time points, the middle row shows the activity at the anchoring level. Grey bars indicate anchors with olfactory properties. The bottom row shows the corresponding local perceptual space given the changing representation of visual percepts.

and spatial modules. In the center figure, a map of the environment is shown. Additional interfaces include a live feed of the images viewed from the CCD camera and an olfactory interface which shows clusters and data points for a loaded repository (Figure 3 (*Right*)). Certain points can be interactively selected to generate new plans for perceptual actions. It also shows the symbolic representation preserving the electronic perception of odours used to classify new odours. More information about the olfactory interface and the categorisation of odours can be found in [Loutfi and Coradeschi, 2004; 2005].

5 Experiments

The general experiment is performed in a series of corridors. In each corridor there may be several objects, in this case garbage cans. The robot automatically toggles between the task of patrolling the corridor, inspecting objects and waiting for commands from the user. Patrolling the corridor involves moving from corridor to corridor in a discovery for new objects and recognition of previous objects. When an inspect is invoked, the robot visits each object collecting the odour property. The inspect is usually autonomously invoked when new objects are detected.

The purpose of the experiment is to evaluate the ability of the extended anchoring framework to maintain an internal representation of the objects in the corridors for an extended period of time. Throughout the autonomous activity of patrolling the corridor, a user may interrupt tasks by requesting the acquisition of specific objects. Object requests can be given by using the image feed from the camera and directly selecting a region in the screen. The sensory signature of the object will be matched against current anchors and current execution of the patrol will be interrupted to include a plan to visit and inspect the selected object. Object can be requested

by giving to the robot a sample of the smelling object and request to find similar objects. Finally top-down requests can be given to find objects by giving to the system a symbolic description of the object. The anchor that most matches the description will be returned.

5.1 Results

The a priori information given to the system consists of a rough map of the environment, shown in Figure 3 (*Middle*), and a repository of interesting objects, namely garbage cans placed outside offices. The robot patrolled the corridors for a period of 4 days, with intermittent breaks during the day and longer breaks during the evening for charging the batteries. At any given time, garbage cans would be removed, displaced, or added into the environment. The total distance covered by the robot is approximately 1.2 km without the inclusion of the extra movement caused by smelling actions and over 70 odour samples were collected.

The local space of the robot together with the visual image from the camera as well as the creation, deletion and updating of anchors is depicted in Figure 4. The figure contains four snapshots throughout an experimental run described as follows:

- Scene 1 - The robot begins patrolling the corridor, two visual percepts are detected and two anchors denoted by **Gar-1** and **Gar-2**, are created. An inspect is performed and both anchors obtain olfactory properties, shown in the Figure by the grey colouring. Since the anchors are created in a bottom-up fashion their labels are arbitrary.
- Scene 2 - As the robot continues its patrol, another object is inserted into the environment at a later time. Note however, that the previous two anchors are still maintained by the track functionality. Although the local space shows only the current percepts, the anchoring

module updates the link between the anchor **Gar-1** and the percept *Gar-27*. A new anchor is also created for the third object denoted by **Gar-3** with visual percept *Gar-24*.

- Scene 3 - The robot approaches the object in order to acquire its odour property and the result is stored in the corresponding anchor. Some time later, the object is removed from the environment. The life of the anchor slowly decreases when an expected percept is no longer detected.
- Scene 4 - The anchor is removed from the system and unless it is perceived again, its properties cannot be accessed by the find functionalities described above.

This scenario shows how the anchoring module is used to create an internal structure which can then maintain the perceptual coherence of objects, considering each object has both spatial and olfactory properties. Even when visual properties of anchors are being updated, the stored smell property remains until a new odour character is acquired by the next inspect action. The previous odour character is then stored in the odour repository.

6 Conclusion

“Take a coin, toss it, and catch it again in your hand...you see a single object fly up in the air, twist over and over, and land in one piece on your hand. Bits don’t fly off. The silver doesn’t depart from the shape, and the shape doesn’t lag behind the motion.” [Blackmore, 2003] (p.244).

Maintaining perceptual coherence of objects over extended periods of time involves the management of perceptual information from different sensing sources, tracking over time and maintaining object persistency. In this paper we showed how a modified anchoring framework could be used as a tool to satisfy these requirements. Experiments on a mobile robot were performed where a robot used both spatial and olfactory sensors to monitor an office environment over an extended period of time.

The problem of perception management is far from being solved. In this paper, we have “scratched its surface” by recognizing the need to consider this problem, highlighting important issues that arise on an embedded system and presenting a first implemented solution.

Acknowledgments This work has been supported by: Vetenskapsrådet, and by ETRI (Electronics and Telecommunications Research Institute, Korea) through the project “Embedded Component Technology and Standardization for URC(2004-2008)”. The authors would like to thank Mathias Broxvall and Lars Karlsson for their contribution to the experimental work.

References

[Adrian, 1993] R. Adrian. Sensor management. In *Proc of AIAA/IEEE Conference on Digital Avionics System*, pages 32–37, 1993.

- [Blackmore, 2003] S. Blackmore. *Consciousness: An Introduction*. Hodder & Stoughton, Oxford, 2003.
- [Chella *et al.*, 2004] A. Chella, S. Coradeschi, M. Frixione, and A. Saffiotti. Perceptual anchoring via conceptual spaces. In *Proc. of the AAAI-04 Workshop on Anchoring Symbols to Sensor Data*, Menlo Park, CA, 2004. AAAI Press. Online at <http://www.aass.oru.se/~asaffio/>.
- [Coradeschi and Saffiotti, 2000] S. Coradeschi and A. Saffiotti. Anchoring symbols to sensor data: preliminary report. In *Proc. of the 17th American Association for Artificial Intelligence Conf. (AAAI)*, pages 129–135, 2000.
- [Coradeschi and Saffiotti, 2003] S. Coradeschi and A. Saffiotti, editors. *Robotics and Autonomous Systems, special issue on Perceptual Anchoring*. Elsevier Science, 2003.
- [D.Schulz *et al.*, 2003] D.Schulz, W. Burgard, D. Fox, and A. Cremers. People tracking with mobile robots using sample-based joint probabilistic data association filters. *I. J. Robotic Res.*, 22(2):99–116, 2003.
- [Karlsson, 2001] L. Karlsson. Conditional progressive planning under uncertainty. In *Proc. of the 17th Int. Joint Conferences on Artificial Intelligence (IJCAI)*, pages 431–438, 2001.
- [Kleinhagenbrock *et al.*, 2002] M. Kleinhagenbrock, S. Lang, J. Fritsch, F. Lmker, G. A. Fink, and G. Sagerer. Person tracking with a mobile robot based on multi-modal anchoring. In *Proc. IEEE Int. Workshop on Robot and Human Interactive Communication (ROMAN)*, pages 423–429, 2002.
- [Knoblauch *et al.*, 2004] A. Knoblauch, R. Fay, U. Kaufmann, H. Markert, and G. Palm. Associating words to visually recognized objects. In S. Coradeschi and A. Saffiotti, editors, *Anchoring symbols to sensor data. Papers from the AAAI Workshop. Technical Report WS-04-03*, pages 10–16. AAAI Press, Menlo Park, California, 2004.
- [Loutfi and Coradeschi, 2004] A. Loutfi and S. Coradeschi. Forming odour categories using an electronic nose. In *Proc. of European Conference in Artificial Intelligence (ECAI 2004)*, pages 119–124, 2004.
- [Loutfi and Coradeschi, 2005] A. Loutfi and S. Coradeschi. Improving odour analysis through human robot interaction. In *To Appear Proc of the IEEE International Conference on Robotics and Automation (ICRA 2005)*, Barcelona, Spain, 2005.
- [Ronnie *et al.*, 2003] L. Ronnie, M. Johansson, and N. Xiong. Perception management - an emerging concept for information fusion. *Information Fusion*, 4(3):231–234, 2003.
- [Saffiotti *et al.*, 1995] A. Saffiotti, K. Konolige, and E. H. Ruspini. A multivalued-logic approach to integrating planning and control. *Artificial Intelligence*, 76(1-2):481–526, 1995.

2D Shape Classification and Retrieval

Graham McNeill, Sethu Vijayakumar

Institute of Perception, Action and Behavior

School of Informatics, University of Edinburgh, Edinburgh, UK. EH9 3JZ

Email: G.J.McNeill-2@sms.ed.ac.uk, sethu.vijayakumar@ed.ac.uk

Abstract

We present a novel correspondence-based technique for efficient shape classification and retrieval. Shape boundaries are described by a set of (ad hoc) equally spaced points – avoiding the need to extract “landmark points”. By formulating the correspondence problem in terms of a simple generative model, we are able to efficiently compute matches that incorporate scale, translation, rotation and reflection invariance. A hierarchical scheme with likelihood cut-off provides additional speed-up. In contrast to many shape descriptors, the concept of a mean (prototype) shape follows naturally in this setting. This enables model based classification, greatly reducing the cost of the testing phase. Equal spacing of points can be defined in terms of either perimeter distance or radial angle. It is shown that combining the two leads to improved classification/retrieval performance.

1 Introduction

In many object recognition problems, the examples are most easily disambiguated on the basis of *shape* - as opposed to features such as color or texture. Often the classification of objects extracted from an image database are most intuitively formulated as a shape classification task. Here we present a fast, general algorithm for classification and retrieval of 2D objects.

Much of the recent work in this area uses a finite set of points taken from the object’s boundary as the *shape representation*. Points can be selected on the basis of maximal curvature [Super, 2004], distance from the centroid [Zhang *et al.*, 2003] or any criteria deemed suitable to the class of shapes involved. More sophisticated approaches parameterize the boundary as a closed curve and slide points along the outline to minimize an objective function (e.g. [Wang *et al.*, 2004]). These methods produce good results but generally use expensive optimization algorithms. An alternative to finding the ‘correct points’ is to simply place points at roughly equal intervals along the boundary. Belongie *et al.* [Belongie *et al.*, 2002] used this approach effectively in their work on shape contexts.

Having chosen how to represent the objects, one must solve the *correspondence problem* in order to compare shapes. This is illustrated in Fig.1. Also, a valid set of transformations for

shape matching is required. These transformations represent the chosen definition of shape, i.e. they are the operations that leave shape unchanged.

In this paper, we introduce a simple, generic technique for shape classification and retrieval. Shapes are represented by a potentially large number of points from their boundaries. The points lie at fixed intervals in terms of distance along the boundary or radial angle. This gives an accurate and robust description of shape that avoids the somewhat arbitrary decision of what constitutes a good point. The high computational cost associated with using many points is reduced in three ways. Firstly, a hierarchical approach avoids the need to consider all possible label assignments. Secondly, potential correspondences are considered simultaneously and a clear winner often emerges early in the computation. Finally, for classification problems, the algorithm produces class means with associated variances at each point. This allows model based classification, requiring fewer shape comparisons at the testing stage. By considering both the distance and angle based descriptors, we can accurately describe a wider range of shapes and increase the classification/retrieval accuracy. The effectiveness of the method is demonstrated on a benchmark data set and compared to other state-of-the-art methods in the field.

2 Method

We wish to develop generic techniques for classifying or retrieving shapes. There are two important issues to consider: Firstly, the ultimate goal is to work with large data sets and hence, computational expense is an important factor. Secondly, the shapes involved may be very diverse, with large differences between some classes. For example, one class might consist of very jagged shapes and the other smoothly curved shapes. This implies that overly specific models of shape variation should be avoided.

2.1 Shape Correspondence

Given two shapes, we wish to optimally align them in order to observe the genuine differences in shape. When shapes are represented by a set of points, it is necessary to find the best match over all possible label assignments (Fig.1). In this paper, we use a definition of shape from statistical shape analysis (e.g. [Dryden and Mardia, 1998]): The *shape* of a set of labelled points is all the information that is invariant to rotation, scaling and translation. In addition to this, we also consider reflection to be a valid transformation. Allowing

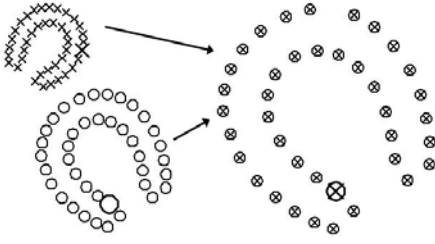


Figure 1: Two identical shapes (left). Large symbols indicate corresponding points under a random assignment of labels. To align the shapes, we must find the *correct labels* as well as the *correct transformations* (right).

only basic transformations reflects our lack of prior knowledge about the shape classes involved. We now formulate the correspondence problem in a probabilistic fashion. This will enable correspondences to be chosen at minimal computational cost.

Correspondence Algorithm

Consider two shapes, each represented by k points from their respective boundaries: $\mathbf{X} = (\mathbf{x}_1 \dots \mathbf{x}_k)^T$ and $\mathbf{Y} = (\mathbf{y}_1 \dots \mathbf{y}_k)^T \in \mathbb{R}^{k \times 2}$. Both shapes are *centered*: $\sum_{j=1}^k \mathbf{x}_j = \sum_{j=1}^k \mathbf{y}_j = \mathbf{0}$. We assume that the $\mathbf{y}_j$ are generated independently from

$$\mathbf{y}_j \sim N(s\Gamma\mathbf{x}_{j+m|k} + \mathbf{t}, \sigma^2\mathbf{I}_2) \quad (1)$$

for $j = 1, \dots, k$, where s is a scaling parameter, $\mathbf{t}$ a translation vector and Γ a 2D rotation matrix - reflection is dealt with later. $m \in \{0, \dots, k-1\}$ cycles the labels of the points in $\mathbf{X}$. Finding the maximum likelihood estimates (MLEs) of the parameters is a restricted version of the matching problem for two unlabelled sets investigated in [Kent *et al.*, 2004]. For the general case, the large number of potential correspondences ($k!$) can make finding the MLEs prohibitively expensive. However, here we are dealing with boundaries and need only consider cycling of the labels. This means that there are only k potential label assignments and the computational cost of computing the MLEs $\hat{\Gamma}$, $\hat{s}$, $\hat{\mathbf{t}}$ and $\hat{m}$ is more acceptable.

The likelihood of the parameters can be written as

$$L(m, s, \mathbf{t}, \Gamma) = \frac{1}{(2\pi\sigma^2)^k} \exp\left\{-\frac{1}{2\sigma^2}d^2(\mathbf{Y}, \mathbf{X})\right\} \quad (2)$$

where

$$d^2(\mathbf{Y}, \mathbf{X}) = \sum_{j=1}^k \|\mathbf{y}_j - (s\Gamma\mathbf{x}_{j+m|k} + \mathbf{t})\|^2.$$

For a fixed $m \in \{0, \dots, k-1\}$, $\hat{s}$, $\hat{\mathbf{t}}$ and $\hat{\Gamma}$ take values that minimize $d^2(\mathbf{Y}, \mathbf{X})$ - a consequence of assuming isotropic variance and independence between points. We now exploit a closed form solution for the minimum of $d^2(\mathbf{Y}, \mathbf{X})$ which can be written compactly using complex notation. Letting $\mathbf{x}_j \equiv (x_{j1}, x_{j2})^T \rightarrow x_{j1} + ix_{j2} \equiv z_j \in \mathbb{C}$, the shapes become complex vectors: $\mathbf{X} \rightarrow \mathbf{z}$, $\mathbf{Y} \rightarrow \mathbf{w} \in \mathbb{C}^k$. Following

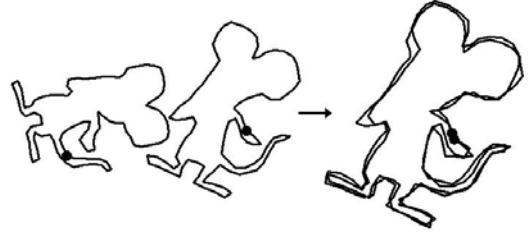


Figure 2: Two shapes from the same class, each described by 100 points (left) - linear interpolation is used to aid visualization. The correspondence algorithm correctly cycles the labels and the shapes can be aligned (right).

[Dryden and Mardia, 1998] we have, for a fixed cycling m , and unit sized shapes ($\sum_{j=1}^k |z_j|^2 = \sum_{j=1}^k |w_j|^2 = 1$)

$$\min_{\Gamma, s, \mathbf{t}} d^2(\mathbf{X}, \mathbf{Y}) = 1 - |\mathbf{w}^* \mathbf{z}|^2$$

where $\mathbf{w}^*$ denotes the conjugate transpose of $\mathbf{w}$. Absorbing the size normalization into the distance expression gives the expression for the *full Procrustes distance*:

$$d_F(\mathbf{z}, \mathbf{w}) = 1 - \frac{|\mathbf{w}^* \mathbf{z}|^2}{\mathbf{w}^* \mathbf{w} \mathbf{z}^* \mathbf{z}}. \quad (3)$$

This is a standard metric in shape analysis. Substituting $d_F^2(\mathbf{z}, \mathbf{w})$ into eq.(2) allows us to evaluate $L(m, \hat{s}, \hat{\mathbf{t}}, \hat{\Gamma})$ for all $m \in \{0, \dots, k-1\}$ and hence, find $\hat{m}$. Fig.1 demonstrates the correspondence algorithm applied to two identical shapes - alignment is achieved using the full Procrustes fit (Section 2.2). As expected, the shapes can be perfectly matched. In Fig.2, the shapes are different examples from the same class. A good match is possible after finding the best correspondence.

Invariance with respect to reflection is not yet incorporated into the distance computation. It is worth mentioning that Procrustes distances can directly incorporate reflection invariance [Mardia *et al.*, 1979]. Unfortunately, this idea is not compatible with our formulation of the correspondence problem. A simple example demonstrates this: Imagine $\mathbf{z}$ is the reflection of $\mathbf{w}$. When faced with the task of corresponding the two shapes, we are obviously unaware of the reflection and proceed to label both shapes in an anticlockwise direction. $\mathbf{w}$ will only be the same as its reflection (in the Procrustes sense) if the labels are reflected along with the points. Doing this would flip the direction of the labels to clockwise. $\mathbf{z}$ cannot be matched to the original $\mathbf{w}$ (due to reflection) nor to the reflected $\mathbf{w}$ (because the labels run in opposite directions). The reflection of $\mathbf{z}$ is given by $\mathbf{z}^* \equiv (\bar{z}_1, \dots, \bar{z}_k)$. This operation changes the direction of the labels to clockwise. Reversing the order of the entries: $\mathbf{z}_R \equiv (\bar{z}_k, \dots, \bar{z}_1)^T$, switches the direction back to anticlockwise. The optimal correspondence between $\mathbf{z}_R$ and $\mathbf{w}$ can now be found as normal. This apparent doubling in computational expense can be significantly reduced by tracking the correspondence calculations for the non-reflected shapes.

Reducing the Computational Cost

Any classification or retrieval problem will require a large number of correspondences to be found. The cost of computing correspondences is dominated by k - the number of

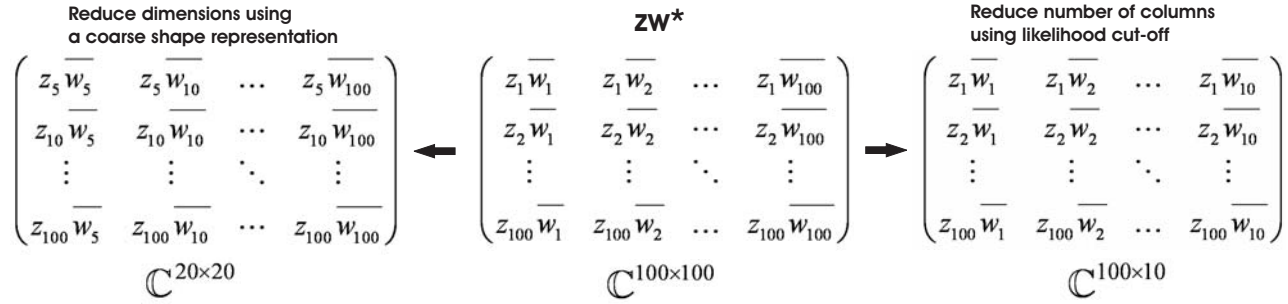


Figure 3: Example of how the speed-up techniques can affect the dimensions of the outer product matrix - see text for details.

boundary points used. Firstly, the number of cycles to consider increase linearly with k . Secondly, the distance computation used to evaluate a correspondence is essentially a dot product of vectors in $\mathbb{C}^k$. From an accuracy perspective, it may be necessary to have a large k . Fortunately, for any given k , there are options for reducing both the number of cycles that are considered and the cost of each distance computation.

To evaluate $d_F^2(\mathbf{z}, \mathbf{w})$ it is necessary to compute the dot products $\mathbf{w}^* \mathbf{w}$, $\mathbf{z}^* \mathbf{z}$ and $\mathbf{w}^* \mathbf{z}$. The values of $\mathbf{w}^* \mathbf{w}$ and $\mathbf{z}^* \mathbf{z}$ do not change with the cycle value and need only be computed once. The cost of corresponding $\mathbf{z}$ and $\mathbf{w}$ is therefore dominated by calculating the dot product between $\mathbf{w}^*$ and all cycled versions of $\mathbf{z}$. This is equivalent to summing along the diagonals of the outer product matrix $\mathbf{z}\mathbf{w}^*$ (for example, summing the leading diagonal gives the value of $\mathbf{w}^* \mathbf{z}$ when the cycle value is zero - see Fig.3). The following procedures reduce the dimensions of $\mathbf{z}\mathbf{w}^*$ and hence, the computational cost of the algorithm.

The number of points used, k , is assumed to be large (say $k = 100$). To avoid considering all k possible values of the cycling parameter m , we initially consider a coarse representation of the shapes using $k' < k$ points (say $k' = 20$ and the points are found by selecting every fifth point of the 100). The likelihoods for all 20 values of the cycling parameter m' are then evaluated as normal. It seems reasonable to assume that the most likely m will correspond the shapes in a similar way to that which would be achieved using all 100 points. Since the 20 points come from the original 100, it is easy to relate these approximate correspondences back to the true shapes (those described by 100 points). The final correspondences are found in the standard way, but only cycle values close to the approximated value are investigated. The outer product matrix is in $\mathbb{C}^{20 \times 20}$ rather than $\mathbb{C}^{100 \times 100}$ for the first round of correspondences (Fig.3). At the fine tuning stage, only a small number of the potential 100 correspondences are evaluated. Thus, the overall cost is greatly reduced.

Regardless of the number of points used to describe a shape, it seems that the full matrix $\mathbf{z}\mathbf{w}^*$ must be evaluated and used to compute the Procrustes distances. However, by computing the columns of $\mathbf{z}\mathbf{w}^*$ incrementally, the likelihoods over all k possible cycles can be monitored and a correspondence chosen when one of them exceeds a given threshold (c.f. Fig.3). This approach captures the intuitive idea that some correspondences will be much easier to find than others. The likelihood threshold is reached after

fewer iterations when ‘new’ points are selected at random (e.g. $w_{137}\mathbf{z}, w_{49}\mathbf{z}, \dots$) so that the full boundary is explored quickly.¹ Clearly if the threshold is never reached and all the points are needed, this incremental method will be more costly than the basic algorithm. The additional cost comes from updating the k different $\mathbf{z}^* \mathbf{z}$ associated with each labelling of $\mathbf{z}$ and from evaluating the likelihood at each stage.² It can be partly avoided by evaluating $\mathbf{z}\mathbf{w}^*$ in blocks, rather than single columns. Often very few points are required to find the best correspondence (Section 3.3). It is worth noting that our algorithm could assign correspondences on the basis of minimum Procrustes distance (d_F) - so why bother with likelihoods? Fig.4 shows the evolution of the likelihoods compared to the squared full Procrustes distances for a typical problem. The likelihood approach magnifies the best correspondences and kills-off the worst, leading to a quicker choice of cycle parameter m . Recall from eq.(1) that there is isotropic variance, σ^2 , at each point. σ^2 is a free parameter that affects how quickly the likelihoods peak to a single cycle value. A small variance encourages fast correspondence selection (Fig.4 (right)) but assumes little within class variability. Note that this thresholding technique is applicable whenever the correspondence algorithm is used, i.e. it is independent of the above method for reducing the number of cycle values considered.

Shape Descriptors

The correspondence algorithm is independent of how the k boundary points are selected in the first place. To keep our approach as generic as possible, very simple point selection techniques are used. We either choose points at roughly equal intervals along the boundary or points at equal increments of the radial angle (the centroid radii model [Chang *et al.*, 1991]). We refer to the former method as the *perimeter* shape descriptor and the latter as the *radial* descriptor. In some cases, a line from the origin can intercept the boundary at more than one point. To make the radial descriptor consistent, all boundary points close to these intercepts are considered and the point most distant from the origin is selected. This can lead to parts of the shape being ignored. However, the results in Section 3 demonstrate the advantage of using

¹Discontinuing the likelihood computation for any m whose likelihood drops below a given threshold would further increase the speed of the algorithm.

²A random subset of a centered shape will be approximately centered so there is no need to update the center incrementally.

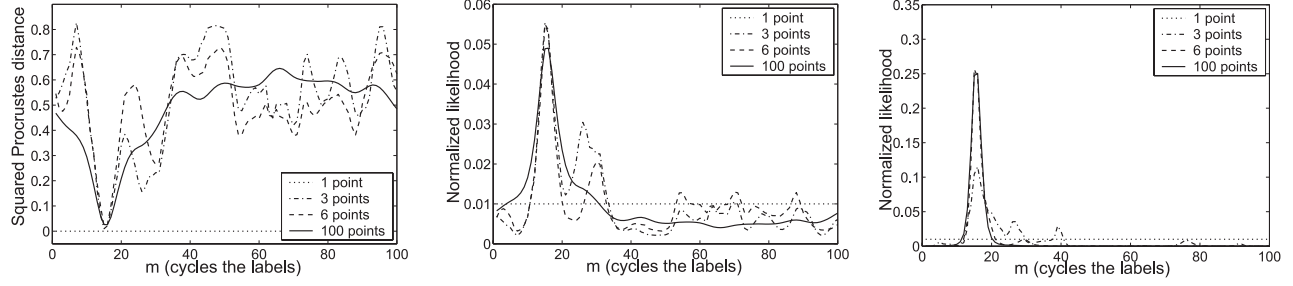


Figure 4: Squared full Procrustes distances (left) and normalized likelihoods (middle) between two shapes from the mouse class. The different lines indicate the distance/likelihood after using a given number of points - or equivalently, computing a given number of columns of $\mathbf{z}\mathbf{w}^*$. Decreasing σ^2 exaggerates the peaked profile of the likelihoods (right).

this sub-optimal descriptor.

2.2 Shape Model

The correspondence technique discussed above can be incorporated into a nearest neighbor classifier: given a test shape, find its optimal correspondence with every training shape and classify on the basis of minimum Procrustes distance. This requires finding a large number of correspondences, a problem that can be avoided using prototypes. Ideally, the *full Procrustes mean* of each class would form the set of prototypes. The definition of this mean follows naturally from the definition of d_F . It is defined as $\bar{\mu} \in \mathbb{C}^k$ such that $\|\bar{\mu}\| = 1$ and

$$\bar{\mu} = \arg \min_{\mu} \sum_{l=1}^n d_F^2(\mathbf{z}_l, \mu) \quad (4)$$

where $\mathbf{z}_1, \dots, \mathbf{z}_n$ are all the shapes in the class. Bookstein [Bookstein, 1996] has proposed an iterative method for finding $\bar{\mu}$. By incorporating the correspondence algorithm into this method, we can correspond shapes directly onto a running class mean.³ This procedure relies on the *full Procrustes fit* for aligning two shapes. Fitting $\mathbf{z}$ to $\mathbf{w}$ transforms $\mathbf{z}$ so that the sum of squared distances between corresponding points is equal to $d_F(\mathbf{z}, \mathbf{w})$. In Fig.1, $d_F(\mathbf{z}, \mathbf{w})$ is zero and a perfect fit is achieved. In Fig.2, $d_F(\mathbf{z}, \mathbf{w})$ is small because the shapes are similar and a reasonable fit is possible. The fit of $\mathbf{z}$ onto $\mathbf{w}$ is given by

$$\mathbf{z}_{fit} = \mathbf{z}^* \mathbf{w} \mathbf{z} / (\mathbf{z}^* \mathbf{z}). \quad (5)$$

The algorithm for finding the correspondences and the mean within a given class is shown in Table 1. Shapes are successively corresponded and fitted to a running mean.

Given a mean, the sample covariance matrices can be calculated at each point. The final *class model* consists of a mean shape - k independent 2D points, each with an associated covariance matrix. Fig.5 shows the algorithm applied to twenty objects from the same class.

2.3 Shape classification

The most likely correspondence and fit between a test shape and each class mean is found as described in Section 2.1. For classification, the model for finding correspondences (eq.(1)) is modified. Points are still assumed independent, but the

³ $\bar{\mu}$ can be calculated analytically when the correct labels are known (see e.g. [Dryden and Mardia, 1998]).

Table 1: Algorithm for corresponding all shapes from a given class to a running class mean.

```

Initialize:  $\mu$  = final shape in class
for i = 1:cycles      for j = 1:final shape in class
  1. get corresponding labels of shape  $j$  and  $\mu$ 
  2. fit shape  $j$  to  $\mu$ 
  3. recompute  $\mu$  using arithmetic mean at each point
end
end Fit all shapes to the final  $\mu$ 

```

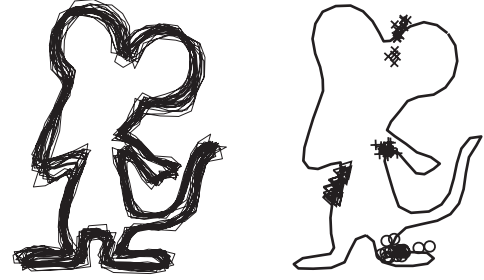


Figure 5: Left: class mean (bold) and the twenty class examples fitted to the mean using 100 points (interpolated for visualization); Right: class mean and scatter of corresponding points at four positions.

isotropic covariance is replaced by the sample covariance matrix⁴ at each point. Thinking of shapes as sets of points in $\mathbb{R}^2$, a test shape $\mathbf{X} = (\mathbf{x}_1, \dots, \mathbf{x}_k)^T$ is assigned to a class $c(\mathbf{X})$ using maximum likelihood (ML)

$$c(\mathbf{X}) = \arg \max_{c=1, \dots, C} \left(\prod_{j=1}^k f(\mathbf{x}_j; \mu_{cj}, \Sigma_{cj}) \right) \quad (6)$$

where C is the number of classes, $(\mu_{c1}, \dots, \mu_{ck})^T \in \mathbb{R}^{k \times 2}$ is the mean of class c , $\Sigma_{cj} \in \mathbb{R}^{2 \times 2}$ is the sample covariance at μ_{cj} and $f(\cdot; \mu_{cj}, \Sigma_{cj})$ is the pdf of the bivariate normal distribution $N(\mu_{cj}, \Sigma_{cj})$.

⁴Computed during the training phase.



Figure 6: Example boundaries from the MPEG-7 data set.

3 Evaluation and Results

The proposed technique is tested on the benchmark MPEG-7 shape database [Latecki *et al.*, 2000]. This consists of 70 classes with 20 observations in each class. Pixels lying on a closed boundary are extracted in a clockwise direction using Matlab’s image processing toolbox. Some of these boundaries are shown in Fig.6 (the same data was used for Figs. 1, 2 and 5). The first principle component of these (2D) points is aligned with the x-axis. This helps reduce the problem of ‘sub-interval shifts’ producing an artificially high Procrustes distance between similar shapes – particularly for small k . The shape is described by k points using either the perimeter or radial descriptor as described in Section 2.1. Initially we present results without using any speed-up techniques. The impact of these are explored in Section 3.3.

3.1 Leave-one-out Shape Classification

A thirty class subset of the MPEG-7 database was recently used by [Kunttu *et al.*, 2003] in their work on multiscale Fourier descriptors. Their approach outperformed the standard contour Fourier descriptor in various classification tasks. Using leave-one-out classification with a nearest neighbor classifier, they achieved 94.2-96.3% accuracy (depending on the length of shape descriptor). We use the perimeter descriptor with the same data set and testing procedure (i.e. leave-one-out and nearest neighbor) to enable a direct comparison with their results. Test shapes are fitted to training shapes using the correspondence method described in Section 2.1. The nearest neighbor is the training shape closest to the test shape in terms of full Procrustes distance. Results are shown in the first row of Table 2. Classification accuracy is slightly better than that of [Kunttu *et al.*, 2003] and the results are consistent over a wide range of k . The same tests were carried out using the ML method described in Section 2.3. Results are given in the bottom row of Table 2. This prototype based technique is almost as accurate as the nearest neighbor classifier and requires that many fewer correspondences are found - 30 versus 599 per test shape.

The first column of Table 3 shows the corresponding results for the full seventy class data set with $k = 48$.⁵ One

⁵Classification using distance to the nearest prototype is less accurate than the ML classifier. This indicates that the (point-wise) scatter about the mean is sufficiently ‘tight’ so as to produce covariance matrices that accurately reflect the within-class variability.

Table 2: Classification performance(%) for the thirty class data set (k = number of points, perimeter descriptor used).

k	12	24	48	72	96
N. neighbor	95.67	97.83	97.83	97.50	98.00
ML	93.17	96.00	96.83	96.67	87.17

Table 3: Classification performance(%) for the seventy class data set with $k = 48$.

	Perimeter	Radial	Combined
N. neighbor	95.71	91.00	96.29
ML	90.36	83.79	92.64

reason for the slightly poorer results on the full data set is that there are now classes for which the perimeter descriptor is definitely not suitable. Despite the generally inferior performance of the radial descriptor (Table 3, middle column), it performs significantly better for some classes. This suggests that using both descriptors may improve classification accuracy. Here, we demonstrate how a naive combination of the descriptors can enhance performance. The ML classifier is modified as follows: Each descriptor (perimeter and radial) gives a vector of class-conditional likelihoods for the test shape. The two vectors are normalized so that they each sum to one. For a given class, the maximum of the two candidate likelihoods is used for classification. This reflects an assumption that the descriptor with the most confident prediction is correct. It is less clear how to normalize the distances for the nearest neighbor classifier. We assume that the majority of shapes bear no resemblance to the test shape and hence, give rise to meaningless correspondences and distances. Normalization is therefore carried over the minimum q entries in each distance vector ($q \ll$ total number of shapes). The distance used for classification is the minimum of the two candidate distances. Here, and in Section 3.2, we fix $q = 200$. The results in Table 3 show a small rise in classification accuracy for the combined case over the perimeter only case. Evidently, the increase in performance is limited by the already high performance of the perimeter descriptor and the simplicity of the normalization technique.

3.2 Bullseye Test for Shape Retrieval

This is a frequently used test in shape retrieval and enables us to compare our algorithm against many of the best performing shape retrieval techniques. A single shape is presented as a query and the top forty matches are retrieved (from the entire data set - the test shape is not removed). The task is repeated for each shape and the number of correct matches (out of a maximum possible 20) are noted. A perfect performance results in $1400 \times 20 = 28000$ matches. Results are given as a percentage of this perfect score. The first two rows of Table 4 show the result of the proposed technique using the perimeter and radial descriptors when no speed-ups were used. The bullseye score was also computed using the combined approach as described for nearest neighbor classification in the previous section. Results are shown in the bottom row of Table 4. [Super, 2004] notes that the best performing algorithms in this test have scored between 75.44% and 78.17%. A naive combination of two simple descriptors

Table 4: Scores for the bullseye test(%).

k	12	24	48	72	96
Perimeter	67.29	74.83	76.28	76.00	74.73
Radial	53.33	64.02	67.57	68.03	68.08
Combined	70.67	77.77	79.14	79.19	78.56

Table 5: Classification performance using speed-ups.

	Standard	LC	C-F	C-F/LC
NN	97.83	97.13 (8.53)	97.83	97.17 (3.91)
ML	96.83	88.75 (9.73)	94.33	85.00 (4.13)

outperforms all of the previously tested algorithms. It should be noted that these results depend on the free parameter q , the number of shapes used in normalization. We are investigating methods for automatically selecting this value.

3.3 Fast Correspondence

We repeat the classification tests on the thirty class data set using the speed-ups described in Section 2.1. The perimeter descriptor with $k=48$ is used throughout. The results are given in Table 5 - results are averaged over 10 runs where likelihood cutoff is used. Column one shows the results when no speed-ups are used. The second column shows the results using likelihood cut-off. A correspondence is chosen when the highest likelihood exceeds the second highest likelihood by $2/(2 \times k)$ (i.e. twice the probability assigned to all correspondences under a uniform distribution - recall reflection must be considered). The average number of points needed to choose a correspondence is given in brackets. Column three gives results when the coarse-to-fine correspondence technique is implemented with $k'=12$. An approximate shape correspondence is mapped back to the 48-point shapes and the best correspondence is chosen over three shifts of the labels in either direction. The final column also shows results using the coarse-to-fine technique but with likelihood cut-off implemented when corresponding the coarse shapes. Implementing the speed-ups has little impact on the accuracy of nearest-neighbor classification. The ML classifier is significantly affected; the coarse-to-fine method produces a moderate reduction in accuracy (94.33 from 96.83) whilst likelihood cut-off leads to a dramatic decrease in performance. Since a test-shape is only compared to its correct class once in the ML case (the class prototype), it is not robust to the increased probability of the correspondence algorithm failing when a low threshold likelihood cut-off is used.

4 Discussion

We have proposed an efficient, correspondence-based shape classification algorithm for 2D boundaries. Shapes are represented by points placed at equal intervals along the boundary; hence, there is no heuristic procedure for choosing the points. A hierarchical matching (going from coarse to fine matches) and likelihood thresholding significantly reduce the cost of finding correspondences with minimal impact on classification accuracy. This ensures that easier to match shapes do not have to go through the complete correspondence computations. The algorithm was tested on the MPEG-7 shape database and was shown to outperform the popular contour

Fourier method. Our approach leads to an intuitive class model that can be used for fast prototype-based classification. The results using this method were comparable to that of standard nearest-neighbor classification. The approach was also shown to perform well in the commonly used bullseye retrieval test. Combining two simple shape descriptors further improved retrieval accuracy.

The ideas presented are most relevant to general shape classification and retrieval problems where there is insufficient knowledge to construct a priori models. Combining/choosing between different shape descriptors is difficult in retrieval since there is no training data available. However, relevance feedback (common in document-based information retrieval) is a realistic mechanism for gaining additional information and can be used to guide dynamic selection and weighting of shape descriptors. Future work will investigate this possibility.

References

- [Belongie *et al.*, 2002] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Trans. on Patt. Anal. and Machine Intell.*, 24:509–522, 2002.
- [Bookstein, 1996] F.L. Bookstein. Landmark methods for forms without landmarks: morphometrics of group differences in outline shape. *Med. Im. Anal.*, 1(3):225–243, 1996.
- [Chang *et al.*, 1991] C.C. Chang, S.M. Hwang, and Buehrer. A shape recognition scheme based on relative distances of feature points from the centroid. *Patt. Recog.*, 24(11):1053–1063, 1991.
- [Dryden and Mardia, 1998] I.L. Dryden and K.V. Mardia. *Statistical Shape Analysis*. Wiley, 1998.
- [Kent *et al.*, 2004] J.T. Kent, K.V. Mardia, and C.C. Taylor. Matching problems for unlabelled configurations. In *Bioinformatics, Images, and Wavelets*, pages 33–36, 2004.
- [Kunttu *et al.*, 2003] I. Kunttu, L. Lepisto, J. Rauhamaa, and A. Viss. Multiscale fourier de scriptor for shape-based image retrieval. In *Proc. of the 12th Int. Conf. on Patt. Recog.*, pages 536–541, 2003.
- [Latecki *et al.*, 2000] L. J. Latecki, R. Lakämper, and U. Eckhardt. Shape descriptors for non-rigid shapes with a single closed contour. In *IEEE Conf. on Comp. Vis. and Patt. Recog.*, pages 424–429, 2000.
- [Mardia *et al.*, 1979] K.V. Mardia, J.T. Kent, and J.M. Bibby. *Multivariate Analysis*. Academic Press, 1979.
- [Super, 2004] B.J. Super. Fast correspondence-based system for shape-retrieval. *Patt. Recog. Lett.*, 25:217–225, 2004.
- [Wang *et al.*, 2004] S. Wang, T. Kubota, and T. Richardson. Shape correspondence through landmark sliding. In *IEEE Conf. on Comp. Vis. and Patt. Recog.*, volume 1, pages 143–150, 2004.
- [Zhang *et al.*, 2003] J. Zhang, X. Zhang, H. Krim, and G.G. Walter. Object representation and recognition in shape spaces. *Patt. Recog.*, 36:1143–1154, 2003.

Efficient Distributed “Hormone” Graph Gradients.

Esben Hallundbæk Østergaard

Maersk Mc-Kinney Moller Institute for Production Technology

University of Southern Denmark

esben@mip.sdu.dk

Abstract

Several researchers have attempted extract the principles behind hormone gradients found in biological system, and apply them to control physically manifested distributed systems. This paper presents an efficient implementation of a graph-based version of hormone gradient mechanisms. The algorithm is based on hop-counting of the topological distance between any given vertex and the hormone emitting vertex, and can deal with dynamic changes to the topology of the system. Performance of the described algorithm is documented through a number of experiments. The algorithm was developed for use in self-reconfigurable robotics, but might very well be useful for many other applications. The use of the algorithm to provide a common coordinate system for a collection of self-reconfigurable robot modules is described, that provides pose relative to the emitting module for all modules affected by the hormone.

1 Introduction

Hormone gradients have a central role in the coordination of growth and self-repair in biological multicellular systems. Several researchers have attempted to use these same mechanisms to control physically manifested distributed systems, by using some abstraction over the perceived functioning of biological hormone gradients. This paper presents an efficient implementation of a graph-based version of hormone gradient mechanisms, based on hop-counting and spanning a tree over the topology, that can deal with dynamic changes to the topology of the system. The paper will also describe how the algorithm can be used to relay a common coordinate system for modules in a self-reconfigurable robot. Even though the focus was on self-reconfigurable robots during development, the algorithm is believed to have much wider applicability.

2 Graph-Based Hormone Gradients

As a first approximation, simple gradients are fairly easy to implement. Figure 1 illustration **a)** shows a graph representing a system of 11 vertices, with 11 edges. In **b)** a simple

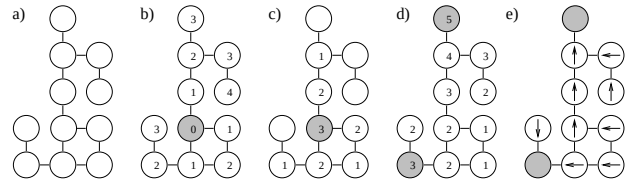


Figure 1: A few simple graph gradients

width first graph traversal algorithm (starting at the gray vertex) has been applied to assign a value to each vertex, where the value corresponds to the recursion depth. The result is that each vertex holds its distance to the gray vertex. In **c)** a similar algorithm is run, where a limit has been imposed on the recursion depth. The value assigned to vertices is the limit minus the recursion depth, resulting in the held value decreasing the further we get away from the gray module. This can be regarded as an abstract implementation of a hormone gradient emitted from the gray vertex. The hormone gradient is emitted with a given strength that defines the depth of the recursion (the maximum topological distance) and thus the number of affected vertices. The **c)** approach also allows us to have several emitting vertices, as shown in **d)** where two vertices are emitting a gradient, one with strength 3, and one with strength 5. In **d)** the graph algorithm is designed such that once a vertex has been assigned a value, only higher values can be assigned to that vertex.

To illustrate how this can be used as a gradient, in **e)** arrows point to the edge leading to the vertex with highest value. We see that if we start at any vertex in the graph and follow the arrows, we end up at one of the emitting vertices. Also we see that the vertex with a higher emission strength has a larger basin of attraction than the vertex with lower emission strength.

3 Agent-Based Hormone Gradients

The previous section briefly described how traditional graph traversal algorithms can produce a loose analogy to biological hormone gradients. However, in a physically distributed system we cannot directly use graph traversal algorithms.

This paper presents a distributed algorithm, providing information on topological distance in communication networks with dynamic changing topologies. Consider an agent

for each vertex in the graph, where agents asynchronously and without any global knowledge, efficiently realizes such graph hormone gradient information.

3.1 Related Work

Hormone gradients have been studied by biologists and have been found to be a key player in coordinating and controlling growth in plants and animals. Mainhardt and Gierers' [Meinhardt, 1993] [Meinhardt and Gierer, 2000] work on how hormone gradients control the forming of the tentacles of the Hydra animal describes how hormones are responsible for almost all aspects of shaping the animals body. Other relevant work includes Nishimura's [Nishimura and Sasai, 2004] work on chemo-taxis on cells in solutions with chemical gradients.

Several researchers have extracted and abstracted these mechanisms for use in the multi-robot domain. Amorphous computing [Abelson et al., 2000][Nagpal, 1999][Nagpal et al., 2003] is one example of research relying heavily on simulated hormone gradients, and many more examples exist, including work on cell division in artificial evolution [Hotz, 2004], and in self-reconfigurable robotics, such as Bojinov, Casal and Hogg's scents [Bojinov et al., 2000].

All the implementations of digital hormone gradients in the distributed robotics literature rely on continuous broadcasting of a hop-count number. When an agent receives a message with a hop count number, it remembers this number and starts emitting a message with an incremented or decremented hop count number, at regular intervals. A timer keeps track on when the last message matching the stored hop count number was received. If too long time has expired, the stored hop count number is erased, and the agents stop sending messages.

As a first approximation, this continuous broadcasting approach works fine. However, there are two main problems with this approach:

- Frequent emission of messages, even if there is no change to the topology of the agents.
- Continuous broadcast style approaches can fail to deal adequately with dramatic changes to the topology.

The first point has to do with efficiency in terms of power consumption and processing power. The second point, illustrated in figure 2, is more serious.

For these two reasons, it is desirable to develop a better algorithm.

4 Features of the Algorithm

The algorithm described in this paper is an extension of E. Gafni and D. Bertsekas' work described in the paper "Distributed Algorithms for Generating Loopfree Routes in Networks with Frequently Changing Topology" [Gafni and Bertsekas, 1981]. Their algorithm has a number of properties that makes it interesting for the purpose of this paper:

1. The algorithm is distributed and works on unknown communication topologies.
2. Their algorithm spans a directed acyclic graph (DAG) over the network topology, where all directed paths lead to the destination.

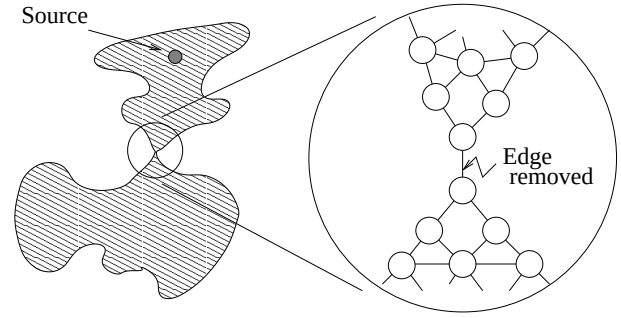


Figure 2: Illustration of a case where continuous broadcast-style graph gradients is problematic. Assume a gradient source in the upper part of the illustrated graph, emitting a decrementing gradient (type c) from figure 1) with high enough intensity to cover the graph. When the edge is removed, we would like gradient information to disappear from the amputated part of the graph. However, depending on the exact implementation of the broadcast-style approach, the gradient hop-count number might instead drift (slowly) towards infinity or zero. The problem is, that when one vertex has timed out, it might receive a new gradient message from one of its neighbors, keeping the gradient "alive".

3. In the event of changes to the topology of the network, the algorithm creates a new DAG using an iterative method.

From a functional point of view, the algorithm presented in this paper is an improved version of the algorithm presented by Gafni and Bertsekas, with two additional features;

4. From a given vertex, the length of any directed path to the destination is equal to the shortest undirected path to the destination.
5. Each vertex knows its topological distance from the destination.

4.1 Definitions

The algorithm described below works on a graph, to create a loose analogy to chemical/hormone gradients. The algorithm is completely distributed, and is able to deal with frequently changing topology, while at the same time using very little communication. Consider an *agent* for each vertex v in the graph $G = (V, E)$. Agents have a number of ports, through which they can communicate to other agents. The algorithm uses two features per agent. Agents can emit a hormone with strength s and sense the intensity i of the hormone. Both s and i are in $\mathbb{N}$, the set of natural numbers. We require agents to know which of their ports are connected to other agents, and which are not. Given this, a digital hormone gradient can be defined as a four-tuple, $\{G, I, S, P\}$, where:

- $G = (V, E)$ is a graph, where vertices are elements of the configuration, and edges are physical connections between the elements.
- $I : V \rightarrow \mathbb{N}$ is the hormone intensity (concentration) mapping, so that $i = I(v)$ is the hormone intensity for vertex v .

- $S : V \rightarrow \mathbb{N}$ is the hormone emission strength mapping, where $s = S(v)$ is the strength s with which vertex $v \in V$ emits the hormone.
- $P : (V \times \mathbb{N}) \rightarrow (\{\bar{\varnothing}\} \cup V)$ is the port function. The port function holds information for the state of each of the agent's ports, so that $P(v, n)$ is the state of the agent for vertex v 's n 'th port. The symbol $\bar{\varnothing}$ is used to specify a port that is not connected, such that $\forall n \in \mathbb{N} : \forall v \in V : \forall v' \in P(v, n), \{v, v'\} \in E \Leftrightarrow P(v, n) \in V$.

The four-tuple $\{G, I, S, P\}$ describes the global state of the hormone gradient, but we are aiming at a distributed algorithm with only local information. For any given vertex $v \in V$, the local gradient state information available is $i = I(v)$, $s = S(v)$ and for each of the j ports, $p_n = P(v, n)$ where $0 \leq n < j$. Besides this information, the algorithm requires agents to remember their parent in the spanning tree $-1 \leq f < j$ (using -1 for no parent), a timer/counter $t \in \mathbb{N}$ and an array for storing information on which ports to send messages to, $m[n] \in \{NONE, PROPAGATE\}$.

The algorithm basically works by maintaining local information on the spanning tree, such that the variable f always points to the port from which the gradient is flowing. The timer/counter is used to adjust the relative speeds of propagating gradient and deleting the gradient. Without this adjustment continuous propagation and deletion of the gradient in local cyclic areas of the graph might arise, causing much communication and slow hormone deletion. However, this adjustment introduces a parameter that is the number of time steps to delay propagate messages when spreading the gradient. Lower values cause the algorithm to be faster but more unstable.

The algorithm for each agent, implementing the desired graph-based hormone gradient behavior, is given below. Note that neither global communication nor synchronization between agents is assumed.

The **—initialize—** function resets the variables of the agent. **—emitGradeint—** is used to set the hormone emission strength of the agent. Setting to zero clears the gradient. **—updateGradient—** holds the central functionality such as sending messages, and should be called at regular time intervals. The frequency at which it is called determines the hormone propagation speed in the system. The four event functions **—connectEvent—**, **—disconnectEvent—**, **—propagateMessageRecievedEvent—** and **—deleteMessageRecievedEvent—** are very light-weight and can be called in some interrupt routine whenever the corresponding event is detected.

```

—initialize—
i := 0; s := 0; f := 0; t := 0;
foreach n ∈ {0..j - 1} do
    m[n] := NONE;
end

```

```

—emitGradient(s')—
s := s';

```

```

—connectEvent(n)—
m[n] := PROPAGATE;

```

```

—disconnectEvent(n)—
if n = f then
    f := -1;
end

```

```

—updateGradient—
if s > i then
    i := s; f := -1;
    foreach n ∈ {0..j - 1} do
        m[n] := PROPAGATE;
    end
else if s < i ∧ f = -1 then
    foreach n ∈ {0..j - 1} do
        if pn ≠  $\bar{\varnothing}$  then
            sendDeleteMessage(n, i-1);
        end
        m[n] := NONE;
    end
    i := 0;
else if t > 0 then
    t := t - 1;
    if t = 0 then
        foreach n ∈ {0..j - 1} do
            if n ≠ f then
                m[n] := PROPAGATE;
            end
        end
    end
else
    foreach n ∈ {0..j - 1} do
        if pn ≠  $\bar{\varnothing}$  ∧ m[n] = PROPAGATE ∧ i > 1 then
            sendPropagateMessage(n, i-1); t := 0;
        end
        m[n] := NONE;
    end
end

```

```

—propagateMessageRecievedEvent(n', i')—
if i' > i then
    i := i'; f := n'; t := 3; // This constant (3) defines the
    delay before propagating the gradient, to allow delete
    messages to "catch up".
else if i' < i - 1 then
    m[n'] := PROPAGATE;
end

```

```

—deleteMessageRecievedEvent(n', i')—
if i' ≥ i then
    f := -1;
else
    m[n'] := PROPAGATE;
end

```

4.2 Tests of the Algorithm

The algorithm was implemented and tested on graphs with varying sizes and numbers of loops, shown in figure 3. To test the robustness of the algorithm, the different activation schemes described in [Butler *et al.*, 2002], D_0 , D_1 and D_∞ , were used for the agents. In D_0 , agents are activated in a fixed order each time step, in D_1 the agents activation order is permuted for each time step, and in D_∞ a random agent is picked and activated n times each time step, where n is the number of agents. Figure 4 shows the number of messages passed as a function of the number of modules for three different graph types, using the D_∞ activation scheme. The

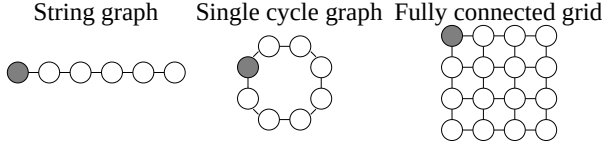


Figure 3: The three graph types used to test performance of the distributed hormone algorithm. The gray vertex is the source of the gradient.

data shows that the algorithm has good performance, in that the number of messages passed is linearly proportional to the number of edges in the graphs, and that the number of time steps used to propagate the hormone is proportional to the diameter of the graph.

A different set of experiments were performed for counting the number of messages used to remove the gradient again, by setting the emission strength of the emitting agent to zero. These results are shown in figure 5. The results show that the number of messages generated for removing a hormone is linearly proportional to the number of involved graph edges. Notice that removing the hormone is faster than covering the graph, due to the delay on propagating the gradient.

5 Pose Hormone Gradients

The previous section described the basic graph hormone gradient algorithm. This section describes how the algorithm has been used for controlling two self-reconfigurable robot systems, shown in figure 6. Self-reconfigurable robot control is a good application test bed for this kind of hormone gradient algorithms, because of the constantly changing topology in such a robot, and because of similarities between self-reconfigurable robots and multi-cellular organisms.

5.1 Common Pose for Self-Reconfigurable Robot Modules

The distributed hormone gradient algorithm described above can be expanded slightly to be used for propagating pose (position and orientation) relative to the emitting vertex in a self-reconfigurable robot. Basically, the expanded algorithm uses feed-forward kinematics in the generated directed acyclic graph to compute the relative pose to the emitting vertex. To realize this, it is a requirement that the relative pose between two connected vertices in the configuration can be found. We let the space $\mathbb{P}$ be the space of relative poses in a 3D vector space.

Then, a relative pose hormone is a four-tuple, $\{G, \mathcal{I}, \mathcal{S}, \mathcal{P}\}$ where:

- $G = (V, E)$ is a graph, where vertices are elements of the configuration, and edges are physical connections between the elements.
- $\mathcal{I} : V \rightarrow (\mathbb{N} \times \mathbb{P})$ is the hormone intensity (concentration) mapping, so that $i = I(v), i = \langle i_n, i_p \rangle$, where $i_n \in \mathbb{N}, i_p \in \mathbb{P}$ is the hormone intensity i_n and relative pose i_p for vertex v .

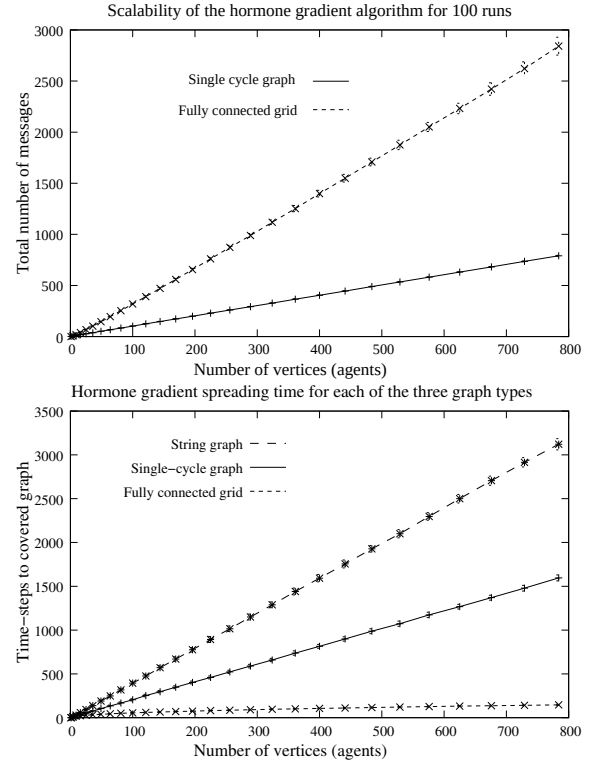


Figure 4: **Top:** Message count for covering a graph with the hormone gradient, for varying number of vertices. Results for the string graph type is not shown, since they turned out to be the same as the results for the single-cycle graph. The variance is caused by using the D_∞ activation scheme. **Bottom:** The number of time steps it takes to fully cover the graph for each of the three graph types and for varying graph sizes.

- $\mathcal{S} : V \rightarrow (\mathbb{N} \times \mathbb{P})$ is the hormone emission mapping, where $s = \mathcal{S}(v), s = \langle s_i, s_p \rangle$ is the emission strength (s_i) and the emitted pose (s_p) for vertex v .
- $\mathcal{P} : (V \times \mathbb{N}) \rightarrow (\{\bar{\varnothing}\} \cup (V \times \mathbb{P}))$ is the port function. The port function holds information for the state of each of agent's ports, so that $P(v, n)$ is the state of vertex v 's n 'th port. The symbol $\bar{\varnothing}$ is used to specify a port that is not connected, such that $\forall n \in \mathbb{N} : \forall v \in V : \forall \langle v', p' \rangle \in P(v, n) : \{v, v'\} \in E \Leftrightarrow P(v, n) \in V$. Here, p' is the relative orientation between the two connected elements, such that if $p_{v'}$ is the pose of v' and p_v is the pose of v , then $p_{v'} \times p' = p_v$.

For any given module $v \in V$, the local gradient state information available is $i = \langle i_n, i_p \rangle = \mathcal{I}(v), s = \langle s_i, s_p \rangle = \mathcal{S}(v)$ and for each of the modules j connectors, $p_n = P(v, n)$ is either $\bar{\varnothing}$ or $\langle v', p' \rangle$, for $0 \leq n < j$.

The principle is to do feed-forward kinematics in the directed acyclic graph generated by the graph hormone algorithm. As a consequence, each module affected by the hormone knows its relative position and orientation to the hormone source. It turns out that by exploiting that each module knows its relative position to the source, it is possible to re-

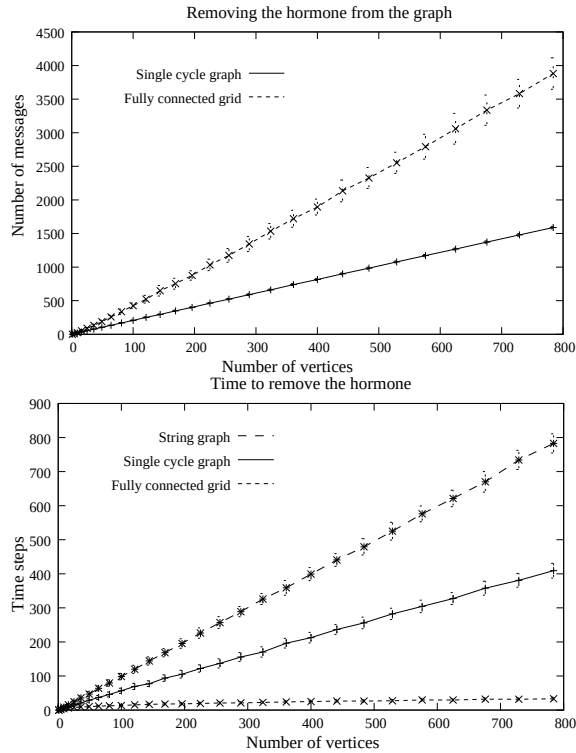


Figure 5: Tests on removing the hormone from the graph by setting the emission strength of the emitting vertex to zero, after the hormone has fully propagated. The two graphs show the number of messages generated and the time used for the three graph families and for varying graph sizes. In the top graph the string graph results were omitted since they are identical to the single cycle results. Notice that removing the hormone is faster than propagating the hormone, but produces more communication and has higher variance. This relationship can be adjusted by changing the propagation delay.

move the delay for propagate messages by adding an extra condition for the case of receiving a propagate message. The implementation details of the pose hormone gradient algorithm has been omitted due to space constraints.

6 Implementation and Experiments

The pose hormone algorithm has been implemented and tested on simulations of the ATRON system, figure 6, left, and on the M-TRAN system, figure 6, right.

Both the ATRON and the M-TRAN modules are composed of two parts, where the parts are connected by an actuated joint. The algorithm is implemented, such that each part of the modules corresponds to a vertex in the graph. The following will describe some experiments done on the pose hormone algorithm on the simulated M-TRAN robots.

6.1 M-TRAN, Message Count

Three test configuration designs are shown in figure 7. Each configuration comes in a form with 4, 32, 108, 256, 500 and

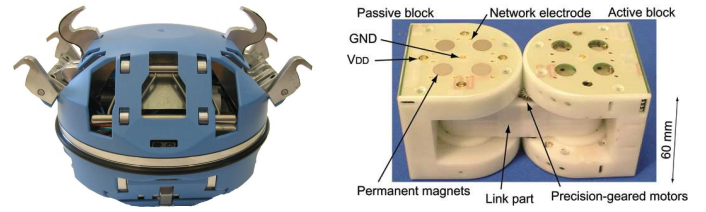


Figure 6: Module for two self-reconfigurable robot systems. **Left:** The ATRON system. **Right:** The M-TRAN System.

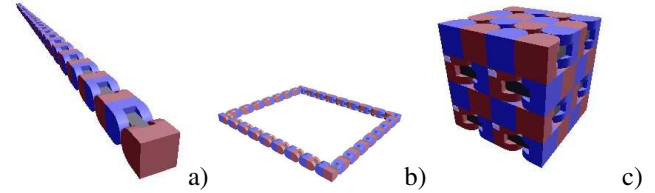


Figure 7: The three test configurations used in the synthetic tests. Here composed of 32 modules each.

864 modules. The three designs differ in the number of loops in the topology, where a) has no loops, b) a single loop and c) has many loops.

An experiment proceeds as follows:

1. A module in one end/corner is set to emit a hormone gradient with an emission strength of 1000.
2. Data is collected on the number of messages used to fully propagate the gradient, and on how many time steps it took.
3. Then the emission strength of the emitting module is set to zero.
4. Similarly to step two, data is collected.

The algorithm was tested using the D_0 , D_1 and D_∞ activation schemes. Ten experiments were performed for each combination of configuration and activation scheme, and mean and variance were computed. The correctness of the pose information is easily verified by placing the emitting module such that the emitted pose is identical to the pose in the simulator. In this case, the relative pose hormone for any module should be identical with its pose in the simulator (provided appropriate scaling). Figure 8 shows the average number of messages, and figure 9 the communication load per connector, using the D_1 activation scheme. For deleting the hormone gradient, the number of messages passed by any connected connector was constant two.

Besides these simple tests, the algorithm was used a lot during work on control of the two self-reconfigurable robot systems, to produce a common coordinate frame of reference for locally cooperating groups of modules.

7 Discussion & Conclusion

This paper describes an efficient algorithm for realizing an abstract implementation of hormone gradients. Hormone gra-

Test Configuration	a)			b)			c)		
# modules	4	32	108	4	32	108	4	32	108
# connections	3	31	107	4	32	108	4	64	252
# messages for propagating	7	63	210	8	66	221	9	115	455
# message for deleting	14	126	420	16	128	432	16	193	720

Figure 8: The average number of messages (rounded to nearest integer) used to propagate and delete a digital hormone gradient for nine different configurations using the D_1 activation scheme. The messages passed between “agents” within the same module are included in the data, so since each module has two “agents”. The number of messages to propagate the gradient in configuration a) is exactly the number of modules plus the number of connections between modules. The data also shows that the number of messages for deleting a gradient is approximately double that of propagating the gradient.

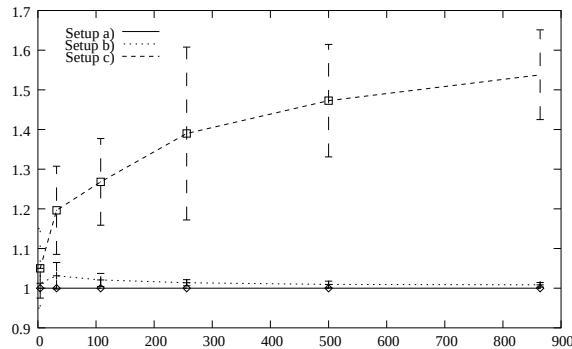


Figure 9: Average message count per connector for propagating the hormone gradient for each of the configurations, using the D_1 activation scheme. The graph shows that configuration design c) with many topology graph cycles causes more messages per connector than a) and b). However, the message count per connector is still lower than two.

dients are widely used in biology to coordinate and differentiate cellular behavior.

The described algorithm is based on topological distance, and is capable of dealing with changing topology in the connectivity graph using only local information and without requiring global synchronization.

The algorithm has properties similar to that of Gafni and Bertsekas, with the added benefit that the spanning DAG is, in a sense, optimal, in that any directed path to the source has the same length as the shortest undirected path. Additionally, in contrast to the commonly used continuous broadcast method, the presented algorithm only generates communication when the topology of the graph changes, or when the gradient information changes. Furthermore, only a constant (and small) number of messages is generated per graph edge, in the event of any change to the hormone gradient. Less communication results in reduced hardware requirements, that again can reduce requirements to power, processing, price and size.

Performance of the described algorithm is documented through a number of experiments. An expansion of the hormone gradient algorithm is given which can be used to provide all self-reconfigurable robot modules affected by the hormone their pose relative to the emitting module. The algorithm was developed for use in self-reconfigurable robotics, but might very well be useful for many other applications. Future work will include further analysis of performance and analytical proofs.

References

- [Abelson *et al.*, 2000] H. Abelson, D. Allen, D. Coore, C. Hanson, G. Homsy, T. Knight, R. Nagpal, E. Rauch, G. Sussman, and R. Weiss. Amorphous computing. *Communications of the ACM*, 43(5):74–82, 2000.
- [Bojinov *et al.*, 2000] H. Bojinov, A. Casal, and T. Hogg. Multiagent control of self-reconfigurable robots. In *Fourth International Conference on MultiAgent Systems (ICMAS)*, pages 143–150, Boston, Massachusetts, USA, July 2000.
- [Butler *et al.*, 2002] Z. Butler, K. Kotay, D. Rus, and K. Tomita. Generic decentralized control for a class of self-reconfigurable robots. In *Proceedings, IEEE Int. Conf. on Robotics and Automation (ICRA)*, volume 1, pages 809–815, Washington, DC, USA, 2002.
- [Gafni and Bertsekas, 1981] E. Gafni and D. P. Bertsekas. Distributed algorithms for generating loopfree routes in networks with frequently changing topology. *IEEE Trans. on Comm.*, COM-29:11–18, 1981.
- [Hotz, 2004] P. E. Hotz. Asymmetric cell division in artificial evolution. In *Congress on Evolutionary Computation (CEC)*, 2004.
- [Meinhardt and Gierer, 2000] H. Meinhardt and A. Gierer. Pattern formation by local self-activation and lateral inhibition. *Bioessays*, 22(8):753–60, Aug 2000.
- [Meinhardt, 1993] H. Meinhardt. A model for pattern formation of hypostome, tentacles, and foot in hydra: how to form structures close to each other, how to form them at a distance. *Developmental Biology*, 157(2):321–333, June 1993.
- [Nagpal *et al.*, 2003] R. Nagpal, A. Kondacs, and C. Chang. Programming methodology for biologically-inspired self-assembling systems. In *Proceedings, AAAI Spring Symposium on Computational Synthesis: From Basic Building Blocks to High Level Functionality*, 2003.
- [Nagpal, 1999] R. Nagpal. Organizing a global coordinate system from local information on an amorphous computer. *AI Memo 1666*, 1999.
- [Nishimura and Sasai, 2004] S. I. Nishimura and M. Sasai. Chemotaxis of an eukaryotic cell in complex gradients of chemoattractants. In *The ninth international symposium on artificial life and robotics (AROB)*, Beppu, Oita, Japan, January 2004.

Building Patterned Structures with Robot Swarms

Justin Werfel^{1,2}

jkwerfel@mit.edu

Yaneer Bar-Yam^{2,3}

yaneer@neesi.org

Radhika Nagpal³

rad@eecs.harvard.edu

¹MIT CSAIL

Cambridge, MA 02139

²New England Complex Systems Institute

Cambridge, MA 02138

³Harvard University

Cambridge, MA 02138

Abstract

We describe a system in which simple, identical, autonomous robots assemble two-dimensional structures using prefabricated modules as building blocks. Modules are capable of some information processing, enabling them to share long-range structural information and communicate it to robots. This communication allows arbitrary solid structures to be rapidly built using a few fixed, local robot behaviors. Modules are identical in shape but may be functionally distinct, with constraints governing the location of different classes. We present algorithms for assembly of solid structures of any shape, both when the layout of module classes is fully specified in advance, and when functional constraints are satisfied during the building process, allowing for adaptive structures. This approach demonstrates a decentralized, autonomous, flexible, simple, and adaptive approach to construction.

1 Introduction

In this paper we discuss the design of a system for automated construction, in which elements are separated into mobile and structural components (robots and modular blocks, respectively). Our approach is to use robots that are simple in capabilities and behavior, and that communicate indirectly through information stored in the environment; that information is embodied by the blocks, which themselves can communicate with one another when attached. Here we focus on construction when the blocks are identical in shape but heterogeneous in function, and a given set of functional constraints governs how they can be put together. We first show how specific structures can be built, and then discuss the assembly of structures which have no prespecified plan but satisfy a given set of constraints.

Our approach relies on very simple procedures and therefore should be practical for the complexities and difficulties of real robots in physical environments. The limited requirements for implementation are that robots be able to follow a perimeter, communicate with immediately adjacent blocks, and push together blocks with self-aligning connectors. Problems, from unexpected environmental influences to break-

down of some robots, need not prevent completion of a structure; temporary failures at any step can be corrected.

The ability to automate construction would be useful particularly in settings where human presence is dangerous or problematic; for instance, robots could be initially sent to underwater or extraterrestrial environments, to create habitats to await later human travelers. Swarm approaches, involving larger numbers of simpler robots rather than one or a few with greater sophistication, have advantages for such a goal, in particular with respect to decentralization and robustness. Such systems can typically absorb the loss of many components without a significant impact on task completion; similarly, they tolerate components acting in no specified order, a useful quality since it is difficult to preplan robot behavior in detail in uncertain environments.

If blocks can be arbitrarily custom-designed for particular structures, then any structure can be built using only local rules for assembly; a trivial, inefficient way to do it is to designate specific blocks for specific locations in a blueprint, and engineer the blocks so that each one can be attached solely to the neighboring blocks in that layout.¹ However, we take the opposite approach, assuming that the block types are defined by the application, and our task is to work within those limits.

An example of such an application is the building of underwater structures (for marine research, oil drilling, etc.), where the building blocks are high-level prefabricated pods that each fill one of several distinct and predefined roles—living quarters, power generation, emergency escape centers, laboratory space, etc. There may be desired constraints on which classes are attached relative to which other classes—for instance, using the previous example, we might want all living pods to be located in a contiguous block, or no two airlock pods to be adjacent to one another.

In §2 we review related work, and in §3 describe the problem we address and our assumptions about the system. §4 gives algorithms the system can use to assemble structures. §5 covers geometric constraints on block placement. §6 and §7 describe structures which are fully prespecified or adaptive, respectively, and elaborate on the satisfaction and specification of functional constraints. §8 discusses issues associated with the use of multiple robots.

¹Geometric constraints will still restrict the order in which blocks may be attached, as discussed in §5.

2 Related work

Several previous papers address topics related to construction, though not with the goal of producing user-specified building designs. For example, [Wawerla *et al.*, 2002; Jones and Matorić, 2004] study aspects of the use of communication among robots, to increase their effectiveness on tasks involving rearrangement of functionally heterogeneous blocks. [Bowyer, 2000] discusses issues of mechanical design for robots meant to build arches, towers, and walls, using extruded foam. [Melhuish *et al.*, 1999] deals with minimization of capabilities for behavior-based robots arranging pucks into walls, using environmental cues.

[Werfel, 2004] describes a framework for arranging blocks into arbitrary lines and curves, specified geometrically; [Mason, 2002] describes one in which structures can be specified through the use of static, global ‘pheromone’ signals. Both make assumptions about the precision with which their mobile agents operate (the former having to do with odometry, the latter with ability to evaluate the strength of distant signals) that may cause the shapes they generate to suffer from lack of robustness to noise.

Our approach is particularly motivated by research in two areas: insect and insect-inspired construction [Théraulaz and Bonabeau, 1995], and self-reconfigurable modular robotics [Rus and Chirikjian, 2001]. Social insects which build do so with the principle of stigmergy, communicating indirectly by storing information in the environment. While this approach can be used to produce structures with given qualitative characteristics, it does not easily allow the consistent production of *specific* structures (potentially arbitrary and complicated ones); nor has a general principle been described for taking a particular desired structure and extracting a set of low-level behaviors that building agents can follow to produce it. Work in modular robotics has produced hardware systems with many capabilities we will take advantage of here, including connections that self-adjust so that inexact alignment is corrected, and communication links between physically attached modules. Such communication is reliable, unambiguous, rapid, and scalable, compared to the communication by external signaling that robots typically use. Modular robot algorithms typically require that all modules remain connected at all times, and that individual modules be capable of mobility, neither of which are appropriate for automated construction applications. Very recently, modular robotics researchers have begun to describe hardware systems for automated assembly of structures [Terada and Murata, 2004; Everist *et al.*, 2004]. These systems are based on inert, identical modules, and have not yet explored issues of specifying distributed robot behavior and creating complex user-specified structures.

Elsewhere we describe a similar system to that presented here [Werfel *et al.*, 2005], concerning identical blocks only. There we show that giving blocks limited communication abilities is an effective way to enable the system to construct arbitrary solid structures, while allowing robots to remain simple and with fixed, local behaviors. The block communication provides the nonlocal knowledge about the structure necessary to assemble arbitrarily complicated shapes.

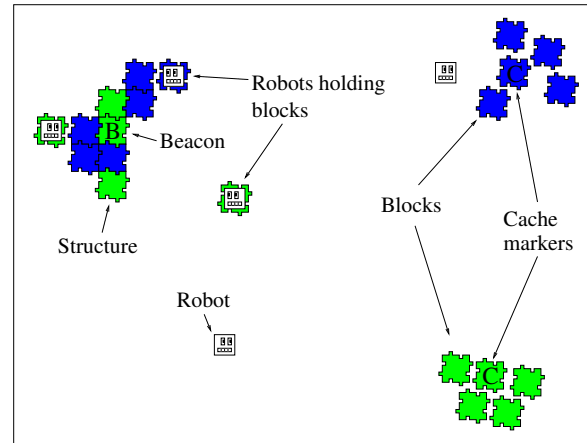


Figure 1: A sketch of the kind of system described here. As the structure is assembled, it forms a lattice with an implicit coordinate system, of which the beacon can be taken to be the origin.

This approach provides other benefits, including increased efficiency of construction, alleviation of interference between robots, and better exploitation of the parallelism of the swarm, without requiring additional capabilities from the robots. As computation becomes less expensive, it has been proposed to make it more pervasive [Hill *et al.*, 2000]. Particularly if the building blocks in question are high-level units, as in the underwater application suggested above, it should be feasible to incorporate these computation capabilities into the blocks without making the system prohibitively more expensive.

3 General problem and assumptions

We consider the problem where mobile robots and caches of building blocks are deployed at random into an obstacle-free workspace, along with a beacon indicating the location for the start of construction. The goal is for robots to collect blocks from the caches and arrange them into a structure, satisfying some set of criteria, starting at the beacon (Fig. 1). The beacon and caches send out distinct long-range signals which robots can use to navigate to them. Each cache contains a single block type.

We will work with square blocks, to be assembled in the horizontal plane. An important constraint is that a block can be added to the growing structure if and only if the potential attachment site has at least two adjacent sides open; otherwise there is insufficient room to maneuver to add the new block (Fig. 2). Though a system has been demonstrated in which cubical blocks can be slid into spaces like that at (D) in that figure [Terada and Murata, 2004], the task of mechanical design will be simplified and the precision with which robots must operate reduced if we maintain this constraint. In particular, if we prevent gaps like that at (D), then situations where a robot needs to maneuver a block down a longer ‘tunnel’, like that around (C), will also be prevented. A block attached to the structure can immediately obtain from its neighbors its

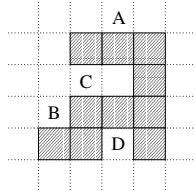


Figure 2: Examples of valid and invalid prospective docking sites for a sample structure (shaded squares represent grid sites already occupied by blocks). A new block can be attached at A or B; C and D are too spatially constrained to allow a block to be maneuvered into position.

position in a common coordinate system, along with information about the present and desired final structure. Blocks along the perimeter of the structure and robots traversing the perimeter can also communicate, with lower bandwidth; this communication is very short-range, avoiding problems associated with signaling over distances, interference when many robots are active, and ambiguity in which agents are signaling.

Robots can move in any direction in at least the two dimensions of the plane, alone or while holding a block. They can detect the presence of other robots, for collision avoidance, either by active short-range signaling or by passive perception. Once in the immediate vicinity of the growing structure, they can follow its perimeter, and communicate with adjacent blocks.

4 Algorithms for blocks and robots

Blocks attached to each other to form the structure have access to information about the entire structure, via communication with other blocks. Our approach will be for blocks to be the agents responsible for ensuring that the structure remains consistent with all the constraints. Robots can then simply circle the perimeter of the structure, and rely on it to tell them when a site is acceptable for attachment of their payload.

We can divide the constraints on block placement into two classes: those independent of block type, and those based solely on block type. Geometric constraints, the former, ensure that the structure is free of unwanted internal gaps and has a perimeter of the desired shape. Functional constraints, the latter, encompass whatever restrictions based on block types may be dictated by the particular application. Both classes are discussed further below (§5–7).

Algorithm 1 summarizes high-level procedures for robots and blocks to follow to build structures using this approach.

4.1 Robustness

Physical robots in real-world environments are subject to a wide variety of internal and environmental perturbations. A hardware implementation would need to be robust to such factors. Our approach is intended to minimize the effect or likelihood of problems that robots and blocks might encounter. It relies on simple basic capabilities that can be made robust and self-correcting: directed movement, block

Algorithm 1 High-level pseudocode procedure for assembly of a close-packed structure by a single robot.

A: Robots

```

while structure not complete do
  choose a block type  $t$ 
  get  $t$  from cache
  go to structure perimeter
  while still holding block do
    ask adjacent structure blocks if  $t$  can be attached here
    if all structure blocks answer yes then
      attach block here
    else if taking too long to find a place for block then
      return block to cache
    else
      move one site counterclockwise along perimeter

```

B: Blocks

```

loop
  if neighboring robot asks to attach a block of type  $t$  at site  $s$  then
    if (block at  $s$  consistent with geometric constraints)
      and ( $t$  at  $s$  consistent with functional constraints)
    then
      answer yes
      update map with  $t$  block at  $s$ 
      pass message about  $(t, s)$  to neighboring blocks
    else
      answer no
  if neighboring block sends message about  $t$  at  $s$  then
    if message represents new information then
      update map with  $t$  block at  $s$ 
      pass message about  $(t, s)$  to neighboring blocks

```

pickup, perimeter following, block joining, limited communication with immediate neighbors.

Noise affecting robot movement, e.g., due to motor unreliability or currents in an underwater setting, can be compensated for. For instance, if robots lose track of the perimeter during their tour, they need only revert to a previous step of their algorithm, find the perimeter again and proceed from there. The lack of robot internal state means no important information is lost in such an event. Self-aligning connectors drawn from modular robotics effect the fine details of block attachment. Communication between robots and blocks can be physically limited, e.g., through appropriate geometric design of an optical interface, to ensure that robots only communicate with immediately neighboring blocks and that crosstalk with more distant blocks is prevented. Interblock communication is via a physical interface.

More serious robot failures can also be absorbed. In keeping with the swarm approach, the algorithm does not depend on particular robots completing particular assigned tasks, nor on blocks being attached in a particular order. Thus breakdowns of individual robots almost anywhere in the workspace need not impede the system's ability to complete the structure. The exception is at the edge of the growing structure, where a broken robot could serve as an obstacle to others following the perimeter, and prevent growth of the structure in

that area. Such perimeter breakdowns would need to be detected and towed away by other robots.

Blocks, less complicated mechanically, may be less subject to failure. Malfunctions that do occur could be detected by neighboring blocks, which could alert robots to the need for replacement. Similarly, in the event that a block is mistakenly attached to the structure so as to violate a constraint, the active nature of the blocks makes it easier to detect and report that error. §9.2 outlines a scheme for disassembly and reconstruction of the necessary part of the structure.

5 Geometric constraints

Positions can be specified by reference to the common coordinate system which blocks share via their communication.

5.1 Rectangular structures

The simplest possible perimeter is a rectangle; the geometric constraint enforcing that shape is simply a restriction against robots attaching blocks at sites with coordinates outside the desired rectangle.

The other geometric constraint to be followed is the generally applicable one that the structure should be solid, without internal holes. To avoid unwanted gaps, it is necessary to avoid situations where two blocks end up in the same row, unless they are adjacent. A block can check, based on its communication with the other blocks in its row [Werfel *et al.*, 2005], whether a potential attachment site satisfies this criterion, and forbid robots from attaching further blocks at that site if it does not. This rule is sufficient to generate solid structures.

5.2 Complex shapes

With only a small modification, the system extends to solid structures with perimeters which are not necessarily rectangular, nor even convex, but can be of any arbitrary shape.² As usual, the desired shape is specified with respect to the shared coordinate system and downloaded by each new block attached to the structure. To achieve the extension to arbitrary shapes without letting unwanted gaps creep in, it is sufficient to relax the geometric rule against attaching two non-adjacent blocks in the same row, to allow two such blocks so long as they are separated by space intended never to be occupied by blocks. The desired shape can be represented compactly, e.g., as a collection of rectangles whose superposition gives the desired structure [Støy and Nagpal, 2004]. Fig. 3 gives an example of an H-shaped structure being built, with geometric constraints only.

6 Functional constraints: Patterned structures

An important aspect of such a construction system, hitherto unaddressed, is the ability to create predefined patterns of block types. This can be achieved with functional constraints that act as a blueprint, specifying exactly what block type

²We will require that any spaces between blocks in the desired final structure be wide enough for two block-carrying robots to pass each other, so that perimeter-following can continue unimpeded.

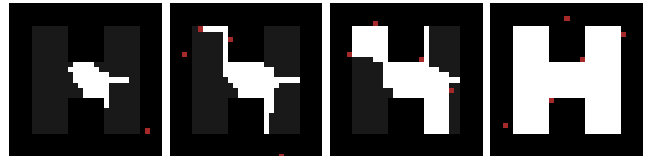


Figure 3: Snapshots of five robots (brown) building a structure of a single block type (white), using only geometric constraints. Sites that the structure plan specifies should eventually be occupied by blocks are shown in dark gray. Robots not appearing in a particular frame are off fetching blocks.

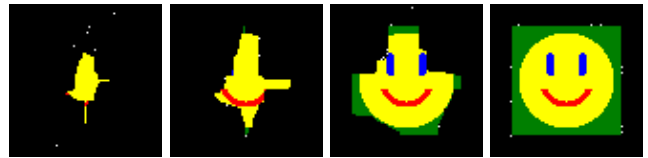


Figure 4: Snapshots of ten robots (white) building a structure of four block types (yellow, red, blue, green).

should be attached at every location in the desired structure. This specification can be more compact than a full enumeration, just as an image specification can be more compact than a bitmap. Structure blocks forbid robots to attach the blocks they carry unless the latter are the single type allowed at the site in question. Figs. 4 and 5 shows rectangular and non-rectangular structures built using such functional constraints.

7 Functional constraints: Adaptive structures

In many cases the design of structures does not require specific patterns, but rather requires functional relationships between the locations of blocks of various types. For instance, with the underwater habitat example we gave above, it may be that the exact locations of escape pods are unimportant, but safety considerations require that no pod be further than three steps away from one. Any structure that satisfies that constraint will be considered acceptable. The more relevant way to specify the constraint in this case is by reference not to absolute coordinates, but to relationships between the block types. Our approach extends naturally to building structures in this flexible way. Figs. 6 and 7 show structures built using relative constraints of this sort.

It is always a valid approach to come up with a blueprint ahead of time that satisfies all such constraints, give that to the system, and have it produce that particular structure. But satisfying constraints on the fly during the building process instead can let the structure be adaptive, and respond to changing circumstances or conditions unknown in advance.

Satisfying constraints on the fly can have additional benefits, such as increased speed of construction. As an example, consider the class of 21×21 -block structures of Fig. 6. If we assume that robots take no time to bring blocks from caches to the structure perimeter, and moving one block-length and attaching a block each take one time step, then the time required to build a structure will reflect primarily the



Figure 5: Example of a patterned structure with a complex shape.

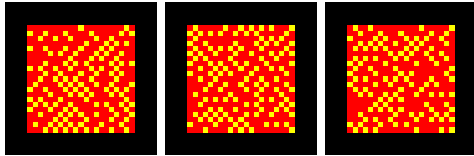


Figure 6: Examples of square structures built with two block types and the constraint that no two yellow blocks be adjacent.

time robots spend following the perimeter looking for a place to put their block. Averaging over 100 independent runs with 10 robots, on-the-fly construction completes in 650 ± 140 time steps, while building exactly the same 100 structures as pre-specified patterns takes 1100 ± 300 steps, or 70% slower on average. Robots come to the structure already carrying some type of block; the speedup for on-the-fly construction occurs because sites can accept any type of block consistent with the constraints, rather than having to wait for a particular block type to appear, and robots can accordingly find places to attach their blocks more readily.

7.1 Satisfiability

A task involving fitting a given set of blocks into a given shape subject to given constraints may have no solution; and if one exists, it may be hard to find. In general, the problem of finding a solution to such a scenario, or showing that none exists, is NP-complete [van Emde Boas, 1997]. However, many cases of interest will have the property that a solution can be found quickly by a straightforward trial, and no exhaustive search is necessary. The system here provides a natural and effective way of assembling structures for constraint sets that are easy to satisfy in this sense. Constraint sets difficult to satisfy would best be handled by finding a solution prior to construction, and building that solution as a blueprint.

7.2 Types of functional constraints

In general, there are several classes of functional constraints we can consider:

- *Absolute*, without reference to neighboring blocks: e.g., type A may be attached; B is forbidden.
- *Proximity*, where the types of neighboring blocks are important but their locations are not: e.g., every A must have at least one B somewhere within a neighborhood of radius 3; no C can be adjacent to a D.

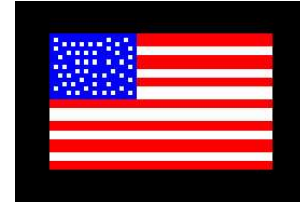


Figure 7: Example of a structure assembled according to the following list of constraints:

Region 1, border of upper-left area: only blue blocks allowed.
Region 2, interior of upper-left area: blue and white blocks both allowed; no white block may be in the eight-neighborhood of another white block.
Region 3, elsewhere: blocks must be either red or white depending on their y-coordinate.

- *Relational*, involving both types and locations of neighboring blocks: e.g., every A must have a B bordering its west edge and a C bordering its north edge. Rotation and reflection may each be allowed or forbidden in such a constraint.

Proximity and relational constraints imply an associated distance d within which blocks are relevant to satisfaction of the constraint. One approach is for blocks to maintain a local map of their area of the structure, whose size is at least of radius d . Blocks refer to this map to determine whether robots should be allowed to attach blocks at a neighboring site; when a new block is attached, a message to that effect is sent to blocks within d , for them to update their maps accordingly. The larger the value of d , the more memory is required of blocks, but the farther-reaching constraints may be.

Another type of functional constraint can result from finite numbers of each block type. For instance, there might be only two blocks of type A, which must be attached adjacent to one another. The first A attached to the structure could be added anywhere, but the second would have to be next to the first. In order to give an A-carrying robot the proper instruction, blocks would need to establish whether another A had already been attached anywhere, a global property of the structure. Having blocks maintain a global map of the structure would be the most straightforward way to ensure being able to handle all possible constraints, though not the most memory-efficient.

7.3 Multiple levels

Constraints may also be specified on multiple levels of organization. We can define different *regions* of the structure, in each of which some different set of constraints as listed above is to be imposed. Figs. 4 and 7 give examples where such regions are designated according to specific coordinates. The former designates regions compactly by reference to geometric shapes and lines (e.g., a region of red blocks is defined as those cells within a given distance from an arc with given center, radius, and endpoints, etc.). The latter combines patterned and adaptive constraints into a single structure.

Regions can be designated in this rigid way, or more fluidly: it may be permissible for a region to occur anywhere

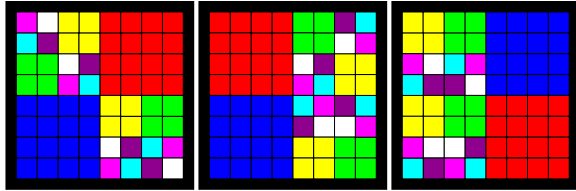


Figure 8: Three examples of structures that can all equally well be built based on the constraints given in the text.

in the structure, with variable size, aspect ratio, orientation, multiplicity, etc. Just as blocks can have proximity and relational constraints restricting their placement, so can regions occur in the structure in locations restricted based on the locations of other regions. Dimensions, etc. of regions may also be specified with reference to each other (e.g., region 1 must be larger than region 2).

Moreover, regions can be nested in this framework. An application might, for instance, call for a complex, which is composed of wings. One wing might be residential, made up of apartments; another might be made up of lab spaces. An apartment can be broken down into bedroom, bathroom, kitchen, etc.; and so on until the lowest level, which is decomposed into blocks as before. Each region may have its own constraints governing the placement of its sub-regions.

Figure 8 gives an explicit, abstract example. An 8×8 structure is constrained to have four 4×4 regions, one of type A, one B, two C. In a region of type A, only red blocks are allowed; in B, only blue blocks are allowed. In C, there must be four 2×2 subregions, one of type D, one E, two F. In a region of type D, only yellow blocks are allowed; in E, only green blocks are allowed. Finally, a region of type F must contain an arrangement of {purple, cyan, magenta, white} blocks, in that order clockwise; rotation is permitted, reflection is not. A great variety of structures can be built consistent with this set of constraints, all equally valid; the figure shows three.

7.4 Short- and long-term constraint satisfaction

There are two ways to satisfy the condition that no functional constraint be violated by attaching a block of type t at a site s . One is to consider only blocks that are part of the structure at that time. The condition is simply: if attaching t at s would result in a structure that violates any constraint, then forbid the attachment. The examples of Figs. 4–7 were assembled in this way.

In many cases, however, it will be desirable to look further ahead in time while ensuring constraints are met. For some constraint sets, situations may occur where attaching a block gives an acceptable structure, but one to which no further blocks can legally be attached. Looking ahead can prevent becoming trapped in such dead ends. A structure like that of Fig. 7 gives one example: guaranteeing exactly 50 ‘stars’ would require looking ahead during assembly of Region 2, to ensure sufficient remaining room for the white blocks not yet attached. It may also be desirable to allow constraints to be violated temporarily, so long as some future way of satisfying them remains possible. Consider, e.g., a structure of two

block types, each of which is constrained always to have at least one neighbor of its own type. Without the possibility of violating the constraint temporarily, the structure would necessarily turn out to consist entirely of one block type: no block of the other type could ever be attached, since it would have no neighbor of its own type until a second such block were attached to it.

This looking ahead may be limited to a fixed number of steps f into the future. Like functional constraints associated with larger d , larger f will be more powerful but more difficult for the blocks. Putting no limit on f , so that blocks always look ahead to the completion of some finished structure, will be slow but will prevent the structure from getting trapped in dead ends of partial completion.

8 Multiple robots

Algorithm 1 gives a procedure which will work for a single robot. However, when more than one robot is working on construction at once, additional precautions need to be taken. Finite speed of message propagation between blocks means that two robots can request to attach at mutually exclusive sites at about the same time, and both can be given permission by blocks working with outdated information.

The violation of geometric constraints (two non-adjacent blocks being attached in the same row) can be prevented, e.g., by sending messages to lock out a previously empty row when a robot requests to attach a block in it, and not giving the robot permission until all blocks in the adjacent row have indicated they have not and will not let other robots attach.

For functional constraints with finite d , the corresponding approach is to lock out all blocks within radius d before giving a robot permission to attach. When d is large (and, in particular, since for some constraints d may extend over the entire structure), so doing can become unwieldy: considerable message-passing over the entire structure will take place for every block attached, and robots may be turned away from attaching more often than necessary.

However, message-passing within the structure will be much faster than physical movement of robots. Geometric constraints only require consulting with other blocks when the first block in a row is being attached (and typically, only a few blocks will need to be consulted); thereafter, allowing or forbidding other blocks to be attached in the same row can be reliably done without communication within the structure [Werfel *et al.*, 2005]. And d is small for many functional constraints of interest.

All this means that, even without this strategy of locking out before attaching, errors should be infrequent. An alternative approach, then, is to correct errors when they arise rather than being scrupulous about preventing them in the first place. Blocks can allow or forbid robots to attach based solely on their own knowledge; if that causes a conflict which is only detected later, the blocks can direct robots to correct the error by removing the offending blocks, as described in §9.2 below.

9 Additional construction procedures

We briefly consider how two related procedures, disassembly and error correction, could be achieved with such a system.

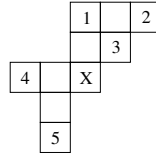


Figure 9: A structure in the process of disassembly. Numbered blocks can be removed. The block marked X meets criterion 1 in the text but not 2; if removed, it would split the structure into two parts. The beacon may be at the location of any of the unmarked blocks.

9.1 Disassembly

Many applications will involve structures intended to be temporary; the modular nature of this system is especially suited to such cases, allowing a structure to be dismantled following use and its parts reused for another structure. Disassembly can be accomplished by having robots follow the structure perimeter, removing any blocks they find that satisfy the following criteria:

1. It has at most two neighbors, bordering adjacent sides; i.e., if that site were unoccupied, it could physically accommodate a block, as in Fig. 2.
2. If it does have two neighbors along adjacent sides, then the other site which those two neighbors border is occupied by a block (see Fig. 9).
3. It is not attached at the beacon location, unless no other blocks remain; the block at the beacon must be the last one removed.

Again, blocks can determine whether they satisfy these criteria, and indicate that to robots. The second and third criteria ensure that throughout disassembly, the structure remains one contiguous piece, which a robot homing in on the beacon signal will be able to find. Violating either of those rules can result in part or all of the remaining structure becoming isolated, and thereby lost to robots no longer able to reach it by the simple procedure of following the beacon signal to the structure and following the perimeter thereafter. If the structure is physically tethered to some anchor point via the first block attached (as would likely be the case, e.g., in space-based applications), then violating those criteria would mean that blocks could drift off and be lost in a more literal sense. Care must also be taken with multiple robots that two blocks not be chosen for removal at the same time, such that each block considered alone satisfies the three criteria, but would violate criterion 2 if the other block were removed. Blocks 1 and 3 in Fig. 9 illustrate such a pair.

9.2 Error correction

Suppose that a block mistakenly ends up attached at a location where it should not be, or needs to be replaced due to malfunction or some other reason. It may be some time before this error arises or is detected; we will thus consider the problem of replacing any arbitrary block in the structure. We outline the high-level approach and omit elaboration of an algorithm to be carried out by individual blocks.

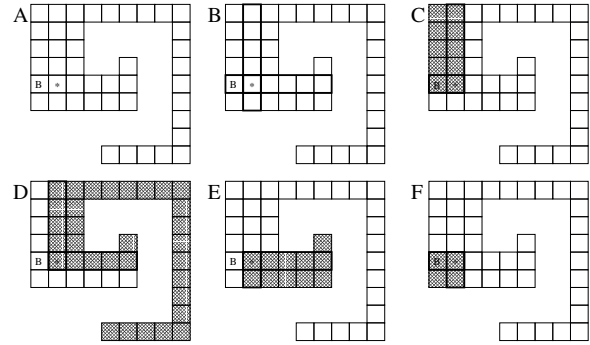


Figure 10: Error correction in a compound structure. The beacon is marked with the letter B; the block to be replaced is marked *. See text for discussion.

Because of our assumption that a block can only be removed from the structure if it is already free on two sides, it is necessary to remove all blocks in an entire ‘quadrant’ of the structure in order to free up the target block. Fig. 10 demonstrates the demarcation of quadrants. (A) shows a completed compound structure; the beacon is marked with the letter B, and the block to be replaced is marked *. Extending horizontal and vertical arms from the target block, as in (B), defines the four (overlapping) quadrants, crosshatched in (C)–(F). A quadrant is constructed essentially by flood-filling the area including and to one side of a pair of arms.

Note that in a compound structure, a quadrant may contain blocks whose absolute coordinates go beyond those of the associated arms (as in (D), where a quadrant nominally above the right-hand arm includes blocks that extend below it). Also, any blocks that would otherwise be left isolated from the rest of the structure if a quadrant were removed must be included in that quadrant. This issue arises if an arm lies along an edge of the structure for all or part of its length. In this example, the right-hand arm lies along such an edge, at the top side of the second and third blocks from the target block; any blocks above and further along that arm must therefore be included in the quadrant nominally below it, as shown in (E).

For efficiency, the quadrant containing the fewest blocks should be chosen for removal. As with disassembly, the block attached at the beacon must remain intact, and so any quadrants containing that beacon are omitted from consideration in this choice. Here the quadrant in (F) contains the fewest blocks, but cannot be removed because it includes the beacon (as does the quadrant in (C)). The quadrant chosen for removal will be that in (E).

Blocks in the quadrant chosen forbid robots from attaching any further blocks to them, and ask to be removed. Upon receiving such a request, some robots may switch their role temporarily from ‘assembly’ to ‘disassembly’. Once the target block has been removed, that section of the structure can be restored by construction as usual. Construction may also progress as usual in other parts of the structure throughout the correction process.

10 Conclusions

We have demonstrated that this approach, of building structures with communicating blocks and multiple simple robots, can achieve a variety of construction objectives. Structures can be built with perimeters of any shape, and with prespecified patterns of block types or adaptive satisfaction of constraints. Other approaches using swarms of simple robots have demonstrated the ability to create structures which satisfy locally specified constraints [Théraulaz and Bonabeau, 1995; Melhuish *et al.*, 1999]. However, this system with communicating blocks is unique in its ability to handle fully specified and locally adaptive constraints alike.

We are currently working on implementing a version of this system using the AMOUR robot of the Rus Robotics Lab at MIT [Vasilescu *et al.*, 2005].

Acknowledgments

We thank Daniela Rus, Jason Redi, and Jake Beal for useful discussions. This work was supported by NSF grant EIA-0130391.

References

- [Bowyer, 2000] Adrian Bowyer. Automated construction using co-operating biomimetic robots. Technical report, University of Bath Department of Mechanical Engineering, Bath, UK, 2000.
- [Everist *et al.*, 2004] Jacob Everist, Kasra Mogharei, Harshit Suri, Nadeesha Ranasinghe, Berok Khoshnevis, Peter Will, and Wei-Min Shen. A system for in-space assembly. In *Proceedings of 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2356–2361, Sendai, Japan, 2004.
- [Hill *et al.*, 2000] Jason Hill, Robert Szewczyk, Alec Woo, Seth Hollar, David Culler, and Kristofer Pister. System architecture directions for networked sensors. In *ASPLOS-IX*, Cambridge, MA, 2000.
- [Jones and Matarić, 2004] Chris Jones and Maja Matarić. Automatic synthesis of communication-based coordinated multi-robot systems. In *Proceedings of 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 381–387, Sendai, Japan, 2004.
- [Mason, 2002] Zachary Mason. Programming with stigmergy: using swarms for construction. In *Proceedings of Artificial Life VIII*, pages 371–374, Sydney, Australia, 2002.
- [Melhuish *et al.*, 1999] Chris Melhuish, Jason Welsby, and Charles Edwards. Using templates for defensive wall building with autonomous mobile ant-like robots. In *Proceedings of Towards Intelligent Autonomous Mobile Robots 99*, Manchester, UK, 1999.
- [Rus and Chirikjian, 2001] Daniela Rus and Gregory Chirikjian, eds. *Autonomous Robots*, 10(1):5–124, 2001.
- [Støy and Nagpal, 2004] Kasper Støy and Radhika Nagpal. Self-reconfiguration using directed growth. In *Proceedings of Distributed Autonomous Robotic Systems 2004*, Toulouse, France, 2004.
- [Terada and Murata, 2004] Yuzuru Terada and Satoshi Murata. Automatic assembly system for a large-scale modular structure: hardware design of module and assembler robot. In *Proceedings of 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2349–2355, Sendai, Japan, 2004.
- [Théraulaz and Bonabeau, 1995] Guy Théraulaz and Eric Bonabeau. Coordination in distributed building. *Science*, 269:686–688, 1995.
- [van Emde Boas, 1997] Peter van Emde Boas. The convenience of tilings. In Andrea Sorbi, editor, *Complexity, Logic and Recursion Theory*, volume 187 of *Lecture Notes in Pure and Applied Mathematics*, pages 331–363. Marcel Dekker Inc., 1997.
- [Vasilescu *et al.*, 2005] Iuliu Vasilescu, Paulina Varshavskaya, Keith Kotay, and Daniela Rus. Autonomous modular optical underwater robot (AMOUR): design, prototype and feasibility study. In *Proceedings of 2005 IEEE International Conference on Robotics and Automation*, Barcelona, Spain, 2005.
- [Wawerla *et al.*, 2002] Jens Wawerla, Gaurav Sukhatme, and Maja Matarić. Collective construction with multiple robots. In *Proceedings of 2002 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Lausanne, Switzerland, 2002.
- [Werfel *et al.*, 2005] Justin Werfel, Yaneer Bar-Yam, and Radhika Nagpal. Construction by robot swarms using extended stigmergy. AI Memo AIM-2005-011, MIT CSAIL, Cambridge, MA, 2005.
- [Werfel, 2004] Justin Werfel. Building blocks for multi-agent construction. In *Proceedings of Distributed Autonomous Robotic Systems 2004*, Toulouse, France, 2004.

POSTER PAPERS

AI AND COGNITIVE ARCHITECTURES

Motivated Agents

Kathryn Kasmarik¹, William Uther, Mary-Lou Maher¹

National ICT Australia*, ¹University of Sydney

kkas0686@it.usyd.edu.au, william.uthier@nicta.com.au, mary@arch.it.usyd.edu.au

1 Introduction

This poster presents a model for motivated agents governed by an internal, domain independent motivation process rather than domain specific rewards, goals, or examples provided by an external teacher. Internal motivation is desirable in complex, dynamic environments where hard-coded domain theories can only approximate the true state of the world and where pre-programmed goals can become obsolete.

Early work with motivated agents focused on the use of domain specific motives [Sloman and Croucher, 1981] or environment modelling by focusing attention on situations with the highest potential for learning [Kaplan and Oudeyer, 2004]. More recent work with intrinsically motivated reinforcement learning agents [Singh et al., 2005] has produced agents that, rather than learning a model, learn new behavioural *options* [Precup et al., 1998]. An option or simply a *behaviour* is a whole course of action that achieves some sub-goal. Our model extends previous work by defining general structures for events, attention focus and motivation.

2 The Motivated Agent Model

Our model for a motivated agent is shown in Figure 1. The agent has three types of structures: sensors, memory and effectors. These structures are connected by three processes: sensation, motivation and action. The agent functions in a continuous sensation-motivation-action loop.

Sensors receive raw data in the form of an n-tuple of state variables $\{x_1, x_2, x_3, \dots, x_n\}$ describing the current state of the agent and its environment. The n-tuple has a particular format. Variables x_1 to x_k comprise data about the state of the agent. Variables x_{k+1} to x_n comprise data about other objects sensed by the agent. This includes absolute data as well as relative data such as the location of an object relative to the agent. This general format is domain independent and the inclusion of relative values makes it possible for the agent to learn general behaviours relative to itself.

Sensation transforms raw data received from sensors into structures called *events*. Events encapsulate two recognisable occurrences among state variables values: increases and decreases between one state and the next. Individual events and the current state n-tuple are incorporated into *memory*.

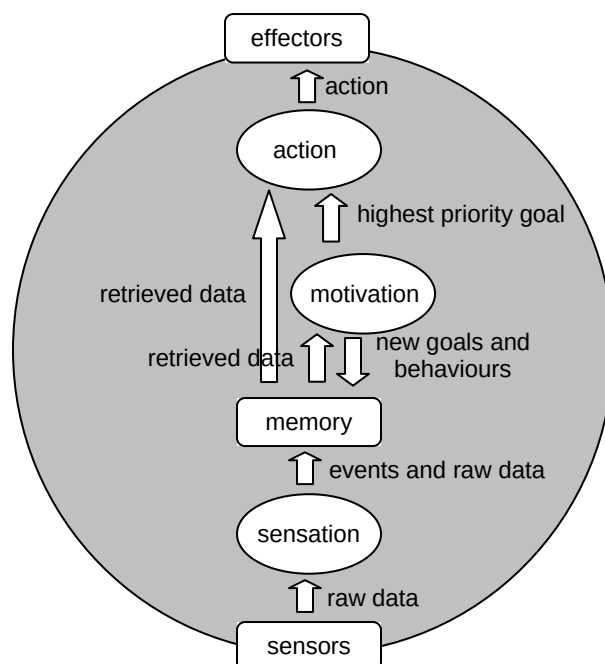


Figure 1 – The motivated agent model.

Memory is a cumulative record of events, actions, and goals. Initially, memory is empty save for a set of primitive actions that the agent can perform in the world.

Motivation motivates agents to understand and repeat interesting events that occur in their environment. We characterise interesting events as changes in the world that occur infrequently. Agents are motivated to create goals to understand and repeat interesting events by identifying situations in which they can learn how to make the event recur using a temporal difference Q-learner (TDQL) [Sutton and Barto, 2000]. Goals are *masked* so that events that occur with equal or lesser frequency than the one being pursued are ignored. Masking increases learning efficiency by focusing

* National ICT Australia is funded by the Australian Government's Backing Australia's Ability initiative, in part through the Australian Research Council.

attention and reducing the size of the state space. Once an agent can repeat an interesting event at will, it can encapsulate its new knowledge as a *behaviour*. Masking ensures that behaviours are independent of the situation in which they were learned. A behaviour can be reused either as a pre-planned course of action for achieving new goals similar to the one from which it was originally created or as a building block when learning to solve more complex goals.

Action reasons about which *effectors* will further the agent's progress towards its current goal.

Effectors are the means by which actions are achieved. They allow the agent to cause a direct change to the world.

3 Motivated Agents in Practice

We demonstrate our agent model using a robot guide. This domain, based on Dietterich's taxi domain [2000] and illustrated in Figure 2, contains avatars who need to be guided to particular locations. Such situations are common in large scale virtual worlds where new citizens can easily become lost. There are four possible sources and destinations for avatars. The robot has twelve effectors controlling the movements of its legs and can choose to start or stop guiding an avatar. The robot's sensors can perceive the absolute co-ordinates of the agent, the elevations of its legs, the co-ordinates of its legs relative to its body and the absolute and relative co-ordinates of an avatar and its destination.

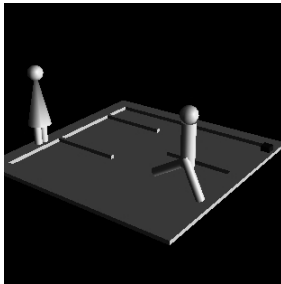


Figure 2 – A guide robot.

After a period of exploration, the robot notices that increases and decreases in its location and the location of other avatars are infrequent. The agent is motivated to create and pursue goals to repeat these changes. Over time the agent learns walking behaviours, such as Behaviour-1, to change its location. These behaviours are independent of the location in which they were learned and the position of other avatars. The agent uses these walking behaviours to develop path-following behaviours, such as Behaviour-2, to change the location of the avatar.

Behaviour-1 [lift left foot, move left foot forwards, put-down left foot, lift right foot, move right foot forwards, put-down right foot]

Behaviour-2 [guide, **Behaviour-1**, **Behaviour-1**, **Behaviour-1**, **Behaviour-1**, **Behaviour-1**, stop-guiding]

4 Empirical Results

We measured the performance of the robot guide by graphing the number of primitive actions taken to produce Behav-

our-2 in response to some goal G . We then implemented a flat TDQL with a pre-programmed reward for achieving a similar behaviour that satisfies G . Figure 3 shows that the motivated agent learns Behaviour-2 more quickly than a flat TDQL can learn a similar behaviour. This is because the motivated agent can make use of abstract behaviours such as Behaviour-1 while constructing Behaviour-2.

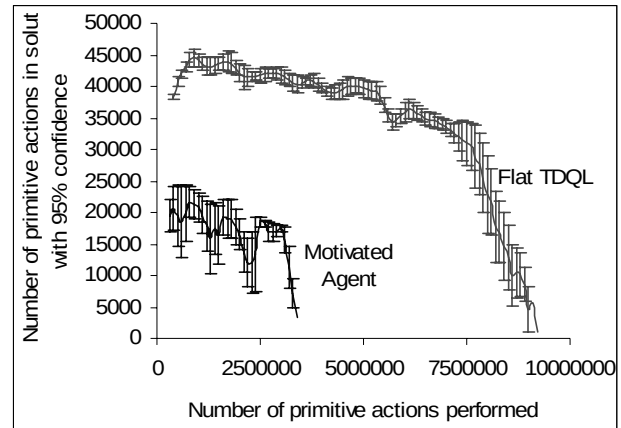


Figure 3 – Learning progress in a guide task.

5 Conclusion

Motivated agents are autonomous agents governed by internal, domain independent motivation. In addition to being able to choose their own goals they are able to learn behaviours to satisfy their goals more quickly than agents using the TDQL algorithm without motivation.

References

- [Dietterich, 2000] T. Dietterich. Hierarchical reinforcement learning with the MAXQ value function decomposition. *Journal of Artificial Intelligence Research*, 13:227-303, 2000.
- [Kaplan and Oudeyer, 2004] F. Kaplan and P-Y. Oudeyer. Intelligent adaptive curiosity: a source of self-development. In *Proceedings of the 4th International Workshop on Epigenetic Robotics*, pages 127–130, 2004.
- [Precup et al., 1998] D. Precup, R. Sutton, and S. Singh. Theoretical results on reinforcement learning with temporally abstract options. In *Proceedings of the 10th European Conference on Machine Learning*, 1998.
- [Singh et al., 2005] S. Singh, A. G. Barto, and N. Chentanez. Intrinsically motivated reinforcement learning. To appear in *Proceedings of Advances in Neural Information Processing Systems 17 (NIPS)*, 2005.
- [Sloman and Croucher, 1981] A. Sloman. and M. Croucher. 1981. Why robots will have emotions, The 7th International Joint Conference on Artificial Intelligence, Vancouver, Canada, pp. 197-202.
- [Sutton and Barto, 2000] R. Sutton, and A. Barto. *Reinforcement learning, an introduction*. MIT Press, 2000.

Using AI and simulations to design and control space habitats

David Kortenkamp and Scott Bell

NASA Johnson Space Center

Metrica Inc. and SKT Inc.

1101 NASA Road 1 Mail Stop ER2

Houston TX 77058

kortenkamp@jsc.nasa.gov and scott.e.bell1@jsc.nasa.gov

Abstract

This paper describes a dynamic simulation of a space habitat. The simulation is configurable and controllable via external programs. Several groups have been using the simulation to study the impact of artificial intelligence tools on space habitat design and control. We outline some of the AI challenges and invite the AI community to use our simulation to further NASA's exploration goals.

1 Introduction

NASA has embarked on a new exploration strategy that will return people to the moon and eventually to Mars [National Aeronautics and Space Administration, 2004]. Two key aspects of this strategy are sustainability and autonomy. The former means that these will not be single-shot missions, but a continued, evolving presence by humans outside of low-earth orbit. The latter means that the missions will be more self-reliant and less dependent on a standing army of earth-based controllers. Achieving both of these aspects requires significant advances in intelligent software systems. In this paper we present an integrated simulation of a space habitat that allows for testing of artificial intelligence approaches.

1.1 Space habitats

The function of a space habitat is to provide a livable environment to the crew. A typical space habitat will have subsystems that provide oxygen and remove carbon dioxide; that provide potable water; that provide food; that remove solid waste; and that provide power. In addition, a biomass subsystem may supply crops that can be turned into food and also help with other life support functions. Figure 1 shows such a habitat configuration. Many of the subsystems impact each other creating complex interactions and dependencies. Since the cost of placing materials on a lunar or planetary surface is so high, minimizing the consumables required to provide life support functions is important. Recycling or regenerating resources, often achieved by using biological-based subsystems, adds additional complexity but reduces overall habitat mass. In the next section we present a dynamic simulation of such an integrated, closed habitat system.

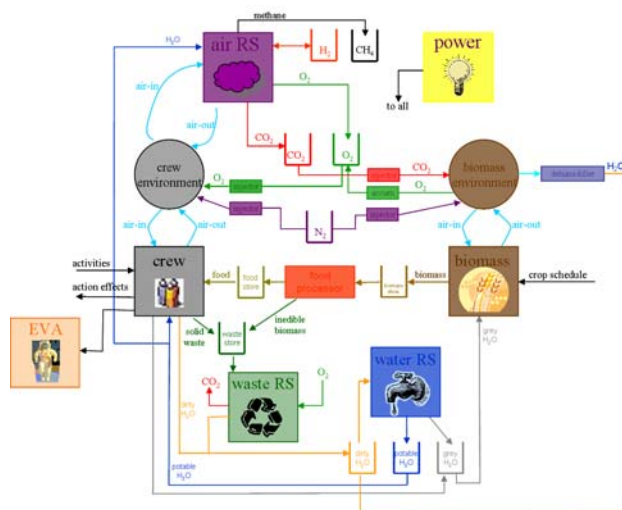


Figure 1: The various modules that comprise a habitat life support system

2 BioSim

BioSim is a discrete-event simulation of a space habitat [Kortenkamp and Bell, 2003]. Each of the life support components of a habitat are modeled as processes that consume certain resources and produce other resources. For example, the water recovery system model consumes dirty water and power and produces potable water. Crew members are also modeled – they consume and produce resources just as other life support components. All of the components, including crew members and crops, if any, exist in one or several environments that consists of a volume of mixed gases. Figure 1 shows the different models that compose our simulation and the resources that pass between the models. BioSim is implemented in Java.

2.1 Configuring BioSim

BioSim can be configured to simulate a wide variety of different habitats. This includes the number, genders and ages of crew members, the size of the habitat environments, atmospheric pressure, capacities of tanks, initial levels of consumables, processing capacity of life support modules and many

other variables. Initial setup is configured via an XML file that is read in when BioSim is started.

2.2 Controlling BioSim

BioSim has sensors and actuators that connect to various controllable elements and allow for real-time control. Sensors read simulation values, such as oxygen levels in the environment. Actuators set flow-rates of resources between components. Higher level actuators control more abstract variables such as the crew schedule, extravehicular activities (EVAs), crop harvesting, crop planting and equipment maintenance. Crop management is a particular focus in BioSim and up to nine different crops can be planted and harvested. Each has their own model for oxygen production, carbon dioxide consumption, water consumption and food production. Sensors and actuators are accessible via CORBA method calls so that control programs can be created in most programming languages and connected as clients to the BioSim server. We have defined BioSim control methods that let you advance BioSim in one hour increments or advance BioSim until the mission has ended. Malfunctions can be injected into any of the components of BioSim at any time.

3 Habitat design

One use of artificial intelligence tools is in the design of an optimal habitat for a specific exploration mission. As an example, let's take a 90 day lunar mission with a crew of four and one eight hour EVA per day. The goal is to find an optimal habitat configuration that meets those mission objectives, i.e., provides sufficient life support to the crew. A habitat configuration includes sizing each of the life support components (e.g., water processing, air processing, crops, power, etc.), sizing tank capacities, deciding on initial levels of consumables, sizing the air volume of the crew habitat, etc. An optimal configuration is defined as one that meets the mission goals with minimum mass. We have defined mass values for all of the variables in a habitat configuration. Essentially, this is a search through a large space of possible habitat designs to find an optimal solution. Many artificial intelligence search techniques would be applicable, including genetic algorithms and reinforcement learning. Several groups are already using BioSim to experiment in this area [Kortenkamp *et al.*, 2005; Wu and Garibay, 2004].

4 Habitat control

A second use of artificial intelligence tools is the control of the habitat. That is, given a predefined habitat configuration develop a control policy that optimizes resource utilization and reduces buffer sizes. The control policy adjusts resource flow rates, crew schedules, crop planting and harvesting schedules and EVA schedules to meet mission objectives with minimum resource utilization. Habitat control will involve dealing with real-time control issues such as water production, air production, etc. as well as planning and scheduling around time and resource constraints. Several groups are looking at different approaches to this problem [Bonasso *et al.*, 2003; Muscettola *et al.*, 2005; Klein *et al.*, 2004]. Habitat control also involves diagnosing

faults in the habitat and devising a recovery strategy. Gautam Biswas at Vanderbilt has been using BioSim to test model-based approaches to fault diagnosis [Biswas *et al.*, 2004]. Another research area is distributed control, for example, determining whether market-based approaches are valid for habitat control and finding what advantages they provide. Distributed control architecture will need to look at integration of control across the various subsystems of the habitat and integration of different artificial intelligence techniques.

5 Conclusion

With a renewed mission to head back to the moon and then to Mars, NASA needs the artificial intelligence community's help in achieving its exploration mission objectives in a cost effective and safe manner. BioSim provides a portable testbed for researchers to test their ideas and demonstrate relevance to NASA. BioSim also provides a baseline in which a standardized set of configurations, malfunctions and metrics can be established and used to compare control approaches. BioSim can be obtained from <http://www.traclabs.com/biosim>.

References

- [Biswas *et al.*, 2004] Gautam Biswas, Eric Jan Manders, John Ramirez, Nagabhusan Mahadevan, and Sherif Ablewahed. Online model-based diagnosis to support autonomous operation of an advanced life support system. *Habitation: International Journal for Human Support Research*, 10(1), 2004.
- [Bonasso *et al.*, 2003] R. Peter Bonasso, David Kortenkamp, and Carroll Thronesbery. Intelligent control of a water recovery system. *AI Magazine*, 24(1), 2003.
- [Klein *et al.*, 2004] Theresa Klein, Devika Subramanian, David Kortenkamp, and Scott Bell. Using reinforcement learning to control life support systems. In *Proceedings International Conference on Environmental Systems*, 2004.
- [Kortenkamp and Bell, 2003] David Kortenkamp and Scott Bell. Simulating advanced life support systems for integrated controls research. In *Proceedings International Conference on Environmental Systems*, 2003.
- [Kortenkamp *et al.*, 2005] David Kortenkamp, Scott Bell, and Luis Rodriguez. Simulating lunar habitats and activities to derive system requirements. In *Proceedings 1st AIAA Space Exploration Conference*, 2005.
- [Muscettola *et al.*, 2005] Nicola Muscettola, Charles Fry, David Kortenkamp, and Scott Bell. Planner-based control of advanced life support systems. In *to appear in Proceedings International Conference on Environmental Systems*, 2005.
- [National Aeronautics and Space Administration, 2004] National Aeronautics and Space Administration. Exploration systems interim strategy. Technical Report NP-2004-07-362-HQ, NASA Headquarters, 2004.
- [Wu and Garibay, 2004] Annie S. Wu and Ivan I Garibay. Genetic algorithm optimization of life support system control. *IEEE Transactions on Systems, Man, and Cybernetics*, 34(3), 2004.

A Universal Measure of Intelligence for Artificial Agents*

Shane Legg and Marcus Hutter
IDSIA, Galleria 2, Manno-Lugano 6928,
Switzerland
{shane,marcus}@idsia.ch

1 The concept of intelligence

A fundamental difficulty in artificial intelligence is that nobody really knows what intelligence is, especially for systems with senses, environments, motivations and cognitive capacities which are very different to our own. If we look to definitions of human intelligence given by experts, we see that although there is no consensus, most views cluster around a few common perspectives and share many key features.

In all cases, intelligence is a property of an entity, which we will call the *agent*, that interacts with an external problem or situation, which we will call the *environment*. An agent's intelligence is typically related to its ability to succeed in environments, which implies that there is some kind of objective, which we will call the *goal*. The emphasis on learning, adaptation and flexibility common to many definitions implies that the environment is not fully known to the agent. Thus intelligence is the ability to deal with a wide range of possibilities, not just a few specific situations. Putting these things together gives us our informal definition of intelligence:

Intelligence measures an agent's general ability to achieve goals in a wide range of environments.

We are confident that this definition captures the essence of many common perspectives on intelligence. It also describes what we want to achieve in machines: A very general capacity to adapt and perform well in a wide range of situations. In this paper we will try to formalise this view of intelligence.

2 A formal framework

We refer to the signals sent from the agent to the environment as *actions*, and the signals sent back as *perceptions*. Our definition requires there to be a goal for the agent to try to achieve which implies that the agent knows what it is. If the goal was known in advance it could be built into the agent, however this would limit each agent to just one goal. The alternative is to inform the agent of what the goal is. Unfortunately, the possession of a high level of language is too strong an assumption to make. Thus we define a communication channel with very simple semantics: A signal that indicates how good the agent's current situation is, called the *reward*. The agent then simply tries to maximise the amount of reward it receives by learning about the structure of the environment and what it needs to accomplish in order to receive large rewards.

*This work was supported by SNF grant 2100-67712.02.

This is the extremely flexible reinforcement learning framework commonly used in AI, and is equivalent to the controller-plant framework in control theory. In our formulation we will include the reward signal as part of the perception generated by the environment. Thus, as the agent's goal is implicitly defined by the environment, to test an agent in any given way it is sufficient to fully define its environment.

Formally, the agent sends information to the environment by sending *symbols* from some finite set called the *action space*, for example, $\mathcal{A} := \{\text{up, down, left, right}\}$. The environment responds with symbols from a finite set called the *perception space*, denoted $\mathcal{P}$. The *reward space*, denoted by $\mathcal{R}$, is a finite subset of $[0, 1] \cap \mathbb{Q}$. Every perception consists of two separate parts; a reward and a non-reward part called an *observation*. For symbols being sent we will use the lower case variable names o , r and a for observations, rewards and actions respectively. We index these in the order in which they occur, thus a_3 is the agent's third action. The agent and the environment take turns at sending symbols, starting with the environment. This produces an increasing history of observations, rewards and actions, $o_1 r_1 a_1 o_2 r_2 a_2 o_3 r_3 a_3 o_4 \dots$

The agent is a function, denoted by π , which takes the current history as input and chooses the next action as output. We represent this as a probability measure over actions conditioned on the current history, for example, $\pi(a_3 | o_1 r_1 a_1 o_2 r_2)$. The workings of the agent is unspecified, though in artificial intelligence the agent will be a machine and thus π will be a computable function. The environment, denoted μ , is similarly defined: $\forall k \in \mathbb{N}$ the probability of $o_k r_k$, given the current history is $\mu(o_k r_k | o_1 r_1 a_1 o_2 r_2 a_2 \dots o_{k-1} r_{k-1} a_{k-1})$.

The agent must try to maximise the total reward it receives, however what this means depends on how we value rewards at different points in the future. The standard way of expressing this is to weight the future reward at time i by a factor γ_i . Thus the future value is $V^{\pi\mu} := \mathbf{E}(\sum_{i=1}^{\infty} \gamma_i r_i)$, where r_i is the reward in cycle i of a given history, and the expected value is taken over all possible interaction histories of π and μ . The choice of γ_i is a subtle issue that controls how greedy or far sighted the agent should be. Here we use the near-harmonic $\gamma_i := 1/i^2$ as this produces an agent with increasing farsightedness of the order of its current age [Hutter, 2004].

As we desire an extremely general definition of intelligence for arbitrary systems, our space of environments should be as large as possible. An obvious choice is the space of all probability measures, however this causes serious problems

as we cannot even describe some of these measures in a finite way. The solution is to require the measures to be computable. This allows for an infinite space of possible environments with no bound on their complexity. It also permits environments which are non-deterministic as it is only their distributions which need to be computable. This space, denoted E , appears to be the largest useful space of environments.

3 A formal measure of intelligence

We want to compute the general performance of an agent in unknown environments. As there are an infinite number of environments, we cannot simply take a uniform distribution over them. If we consider the agent's perspective on the problem, this is the same as asking: Given several different hypotheses which are consistent with the data, which hypothesis should be considered the most likely? This is a standard problem in inductive inference for which the usual solution is to invoke Occam's razor: *Given multiple hypotheses which are consistent with the data, the simplest should be preferred.* As this is generally considered the most intelligent thing to do, we should test agents in such a way that they are, at least on average, rewarded for correctly applying Occam's razor. This means that our a priori distribution over environments should be weighted towards simpler environments.

As each environment is described by a computable measure, one way of measuring the complexity of an environment is by taking its Kolmogorov complexity. If $\mathcal{U}$ is a prefix-free universal Turing machine then the Kolmogorov complexity of an environment μ is the length of the shortest program on $\mathcal{U}$ that computes μ , formally $K(\mu) := \min_p \{l(p) : \mathcal{U}(p) = \mu\}$. Unfortunately, K is not computable and is provably difficult to approximate. For the purposes of Occam's razor, it also seems philosophically unnatural to consider short programs which require an enormous amount of time to compute to be "simple". We can address both of these problems by using a notion of complexity that takes execution time into account, such as Kt complexity [Levin, 1973]. Formally, $Kt(\mu) := \min_p \{l(p) + \log t(p) : \mathcal{U}(p) = \mu\}$ where $t(p)$ is the number of steps required to compute μ on $\mathcal{U}$. This gives us a computable distribution $2^{-Kt(\mu)}$ over our space of possible environments which is consistent with the notion that very simple algorithms should be short and fast to compute. Another alternative is the Speed Prior [Schmidhuber, 2002].

We can now define the *universal intelligence* of an agent π to simply be its expected performance when faced with an unknown environment sampled from this distribution,

$$\Upsilon(\pi) := \sum_{\mu \in E} 2^{-Kt(\mu)} V^{\pi\mu}.$$

4 Properties of universal intelligence

It is clear by construction that universal intelligence measures the general ability of an agent to perform well in a very wide range of environments, as required by our informal definition of intelligence given earlier. The definition places no restrictions on the internal workings of the agent; it only requires that the agent is capable of generating output and receiving input which includes a reward signal. Universal intelligence also reflects Occam's razor in a natural way that respects both

the minimal description and computation time of an environment. Indeed it is similar to intelligence tests for humans which usually define the correct answer to a question to be the simplest consistent with the given information. Although our space of possible environments E is infinite, as Kt is computable, $\Upsilon(\pi)$ can also be computed.

By considering $V^{\pi\mu}$ for a number of basic environments, such as small MDPs, and agents with simple but very general optimisation strategies, it is clear that Υ correctly orders the relative intelligence of these agents in a natural way. If we consider a highly specialised agent, for example IBM's Deep-Blue chess super computer, then we can see that this agent will be ineffective outside of one very specific and complex environment, and thus would have a very low universal intelligence value. This is consistent with our view of intelligence as being a highly adaptable and general ability.

A very high value of Υ would imply that an agent was able to perform well in many environments. Such a machine would obviously be of large practical significance. If we replace near-harmonic discounting with a finite length horizon and ignore computation time, it is possible to define an order relation between agents over interaction histories, known as the Intelligence Order Relation (IOR) [Hutter, 2004]. The maximal agent with respect to this order relation is AIXI, and with minor adjustments, it would also be maximal with respect to Υ . AIXI has been shown to have many optimality properties, including the ability to be self-optimising in environments in which this is at all possible [Hutter, 2004]. These results demonstrate the power of agents which rate highly with respect to the IOR and the related Υ defined here.

Clearly Υ spans simple adaptive agents right up to super intelligent agents like AIXI, unlike the pass-fail Turing test which is useful only for agents with near human intelligence. Moreover, the Turing test is highly anthropomorphic, with many suggesting that it is a test of humanness rather than intelligence. We have avoided this problem of human bias, as well as the need for human judges, by basing our definition on the fundamentals of information and computation theory.

The only related work to ours is the C-Test [Hernández-Orallo, 2000]. While Υ is an interactive test, the C-Test is a static sequence prediction test which always ensures that each question has an unambiguous answer. We believe that these are unrealistic and unnecessary assumptions. The C-Test was able to compute a number of usable test problems which were shown to correlate with real IQ test scores for humans.

References

- [Hernández-Orallo, 2000] J. Hernández-Orallo. Beyond the Turing test. *Journal of Logic, Language and Information*, 9(4):447–466, 2000.
- [Hutter, 2004] M. Hutter. *Universal Artificial Intelligence: Sequential Decisions based on Algorithmic Probability*. Springer, Berlin, 2004.
- [Levin, 1973] L. A. Levin. Universal sequential search problems. *Problems of Information Trans*, 9:265–266, 1973.
- [Schmidhuber, 2002] J. Schmidhuber. The Speed Prior: A new simplicity measure yielding near-optimal computable predictions. In *Proc. COLT 2002, Lecture Notes in Artificial Intelligence*, pages 216–228, July 2002. Springer.

It's About Time

Neil Madden and Brian Logan

University of Nottingham

School of Computer Science and Information Technology

Jubilee Campus, Triumph Road, Nottingham, NG8 1BB.

{nem,bsl}@cs.nott.ac.uk

1 Introduction

A number of experimental studies, e.g., [Geldard and Sherrick, 1972; Kolers and von Grünau, 1976] have suggested that interpretations of events can override direct sensory evidence. For example, for some sequences of perceptual events of short duration, the interpretation of individual events in the sequence depends on the characteristics of the sequence as a whole. This ‘backwards referral in time’, in which later events influence the perception of earlier events, is difficult to account for within a serial model of cognition without incorporating implausible delays (basically delaying sensory experience “until all the data is in”).

Dennett and Kinsbourne [1991; 1992] have proposed the *Multiple Drafts* theory as a way of modelling such cognitive processes. The Multiple Drafts theory is based on a parallel, distributed view of cognition, in which large numbers of processes work independently on multiple interpretations of data simultaneously. These are the multiple *drafts*. Eventually a single draft may become dominant, but no draft is ever entirely safe from revision.

2 The Temporal Abstraction Network Architecture

We present a cognitive architecture for perceptual processing which draws on aspects of the Multiple Drafts theory. Our Temporal Abstraction Network (TAN) architecture consists of a set of processes, each with its own state, represented by a time-limited buffer, along with a procedure for drawing inferences based on the current contents of the buffer. These processes are connected together via a bus architecture, allowing the conclusions drawn by one process to form the inputs to other processes (including themselves), see Figure 1.

Each process has an *input buffer* with specified capacity and duration. *Duration* is the maximum length of time elements can remain in the buffer before they are forgotten. *Capacity* is the number of items that may be present in the buffer at any given time. The duration and capacity of a buffer are independent of each other, e.g., a buffer may have large capacity but short duration or small capacity but longer duration. New inputs are added to the buffer in first in first out fashion—items arriving at a full buffer cause the oldest items in the buffer to be overwritten.

Each inference process also contains a set of *production*

rules that are used to spot patterns in input data, and draw conclusions based on these patterns. Conclusions are written to an *output bus* that transmits them to the input buffers of other processes. In this way, data can be abstracted as it progresses through the network of connected processes, with different abstractions persisting for different lengths of time. We envisage that processes further up a chain (further from the initial percepts) would have buffers spanning a larger duration of time than the lower level processes, allowing the system as a whole to remember more abstract conclusions, while most of the details are forgotten.

The bus connection architecture allows a single conclusion from a low-level process to be delivered to multiple higher-level processes, allowing for multiple drafts to be formed based on the same data, potentially producing different conclusions or interpretations.

3 The Cutaneous “Rabbit” Model

We have used the architecture outlined above to model a number of perceptual phenomena. In this section we briefly describe one such model: the cutaneous “rabbit”. Geldard and Sherrick’s cutaneous “rabbit” experiments [1972; 1977] illustrate a perceptual phenomenon called *sensory saltation*. In the experiments a series of short ‘taps’ (of about 2ms duration) were delivered to different locations on the arm of a subject — for instance, five taps at the wrist, followed by five between wrist and elbow, and then five more at the elbow. Subjects reported that the taps had been more or less evenly spaced along their arm — as if a little rabbit was hopping up the arm. Variation in the interval between taps (inter-stimulus interval, *ISI*) causes differences in the perceived effect. If the *ISI* exceeds approximately 200 ms the taps are perceived at their correct locations. With an *ISI* of 20 ms or less, some taps ‘disappear’, with say 15 taps being perceived as just 6.

Our model aggregates information about individual taps into information about a sequence of taps (see Figure 1), allowing the agent to reason about and predict the behaviour of an object over time. At the lowest level, the model processes sensory information to determine the presence of a single tap. This process has a buffer duration of about 20 ms and a capacity of just a single element (in this case, the ‘element’ is actually a collection of low-level data). If more than one tap occurs within this time-frame, then the newer tap simply overwrites any previous tap. This is consistent with the ex-

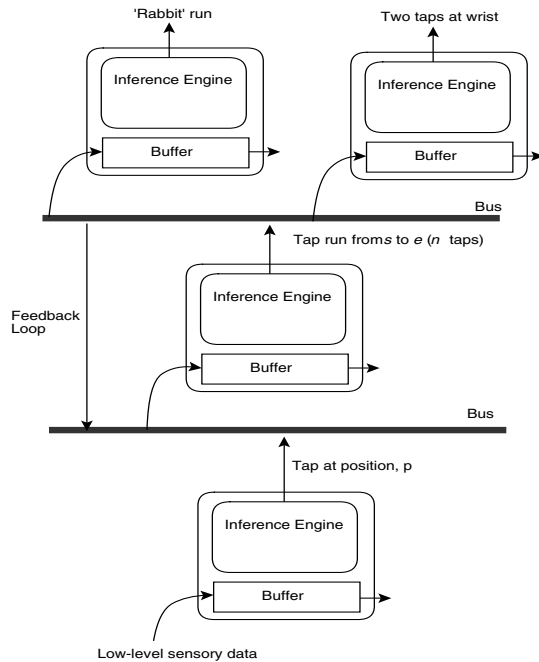


Figure 1: Network for cutaneous “rabbit” experiment.

perimental results which indicate that taps occurring within about 20 ms of each other are merged with the location of the newer tap dominating.

The intermediate level processing has a buffer of duration of 200 ms and a capacity of 2 elements. When a tap arrives at an empty buffer (which can happen at most once every 20 ms) a new aggregate conclusion is generated, taking the position of the tap as the start and end position of the ‘run’, and initialising the count of taps in this run. This conclusion is passed to the output bus, where it is transmitted to other processes, but also, via a feedback loop, back to the input bus of the intermediate level process. If a subsequent tap arrives before this aggregate fact expires from the buffer (i.e., within 200 ms) then a new conclusion is formed which adjusts the end point of the run to the new tap position and increases the tap count by 1. The buffer duration ensures that any gap of 200 ms or more causes the previous ‘run’ to be forgotten, and thus any subsequent tap will be perceived as the start of a new run, which is consistent with the experimental data.

At the highest level of processing (top left in the Figure), a process with a two element buffer detects the end of a tap run by comparing the start position of sequential tap-run inputs. It is the output from this process that is eventually used to generate a report of the experience. It is important to note that although the intermediate-level buffer has a duration of 200 ms it is not necessary to delay conclusions for 200 ms. Instead, the process produces a conclusion whenever a new tap is felt (at most, once every 20 ms), and these conclusions can be acted upon immediately.

4 Discussion

There are a number of parallel models of cognition (e.g. CopyCat, EPIC) which have some similarities to the Multiple

Drafts theory. However these models generally involve some serial component where “everything comes together”. For instance, in CopyCat [Mitchell, 1993] multiple parallel processes operate in a stochastic manner on a single workspace, creating a single solution to a problem; in EPIC [Kieras and Meyer, 1997] there is a single central executive which acts to coordinate the parallel processes.

In contrast, our Temporal Abstraction Network architecture has no global coordinating or integrating facility. However, abandoning a *single* central executive process does not mean that information cannot be brought together *locally* for integration. The TAN architecture is capable of local (serial) integration while maintaining multiple simultaneous drafts: information flow can diverge as easily as converge. This approach is in contrast to that of Dennett and Kinsbourne who suggest that the only alternative to the Cartesian theatre is a strictly parallel architecture, where local integration is replaced by a more chaotic Pandemonium approach (e.g., [Kinsbourne, 1994]).

In future work we plan to concentrate on extending the architecture to account for action selection, as well as expanding on the details of how reports are generated. One interesting area for future research will be to look at Libet’s controversial experimental results [Libet, 1985] on voluntary action.

References

- [Dennett and Kinsbourne, 1992] Daniel C. Dennett and Marcel Kinsbourne. Time and the observer: The where and when of consciousness in the brain. *Behavioral and Brain Sciences*, 15(2):183–247, 1992.
- [Dennett, 1991] Daniel C. Dennett. *Consciousness Explained*. Penguin, London, 1991.
- [Geldard and Sherrick, 1972] F. A. Geldard and C. E. Sherrick. The cutaneous ‘rabbit’: A perceptual illusion. *Science*, 178:178–179, 1972.
- [Geldard, 1977] F. A. Geldard. Cutaneous stimuli, vibratory and saltatory. *Journal of Investigative Dermatology*, 69:83–87, 1977.
- [Kieras and Meyer, 1997] David E. Kieras and David E. Meyer. An overview of the EPIC architecture for cognition and performance with application to human-computer interaction. *Human-Computer Interaction*, 12(4):391–438, 1997.
- [Kinsbourne, 1994] Marcel Kinsbourne. Models of consciousness: Serial or parallel in the brain? In M. S. Gazzaniga, editor, *The Cognitive Neurosciences*, pages 1321–1330. MIT Press, Cambridge, MA, 1994.
- [Kolars and von Grünau, 1976] P. A. Kolars and M. von Grünau. Shape and color in apparent motion. *Vision Research*, 16:329–335, 1976.
- [Libet, 1985] Benjamin Libet. Unconscious cerebral initiative and the role of conscious will in voluntary action. *Behavioral and Brain Sciences*, 8:529–566, 1985.
- [Mitchell, 1993] Melanie Mitchell. *Analogy-Making as Perception: a computer model*. MIT Press, Cambridge, MA, 1993.

Automation Intelligence for the Smart Environment

G. Michael Youngblood, Edwin O. Heierman, Lawrence B. Holder, and Diane J. Cook

Department of Computer Science and Engineering

The University of Texas at Arlington

Arlington, TX 76019-0015

{youngbld,heierman,holder,cook}@cse.uta.edu

Abstract

Scaling AI algorithms to large problems requires that these algorithms work together to harness their respective strengths. We introduce a method of automatically constructing HHMMs using the output of a sequential data-mining algorithm and sequential prediction algorithm. We present the theory of this technique and demonstrate results using the MavHome intelligent environment.

1 Introduction

An important component of an intelligent environment is to anticipate actions of a human inhabitant and then automate them. The decision of which action to execute must be correct in order to avoid creating excess work for humans in the form of correcting wrong automated actions and performing manual actions.

We examine the problem of learning human inhabitant behavioral models in the MavHome intelligent environment and using this to automate the environment. An event in the environment is described by the time of the event, the device/sensor zone, the device/sensor number, the new value of the device or sensor, the source of the vent (e.g., sensor network, powerline controller), and the inhabitant initiating the event (if known).

2 Solution Strategy

To automate the environment, we collect observations of manual inhabitant activities and interactions with the environment. We then mine sequential patterns from this data using the ED sequence mining algorithm. Finally, a hierarchical Markov model is created using low-level state information and high-level sequential patterns, and is used to learn an action policy for the environment.

2.1 Mining Sequential Patterns Using ED

Our data mining algorithm, ED, mines sequential patterns from observed activities. Data is processed incrementally and sequential patterns are mined according to their ability to compress the data using the Minimum Description Length principle. Periodicity (daily, every other day, weekly occurrence) of episodes is detected using autocorrelation and included in the episode description. If the instances of a pattern

are highly periodic (occur at predictable intervals), the exact timings do not need to be encoded and the resulting pattern yields even greater compression value.

2.2 Predicting Activities Using ALZ

To predict inhabitant activities, we borrow ideas from text compression. By predicting inhabitant actions, the home can automate or improve upon anticipated events that inhabitants would normally perform in the home. Our Active LeZi (ALZ) algorithm [Gopalratnam and Cook, 2005] approaches this problem from an information-theoretic standpoint. ALZ incrementally parses the input sequence into phrases and, as a result, gradually changes the order of the corresponding Markov model that is used to predict the next symbol in the sequence. Frequency of symbols is stored along with phrase information in a trie, and information from multiple context sizes are combined to provide the probability for each potential symbol as being the next one to occur. In our experiments, ALZ proved to be a very accurate sequential predictor. However, accuracy is further improved when the task is restricted by ED to only perform predictions when the current activity is likely to be part of a frequently-occurring pattern.

2.3 Decision Making Using ProPHeT

Work in decision-making under uncertainty has popularized the use of Hierarchical Hidden Markov Models and Partially Observable Markov Decision Processes. Recently, there have been many published hierarchical extensions that allow for the partitioning of large domains into a tree of manageable POMDPs [Pineau *et al.*, 2001; Theocharous *et al.*, 2001]. Although the Hierarchical POMDP is appropriate for an intelligent environment domain, current approaches generally require *a priori* construction of the HPOMDP. Given the large size of our domain, we need to seed our model with structure automatically derived from observed inhabitant activity data.

Unlike other approaches to creating a hierarchical model, our decision learner, ProPHeT, actually automates model creation by using the ED-mined sequences to represent the abstract nodes in the higher levels of the hierarchy. Lowest-level states correspond to an environment state representation together with an ALZ-supplied prediction of the next inhabitant action. To learn an automation strategy, the agent explores the effects of its decisions over time and uses this experience within a reinforcement learning framework to form con-

trol policies which optimize the expected future reward. The current version of MavHome receives negative reinforcement when the inhabitant immediately reverses an automation decision (e.g., turns the light back off) or an automation decision contradicts user-supplied safety and comfort constraints (e.g., do not let the temperature exceed 100 degrees).

3 Environments

All of the algorithms described here are implemented in MavHome and are being used to automate two environments, shown in Figure 1. The MavLab environment contains work areas, cubicles, a break area, a lounge, and a conference room. MavLab is automated using 54 X-10 controllers and the current state is determined using light, temperature, humidity, motion, and door/seat status sensors. The MavPad is an on-campus apartment hosting a full-time student occupant. MavPad is automated using 25 controllers and provides sensing for light, temperature, humidity, leak detection, vent position, smoke detection, CO detection, motion, and door/window/seat status sensors.



Figure 1: The MavLab (left) and MavPad (right) environments.

4 Case Study

As an illustration of these techniques, we have evaluated a week in an inhabitant's life with the goal of reducing the manual interactions in the MavLab. The data was generated from a virtual inhabitant based on captured data from the MavLab and was restricted to just motion and lighting interactions which account for an average of 1400 events per day. We trained ALZ and ED on real data and then repeated a typical week in our ResiSim simulator to determine if the system could automate the lights throughout the day in real-time.

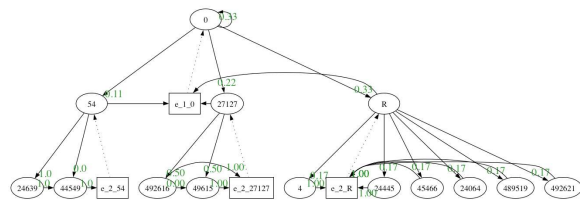


Figure 2: ProPHeT generated HHMM with production nodes abstracted.

ALZ processed the data and converged to 99.99% accuracy after 10 iterations through the training data, and accuracy was

54% on test data. When automation decisions were made using ALZ alone, interactions were reduced by 9.7% on average. Next, ED processed the data and found 3 episodes to use as abstract nodes in the HPOMDP, as shown in Figure 2. The HHMM model with no abstract nodes reduced interactions by 38.3%, and the combined-learning system (ProPHeT bootstrapped using ED and ALZ) was able to reduce interactions by 76%, as shown in Figure 3.

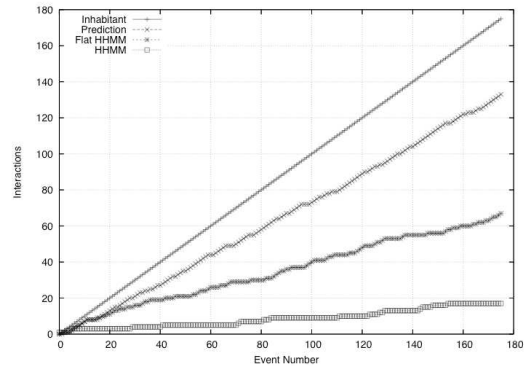


Figure 3: Interaction reduction.

Experimentation in the MavPad using real inhabitant data has yielded similar results. In this case, ALZ alone reduced interactions from 18 to 17 events, the HPOMDP with no abstract nodes reduced interactions by 33.3% to 12 events, while the bootstrapped HPOMDP reduced interactions by 72.2% to 5 events.

In this research we have shown that learning algorithms can successfully automate an intelligent environment. We see that synergy between these algorithms can improve performance, as ED-produced abstractions in the hierarchy coupled with a prediction produced by ALZ improved automation performance for ProPHeT. A full system deployment in the MavPad is currently being conducted.

References

- [Gopalratnam and Cook, 2005] K. Gopalratnam and D. J. Cook. Online sequential prediction via incremental parsing: The Active LeZi algorithm. *IEEE Intelligent Systems*, 2005.
- [Pineau *et al.*, 2001] J. Pineau, N. Roy, and S. Thrun. A Hierarchical Approach to POMDP Planning and Execution, 2001. Workshop on Hierarchy and Memory in Reinforcement Learning (ICML).
- [Theocharous *et al.*, 2001] G. Theocharous, K. Rohanimanesh, and S. Mahadevan. Learning Hierarchical Partially Observable Markov Decision Processes for Robot Navigation, 2001. IEEE Conference on Robotics and Automation.

POSTER PAPERS

CONSTRAINT SATISFACTION AND SEARCH

Allocation and Scheduling for MPSoCs via decomposition and no-good generation

Luca Benini, Davide Bertozzi, Alessio Guerri, Michela Milano

DEIS, University of Bologna

V.le Risorgimento 2, 40136, Bologna, Italy

{lbenini, dbertozzi, aguerri, mmilano}@deis.unibo.it

This paper proposes a decomposition approach to the allocation and scheduling of a multi-task application on a multi-processor system-on-chip (MPSoCs) [Wolf, 2004]. This is currently one of the most critical problems in electronic design automation for Very-Large Scale Integrated (VLSI) circuits. With the limits of chip integration reaching beyond one billion of elementary devices, current advanced integrated hardware platforms for high-end consumer application (e.g. multimedia-enabled phones) contain multiple processors and memories, as well as complex on-chip interconnects. The hardware resources in these MPSoCs need to be optimally allocated and scheduled under tight throughput constraints when executing a target software workload (e.g. a video decoder).

The multi-processor system consists of a pre-defined number of distributed computation nodes, as depicted in Figure 1. Each node is made by a processing core and by a tightly coupled local memory. Unfortunately, the scratchpad memory is of limited size, therefore data in excess must be stored externally in a remote on-chip memory, accessible via the bus. The bus for state-of-the-art MPSoCs is a shared communication resource, and serialization of bus access requests of the processors (the bus masters) is carried out by a centralized arbitration mechanism.

Whenever predictable performance is needed for applications, it is important to avoid high levels of congestion on the bus, since this makes completion time of bus transactions much less predictable. Moreover, under a low congestion regime, performance of state-of-the-art shared busses scales almost in the same way as that of advanced busses with topology and communication protocol enhancements. Finally, bus modelling is simpler under these working conditions (e.g., additive models). Communication cost is therefore critical for determining overall system performance, and will be minimized in our task allocation framework.

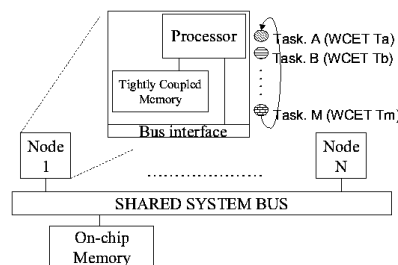


Figure 1: Single chip multi-processor architecture.

Based on our methodology, the target application running on top of the hardware platform is pre-characterized and abstracted as a task graph, with specification of computation, storage and communication requirements. More in detail, the worst case execution time (WCET) is specified for each task and plays a critical role whenever application real time constraints (expressed here in terms of minimum required throughput) are to be met. In fact, tasks are scheduled on each processor based on a time-wheel. The sum of the WCETs of the tasks for one iteration of the time wheel must not exceed time period RT (i.e., the minimum task scheduling period ensuring that throughput constraints are met), which is the same for each processor since the minimum throughput is an application (not single processor) requirement.

The problem we are facing is a scheduling problem with alternative resources. Each task should be allocated to one of the processors (Node i in Figure 1). Each task also needs 3 memory areas for executing, that should be allocated to a memory device: **internal task state** and **program data** can be allocated either on the local scratchpad memory or on the remote on-chip memory, while **communication queue** (the memory area used by the tasks to communicate one other) must be allocated on the local scratchpad. Tasks duration depends on where memory slots are allocated; tasks need time to access the bus and use remote memory. Clearly, tasks should be scheduled in time subject to real time constraints, precedence constraints, and capacity constraints on all unary and cumulative resources. However, on a different perspective, the problem decomposes into two problems: the allocation of tasks to processors and the memory slots required by each task to the proper memory device; a scheduling problem with static resource allocation.

The objective function of the overall problem is the minimization of communication cost. This function involves only variables of the first problem. In particular, we have a communication cost each time two communicating tasks are allocated on different processors, and each time a memory slot is allocated on a remote memory device. Once the communication cost has been minimized, among feasible schedules we prefer those having minimum makespan.

The allocation problem is difficult to solve with Constraint Programming (CP). CP has a naive method for solving optimization problems: each time a solution is found, an additional constraint is added stating that each successive solution

should be better than the best one found so far. If the objective function is strongly linked to decision variables, CP can be effective, otherwise it is hopeless to use CP to find the optimal solution. In case the objective function is related to a single variable, like for makespan in scheduling problems, CP works quite well. However, if the objective function is a sum of cost variables, CP is able to prune only few values, deep in the search tree since the connection between the objective function and the problem decision variables is weak. If the objective function relates to pairs of assignments the situation is even worse. This is the case of our application where the objective function depends on pairs of assignments. In fact, a contribution to the objective function is added when two communication tasks are allocated to different processors.

On the contrary, Integer Programming (IP) is extremely good to cope with these problems, while is weak in coping with time. Scheduling problems require to assign tasks to time slots, and each slot should be represented by an integer variable, and the number of variables increases enormously. CP, instead, is very effective to cope with time constraints.

Therefore, the first problem could be solved with IP effectively, while for the second CP is the technique of choice. The question is now: how do these problems interact?

We solve them separately, the allocation problem first (called master problem), and the scheduling problem (called subproblem) later. The master is solved to optimality and its solution passed to the subproblem solver. If the solution is feasible, then the overall problem is solved to optimality. If, instead, the master solution cannot be completed by the subproblem solver, a no-good is generated and added to the model of the master problem, roughly stating that the solution passed should not be recomputed again (it becomes infeasible), and a new optimal solution is found for the master problem respecting the (set of) no-good(s) generated so far. Being the allocation problem solver an IP solver, the no-good has the form of a linear constraint.

A similar method is known in Operations Research as Benders Decomposition [Benders, 1962], where the overall problem can be decomposed in two parts connected by some variables. Some related approaches are [Hooker, 2004] and [Grossmann and Jain, 2001].

We show that this method is extremely effective if compared to the approaches considering the problem as a whole.

The methodology proposed in this paper has been applied to a video signal processing pipeline, wherein each task processes output data of the preceding task in the pipeline. Functional pipelining is widely used in the domain of multimedia applications. Task parameters have been derived from a real video graphics pipeline processing pixels of a digital image. The proposed allocation and scheduling techniques can be easily extended to all applications using pipelining as workload allocation policy, and aim at providing system designers with an automated methodology to come up with effective solutions and cut down on design time. To do that, we schedule several repetitions of the pipeline processes in order to achieve a working rate configuration.

To validate the strength of our approach, we now compare the results obtained using this model (Hybrid in the following) with results obtained using only a CP or IP model to

solve the overall problem. Actually, since the first experiments showed that both CP and IP were not able to find a solution, except for the easiest instances, within 15 minutes, we simplified these models removing some variables and constraints. In CP, we fixed the activities execution time not considering the execution time variability due to remote memory accesses; in IP, we do not consider all the variables and constraints involving the bus: we do not model the bus resource and we therefore suppose that each activity can access data whenever it is necessary.

We generate a large variety of problems, varying both the number of tasks and processors. All the results presented are the mean over a set of 10 problems for each task or processor number. All problems considered have a solution. Experiments were performed on a 2GHz Pentium 4 with 512 Mb RAM. We used ILOG CPLEX 8.1 and ILOG Solver 5.3 as modelling and solving tools.

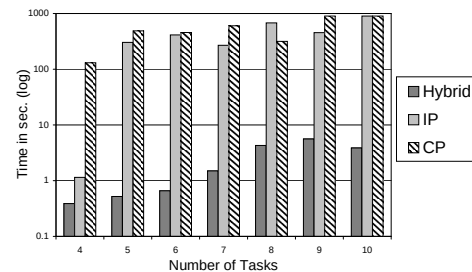


Figure 2: Comparison between algorithms search times for different process number.

the y-axis has a logarithmic scale. For space reasons we omit the search time figure varying the number of processors, but it has very similar behaviours to figure 2.

Although CP and IP deal with a simpler problem model, we can see that these algorithms are in general not comparable with Hybrid and, as the number of tasks grows, IP and CP performances worsen and their search times become orders of magnitude higher w.r.t. Hybrid. Furthermore, we considered in the figures only instances where the algorithms are able to find the optimal solution within 15 minutes, and, for problems with 6 tasks or more, IP and CP can find the solution only in the 50% or less of the cases.

References

- [Benders, 1962] J. F. Benders. Partitioning procedures for solving mixed-variables programming problems. *Numerische Mathematik*, 4:238–252, 1962.
- [Grossmann and Jain, 2001] I. E. Grossmann and V. Jain. Algorithms for hybrid milp/cp models for a class of optimization problems. *INFORMS Journal on Computing*, 13:258–276, 2001.
- [Hooker, 2004] J. N. Hooker. A hybrid method for planning and scheduling. In *Procs. of the 10th Intern. Conference on Principles and Practice of Constraint Programming - CP 2004*, pages 305–316, Toronto, Canada, Sept. 2004. Springer.
- [Wolf, 2004] W. Wolf. The future of multiprocessor systems-on-chips. In *In Procs. of the 41st Design and Automation Conference - DAC 2004*, pages 681–685, San Diego, CA, June 2004. ACM.

Minimizing a Makespan Under Uncertainty

Jérôme Fortin

UPS, IRIT,
118 route de Narbonne,
31062 Toulouse, France
fortin@irit.fr

Paweł Zieliński

Institute of Mathematics,
University of Technology
Wyb. Wyspiańskiego 27,
50-370 Wrocław, Poland
pziel@im.pwr.wroc.pl

Didier Dubois

UPS, IRIT,
118 route de Narbonne,
31062 Toulouse, France
dubois@irit.fr

Hélène Fargier

UPS, IRIT,
118 route de Narbonne,
31062 Toulouse, France
fargier@irit.fr

Abstract

This paper reconsiders the most basic scheduling problem, that of minimizing the makespan of a partially ordered set of activities, in the context of incomplete knowledge. After positioning this paper in the scope of temporal networks under uncertainty, we provide a complete solution to the problem of finding floats of activities, and of locating surely critical ones, as they are often isolated. The minimal float problem is NP-hard while the maximal float problem is polynomial. New complexity results and efficient algorithms are provided for the interval-valued makespan minimization problem.

1 Introduction and Motivation

Temporal Constraint Networks (TCN) represent relations between dates of events and also allow to express constraints on the possible durations of activities from intervals of values [Dechter *et al.*, 1991]. TCN have been extended to take into account uncertainty of durations of some tasks in realistic applications. A distinction is made between so-called contingent constraints (for example, when the duration of a task cannot be known before its execution) and controllable ones (for example a time interval to be chosen between starting times of two tasks). The resulting network (Simple Temporal Network with Uncertainty) becomes a decision-making problem under uncertainty, and notions of consistency must be refined so as to ensure controllability, that is, ensured consistency despite uncertainty [Morris *et al.*, 2001; Khatib *et al.*, 2001]. As far as we know, the TCN community has extensively worked on the controllability of a network, but the question of optimizing the total duration of set of tasks described by a STPU has not been studied. Nevertheless, not all solutions to an STPU are equally valuable, and solutions minimizing the makespan are of obvious practical interest.

This paper provides a full picture of the complexity of the makespan minimization problem under interval-based uncertainty. It is shown that the only NP-hard problem is the one of finding the greatest lower bound of the float, which is closely related to asserting the possible criticality of a task. All other problems turn out to be polynomial. The fact that the two problems of asserting if an arc is necessary critical or possibly critical do not have the same complexity is rather unexpected.

2 Preliminaries

An activity network is classically defined as a set of activities (or tasks) with given duration times, related to each other by means of precedence constraints. When there are no resource constraints, it can be represented by a directed, connected and acyclic graph. Of major concern, is to minimize the ending time of the last task, also called the makespan of the network. For each task, three quantities have practical importance for the management of the activity network: The *earliest starting time* est_{ij} of an activity (i, j) is the date before which the activity cannot be started without violation of a precedence constraint. The *latest starting time* lst_{ij} of an activity (i, j) is the date after which the activity cannot be started without delaying the end of the project. The *float* f_{ij} of an activity (i, j) is the difference between the latest starting time lst_{ij} and the earliest starting time est_{ij} . An activity is *critical* if and only if its float is equal to zero.

Now, activity durations (weights of the arcs) $(i, j) \in A$ are only known to belong to time intervals $D_{ij} = [d_{ij}^-, d_{ij}^+]$, $d_{ij}^- \geq 0$. We define a configuration as a precise instantiation of the duration of each task $(i, j) \in A$. Ω denotes a configuration, while $d_{ij}(\Omega)$ denotes the duration of activity (i, j) in configuration Ω . $G(i, j)$ is the subgraph of G composed of nodes succeeding i and preceding j .

Computing earliest starting dates is not a difficult issue. Here we solve four problems, stated in [Chanas *et al.*, 2002]: 1) determining the widest intervals LST_{kl} (bounds) of possible values of the latest starting times lst_{kl} of a given activity $(k, l) \in A$, i.e. the interval $LST_{kl} = [lst_{kl}^-, lst_{kl}^+]$ defined by $lst_{kl}^- = \min lst_{kl}(\Omega)$ and $lst_{kl}^+ = \max lst_{kl}(\Omega)$; the problem of computing lst_{kl}^- (resp. lst_{kl}^+) is denoted GLBLST (resp. LUBLST); 2) determining the widest intervals F_{kl} of possible values of floats (total floats) f_{kl} of a given activity $(k, l) \in A$, i.e. the interval $F_{kl} = [f_{kl}^-, f_{kl}^+]$ bounded by $f_{kl}^- = \min f_{kl}(\Omega)$ and $f_{kl}^+ = \max f_{kl}(\Omega)$. The problem of computing f_{kl}^- (resp. f_{kl}^+) is denoted GLBF (resp. LUBF). In both problems minimization and maximization are taken over all possible configurations. The solutions to problems GLBLST, LUBLST, LUBF and GLBF come down to finding an extreme configuration [Dubois *et al.*, 2005] where such bounds are attained. As there are 2^n extreme configurations, it explains the potentially exponential nature of the problem. GLBLST, LUBLST and GLBF have recently been solved in [Chanas

and Zieliński, 2002; Zieliński, 2005], and only GLBF is NP-hard. In this paper, we recall the solutions of these three problems, and present the solution to the last one LUBF, thus providing a full picture of the makespan minimization problem under incomplete information.

3 Evaluating Necessary Criticality

This section presents a new method which can decide if a given task (k, l) is necessary critical.

First, under the assumption that the durations of the predecessors of task (k, l) are precisely known, we already know an algorithm that asserts if (k, l) is necessary critical. It constitutes the basis for computing the LUB and GLB of the latest starting dates in polynomial time in [Zieliński, 2005]. We have extended these results and can provide a general algorithm which asserts if (k, l) is necessary critical in a network G in polynomial time without any consideration of the durations of tasks preceding (k, l) .

Suppose that (k, l) is necessary critical in the sub-network $G(i, n)$, then we proved that we can fix the duration of tasks preceding node i at their lower bounds, without changing the necessary criticality of (k, l) in G . On the contrary, if (k, l) is not necessary critical in $G(i, n)$, then we proved that we can fix the duration of tasks preceding node i at their upper bounds. Applying this idea on networks $G(i, n)$ for i from k down to 1, for each network, tasks preceding (k, l) have always precise duration in $G(i, n)$, and so we can easily assert the necessary criticality in polynomial time. The resulting algorithm requires $O(mn)$ steps to assert if (k, l) is necessary critical in G .

4 Computing the Least Upper Bound on the Floats

To compute the LUB of the floats, we are going to increase step by step the duration of (k, l) from $D_{kl} = d_{kl}^-$ until (k, l) becomes necessary critical with a duration $d_{kl}^- + f^*$. We proved that the value f^* is the LUB of the float of (k, l) .

If we assume that activities preceding (k, l) have precise durations, then $f_{kl}^+ = l_{st_{kl}}^+ - est_{kl}^+$. So at each step of the computation, we can easily compute f_i^+ representing the float of (k, l) in network $G(i, n)$. After initialization of durations in the network by the algorithm which asserts if (k, l) is necessary critical, if (k, l) is not necessary critical, we have to increase the duration of (k, l) by the minimal non-null f_i^+ for i preceding k , then go back to the test of asserting necessary criticality. When (k, l) becomes necessary critical, we have found f_{kl}^+ . At each iteration, (k, l) becomes necessary critical in at least one new network $G(i, n)$, and so the resulting algorithm is polynomial and takes $O(n^3m)$.

The following table summarizes the complexity of the different problems of the PERT/CPM on intervals:

Earliest starting date (all tasks)	GLB	P	$O(n + m)$
	LUB	P	$O(n + m)$
Latest starting date (one task)	GLB	P	$O(mn)$
	LUB	P	$O(mn)$
Float (all tasks)	GLB	NP-Hard	$O((n + m) P)$
Float (one task)	LUB	P	$O(n^3m)$

In this table we give the complexity of the theoretically best known algorithm which computes the quantity of interest. In particular, the path algorithm [Dubois *et al.*, 2005] which computes the GLB and LUB of floats and latest starting dates runs in $O((n + m)|P|)$ time, where $|P|$ is the number of path of the graph G . The real complexity thus depends of the topology of the network.

5 Conclusion

This paper proposes a complete solution to the criticality analysis of a network of activities when durations of tasks are ill-known. It is shown that moving from precise to imprecise durations radically changes the complexity of the problem, ruining the traditional critical path method. Part of the problem, pertaining to the GLB of the float, becomes NP-hard, the other questions remaining polynomial, although not straightforwardly so. These complexity results shed light on reasons why the more familiar stochastic counterparts to this basic scheduling problem are so difficult to handle, part of the difficulty being already present in the interval analysis. The proposed algorithms can be of obvious relevance to practical predictive project scheduling problems where durations of tasks are not known in advance. Clearly, due to the basic nature of the addressed problem, several lines of research can be envisaged for future research. For instance one may assume that part of the tasks durations are controllable and additional constraints relating durations and starting or ending times may be available. Then one obtains a makespan minimization problem in the more general setting of STPU's.

References

- [Chanas and Zieliński, 2002] S. Chanas and P. Zieliński. The computational complexity of the criticality problems in a network with interval activity times. *EJOR*, 136:541–550, 2002.
- [Chanas *et al.*, 2002] S. Chanas, D. Dubois, and P. Zieliński. On the sure criticality of tasks in activity networks with imprecise durations. *IEEE Transactions on Systems, Man, and Cybernetics*, 34:393–407, 2002.
- [Dechter *et al.*, 1991] R. Dechter, I. Meiri, and J. Pearl. Temporal constraint networks. *AI*, 49:61–95, 1991.
- [Dubois *et al.*, 2005] D. Dubois, H. Fargier, and J. Fortin. Computational methods for determining the latest starting times and floats of tasks in interval-valued activity networks. *J. of Intelligent Manufacturing*, 2005. To appear.
- [Khatib *et al.*, 2001] L. Khatib, P. Morris, R. Morris, and F. Rossi. Temporal constraint reasoning with preferences. In *IJCAI*, pages 322–327, 2001.
- [Morris *et al.*, 2001] Paul Morris, Nicola Muscettola, and Thierry Vidal. Dynamic control of plans with temporal uncertainty. In *IJCAI*, pages 494–502, 2001.
- [Zieliński, 2005] P. Zieliński. On computing the latest starting times and floats of activities in a network with imprecise durations. *Fuzzy Sets and Systems*, 150:53–76, 2005.

Scaling up WA* with Commitment and Diversity*

David Furcy

University of Wisconsin Oshkosh
Computer Science Department
Oshkosh, WI 54901-8643
furcyd@uwosh.edu

Sven Koenig

University of Southern California
Computer Science Department
Los Angeles, CA 90089-0781
skoenig@usc.edu

Abstract

Weighted A* (WA*) is a popular search technique that scales up A* while sacrificing solution quality. Recently, researchers have proposed two variants of WA*: KWA* adds diversity to WA*, and MSC-WA* adds commitment to WA*. In this paper, we demonstrate that there is benefit in combining them. The resulting MSC-KWA* scales up to larger domains than WA*, KWA* and MSC-WA*, which is rather surprising since diversity and commitment at first glance seem to be opposing concepts.

1 Introduction

Weighted A* (WA*) is a popular search technique that scales up A* while sacrificing solution quality, by weighing the heuristic values more strongly than A*. In this paper, we study how to scale up WA* to even larger domains, building on two variants of WA* that have been proposed recently: First, WA* expands only one state per iteration. KWA* is a variant of WA* that expands several states per iteration, which adds diversity to WA* to give it a stronger breadth-first component. Second, WA* considers all states in the OPEN list as candidates for expansion. Multi-State Commitment WA* (MSC-WA*) is a variant of WA* that considers only a small number of states in the OPEN list as candidates for expansion, which adds commitment to WA* to give it a stronger depth-first component. In this paper, we demonstrate that there is benefit in combining them. Our experimental results, for example, show that the resulting MSC-KWA* with the Manhattan heuristic scales up to the 48-Puzzle, whereas WA*, KWA* and MSC-WA* do not.

2 WA*, KWA* and MSC-WA*

WA*: WA* is a variant of A* that scales it up by making its search more greedy and thus more focused [Pohl, 1970]. While A* calculates the f-value of a state s as $f(s) = g(s) + h(s)$, WA* calculates it as $f(s) = (1 - W)g(s) + W \times h(s)$, where $1 \geq W \geq 0.5$ is the only parameter of WA*. **KWA*:** Due to its strong focus, WA* is likely to be led into goal-free regions of the search space by misleadingly small heuristic

values. KWA* tackles this problem by adding a breadth-first component to WA* (diversity) to make its search less focused. It expands the $K \geq 1$ states with the smallest f-values in the OPEN list (rather than just one) in parallel during each iteration [Felner *et al.*, 2003]. Thus, KWA* has two parameters, namely W and K . If $K = 1$, then KWA* reduces to WA*. Larger values of K increase the probability that WA* selects a state on the shortest path from the start state to the goal state during the next iteration at the risk of generating more states than necessary during each iteration. **MSC-WA*:** WA* focuses the search more than A*. However, the state that WA* expands next can be any state in the OPEN list (depending on the f-values). Multi-State Commitment WA* (MSC-WA*) increases the focus of WA* even more by focusing the search on a subset of the states in the OPEN list (commitment) [Kitamura *et al.*, 1998]. MSC-WA* splits the OPEN list into a COMMIT list of at most C states and a RESERVE list. Thus, MSC-WA* has two parameters, namely W and C . The COMMIT list plays the role of a smaller OPEN list, whereas the RESERVE list stores the remaining states of the original OPEN list. MSC-WA* expands the state with the smallest f-value in the COMMIT list. MSC-WA* never re-expands a state, even if it finds a shorter path from the start state to the state in question. The number of states in the COMMIT list can thus get smaller than C . In this case, MSC-WA* uses the states with the smallest f-values in the RESERVE list to refill the COMMIT list. If $C = \infty$, then the RESERVE list remains empty and MSC-WA* reduces to a variant of WA* that never re-expands a state. Smaller values of C make MSC-WA* focus the search on a smaller subset of the states in the original OPEN list.

3 Comparison

Table 1 reports the results of our experiments with WA*, KWA* and MSC-WA* on the N -Puzzle with $N = 15, 24, 35$ and 48, using the Manhattan distance as the heuristic function. The goal state has the blank in the upper left corner. For each combination of parameters, we average over 50 experiments (100 experiments for $N = 15$) with a memory capacity of six million states. (For W , for example, we use the values 0.50, 0.56, 0.60, 0.67, 0.75, 0.80, 0.86, 0.90, 0.95 and 0.99.) Each row of the table then reports the parameter combinations that optimize the average performance for each one of four performance measures, namely the solution cost, the num-

We thank Richard Korf for providing us with WA code and acknowledge funding from NSF under contracts IIS-0098807 and IIS-0350584. The views and conclusions contained in this document are those of the authors and not of NSF.

Table 1: Comparison of WA*, KWA*, MSC-WA*, and MSC-KWA* for the N -Puzzle

N	Performance Measure	WA*			KWA*				MSC-WA*				MSC-KWA*			
		Value	W	Best	Value	W	K	Best	Value	W	C	Best	Value	W	$K = C$	Best
15	Solution Cost	63.51	0.67		53.85	0.67	50,000	✓	56.29	0.60	80,000		53.89	0.99	50,000	✓
	Stored States	6,050	0.99		6,028	0.99	8		4,113	0.95	20		3,223	0.99	5	✓
	Generated States	6,972	0.99		6,704	0.99	8		4,191	0.95	20		3,259	0.99	5	✓
	Runtime (Seconds)	0.003	0.99		0.003	0.99	5		0.002	0.99	6		0.001	0.95	3	✓
24	Solution Cost	165.16	0.75		113.56	0.99	20,000	✓	164.56	0.75	90,000		116.32	0.99	20,000	
	Stored States	44,097	0.99		32,567	0.99	5		36,907	0.99	300		16,178	0.99	6	✓
	Generated States	56,070	0.99		43,578	0.99	4		37,832	0.99	300		16,331	0.99	6	✓
	Runtime (Seconds)	0.027	0.99		0.021	0.99	4		0.021	0.99	50,000		0.007	0.99	6	✓
35	Solution Cost				236.50	0.99	7,000	✓	472.10	0.90	3,000		244.14	0.99	7,000	
	Stored States				417,675	0.95	20		456,777	0.99	90		56,807	0.99	5	✓
	Generated States				652,100	0.95	500		467,586	0.99	90		57,291	0.99	5	✓
	Runtime (Seconds)				0.377	0.95	500		0.297	0.99	90		0.033	0.99	5	✓
48	Solution Cost												18,379.32	0.60	5	✓
	Stored States												275,293	0.80	4	✓
	Generated States												277,282	0.80	4	✓
	Runtime (Seconds)												0.181	0.80	4	✓

ber of stored states, the number of generated states, and the actual runtime. A check mark indicates that the algorithm is within one percent of the optimal performance reported in the row. An empty cell indicates that the algorithm does not solve all of our random instances. The table shows the following trends: First, all runtimes are small because the algorithms quickly either solve a search problem or run out of memory. Second, KWA* has a smaller solution cost than WA* and MSC-WA*. The solution cost of MSC-WA* remains roughly the same as the one of WA*, whereas the one of KWA* improves with N compared to the one of WA*, which can be attributed to the breadth-first component (diversity) of KWA*. Third, MSC-WA* has a smaller number of generated states than WA* and KWA*, which can be attributed to the depth-first component (commitment) of MSC-WA*. Fourth, both KWA* and MSC-WA* store a smaller number of states than WA* although neither KWA* nor MSC-WA* dominates the other one in this respect. (KWA* stores a smaller number of states than WA* only for large values of N .) The small memory consumption of KWA* and MSC-WA* allows them to solve all of our random instances of the 35-Puzzle, while WA* solves only 98 percent of our random instances, even for the best possible value of W . For KWA*, this result is partly due to our improved implementation of KWA* that stores each state at most once on the OPEN and CLOSED list. However, it turns out that the low memory consumption of our implementation of KWA* is *not* a systematic effect but due to noise since it requires one to hit the correct value of K precisely, which is not possible in practice.

4 MSC-KWA*

Although the concepts of diversity and commitment seem to be opposing at first glance, we now combine them: MSC-WA* splits the OPEN list into a COMMIT list of C states and a RESERVE list. This makes MSC-WA* even more focused than WA*, and thus makes it more likely that MSC-WA* is led into goal-free regions of the search space by misleadingly small heuristic values. We alleviate this problem by adding to MSC-WA* the mechanism used by KWA*. MSC-KWA*, the resulting algorithm, expands in parallel during each iteration all $K \leq C$ states with the smallest f -values in the COMMIT list (rather than just one). Thus, MSC-KWA* has three parameters, namely W , C and K . We eliminate

one of them by setting $K = C$ to simplify the experimental conditions, which is reasonable since decreasing K and C both increase the focus of MSC-KWA*. Table 1 shows that MSC-KWA* has a smaller number of stored states, a smaller number of generated states, and a smaller actual runtime than WA*, KWA* and MSC-WA*. The savings are significant and increase with N . For example, the number of stored states of MSC-KWA* is at least a factor of seven smaller for the 35-Puzzle than the ones of WA*, KWA* and MSC-WA*. The small memory consumption of MSC-KWA* allows it to solve all of our random instances of the 48-Puzzle, which WA*, KWA* and MSC-KWA* cannot. (KWA* solves only 76 percent and MSC-WA* solves only 78 percent of our random instances, even if their parameter values are optimized.) Furthermore, the solution cost of MSC-KWA* is always within about three percent of the solution cost of KWA*, the algorithm with the smallest solution cost. Additional experiments are reported in [Furcy, 2004]. For example, MSC-KWA* (with $K \neq C$) solves all of our random instances of the 4-Peg Towers of Hanoi, which WA*, KWA* and MSC-WA* cannot. Note that beam search is a special case of MSC-KWA* (namely, with $K = C$ and no RESERVE list) and thus can be understood as a variant of best-first search with commitment and diversity, not just commitment (the current view). It is future work to determine the best parameters values of MSC-KWA* automatically.

References

- [Felner *et al.*, 2003] A. Felner, S. Kraus, and R. Korf. KBFS: K-best-first search. *Annals of Mathematics and Artificial Intelligence*, 39:19–39, 2003.
- [Furcy, 2004] D. Furcy. *Speeding Up the Convergence of Online Heuristic Search and Scaling Up Offline Heuristic Search*. PhD thesis, College of Computing, Georgia Institute of Technology, Atlanta (Georgia), 2004.
- [Kitamura *et al.*, 1998] Y. Kitamura, M. Yokoo, T. Miyaji, and S. Tatsumi. Multi-state commitment search. In *Proceedings of the International Conference on Tools for Artificial Intelligence*, pages 431–439, 1998.
- [Pohl, 1970] I. Pohl. First results on the effect of error in heuristic search. In B. Meltzer and D. Michie, editors, *Machine Intelligence*, volume 5, pages 219–236. American Elsevier, New York, 1970.

A Covering Problem for Hypercubes

Jörg Hoffmann

Max-Planck-Institut für Informatik
Saarbrücken, Germany
hoffmann@mpi-sb.mpg.de

Sebastian Kupferschmid

Universität Freiburg, Institut für Informatik
Freiburg, Germany
kupfersc@informatik.uni-freiburg.de

Abstract

We introduce a new NP-complete problem asking if a “query” hypercube is (not) covered by a set of other “evidence” hypercubes. This comes down to a form of constraint reasoning asking for the satisfiability of a CNF formula where the logical atoms are inequalities over single variables, with possibly infinite variable domains. We empirically investigate the location of the phase transition regions in two random distributions of problem instances. We introduce a solution method that iteratively constructs a representation of the non-covered part of the query cube. In particular, the method is not based on backtracking. Our experiments show that the method is, in a significant range of instances, superior to the backtracking method that results from translation to SAT, and application of a state-of-the-art DP-based SAT solver.

This paper is an extended abstract. More details can be found in the long version of the paper [Hoffmann and Kupferschmid, 2005].

We introduce a new NP-complete problem asking if there is a point in a given n -dimensional “query” hypercube that is not covered by – contained in the union of – a set of other n -dimensional “evidence” hypercubes. An n -dimensional hypercube is a cross product of n intervals. Intervals in our context are defined as statements of the form $l < [\leq] x < [\leq] u$ where “ x ” is a variable, “ l ” and “ u ” are members of x ’s domain, and “ $<$ ” is a total order defined over this domain.¹

Definition 1 Let *QCOVER* denote the following problem: Given an n -dimensional hypercube Q , and a set $\mathcal{E}$ of n -dimensional hypercubes, is there a point in Q that is not contained in $\bigcup_{E \in \mathcal{E}} E$?

Covering problems of this kind arise, e.g., in the context of regression planning with numeric state variables [Koehler, 1998]. More generally, QCOVER is a form of constraint reasoning asking for the satisfiability of a CNF formula where the logical atoms are inequalities over single variables, with possibly infinite variable domains. The correspondence is the

¹By square parentheses “*syml*1 [*syml*2]” we denote alternative possibilities, i.e. that *syml*2 can be substituted for *syml*1.

following. A QCOVER instance is a constraint problem with n variables x^d . The query hypercube specifies a region inside which the solution must lie, the evidence cubes specify regions inside which the solution must *not* lie. A hypercube corresponds to a conjunction of inequalities of the form $c < [\leq, >, \geq] x^d$. So the complement of a hypercube (of an evidence hypercube) corresponds to a *disjunction* of such inequalities, and the overall problem is a conjunction of disjunctive constraints. Vice versa, any conjunction of such disjunctive constraints can be expressed as hypercubes (if a disjunctive constraint does not mention a variable x^d , then the interval in dimension d is the whole variable domain).

Proposition 1 *QCOVER* is NP-complete.

We empirically explore two random distributions of QCOVER instances. The first one, which we call *Random QCOVER*, chooses the end points for all intervals uniformly from a set of m possible values. The second one, which we call *Random 3-QCOVER*, is similar to the fixed clause-length model for generating random 3SAT instances [Mitchell *et al.*, 1992]. It always selects the query cube to be the cross-product of the (whole) variable domains, and, in the evidence cubes, assigns the whole variable domains to all but 3 randomly chosen dimensions. For both distributions, we investigate the location of the phase transition regions. As it turns out, Random 3-QCOVER shows a typical phase transition behaviour while Random QCOVER shows no such behaviour, see Figure 1.

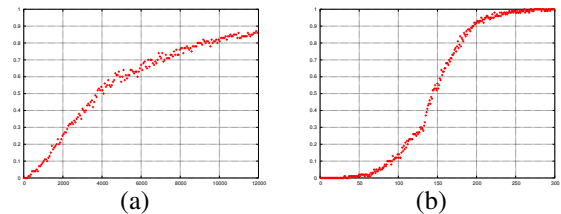


Figure 1: Typical plot of the proportion of unsatisfiable instances against k for (a) Random QCOVER and (b) Random 3-QCOVER.

By 50% point, we denote the number k of evidence cubes at which our random instances have (empirically) equal probability of being satisfiable or unsatisfiable. The 50% point depends on n and m . We use instances from the 50% points to evaluate different solution methods.

Today, in most cases the empirically most efficient solution methods for satisfiability problems are backtracking methods. Such methods are depth-first searches that split, in each search node, the search space along the possible values of a variable. A polynomial propagation of constraints is used to determine conflicts early on in the search tree, and analysis of conflicts is used to prune unnecessary branches. Methods of this kind have proved successful for solving constraint satisfaction problems. In particular, the modern descendants of the Davis Putnam procedure still constitute the state-of-the-art in determining the solvability of propositional CNF formulas. In our work, we have implemented a backtracking method for QCOVER by plugging a straightforward translation to SAT into Chaff [Moskewicz *et al.*, 2001].² To contrast this method, we have also developed an algorithm, named *cube elimination*, that, instead of backtracking over possible variable values, iteratively constructs a representation of the non-covered part of the query cube. See Figure 2.

```

procedure cube elimination( $Q, \mathcal{E}$ )
 $\mathcal{Q} := \{Q\}$ 
for all  $E \in \mathcal{E}$  do
   $\mathcal{Q}' := \emptyset$ 
  for all  $Q' \in \mathcal{Q}$  do
     $\mathcal{Q}' := \mathcal{Q}' \cup \text{minimal cover}(Q' \setminus E)$ 
  endfor
   $\mathcal{Q} := \mathcal{Q}'$ 
  if  $\mathcal{Q} = \emptyset$  then answer “unsatisfiable”, return  $\emptyset$  endif
endfor
answer “satisfiable”, return  $\mathcal{Q}$ 

```

Figure 2: Cube elimination.

The algorithm maintains a set $\mathcal{Q}$ of hypercubes that initially contains only the query cube itself. Then iteratively all evidence cubes E are “eliminated” by subtracting them from all cubes Q in $\mathcal{Q}$. The result $Q \setminus E$ of such a subtraction is not necessarily a hypercube; we represent it as a *set* of hypercubes, computed by the *minimal cover* procedure. The latter is a simple **for**-loop over all dimensions, returning a set of hypercubes that covers exactly $Q \setminus E$, and that is minimal in the sense that there is no smaller set of hypercubes covering exactly $Q \setminus E$ (the worst-case size of the set is $2n$).

In SAT, where all variables are boolean and have only two possible values, cube elimination comes down to transforming the CNF into a DNF. This seems a hopeless approach, but, for our Random QCOVER distribution, our experiments show that cube elimination is often superior to the backtracking method implemented by Chaff. Figure 3 shows our results for the Random QCOVER distribution, in terms of (nr. of search decisions made by Chaff) divided by (total nr. of cubes generated by cube elimination). Clearly, cube elimination becomes superior as the value of m increases. This is particularly true in the unsatisfiable instances where the cube elimination search space is, in the largest instances, around 5 orders of magnitude smaller than that of backtracking. It

²The translation implements the multiple-valued variable for each dimension by a set of boolean variables a that cover (only) the relevant case distinctions in that dimension. Binary exclusion clauses of the form $\neg a \vee \neg a'$ ensure that, by unit propagation, only one variable a can be set to 1 at a time.

should be noted that cube elimination produces a representation of *all* satisfying assignments in the satisfiable cases.

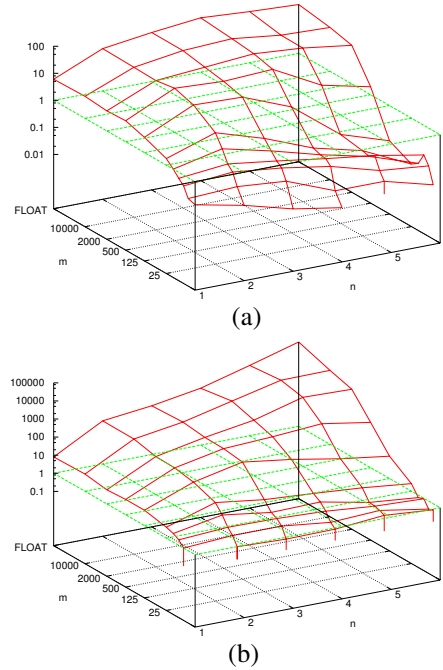


Figure 3: Search space size quotient Chaff vs. cube elimination in Random QCOVER, averaged over (a) all instances, (b) only unsatisfiable instances. The z -axis is log-scaled, the plain $z = 1$ is included for orientation.

In the Random 3-QCOVER distribution, we found backtracking to be generally superior to cube elimination. For lack of space, we omit the details. In spirit, cube elimination is somewhat similar to the “bucket elimination” framework defined by Rina Dechter [1999]. The cube elimination algorithm is easiest to understand, and was originally motivated by, viewing the satisfiability problem we consider as a *geometrical* problem. This opens up the question if geometrical interpretations of other satisfiability problems can lead to interesting new methods for these problems.

References

- [Dechter, 1999] R. Dechter. Bucket elimination: A unifying framework for reasoning. *AI*, 113:41–85, 1999.
- [Hoffmann and Kupferschmid, 2005] J. Hoffmann and S. Kupferschmid. A covering problem for hypercubes. TR 218, Universität Freiburg, 2005. At [ftp://ftp.informatik.uni-freiburg.de/documents/reports/report218/report00218.ps.gz](http://ftp.informatik.uni-freiburg.de/documents/reports/report218/report00218.ps.gz).
- [Koehler, 1998] J. Koehler. Planning under resource constraints. In *Proc. ECAI-98*, pages 489–493.
- [Mitchell *et al.*, 1992] D. Mitchell, B. Selman, and H. Levesque. Hard and easy distributions of SAT problems. In *Proc. AAAI-92*, pages 459–465.
- [Moskewicz *et al.*, 2001] M. Moskewicz, C. Madigan, Y. Zhao, L. Zhang, and S. Malik. Chaff: Engineering an efficient SAT solver. In *Proc. DAC’01*.

Predicate-Oriented Isomorphism Elimination in Model Finding*

Xiangxue Jia and Jian Zhang
Laboratory of Computer Science
Institute of Software
Chinese Academy of Sciences
Beijing 100080, China
Email: {jxx, zj}@ios.ac.cn

Abstract

Finding models of logical formulas is a challenging problem. For first-order formulas, a finite model can be found by exhaustive search. For many structured problem instances, there is much isomorphism in the search space. This paper proposes general-purpose techniques for eliminating isomorphic subspaces, which can be helpful when the formulas have many predicates. The techniques are based on inherent symmetries in first-order clauses.

1 Introduction

Satisfiability checking (or model finding) is an important problem in logic and Artificial Intelligence. If restricted to the propositional logic, the problem is well-known as SAT, and has been studied by many people. In this paper, we focus on first-order satisfiability problems. More specifically, we study how to find finite models of first-order formulas. We propose two methods which use symmetries to eliminate isomorphic subspaces and prune the search tree. One technique works row-by-row; the other one processes submatrices, based on Ramsey numbers.

Symmetry breaking can be very helpful to improve the efficiency of combinatorial searching. For example, it has played a key role in the solution of some previously open questions on quasigroups [Fujita *et al.*, 1993]. But the technique used there is problem-specific. Other researchers have studied more general methods for using symmetries to improve the efficiency of model finding. The Least Number Heuristic (LNH) [Zhang and Zhang, 1995] is such a technique. It works well on many problems (including the quasigroup problems). But it is not effective on predicates. The techniques proposed in this paper can remedy this weakness to some extent.

2 Background

We assume that the reader is familiar with the basic concepts of classical logics (including propositional and first-order logics). As mentioned earlier, this paper deals with the problem of finding finite models of first-order formulas. Currently, there are some automated tools available for doing this. Tools

like SEM [Zhang and Zhang, 1995] search for the models directly, while other tools (like MACE2 [McCune, 2001]) translate the problem into SAT and use efficient SAT solvers.

For simplicity, we assume that there is only one domain: $D_n = \{0, 1, \dots, n-1\}$. (But the methods can also be used on many-sorted formulas.) Suppose there is a binary predicate, P , in the formulas. To give an interpretation for P , we need to find an $n \times n$ matrix, in which each entry (also called a *cell*) has a truth value (TRUE or FALSE).

A set of ground clauses, Φ , is *symmetric* with respect to a set of domain elements, if Φ remains the same under any permutation of the elements. When Φ is obvious from the context, we also say that the elements are *symmetric* or *interchangeable* with each other. Clearly, this is an equivalence relation defined on a set of domain elements.

For convenience, we assume that all the domain elements are symmetric initially. We may think that there is only one equivalence class on the domain elements. This class includes every element of D_n .

Ramsey Numbers

Ramsey numbers are well-known in graph theory. For a pair of positive integers p and q ($p > 1, q > 1$), the Ramsey number $r = R(p, q)$ is the minimum number of vertices such that all undirected simple graphs of order r contain a clique of order p or an independent set of order q .

Currently we do not know the exact values of many Ramsey numbers. But we know that

$$\begin{aligned} R(2, q) &= q \quad \text{for any } q, \\ R(3, 3) &= 6, \quad R(3, 4) = 9, \quad R(3, 5) = 14, \dots \\ R(4, 4) &= 18, \quad R(4, 5) = 25 \end{aligned}$$

and obviously, for any p and q , $R(p, q) = R(q, p)$.

3 Row-by-row Isomorphism Elimination

Suppose there is a model. Let us consider an interpretation for P . First look at the first row of the matrix, that is, the value of $P(0, y)$ (for $y > 0$). For these $(n-1)$ truth values, either of the following propositions must hold, but not both:

- (1) At least $\lceil (n-1)/2 \rceil$ cells got the value TRUE;
- (2) At least $\lceil n/2 \rceil$ cells got the value FALSE.

Here $\lfloor x \rfloor$ denotes the largest integer i such that $i \leq x$, and $\lceil y \rceil$ denotes the smallest integer j such that $j \geq y$.

*Supported by the NSF of China (No. 60125207).

If (1) is true, without loss of generality, we can assume that the cells

$$P(0, \lfloor (n-1)/2 \rfloor + 1), \dots, P(0, n-1)$$

take the value TRUE. Similarly, if (2) is true, we assume that the second half of the first row got the value FALSE. So we have two subcases.

The above kind of “divide-and-conquer” can be repeated until the size of the equivalence class is reduced to 1. And we can get many subcases.

We call this technique the *Row-by-Row (RR) strategy*. Now we prove its correctness. Let GC denote the set of ground clauses obtained by instantiating the first-order clauses using all the elements in the domain D_n .

Lemma 1. Suppose $1 < t \leq n-1$, $0 \leq k < n-t$, and GC is symmetric with respect to $\{n-t, \dots, n-1\}$. Then any model can be mapped (under certain permutation of domain elements) to one of the following two models:

- (1) a model in which $P(k, n-r_1), \dots, P(k, n-1)$ are TRUE, where $r_1 = \lceil t/2 \rceil$;
- (2) a model in which $P(k, n-r_2), \dots, P(k, n-1)$ are FALSE, where $r_2 = \lceil (t+1)/2 \rceil$.

Theorem 1. If GC is symmetric with respect to D_n , there is a model of GC in D_n if and only if a model exists in one of the subcases.

For a given row, there can be several ways of dividing the assignments into two subcases. For example, we could divide the assignments at the first row into these two subcases:

- (1) $P(0, 1)$ is TRUE;
- (2) $P(0, y)$ is FALSE, for every $y \geq 1$.

4 A Technique Based on Ramsey Numbers

Now we present a search strategy based on the Ramsey numbers, which is called the *RN strategy*. It tries to determine the values of several submatrices, starting from the upper-left corner to the lower-right corner.

If P is commutative, i.e., $P(x, y) = P(y, x)$ for any x, y , then the matrix can be regarded as a complete graph whose edges are colored with TRUE or FALSE. When n is greater than or equal to the Ramsey number $R(p, q)$, there will be a complete subgraph K_p with color TRUE, or a complete subgraph K_q with color FALSE. Without loss of generality, we can divide the model finding process into two subcases:

- (1) For any i, j ($0 \leq i, j < p$, $i \neq j$), $P(i, j)$ and $P(j, i)$ are TRUE.
- (2) For any i, j ($0 \leq i, j < q$, $i \neq j$), $P(i, j)$ and $P(j, i)$ are FALSE.

When n is a bigger number, we can apply this kind of “divide-and-conquer” several times.

If P is not commutative, then $P(x, y)$ ($x < y$) can be regarded as a complete undirected graph whose edges are colored with TRUE or FALSE. Similarly, we can also divide the model searching process into two subcases:

- (1) For any i, j ($0 \leq i < j < p$), $P(i, j)$ is TRUE.
- (2) For any i, j ($0 \leq i < j < q$), $P(i, j)$ is FALSE.

To avoid repeated search, we can put the following condition on the first subcase: There is no K_q colored with FALSE.

5 Experiments

We have tested our techniques on several problems, including

- Pigeon-hole
- Ramsey Number Searching
- Steiner Triple System

Several programs are used, e.g., SEM [Zhang and Zhang, 1995], MACE2 [McCune, 2001] and the SAT solver *siege_v4* (<http://www.cs.sfu.ca/~loryan/personal/>).

Experimental results show that, typically our techniques can reduce the search time by a factor of 2 to 8.

6 Concluding Remarks

For many structured problems, if they are specified naturally using first-order logic, the specification suggests a lot of symmetries which can be used to prune the search space. This paper proposes two techniques for using such symmetries. They are mainly aimed at predicates. One of them works row by row, and the other one processes submatrices from the upper-left corner to the lower-right corner. The two techniques may be regarded as complementary to the least number heuristic and its extension. They can be helpful to both first-order and propositional solvers.

Symmetries have also been studied in the SAT community. For example, Aloul *et al.* [Aloul *et al.*, 2003] have obtained very impressive results on various problems in the EDA field. They have implemented a software package called Shatter which can add certain clauses to a SAT instance to make it easier. Such an approach typically has to search for symmetries in large CNF formulas. For some SAT instances, the search may take a long time. Compared with a propositional formulation, a specification in the first-order logic allows one to use symmetries without searching for them.

In the future, we are going to study how to combine various techniques in a single framework. It is also interesting to compare first-order techniques with symmetry-breaking predicates used for SAT solvers.

References

- [Aloul *et al.*, 2003] Fadi A. Aloul, Arathi Ramani, Igor L. Markov, and Kareem A. Sakallah. Solving difficult instances of boolean satisfiability in the presence of symmetry. *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, 22(9):1117–1137, 2003.
- [Fujita *et al.*, 1993] Masayuki Fujita, John Slaney, and Frank Bennett. Automatic generation of some results in finite algebra. In *Proceedings of the 13th Int'l Joint Conf. on Artif. Intel. (IJCAI)*, pages 52–57. Morgan Kaufmann, 1993.
- [McCune, 2001] William McCune. *MACE 2.0 Reference Manual and Guide*. Technical Memo. ANL/MCS-TM-249, Argonne National Laboratory, Argonne, IL, 2001.
- [Zhang and Zhang, 1995] Jian Zhang and Hantao Zhang. SEM: a system for enumerating models. In *Proceedings of the 14th Int'l Joint Conf. on Artif. Intel. (IJCAI)*, pages 298–303. Morgan Kaufmann, 1995.

Maintaining Arc Consistency using Adaptive Domain Ordering*

Chavalit Likitvivatanavong

Cork Constraint Computation Centre
University College Cork, Ireland

Yuanlin Zhang

Department of Computer Science
Texas Tech University, Texas, U.S.

James Bowen and Eugene C. Freuder

Cork Constraint Computation Centre, University College Cork, Ireland

1 Introduction

Solving a Constraint Satisfaction Problem (CSP) by Maintaining Arc Consistency (MAC) [Sabin and Freuder, 1994] has been one of the most widely-used methods. Since Arc Consistency (AC) is enforced at every node in a search tree, its efficiency is critical to the whole algorithm. We propose a new MAC algorithm based on AC-3.1 [Zhang and Yap, 2001] in which the AC component is capable of starting from where it left off in its previous execution with low overhead. It has the following properties: (1) $O(ed^2)$ worst-case time complexity in any node and any branch of the search tree; (2) $O(ed)$ space complexity; (3) the ability to avoid a type of redundant constraint check called a *negative repeat*; (4) no re-computation and maintenance of its internal data structures upon backtrack.

2 Definitions

In this paper we consider CSPs with binary constraints and use D_X , n , e , and d to denote the current domain of variable X , the number of variables, the number of constraints, and the maximum domain size respectively. We also assume a domain to be totally-ordered and adopt a random-access doubly linked-list implementation. The terms head and tail denote the boundary of a domain. A constraint check between value $a \in D_X$ and value $b \in D_Y$ is denoted by $C_{XY}(a, b)$.

A *propagation-oriented backtrack search* algorithm for CSPs is the standard depth-first backtrack search framework augmented by some process that handles all constraint propagation and the maintenance of the internal data structures involved. The *search tree* is defined by associating each node with a variable assignment of the algorithm. *Node complexity* of the algorithm is the time complexity cost of the constraint propagation performed at each node. *Path complexity* is the aggregate cost for any path in the search tree, summing the cost of every node in succession, starting from the root, to a leaf. During search some constraint checks may be repeated many times even though the constraint processing component is optimal in a single execution. We define the following type of redundant check called a *negative repeat*. Negative repeats can be troublesome for hard problems that require a large amount of backtracking.

Definition 1 (Negative Repeat) A constraint check $C_{XY}(a, b)$ performed at time t during search is called a *negative repeat* with respect to Y if and only if:

- (1) $C_{XY}(a, b) = \text{false}$, and
- (2) $C_{XY}(a, b)$ has been performed at time s where $s < t$, and
- (3) b has been continuously present in the time interval (s, t) .

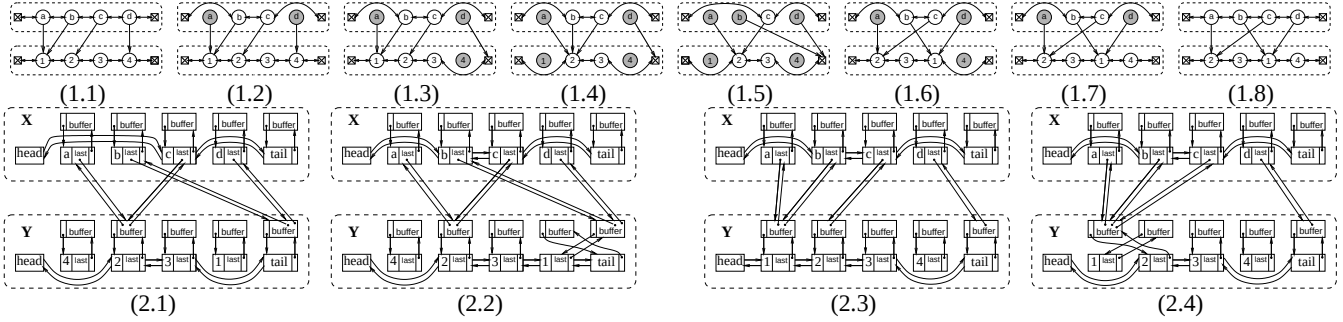
3 Adaptive Domain Ordering

In this section we introduce a new MAC algorithm called Adaptive Domain Ordering (ADO). The key feature is the dynamic rearrangement of variable domains after each backtrack. A pruned value is simply restored to the end of its domain, rather than its initial position. This technique makes the algorithm capable of avoiding *all* negative repeats as well as requiring no maintenance on its internal data structure. Unlike AC-3.1, on which the constraint processing unit of ADO is based, we associate the following two invariants for $\text{last}(X, a, Y)$. First, the *safety invariant*: there exists no support of a in $\{c \in D_Y \mid c < \text{last}(X, a, Y)\}$. Second, the *prospect invariant*: there exists a support of a in $\{c \in D_Y \mid c \geq \text{last}(X, a, Y)\}$. ADO is a propagation-oriented backtrack search algorithm and behaves like MAC-3.1 for the most part. The main differences lie in the routines for removing a value (remove), finding a support, and restoring a value (restore). When a value is removed, any last that points to it will be moved to the next value. When no support for $a \in D_X$ is found in D_Y , $\text{last}(X, a, Y)$ will be made to point to tail of Y . When a value is restored, all the last pointers that point to the tail will be made to point to the restored value instead. It can be proved that, given any path in a search tree, the worst-case aggregate complexity of remove is $O(ed^2)$. Due to space restriction we are not able to give detailed proofs of the complexity cost and correctness of ADO. An example trace of the algorithm is shown as follows.

Example Consider $D_X = \{a, b, c, d\}$, $D_Y = \{1, 2, 3, 4\}$ and $C_{XY} = \{(a, 1), (b, 1), (c, 2), (d, 4)\}$ (allowed tuples). The result after the initial AC processing¹ is shown in Figure (1.1): $\text{last}(X, a, Y)=1$, $\text{last}(X, b, Y)=1$, $\text{last}(X, c, Y)=2$, and $\text{last}(X, d, Y)=4$. Suppose a and d are removed due to some external cause; their last values remain unchanged (1.2). Next, suppose 4 is removed. The result after AC processing is shown in (1.3). At

*This work has received support from Science Foundation Ireland under Grant 00/PI.1/C075.

¹In truth, we only enforce directional AC on the arc (X, Y) to keep the example simple.



the next search level suppose 1 is removed. According to the algorithm, any last that points to 1 — including $\text{last}(X, a, Y)$ — will be shifted to 2 (1.4). Figure (1.5) shows the result after the problem is made arc consistent. Now consider the network after backtrack. Since 1 is removed after 4, it must be restored before. Figure (1.6) shows the result after 1 is restored. All the last pointers that pointed to tail are moved to 1. Notice that b is restored as well because it was pruned at the same level as 1. Figure (1.7) shows the result after 4 is restored. Figure (1.8) shows the network after the search backtracks to the point where we started this example. Note how the last pointers and the domain ordering differ from (1.1). $\square$

To make ADO efficient, we need another data structure associated with each value to account for the last pointers that point to it. This is called buffer. Its elements can be preallocated since a value has only one last pointer in any constraint. We then make a last pointer refer to a value's buffer instead of the value itself — i.e. $(a, X) \in \text{buffer}(b, Y)$ iff $\text{last}(X, a, Y)$ points to $\text{buffer}(b, Y)$. This allows a set of last pointers to be switched all at once just by rearranging related buffers. For example, consider Figure (2.1), a more detailed view of (1.5). When 1 is restored, we want to move all the pointers from tail to 1. This can be done simply by swapping the two buffers (2.2). As a result restore takes $O(1)$ time.

We use a similar technique for remove . We compare the buffer of the value to be removed with that of the next value in the current domain (or tail if there is none) and swap both buffers if the first contains more elements. For an example, consider Figure (2.3), which is the detailed view of Figure (1.3). When value 1 is removed, its buffer size is compared with the buffer size of value 2. Since it has more elements, we move pointers from the buffer for value 2 into that for value 1 and swap both buffers, which results in (2.4).

Given any path in a search tree, the worst-case aggregate complexity of remove using buffer with the above techniques is $O(ed \lg d)$. We can further reduce the cost by representing data in a buffer as a rooted tree and having the root of the smaller tree point to that of the larger one. In fact, this is the union-by-rank operation for the disjoint-set union problem. remove then takes $O(1)$ time. However, locating the value of a last pointer no longer takes constant time. By using path compression, its worst-case aggregate complexity is $O(ed \alpha(ed^2))$ where α is the inverse Ackerman's function, which is almost constant [Tarjan, 1975].

ADO has $O(ed^2)$ for node and path complexity since the cost related to buffer is subsumed by the standard cost of establishing arc consistency. Correctness follows because: (1) the prospect invariant states that there is a support somewhere

for each value, thereby enforcing arc consistency; (2) the safety invariant states that no support is skipped. Note that because the problem is made arc consistent before the search starts, we can always expect a value to have a support. If a value is subsequently removed, it is only because all of its supports are pruned; these supports are still in the original domain. The existence of a support in the original domain is central in the correctness proofs, and without the AC preprocessing before the search starts ADO would not be correct.

4 Conclusions

We have designed ADO to explore the theoretical limits of MAC and, as far as we know, it achieves the best results and outperforms all other existing MAC algorithms. Specifically, ADO has $O(ed^2)$ worst-case time complexity in any node and any branch of the search tree while using only $O(ed)$ space. This resolves the trade-off between the two traditional implementations of MAC-3.1. The first one records every change made to last so that after backtrack the algorithm can start from the exact same state. Although both node and path complexity for this approach is $O(ed^2)$, its space complexity is $O(ed \min(n, d))$. The second approach resets last in every search node to keep the space at $O(ed)$, but this comes at the expense of path complexity, which becomes $O(ned^2)$.

Régis [2004] also aims to create a maintenance-free MAC algorithm that has the best features from the two implementations of MAC-3.1. The algorithm in [Régis, 2004] is both node and path optimal while using $O(ed)$ space. However, the last structure needs to be recomputed and updated after each backtrack besides the normal restoration of pruned values. It cannot avoid negative repeats. By contrast, ADO requires no recomputation and no update of its internal structure and is able to avoid negative repeats.

References

- [Régis, 2004] J.-C. Régis. Maintaining arc consistency algorithms during the search with an optimal time and space complexity. In *CP-04 Workshop on Constraint Propagation and Implementation*, 2004.
- [Sabin and Freuder, 1994] D. Sabin and E. C. Freuder. Contradicting conventional wisdom in constraint satisfaction. In *Proceedings of ECAI-94*, pages 125–129, 1994.
- [Tarjan, 1975] R. E. Tarjan. Efficiency of a good but not linear set union algorithm. *Journal of ACM*, 22(2):215–225, 1975.
- [Zhang and Yap, 2001] Y. Zhang and R. H. C. Yap. Making AC-3 an optimal algorithm. In *Proceedings of IJCAI-01*, pages 316–321, 2001.

Combination of Local Search Strategies for Rotating Workforce Scheduling Problem

Nysret Musliu

Technische Universität Wien
Karlsplatz 13, 1040 Wien, Austria
musliu@dbai.tuwien.ac.at

Abstract

Rotating workforce scheduling is a typical constraint satisfaction problem which appears in a broad range of work places (e.g. industrial plants). Solving this problem is of a high practical relevance. In this paper we propose the combination of tabu search with random walk and min conflicts strategy to solve this problem. Computational results for benchmark examples in literature and other real life examples show that combination of tabu search with random walk and min conflicts strategy improves the performance of tabu search for this problem. The methods proposed in this paper improve performance of the state of art commercial system for generation of rotating workforce schedules.

1 Introduction

Workforce scheduling, in general, includes sub-problems which appear in many spheres of life as in industrial plants, hospitals, public transport, airlines companies, universities etc. There are two main variants of workforce schedules: rotating (or cyclic) workforce schedules and non-cyclic workforce schedules. In a rotating workforce schedule all employees have the same basic schedule but start with different off-sets. For this problem we have given (for formal definition see [Musliu *et al.*, 2002]): number of employees: n , set of m shifts, length of schedule, temporal requirements, constraints about the sequences of shifts permitted to be assigned to employees (the complement of inadmissible sequences), maximum and minimum length of periods of consecutive shifts and maximum and minimum length of blocks of work days. The aim is to find a cyclic schedule (assignment of shifts to employees) that satisfies the requirement matrix and fulfill all given hard constraints.

Workforce schedules have an impact on the health and satisfaction of employees as well as on their performance at work. Therefore, computerized workforce scheduling has interested researchers for over 30 years. For solving the problem of workforce scheduling, different approaches have been used in the literature. Survey of algorithms used for different workforce scheduling problems is given by Alfares [Alfares, 2004]. Balakrishnan and Wong [Balakrishnan and Wong,

1990] solved a problem of rotating workforce scheduling by modeling it as a network flow problem. Musliu *et al.* [Musliu *et al.*, 2002] proposed and implemented a new method for the generation of rotating workforce schedules, which is based on pruning of search space, by involving the decision maker during the generation of partial solutions. The algorithm has been included in a commercial product called First Class Scheduler, which is already being used by several companies in Europe since 2000. Recently, Musliu [Musliu, 2003] proposed a simple tabu search approach for this problem. In this paper we propose improvement of tabu search technique proposed in [Musliu, 2003] by combination of this technique with random walk and min conflicts heuristic.

2 Combination of Local search strategies

In this paper we propose a combination of tabu search with random walk and min conflicts technique. The combination of tabu search with random walk works as follows: For each iteration with probability p randomly the cell (in the schedule) which is in conflict is picked. Another cell is picked in the same column randomly and the contents of two selected cells are swapped. With probability $1 - p$, the tabu search algorithm is applied.

The random walk strategy in our case picks with probability p , first randomly one cell which appears in some violated constraint (conflict). The content of this cell is swapped with the content of other cell which is selected randomly in the same column. As described later the solutions during each iteration fulfill always the workforce requirements. This is the reason why we allow the swap of contents of cells only inside of one column.

Further, we consider also combination of tabu search with min-conflicts strategy. The procedure applies standard min conflicts strategy for each iteration with probability p . With probability $1 - p$, it follows the tabu search algorithm.

The min-conflict strategy for this problem works as follows. Randomly a cell in schedule (variable) which is in a conflict is picked. Further, all solutions are generated which can be obtained by swapping the content of selected cell and other cells in the same column. From the generated neighborhood the solution which minimizes number of conflicts is selected for the next iteration. Note that there is a small difference compare to pure min conflicts heuristic ([Minton *et al.*, 1992]) due to the specifics of this problem. First, in

our case not only one variable in conflict changes the value, but two variables (which represents schedule cells in the same column) exchange their contents. Second, with our procedure also a worst solution can be accepted for the next iteration.

Considering generation of neighborhood in tabu search we apply a simple move, which swaps the contents of two elements in the schedule. However, to reduce the neighborhood of the current solution, we only allow those solutions which fulfill workforce requirements, and thus the swapping of elements is done only inside of a particular column (day of schedule). The whole neighborhood of the current solution is obtained by swapping all possible elements (not identical) in all columns of a table. The applied move is denoted as $swap(j, i, k)$, where j represents the column in which the swap of elements should be done, and i, k represent elements in column j that should swap their contents. Additionally, the above described move was extended to swap the blocks of neighborhood cells.

To calculate the fitness of the solution, for each violation of constraint a determined number of points (penalty) is given, based on the constraint and the degree of the constraint violation. The fitness represents the sum of all penalties caused from the violation of constraints. Since the problem we want to solve only has hard constraints, the solution will be found when the fitness of the solution reaches the value 0.

3 Computational results

In this section we report on computational results obtained with the current implementation of local search strategies. For comparison we collected a set of real life problems which can be used further by other researchers to compare their results with ours. The collection of these problems can be found on <http://www.dbai.tuwien.ac.at/staff/musliu/benchmarks/>. The problems for which we give the results in this paper contains from 7 to 64 groups. Each group can contain one or more employees. The first three examples are benchmark examples we could find in literature with some modification according the definition of problems we solve in this paper. Other 15 examples are real life examples of different size which appeared in different areas in practice. The experiments were performed in a machine Pentium 4, 1,8GHZ, 512 MB RAM. For each problem 10 independent runs were executed. Maximal number of evaluations for each run is 10000000. Note that we experimented with different parameters considering the probability for random walk, length of tabu list and type of tabu. In this paper the best results we could obtain for each strategy are presented.

In the Table 1 a summary of results for tabu search, tabu search with random walk, and tabu search with min conflicts heuristics is presented. For each problem the average time needed (in seconds) for generation of solution is presented. The column named Sol. (solutions) represents number of solutions found in 10 runs.

3.1 Discussion of results

From the Table 1 we can conclude that tabu search approach is improved with combination of tabu search with random walk considering both the average time needed to find the

Table 1: Results for three techniques

Ex.	TS		TS and RW		TS and MC	
	t(sec)	Sol	t(sec)	Sol	t(sec)	Sol
1	1.0	10	1.0	10	0.3	10
2	1.1	10	0.9	10	0.3	10
3	11.1	10	7.5	10	3.4	10
4	1.3	10	1.2	10	0.5	10
5	3.2	10	4.6	10	1.6	10
6	0.3	10	0.3	10	0.3	10
7	239.2	3	69.0	3	196.1	9
8	5.4	10	4.5	10	2.3	10
9	255.7	9	139.1	10	79.5	10
10	52.8	10	22.1	10	12.2	10
11	185.5	6	103.7	10	134.2	10
12	61.4	8	24.7	10	18.3	10
13	14.9	10	7.7	10	6.4	10
14	8.1	10	3.3	10	2.3	10
15	-	0	-	0	738.0	5
16	40.9	10	30.8	10	10.8	10
17	48.5	10	28.6	10	10.9	10
18	411.2	6	322.7	10	169.2	10

solutions and also the number of runs in which the solution is found. Combination of tabu search with min conflicts heuristic further improves tabu search considering the the time for which the solution are generated and the number of successful runs. The combination of tabu search with min conflicts strategy proposed in this paper improves the results obtained by FCS, which to our best knowledge is the state of art commercial systems for generation of rotating workforce schedules.

References

- [Alfares, 2004] H. K. Alfares. Survey, categorization, and comparison of recent tour scheduling literature. *Annals of Operations Research*, 127(1-4):145–175(31), 2004.
- [Balakrishnan and Wong, 1990] Nagraj Balakrishnan and Richard T. Wong. A network model for the rotating workforce scheduling problem. *Networks*, 20:25–42, 1990.
- [Minton *et al.*, 1992] Steven Minton, Mark D. Johnston, Andrew B. Philips, and Philip Laird. Minimizing conflicts: a heuristic repair method for constraint satisfaction and scheduling problems. *Artificial Intelligence*, 58:161–205, 1992.
- [Musliu *et al.*, 2002] Nysret Musliu, Johannes Gärtner, and Wolfgang Slany. Efficient generation of rotating workforce schedules. *Discrete Applied Mathematics*, 118(1-2):85–98, 2002.
- [Musliu, 2003] Nysret Musliu. Applying tabu search to the rotating workforce scheduling problem. In *The 5th Metaheuristics International Conference (MIC'03)*, Kyoto, Japan, 2003.

Corrective Explanation for Interactive Constraint Satisfaction *

Barry O’Sullivan, Barry O’Callaghan and Eugene C. Freuder

Cork Constraint Computation Centre

Department of Computer Science, University College Cork, Ireland

{b.osullivan|b.ocallaghan|e.freuder}@4c.ucc.ie

Abstract

Interactive tasks such as online configuration can be modeled as constraint satisfaction problems. These can be solved interactively by a user assigning values to variables. Explanations for failure in constraint programming tend to focus on conflict. However, what is often desirable is an explanation that is *corrective* in the sense that it provides the basis for moving forward in the problem-solving process. This paper defines this notion of *corrective explanation* and demonstrates that a greedy search approach performs very well on a large real-world configuration problem.

1 Introduction

To demonstrate the distinction between corrective explanations and more traditional conflict-based explanations, consider the example in Figure 1.

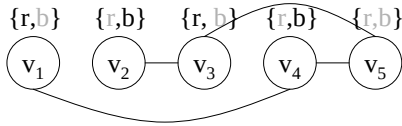


Figure 1: Example of corrective versus conflict-based explanations.

This figure presents assignments of colors to variables in a coloring problem. We have assigned colors to the first 4 variables (highlighted in black), only two colors are available, and we are running a forward checking algorithm, so we have encountered a domain wipeout on variable v_5 at this point. We will say that variables v_3 and v_4 form a “culprit set”. The assignments to those variables account for the wipeout. The variables v_3 and v_4 do not provide a corrective explanation because it is not possible by changing their instantiations alone to restore any choices to v_5 . $\{v_1, v_4\}$ is a *corrective explanation of inconsistency*, as alternative assignments can be found for these variables that enables the user to assign at least one more variable. Making v_1 b(lue) and v_4 r(ed) allows us to proceed to assign a value to v_5 , blue.

*This work has received support from Enterprise Ireland (Grant SC/02/289) and Science Foundation Ireland (Grant 00/PI.1/C075).

A *corrective explanation for value recovery* is useful when the user would like to restore a value that was removed due to some previous decision(s). Such an explanation identifies modifications to the user’s decisions guaranteeing that the desired value is restored and can be selected consistently. A corrective explanation is minimal if no (proper) subset is itself a corrective explanation. We define an optimal corrective explanation as one of minimum cardinality.

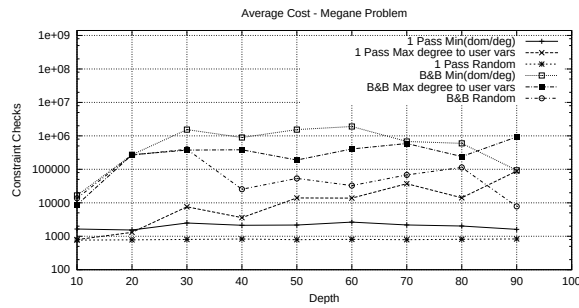
2 Computing Corrective Explanations

Instead of using compilation methods [Amilhastre *et al.*, 2002] or standard model-based diagnosis techniques [Junker, 2004; Reiter, 1987], we advocate the use of a heuristic search approach to find corrective explanations. While heuristic search cannot guarantee that minimum length explanations are found, near optimal explanations can often be found quite efficiently. We generate corrective explanations by returning any differences between the assignments made by the user and the assignments in a solution found using carefully chosen *variable and value ordering heuristics*. These heuristics attempt to maximize the number of user assignments in such a solution. The *value ordering heuristic* favors values chosen by the user during search, by selecting those first whenever possible, and uses standard heuristics for the remaining ones. We partition the set of variables into a set of user-assigned variables and a set of unassigned variables. User-assigned variables are considered first, and within each subset of variables a particular *variable ordering heuristic* is used. The intuition here is that during search the user’s choices are always considered first, so they are more likely to participate in the solution used to generate an explanation.

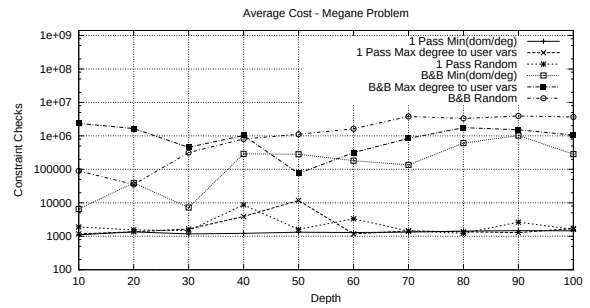
3 Experiments

We ran a suite of experiments based on the Renault Megane Configuration benchmark [Amilhastre *et al.*, 2002]. The problem consists of 101 variables, domain sizes vary from 2 to 43, and there are 113 non-binary constraints. The number of solutions to the problem is in excess of 1.4×10^{12} . The solver we used in our experiments was based on generalized forward checking.

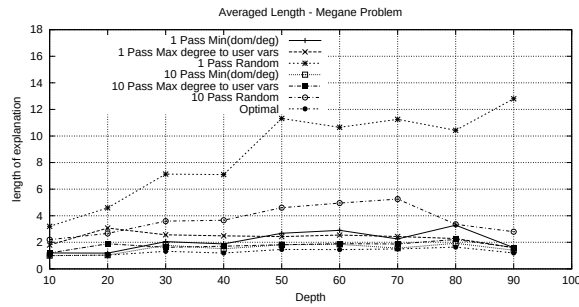
We studied three *variable ordering* heuristics for generating corrective explanations using the approach described above: *minimum ratio of domain over forward degree* (also



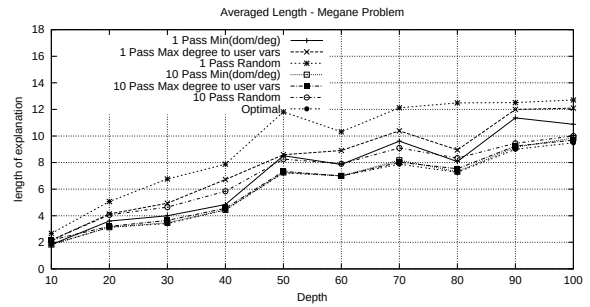
(a) Average cost of inconsistency explanations.



(b) Average cost of value recovery explanations.



(c) Average length of inconsistency explanations.



(d) Average length of value recovery explanations.

Figure 2: Results for finding explanations for the Renault Megane Configuration Benchmark.

referred to as minimum domain/degree), *maximum degree to the user's assigned variables* and *random*. We used a *random value ordering*, since it seemed to perform best overall.

Based on the variable and value ordering heuristics, we studied three different search techniques: 1) a branch-and-bound search seeking to minimize explanation length; 2) a greedy heuristic search for a solution that returns a single explanation; 3) running multiple greedy heuristic searches (10 in these experiments), providing multiple explanations, of which the one of minimum length was returned. The simulated user chose variables randomly and values lexicographically. We considered both the task of generating corrective explanations for inconsistency (over 800 configuration sessions), and corrective explanations for value recovery (over 100 sessions) in which the user chose to restore an inconsistent value with a probability of 0.1. We report averages at intervals of 10 variables.

In Figure 2(a) and Figure 2(b) we see that minimum length explanations tend to be very expensive to compute using a branch-and-bound search, thus not a viable option in an interactive context. In Figure 2(c) and Figure 2(d) we plot average explanation lengths. The greedy approaches, with the possible exception of that based on the random variable ordering, find almost minimum length corrective explanations using a fraction of the search cost of branch-and-bound.

The best approach of those studied was based on minimum domain/degree. For the inconsistency explanations experiment the average length of the explanations found was 2.15 variables, for the one solution version, and 1.60 variables when we proposed the best of 10 solutions. This compares

with an average optimal explanation length of 1.32 variables. The average search effort required to find a single explanation was 2,142 constraint checks. For the value recovery experiment, minimum domain/degree found explanations of length 8.22 and 7.16 for the one solution and best of 10 solutions, respectively. The average optimal length was 7.03. The search effort required to find a single explanation was 1,354 checks.

4 Future Work and Conclusion

The notion of corrective explanation is new in the domain of constraint satisfaction, where existing work on explanation is focused on conflict. Using a greedy search approach with good variable and value ordering heuristics seems an attractive option as it represents a good compromise between search cost and explanation length. Building upon the work of Junker [Junker, 2004] we plan to explore the potential for a preference-oriented relaxation-focused corrective explanation algorithm.

References

- [Amilhastre *et al.*, 2002] J. Amilhastre, H. Fargier, and P. Marguis. Consistency restoration and explanations in dynamic CSPs – application to configuration. *Artificial Intelligence*, 135:199–234, 2002.
- [Junker, 2004] U. Junker. QUICKXPLAIN: Preferred explanations and relaxations for over-constrained problems. In *Proceedings AAAI-04*, pages 167 – 172, 2004.
- [Reiter, 1987] R. Reiter. A theory of diagnosis from first principles. *Artificial Intelligence*, 32(1):57–95, 1987.

CSP Search with Responsibility Sets and Kernels

Igor Razgon and Amnon Meisels

Department of Computer Science,
Ben-Gurion University of the Negev,
Beer-Sheva, 84-105, Israel
{irazgon,am}@cs.bgu.ac.il

Abstract

We introduce data structures called responsibility set and kernel. We present an algorithm FC-RK, which is a modification of FC that maintains these structures and uses them for pruning of the search space. According to our experimental evaluation, FC-RK outperforms FC-CBJ on constraint networks encoding graph k -coloring instances and on non-dense random binary constraint networks.

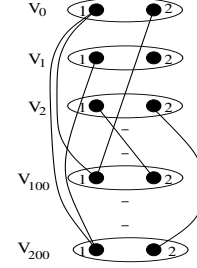


Figure 1: A constraint network.

1 Introduction

The present paper introduces a modification of the forward checking algorithm (FC) [Haralick and Elliott, 1980] that associates every removed value with two data structures: the *responsibility set* and the *kernel*. Having introduced the structures, we describe how they help to prune the search space. We define a notion of *filtering value*. Next, we prove the central theorem of the paper which claims that the existence of a filtering value in some state that occurs during work of FC means that the current partial solution of the state is a nogood. We modify FC so that every time after lookahead or backtrack, it checks whether there is a filtering value in the current state. If there is, it backtracks again. We call the resulting modification FC-RK.

The rest of the paper is organized as follows. Section 2 provides the necessary definitions and notations. Section 3 formulates the main theorem and introduces FC-RK. Section 4 briefly discusses the results of preliminary evaluation.

2 Preliminaries and Notations

A binary *constraint network* (CN) $Z = \langle V, D, C \rangle$ is a triple consisting of a set of *variables* V , a set of *domains* D and a set of *constraints* C . Let $V = \{v_1, \dots, v_n\}$. Then $D = \{D_{v_1}, \dots, D_{v_n}\}$, where D_{v_i} is the domain of values of variable v_i , $C = \{C_{v_i, v_j} \mid i \neq j, 1 \leq i, j \leq n\}$, where $C_{v_i, v_j} \subseteq D_{v_i} \times D_{v_j}$ is the set of all compatible pairs of values of v_i and v_j . We refer to the parts of Z as $V(Z)$, $D(Z)$, and $C(Z)$. To emphasize that a value val belongs to the domain of a variable v , we refer to this value as val^v . An *assignment* of a CN Z is a pair $\langle v_i, val \rangle$ such that $v_i \in V(Z)$, $val \in D_{v_i}$. A consistent set of assignments is a *partial solution* of Z . A partial solution that assigns all the variables of Z is a *solution*

of Z . A CN that has no solution is *insoluble*. Let P be a partial solution of Z . We denote the set of variables assigned by P by $V(P)$. A *nogood* of a CN Z is a partial solution of Z that cannot be extended to a full solution.

Throughout the paper we use the CN shown in Figure 1. The ellipses represent the domains of variables and incompatible pairs of values are connected by arcs. For example, value 2 of V_0 is incompatible with value 1 of V_{100} .

3 Responsibility sets, kernels, and FC-RK

In this section we develop a modification of FC called FC-RK. First we define two structures maintained by these algorithms.

Definition 1 Let P be a nogood of a CN Z . A *responsibility set* R of P is a subset of $V(Z) \setminus V(P)$ such that there is no consistent extension of P that assigns all the variables of R .

Definition 2 Let P be a nogood of a CN Z , let R be a responsibility set of P and let $\langle u, val \rangle \in P$. The *kernel* of val^u with respect to P and R is the subset of R which contains all the variables whose domains have values incompatible with val^u but compatible with the rest of assignments of P .

Consider the CN shown on Figure 1. The nogood $\{\langle V_0, 2 \rangle \langle V_1, 1 \rangle\}$ has a responsibility set $\{V_2, V_{100}, V_{200}\}$, while the kernel of $\langle V_1, 1 \rangle$ is $\{V_{200}\}$.

Now we present a modification of FC that maintains responsibility sets and kernels associated with removed values. The proposed method is a slight modification of the technique described in [Schiex and Verfaillie, 1994].

Recall that FC detects that the current partial solution P is a nogood if the current domain of some future variable is emp-

tied or if the current domain of the variable being assigned is wiped out [Prosser, 1993]. In both of these cases, FC discards the last assignment $\langle u, val \rangle$ of the current partial solution and removes val from the current domain of u . We say that P is the nogood associated with val^u . The modified version of FC associates val^u with two sets denoted by $rs(val^u)$ and $ker(val^u)$. These are the responsibility set and the kernel of the nogood associated with val^u . The additional operations performed by the modified version of FC are the following.

- The rs and ker sets are initialized to $\emptyset$ for all the values of the CN.
- If a value val_1^v is deleted during the lookahead stage performed after assigning val to u then v is inserted into $ker(val^u)$.
- Consider a value val^u that is deleted during backtrack and let v be the variable whose empty domain caused the backtrack. Then $rs(val^u)$ is set to $S \cup \{v\}$, where S is the union of the rs -sets of all values of v . Also $ker(val^u)$ is updated to the intersection of the current value of $ker(val^u)$ with $rs(val^u)$. (Note that the preliminary value of $ker(val^u)$ is computed at the lookahead stage in order to avoid recomputation of values that are incompatible with val^u .)
- Once a discarded value is restored to the current domain, the rs and ker sets of the value are re-initialized to $\emptyset$.

Let us illustrate the method by simulating a few iterations of the modified FC that processes the CN of Figure 1. It starts by assigning $\langle V_0, 1 \rangle$ $\langle V_1, 1 \rangle$ and then $\langle V_2, 1 \rangle$. The last assignment empties the current domain of V_{100} and FC backtracks. The rs -set of 1^{V_2} is set to $\{V_{100}\}$ so is $ker(1^{V_2})$. Note that the assignment $\langle V_2, 1 \rangle$ can remove values from many other variables of the CN but the kernel is restricted to $\{V_{100}\}$ during the backtrack. In the next step, FC assigns V_2 with 2 and backtracks again because V_{200} is emptied. Then the domain of V_2 is emptied therefore FC backtracks yet another time and produces a nogood $\{\langle V_0, 1 \rangle, \langle V_1, 1 \rangle\}$ with $rs(1^{V_1}) = \{V_2, V_{100}, V_{200}\}$ and $ker(1^{V_1}) = \emptyset$.

Consider a state of FC that occurs just after lookahead or backtrack. Denote the current partial solution by P and its last assignment by $\langle u, val \rangle$. For the given state, define a *filtering value*.

Definition 3 A value val_1^v removed by backtrack is a *filtering value* if the following conditions hold.

- **Compatibility condition.** The assignments of variables of $V(P) \cap ker(val_1^v)$ are compatible with val_1^v .
- **Inclusion condition.** For every unassigned $w \in ker(val_1^v)$, all values of the current domain of w are compatible with val_1^v .

Now we are ready to formulate the central theorem.

Theorem 1 Assume that in the given state there is a filtering value val_1^v . Then P is a nogood with a responsibility set $S = rs(val_1^v) \setminus V(P) \cup S^*$, where S^* is the union of rs -sets of all values that belong to the domains of unassigned variables of $ker(val_1^v)$.

Let us illustrate the theorem on the CN in Figure 1. Consider a state of a CSP solver in which the value 1^{V_0} is removed and associated with the rs -set $\{V_2, V_{100}, V_{200}\}$ and the ker -set $\{V_{100}, V_{200}\}$. Continuing the simulation, we assign V_0 with 2 and V_1 with 1. For the obtained partial solution, 1^{V_0} is a filtering value, all values of the current domains of V_{100} and V_{200} are compatible with 1^{V_0} (the inclusion condition).

Theorem 1 suggests a pruning procedure that discards the current partial solution if it finds a filtering value. The procedure is applied every time after lookahead or backtrack. Algorithm 1 presents the pseudocode of the procedure.

Algorithm 1 PRUNING PROCEDURE OF FC-RK

```

1: for every removed value  $val_1^v$  do
2:   if  $val_1^v$  is a filtering value then
3:     Let  $\langle u, val \rangle$  be the last assignment of the current
       partial solution
4:     Backtrack with setting  $rs(val^u)$  to  $S$  ( $S$  is defined
       in Theorem 1)
5:   end if
6: end for

```

We call the modification of FC that computes rs and ker sets and applies Algorithm 1 every time after lookahead and backtrack, FC-RK (RK abbreviates Responsibility sets and Kernels).

4 Preliminary Evaluation

The experimental evaluation shows that, FC-RK outperforms FC-CBJ on the majority of instances of graph k -coloring problem and on binary random CSPs with low density. For dense random CSPs, FC-RK performs more consistency checks than FC-CBJ but visits less nodes on the search tree. We believe that performance of FC-RK could be improved by an algorithm for checking the existence of filtering values, that takes less consistency checks than our current implementation.

References

- [Haralick and Elliott, 1980] R. M. Haralick and G.L. Elliott. Increasing tree search efficiency for constraint satisfaction problems. *Artificial Intelligence*, 14:263–313, 1980.
- [Prosser, 1993] P. Prosser. Hybrid algorithms for the constraint satisfaction problem. *Computational Intelligence*, 9:268–299, 1993.
- [Schiex and Verfaillie, 1994] T. Schiex and G. Verfaillie. Nogood Recording for Static and Dynamic Constraint Satisfaction Problem. *International Journal of Artificial Intelligence Tools*, 3(2):187–207, 1994.

Hypertree- decomposition via Branch- decomposition*

Marko Samer

Institute of Information Systems (DBAI), Vienna University of Technology, Austria
samer@dbai.tuwien.ac.at

Abstract

Hypertree-decomposition is the most general approach in the literature for identifying tractable computation problems encoded as hypergraphs. We show how the heuristic branch-decomposition approach for ordinary graphs of [Cook and Seymour, 2003] can be used for the heuristic construction of hypertree-decompositions.

1 Introduction

Many NP-complete problems (e.g., the constraint satisfaction problem CSP, the homomorphism problem HOM, the Boolean conjunctive query problem BCQ, etc.) can be described by *hypergraphs* in a natural way. It was shown by [Gottlob *et al.*, 2002] that in analogy to tree-width of graphs, hypertree-width of hypergraphs is an appropriate measure for the cyclicity and therefore the tractability of the corresponding computation problems. In particular, they have shown that NP-complete problems become polynomially solvable and even highly parallelizable when restricted to instances with bounded hypertree-width. Moreover, it was shown by [Gottlob *et al.*, 2000] that hypertree-decomposition and the corresponding measure of hypertree-width generalizes all other tractability measures for hypergraphs in the literature.

Recent research focuses on heuristic approaches for fast hypertree-decomposition with small width. In this presentation, we show how the heuristic approach of [Cook and Seymour, 2003] for branch-decomposition (of ordinary graphs) can be used for heuristic hypertree-decomposition.

2 Preliminaries

A *hypergraph* $\mathcal{H}$ is a tuple $(\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is a set of vertices (variables) and $\mathcal{E} \subseteq \wp(\mathcal{V}) \setminus \{\emptyset\}$ is a set of hyperedges. We define $\text{var}(\mathcal{H}) = \mathcal{V}$ and $\text{edges}(\mathcal{H}) = \mathcal{E}$. A *hypertree* for a hypergraph $\mathcal{H}$ is a triple $(\mathcal{T}, \chi, \lambda)$, where $\mathcal{T} = (\mathcal{V}, \mathcal{E})$ is a tree and $\chi : \mathcal{V} \rightarrow \text{var}(\mathcal{H})$ and $\lambda : \mathcal{V} \rightarrow \text{edges}(\mathcal{H})$ are labeling functions.

A *hypertree-decomposition* of a hypergraph $\mathcal{H}$ is a hypertree $(\mathcal{T}, \chi, \lambda)$ for $\mathcal{H}$ satisfying four conditions. Due to space restrictions, we refer the interested reader to [Gottlob *et al.*,

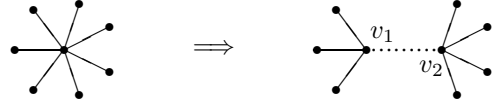


Figure 1: Splitting a vertex

2002] for the formal definition of a hypertree-decomposition. The *width* of a hypertree-decomposition $(\mathcal{T}, \chi, \lambda)$ is given by $\max_{p \in \text{vertices}(\mathcal{T})} |\lambda(p)|$, and the *hypertree-width* of a hypergraph is the minimum width over all its hypertree-decompositions.

A *branch-decomposition* of a hypergraph $\mathcal{H}$ is a tuple $(\mathcal{T}, \tau)$, where $\mathcal{T} = (\mathcal{V}, \mathcal{E})$ is a tree having $|\text{edges}(\mathcal{H})|$ leaves and in which every non-leaf vertex has degree three, and τ is a bijection from the set of leaves of $\mathcal{T}$ to $\text{edges}(\mathcal{H})$. The *order* of an edge $e \in \mathcal{E}$ is the number of vertices $v \in \text{var}(\mathcal{H})$ such that there are leaves l_1 and l_2 of $\mathcal{T}$ in different components of $\mathcal{T}$ when removing e with $v \in \tau(l_1) \cap \tau(l_2)$. The *width* of a branch-decomposition $(\mathcal{T}, \tau)$ is the maximum order of the edges of $\mathcal{T}$, and the *branch-width* of a hypergraph is the minimum width over all its branch-decompositions.

3 The Branch- decomposition Heuristic

In this section, we summarize the basic steps of the heuristic branch-decomposition approach of [Cook and Seymour, 2003]. Note that this approach was developed for ordinary graphs. Its starting point is the construction of the initial data structure which is a star as shown in Fig. 1. This is done in such a way that for each edge in the graph there exists exactly one edge in the star. This edge is then labeled with the set of vertices of the corresponding edge in the graph (i.e., each edge is labeled with exactly two vertices). Afterwards, the vertices of the star are successively split (cf. Fig. 1) according to the decomposition rules below. In each step, the two vertices v_1 and v_2 resulting from the splitting are connected by a new edge which is labeled by the intersection of the set of vertices labeling the edges incident with v_1 and the set of vertices labeling the edges incident with v_2 . This process stops when all non-leaf vertices have degree three; the resulting tree is then a *branch-decomposition* of the underlying graph.

The decomposition rules can be divided into two classes: (i) the application of “safe splits” and (ii) the application of the *Eigenvector heuristic* [Alon, 1986]. *Safe splits* divide the

*This work was supported by the Austrian Science Fund (FWF) project P17222-N04.

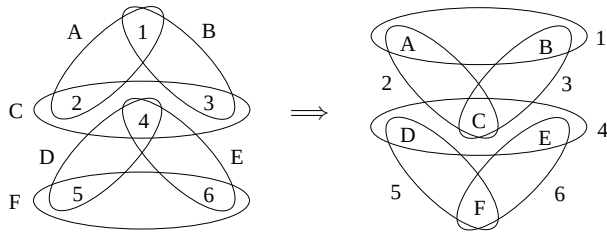


Figure 2: A hypergraph and its dual hypergraph

graph in such a way that it remains *extendible*, i.e., there is some way to (repeatedly) divide the subgraphs to obtain a branch-decomposition of width equal to the branch-width of the original graph. In other words, nothing bad can happen when applying safe splits. There are three kinds of safe splits presented by [Cook and Seymour, 2003]: (i) *pushing*, (ii) *2-separations*, and (iii) *3-separations*. At each stage, these safe splits are applied as long as possible. If none of them is applicable, the Eigenvector heuristic is used.

4 Application to Hypertree- decomposition

We will now show how the branch-decomposition heuristic described in Section 3 can be used for hypertree-decomposition. Recall that this heuristic was developed for ordinary graphs and not for hypergraphs. Thus, our first step is to show that the safe splits as well as the Eigenvector heuristic are also applicable to hypergraphs.

To this aim, recall that the new edge introduced at each vertex splitting (cf. Fig. 1) is labeled with the intersection of two sets of vertices as described above. In general, however, this intersection may contain more than two vertices. Thus, such newly introduced edges represent hyperedges, i.e., the data structure underlying the original branch-decomposition approach starts as an ordinary graph and evolves to a hypergraph during the decomposition process. This can be seen as an intuitive justification for the applicability of the branch-decomposition heuristic to hypergraphs. It is easy to verify that this intuition holds indeed true.

Now, note that given a tree-decomposition¹ of a hypergraph of width k , it is possible to construct a branch-decomposition of width at most $k + 1$, and given a branch-decomposition of a hypergraph of width k , it is possible to construct a tree-decomposition of width at most $3k/2$ [Robertson and Seymour, 1991]. Thus, having small tree-width is equivalent to having small branch-width.

Hence, a simple approach to obtain a hypertree-decomposition via branch-decomposition is to (i) construct a branch-decomposition of the given hypergraph by using the above heuristic, (ii) transform this branch-decomposition into a tree-decomposition, and, in analogy to [McMahan, 2004], (iii) apply set covering heuristics in order to obtain a hypertree-decomposition. In particular, set covering is used to obtain appropriate λ -labels based on the χ -labels given by the tree-decomposition.

¹Intuitively, a tree-decomposition is a hypertree-decomposition without the labeling function λ .

Another approach we investigate is dual to the above one in the sense that we obtain a hypertree where the λ -labels are given and appropriate χ -labels have to be set. To this aim, let us first introduce the *dual hypergraph* of a hypergraph as exemplified in Fig. 2. The dual hypergraph is simply obtained by swapping the roles of hyperedges and vertices. Our procedure is then to (i) construct a branch-decomposition of the *dual* hypergraph by using the above heuristic, (ii) transform this branch-decomposition into a tree-decomposition, and (iii) interpret the labels of this tree-decomposition as λ -labels of a hypertree and set the χ -labels appropriately in a straight-forward way. The resulting hypertree is then a hypertree-decomposition of the original hypergraph.

The attentive reader may have noticed that there are two problems with the latter approach: First, the branch-width and therefore also the hypertree-width is at least the cardinality of the largest edge in the dual hypergraph which is equal to the maximum number of hyperedges having a common vertex in the original hypergraph. Second, since the λ -labels satisfy the conditions of a tree-decomposition by construction, the hypertree-width may be larger than necessary. However, by introducing heuristic pre- and post-processing steps, we are able to overcome both problems.

5 Conclusion

We have tested our approach on hypergraphs representing adder and bridge circuits connected in a line. For all instances of such adder and bridge hypergraphs, we obtained a hypertree-decomposition of almost optimal width within a few seconds. Natural future work will be a full implementation of our approach, the investigation of further improvements, and a systematic comparison with other hypertree-decomposition heuristics. Moreover, an implementation of our second approach in such a way that the tree-decomposition of the dual hypergraph is directly constructed (e.g., by *bucket elimination*) without the intermediate-step of a branch-decomposition seems to be promising.

References

- [Alon, 1986] N. Alon. Eigenvalues and expanders. *Combinatorica*, 6(2):83–96, 1986.
- [Cook and Seymour, 2003] W. Cook and P. Seymour. Tour merging via branch-decomposition. *INFORMS Journal on Computing*, 15(3):233–248, 2003.
- [Gottlob *et al.*, 2000] G. Gottlob, N. Leone, and F. Scarcello. A comparison of structural CSP decomposition methods. *Artificial Intelligence*, 124(2):243–282, 2000.
- [Gottlob *et al.*, 2002] G. Gottlob, N. Leone, and F. Scarcello. Hypertree decompositions and tractable queries. *Journal of Computer and System Sciences*, 64(3):579–627, 2002.
- [McMahan, 2004] B. McMahan. Bucket elimination and hypertree decompositions. Implementation report, Institute of Information Systems (DBAI), TU Vienna, 2004.
- [Robertson and Seymour, 1991] N. Robertson and P. Seymour. Graph minors. X. Obstructions to tree-decomposition. *Journal of Combinatorial Theory, Series B*, 52:153–190, 1991.

Improving Tree Decomposition Methods With Function Filtering

Martí Sánchez⁽¹⁾, Javier Larrosa⁽²⁾ and Pedro Meseguer⁽¹⁾

⁽¹⁾IIIA-CSIC, Campus UAB, 08193 Bellaterra, Spain

⁽²⁾Dep. LSI, UPC, Jordi Girona 1-3, 08034 Barcelona, Spain

marti@iiia.csic.es, larrosa@lsi.upc.es, pedro@iiia.csic.es

Abstract

Tree decomposition can solve weighted CSP, but with a high spatial complexity. To improve its practical usage, we present function filtering, a strategy to decrease memory consumption. Function filtering detects and removes some tuples that appear to be consistent but that will become inconsistent when extended to other variables. We show empirically the benefits of our approach.

1 Introduction

A weighed CSP (WCSP) is defined as $\langle X, D, C, S(k) \rangle$ where X and D are variables and domains as in CSP. C is a finite set of constraints as cost functions; $f \in C$ relates some variables $var(f) = \{x_{i_1}, \dots, x_{i_r}\}$ called its scope, and assigns costs to tuples $t \in \prod_{x_i \in var(f)} D_i$ such that,

$$f(t) = \begin{cases} 0 & \text{if } t \text{ is allowed} \\ 1 \dots k-1 & \text{if } t \text{ is partially allowed} \\ k & \text{if } t \text{ is totally forbidden} \end{cases}$$

$S(k) = \langle [0, 1, \dots, k], \oplus, \geq \rangle$ is a valuation structure such that $a \oplus b = \min\{k, a + b\}$, $\top = k$, $\perp = 0$ [Larrosa, 2002]. We assume that the reader is familiar with assignments or value tuples t_S with scope S , complete tuples ($S = X$), projections over $S' \subset S$, $t_S[S']$, and concatenation of two tuples $t_S \cdot t'_T$, defined only if common variables coincide in their corresponding values. We assume that $f(t_S)$ (with $var(f) \subset S$) always means $f(t_S[var(f)])$. A complete tuple t_S is *consistent* if $\bigoplus_{f \in C} f(t) < k$, else t_S is *inconsistent*. A *solution* is a complete consistent assignment with minimum cost. Finding a solution is NP-hard. With $k = 1$ WCSP reduces to CSP.

We define two operations on functions. *Projecting out* a variable $x \in var(f)$ from f , denoted $f_{\downarrow x}$, is a new function with scope $var(f) - x$ defined as $f_{\downarrow x}(t) = \min_{a \in D_x} (f(a, t))$. *Summing* two functions f and g is a new function $f + g$ with scope $var(f) \cup var(g)$ and $\forall t \in \prod_{x_i \in var(f)} D_i$, $\forall t' \in \prod_{x_j \in var(g)} D_j$ such that $t \cdot t'$ is defined, $(f + g)(t \cdot t') = f(t) \oplus g(t')$. We say that function g is a *lower bound* of f , denoted $g \leq f$, if $var(g) \subseteq var(f)$ and for all possible tuples t of f , $g(t) \leq f(t)$. A set of functions G is a *lower bound* of f iff $(\sum_{g \in G} g) \leq f$. It is easy to check that for any f , $(f_{\downarrow x})$ is a lower bound of f , and $\sum_{f \in F} (f_{\downarrow x}) \leq (\sum_{f \in F} f)_{\downarrow x}$.

```

procedure CTE( $\langle X, D, C, k \rangle, \langle \langle V, E \rangle, \chi, \psi \rangle$ )
1 for each  $(u, v) \in E$  s.t. all  $m_{(i,u)}, i \neq v$  have arrived do
2    $B \leftarrow \psi(u) \cup \{m_{(i,u)} \mid (i, u) \in E, i \neq v\}$ ;
3    $m_{(u,v)} \leftarrow (\sum_{f \in B} f) \downarrow_{elim(u,v)}$ ;
4   send  $m_{(u,v)}$ ;

```

Figure 1: The CTE algorithm.

A *tree decomposition* of a WCSP $\langle X, D, C, S(k) \rangle$ is a triplet $\langle T, \chi, \psi \rangle$, where $T = \langle V, E \rangle$ is a tree, χ and ψ are labeling functions which associate with each vertex $v \in V$ two sets, $\chi(v) \subseteq X$ and $\psi(v) \subseteq C$ such that: (i) for each function $f \in C$, there is exactly one vertex $v \in V$ such that $f \in \psi(v)$ and $var(f) \subseteq \chi(v)$; (ii) for each variable $x \in X$, the set $\{v \in V \mid x \in \chi(v)\}$ induces a connected subtree of T . The *tree-width* of a tree decomposition is $tw = \max_{v \in V} |\chi(v)| - 1$. If $(u, v) \in E$ the *separator* is $sep(u, v) = \chi(u) \cap \chi(v)$, and the *eliminator* is $elim(u, v) = \chi(u) - sep(u, v)$ [Dechter, 2003].

Cluster-Tree Elimination (CTE) is an algorithm that solves WCSP by sending messages along tree decomposition edges [Dechter and Pearl, 1989]. Edge $(u, v) \in E$ has associated two CTE messages $m_{(u,v)}$, from u to v , and $m_{(v,u)}$, from v to u . $m_{(u,v)}$ is a function computed summing all functions in $\psi(v)$ with all incoming CTE messages except from $m_{(v,u)}$ and projecting out variables in $elim(u, v)$. $m_{(u,v)}$ has scope $sep(u, v)$. The CTE algorithm appears in Figure 1. Its complexity is time $O(d^{tw+1})$ and space $O(d^s)$, where d is the largest domain size and s is the maximum separator size.

Mini-Cluster-Tree Elimination (MCTE(r)) approximates CTE. If the number of variables in a cluster is high, it may be impossible to compute $m_{(u,v)}$ due to memory limitations. MCTE(r) computes a lower bound by limiting by r the arity of the functions sent in the messages. A MCTE(r) message, $M_{(u,v)}$, is a set of functions that approximate the corresponding CTE message $m_{(u,v)}$ ($M_{(u,v)} \leq m_{(u,v)}$). It is computed as $m_{(u,v)}$ but instead of summing all functions of set B , it computes a partition $P = \{B_1, B_2, \dots, B_p\}$ of B such that the sum of the functions in every B_i does not exceed arity r . The MCTE(r) algorithm, with time and space complexity $O(d^r)$, is obtained replacing line 3 of CTE by,

```

3.1  $P \leftarrow \text{partitioning}(B, r)$ ;
3.2  $M_{(u,v)} \leftarrow \{(\sum_{f \in B_i} f) \downarrow_{elim(u,v)} \mid B_i \in P\}$ ;

```

2 Function Filtering

A *nogood* is a tuple t that cannot be extended into a complete consistent assignment. Nogoods are useless for solution generation, so they can be eliminated as soon as are detected. We store function f as a set S_f containing all pairs $(t, f(t))$ with cost less than k . The *size* of f , denoted $|f|$, is the number of tuples with cost less than k . Computing $f \downarrow_x$ has time complexity $O(|f|)$. For summing $f + g$, we iterate over all the combinations $(t, f(t)) \in S_f$ and $(t', g(t')) \in S_g$ and, if they match, compute $(t \cdot t', f(t) \oplus g(t'))$, which has complexity $O(|f||g|)$. Efficiency of operations depends on function size. To anticipate the detection of nogoods, we propose *function filtering* on f from a set of functions H , noted $\bar{f}^H$, as

$$\bar{f}^H(t) = \begin{cases} f(t) & \text{if } (\bigoplus_{h \in H} h(t)) \oplus f(t) < k \\ k & \text{otherwise} \end{cases}$$

Tuples reaching the upper bound k are removed, which causes to reduce $|f|$ before operating with it. Therefore, let f (resp. g) be a function and F (resp. G) be a lower bound. When summing f and g , if previously filtered with the lower bound of the other function, the result is preserved,

$$\bar{f}^G + \bar{g}^F = f + g$$

and the sum is done with functions of smaller size, so it is presumably done more efficiently. If $f \notin H, g \notin H$,

$$\overline{f+g}^H = \overline{\bar{f}^H + \bar{g}^H}^H$$

Function filtering easily integrates into CTE. We define a *filtering tree-decomposition* $\langle T, \chi, \psi, \phi \rangle$, where $\phi(u, v)$ is a set of functions associated to edge $(u, v) \in E$ with scope included in $sep(u, v)$. $\phi(u, v)$ must be a lower bound of the corresponding $m_{(u,v)}$ (namely, $\phi(u, v) \leq m_{(u,v)}$). The new algorithms CTEf and MCTEf(r) use a filtering tree decomposition. They are equivalent to CTE and MCTE(r) except in that they use $\phi(u, v)$ for filtering functions before computing $m_{(u,v)}$ or $M_{(u,v)}$. For CTEf, we replace line 3 by,

$$3 \quad m_{(u,v)} \leftarrow \sum_{f \in B} \bar{f}^{\phi(u,v)} \downarrow_{elim(u,v)};$$

Similarly for MCTEf(r) we replace line 3 by two lines,

$$3.1 \quad P \leftarrow \overline{partitioning(B, r)}^{\phi(u,v)};$$

$$3.2 \quad M_{(u,v)} \leftarrow \{(\sum_{f \in B_i} f) \downarrow_{elim(u,v)} \mid B_i \in P\};$$

The effectiveness of the new algorithms will depend on the quality of the lower bound $\phi(u, v)$. It is worth noting that the algorithms will be correct as long as $\phi(u, v)$ is a true lower bound which can be computed using any technique. An option is to include in $\phi(u, v)$ all the original functions used to compute $m_{(u,v)}$ properly projected,

$$\phi(u, v) = \{f \downarrow_S \mid f \in \psi(w), w \in T(u, v), S = var(f) - \chi(u)\}$$

Our CTEf and MCTEf implementations use this lower bound. $T(u, v)$ denote the nodes of the tree decomposition reachable from node v after the elimination of edge (u, v) . Another option for CTEf is to include in $\phi(u, v)$ a message $M_{(v,u)}$ from a previous execution of MCTE(r). Applying

```

procedure IMCTE( $\langle X, D, C, k \rangle, \langle \langle V, E \rangle, \chi, \psi \rangle$ )
for each  $(u, v) \in E$  do  $\phi(u, v) := \{\emptyset\}; r := 1;$ 
repeat
  MCTEf( $r$ );  $r := r + 1;$ 
for each  $(u, v) \in E$  do  $\phi(u, v) := M_{(u,v)};$ 
until exact solution or exhausted resources

```

Figure 2: The IMCTE algorithm.

this idea to MCTEf, we obtain a recursive algorithm which naturally produces an elegant iterative approximating method called IMCTEf (Figure 2). It executes MCTEf(r) using as lower bounds $\phi(u, v)$ the messages $M_{(v,u)}^{r-1}$ computed by MCTEf($r-1$) which, recursively, uses the messages $M_{(v,u)}^{r-2}$ computed by MCTEf($r-2$), and so on.

State of the art CTE based algorithms assume to consume all d^s memory. Here we show that storing only consistent tuples and applying filtering techniques we can greatly reduce memory usage. Experiments focus on (i) showing that CTEf uses less memory than CTE to find the exact solution, and (ii) showing that MCTEf(r), exhausts resources at a smaller arity r and finds worst LB than the iterative version IMCTE. We have tested CTE, CTEf, MCTEf(r) and IMCTE on DIMACS dubois Max-Sat instances, Borchers Weighed Max-Sat instances and SPOT instances. When d^s is small CTE is feasible. When d^s is large, CTE is unfeasible and filtering is very useful as shown by the gain in one order of magnitude in the used memory (see the first Borchers and the firsts SPOT instances). In unfeasible instances for both CTE and CTEf, IMCTE finds better quality lower bounds than MCTEf(r).

References

- [Dechter and Pearl, 1989] R. Dechter and J. Pearl. Tree clustering for constraints networks. *Artificial Intelligence*, 38, 1989.
- [Dechter, 2003] R. Dechter. *Constraint Processing*. Elsevier, 2003.
- [Larrosa, 2002] J. Larrosa. Node and arc consistency in weighted csp. In *Proc. AAAI*, 2002.

	X	C	d	sep	CTE	CTEf	MCTEf(r)		IMCTE		UB
							r	LB	r	LB	
dubois100	75	200	2	3	3k	2k					1
wp2100	50	95	2	9	6k	1k					16
wp2150	50	138	2	15	302k	40k					34
wp2200	50	186	2	19	-	733k					69
wp2250	50	233	2	24	-	-	21	69	25	96	96
wp2300	50	261	2	26	-	-	21	76	21	132	132
wp2350	50	302	2	30	-	-	21	129	21	155	212
wp2400	50	340	2	30	-	-	20	70	20	137	212
wp2450	50	378	2	31	-	-	20	130	20	187	257
wp2500	50	418	2	34	-	-	20	168	20	251	318
spot54	67	271	4	11	754k	16k					37
spot29	82	462	4	14	-	63k					8059
spot503	143	635	4	8	-	26k					11113
spot505	240	2242	4	22	-	-	12	5020	12	14188	21254
spot42	190	1394	4	26	-	-	12	109000	12	123050	155051

Table 1: Columns are: instance, number of variables, number of constraints, maximum domain size, maximum separator size, tuples consumed by CTE algorithm, tuples consumed by CTEf algorithm (- denotes exhausted memory), arity r reached by MCTEf(r), LB computed by MCTEf(r), arity r reached by IMCTE, LB computed by IMCTE (before resources exhausted), optimal UB of the problem.

Streamlining Local Search for Spatially Balanced Latin Squares

Casey Smith, Carla Gomes, and Cesar Fernandez

Computer Science Department

Cornell University

{casey,gomes}@cs.cornell.edu, cesar@eup.udl.es

Abstract

Streamlined constrained reasoning powerfully boosts the performance of backtrack search methods for finding hard combinatorial objects. We use so-called spatially balanced Latin squares to show how streamlining can also be very effective for local search: Our approach is much faster and generates considerably larger spatially balanced Latin squares than previously reported approaches (up to order 35; the previous best results could only generate solutions up to order 18). We also provide a detailed characterization of our streamliner and solution topology for small orders. We believe that streamlined local search is a general technique suitable for solving a wide range of hard combinatorial design problems.

1 Introduction

The idea underlying streamlining constraints [Gomes and Sellmann, 2004] is simple but powerful: In order to increase the effectiveness of propagation, “streamlining constraints” partition the solution space into different portions with different global properties. Since in general the streamlined constraints are not implied by the original problem constraints, the resulting search and solution spaces are subsets of the original spaces. Gomes and Sellmann showed the effectiveness of streamlined constraints for solving hard combinatorial design problems, in particular spatially balanced Latin squares (SBLS, defined in Section 2). Using streamlined backtrack search methods, they solved considerably larger SBLS problem instances than with local search: up to order 18, while a local search method applied to the original problem could only generate solutions up order 9.

This paper shows how streamlining can also be effective for local search. We note that while the addition of streamlining constraints is straightforward in backtrack search methods, it is not always the case for local search methods, since constraints are in general enforced via the objective function. The key is to consider streamlining constraints that lead to an alternate neighborhood representation. We applied streamlined local search to the problem of generating spatially balanced Latin squares. Our streamliner defines a basic swap neighborhood as a column permutation of the input Latin

square. This provides additional interesting structural properties, making our local search procedure scale up considerably better than previously reported approaches. We solve instances up through order 35; the previous best results could only generate solutions up to order 18. Moreover, our streamliner provides useful intuitions on a construction for spatially balanced Latin squares. We believe that streamlined local search is a general technique suitable for finding objects with intricate combinatorial constraints.

2 Spatially Balanced Latin Squares

A Latin Square of order n is an n by n grid where each of the n^2 cells in the grid is assigned one of n symbols such that each symbol appears exactly once in each column and each row. Creating a Latin square is by no means difficult. We can define a *cyclic construction* as an order n Latin square where the element in row i and column j has the value $(i + j) \bmod n$ (see Figure 1). The first row will have all the symbols in order: $[1\ 2\ 3\ \dots\ n]$. Each subsequent row is the same as the previous row with all the values shifted one location to the left, wrapping the first element to the end. Thus, the second row will be $[2\ 3\ \dots\ n\ 1]$; the third will be $[3\ \dots\ n\ 1\ 2]$; and so on. Note that the columns follow the same pattern: the first column is $[1\ 2\ 3\ \dots\ n]^T$; the second is $[2\ 3\ \dots\ n\ 1]^T$; and the third is $[3\ \dots\ n\ 1\ 2]^T$.

One challenge comes in generating *spatially balanced* Latin squares (SBLS) where the sum of the distances between two symbols across all rows is constant for all pairs of symbols. First, we define $d_i(j, k)$ to be the distance between symbols j and k in row i . This distance is calculated as the absolute difference of the column indices of where symbols j and k appear in row i . For example, if row i were $[3\ 5\ 1\ 2\ 4]$, then $d_i(5, 2)$ would be $|2 - 4| = 2$. Then we can define the total distance $d(j, k)$ to be

$$d(j, k) = \sum_{i=1}^n d_i(j, k). \quad (1)$$

A SBLS requires that all pairs of symbols have the same total distance:

$$\forall i \neq j, k \neq l : d(i, j) = d(k, l). \quad (2)$$

SBLS are useful for agronomic field experiment design [van Es and van Es, 1993] and are compelling constraint satisfaction problems because of their high degree of structure. As

1	2	3	4	5
2	3	4	5	1
3	4	5	1	2
4	5	1	2	3
5	1	2	3	4

(A)

1	4	5	3	2
2	5	1	4	3
3	1	2	5	4
4	2	3	1	5
5	3	4	2	1

(B)

Figure 1: (A) A cyclic Latin square (order 5). (B) A SBLS where $\forall j, k : d(j, k) = 10$.

shown in [Gomes *et al.*, 2004], $d(j, k) = \frac{n(n+1)}{3}$ for all $j \neq k$ in any SBLS, implying that no SBLS exist for orders n such that $n \bmod 3 = 1$.

3 Streamlined Local Search

Gomes and Sellmann showed that a standard local search approach performs remarkably poorly at finding SBLS. We hypothesize that this is due to the enormity of the search space and the global nature of the constraints. The search space is so big and the solutions are so few that the odds of local search finding a solution are very slim for all but the smallest orders. If we can cleverly restrict the search space to a small subspace which contains a higher density of solutions, we can increase the odds that local search will find a solution.

Consider the subspace resulting from starting with a Latin square and restricting the search space to be a column order permutation of the original square. That is, take a Latin square and swap entire intact columns to permute the order in which the columns appear. Since this reordering of the columns will never make the square non-Latin (all symbols will appear in all rows and in all columns exactly once), we can try permuting the column order to make the Latin square spatially balanced. Note that the space of column permutations of a particular Latin square is vanishingly small compared to the entire space of Latin squares. The goal is to find an initial Latin square such that this small subspace is dense with SBLS. Cyclically constructed Latin squares empirically provide such a subspace.

If the initial Latin square is generated by the cyclic construction, some structural properties can be exploited to speed the search. First, observe that there will be at most $\lfloor \frac{n}{2} \rfloor$ unique values of the total distance among the $\frac{n(n-1)}{2}$ pairs of symbols. To see this, define

$$d_{mod}(j, k) = \min(|j - k|, |j - k + n|, |j - k - n|). \quad (3)$$

Thus, $d_{mod}(j, k)$ is the distance between symbols j and k in symbol space, modulo the number of symbols (n). For example, for an order 6 Latin square, $d_{mod}(2, 4) = 2$ because the shortest path from 2 to 4 is by counting 2 3 4, revealing a distance of 2. Similarly, $d_{mod}(1, 5) = 2$ because the shortest path from 1 to 5 is by counting 5 6 1, revealing a distance of 2. Examining the cyclic construction reveals that pairs of symbols with identical values of $d_{mod}(j, k)$ will have identical values of $d(j, k)$, even if the column order is permuted. Thus since there are $\lfloor \frac{n}{2} \rfloor$ different nonzero values of $d_{mod}(j, k)$, there are at most $\lfloor \frac{n}{2} \rfloor$ different total distances between pairs of symbols for any column order permutation of the cyclic

A	B	C	D	E
1	2	3	4	5
2	3	4	5	1
3	4	5	1	2
4	5	1	2	3
5	1	2	3	4

(A)

A	E	C	D	B
1	5	3	4	2
2	1	4	5	3
3	2	5	1	4
4	3	1	2	5
5	4	2	3	1

(B)

Col	A	B	C	D	E
A	-	X	Y	Y	X
B	-	-	X	Y	Y
C	-	-	-	X	Y
D	-	-	-	-	X
E	-	-	-	-	-

(C)

Figure 2: (A) A cyclic Latin square (order 5). (B) A column permutation of the cyclic Latin square. Note that no matter how the columns are reordered,

$$d(1, 2) = d(2, 3) = d(3, 4) = d(4, 5) = d(5, 1) = |A - B| + |B - C| + |C - D| + |D - E| + |A - E|$$

$$d(1, 3) = d(2, 4) = d(3, 5) = d(4, 1) = d(5, 2) = |A - C| + |B - D| + |C - E| + |D - A| + |E - B|$$

(C) A matrix showing how the intra-column distances contribute to the pair distances. The 10 pairs of symbols correspond to group X ((1,2), (2,3), (3,4), (4,5), (5,1)) and group Y ((1,3), (2,4), (3,5), (4,1), (5,2)).

construction. As an example, if we check the total distance between symbols 1 and 2, we do not need to check the distance between any other consecutive symbols, i.e., 2 and 3, 3 and 4, and so on because they will all have identical values (see Figure 2). Next, note that since the columns always remain intact no matter how their order is permuted, we need only store the first row as we search for SBLS. From the first row, we can calculate the $d(j, k)$ corresponding to each of the $\lfloor \frac{n}{2} \rfloor$ possibly distinct $d(j, k)$. We also observe that we can fix the first column to be headed by symbol 1 without decreasing our probability of finding a solution: there will be an equal number of solutions with 1 heading the first column as with any other number.

These observations naturally lead to a gradient descent algorithm. First, we define the *imbalance* of a square to be

$$\text{imbalance} = \sum_{i < j} \left| d(i, j) - \frac{n(n+1)}{3} \right|. \quad (4)$$

That is, the imbalance is the sum over all pairs of how far from ideal spacing the total spacing of the pair is. Starting with a random permutation of the first row, we swap the location of two symbols such that it decreases the imbalance of the square. When we reach a local minimum, we randomize and try again.

Algorithm 1 SBLS Streamlined Local Search

- 1: Generate a Latin Square of order n by the cyclic construction
 - 2: Randomly permute the order of the columns
 - 3: Select a column and try swapping it with every other column
 - 4: Keep the swap which minimizes the imbalance
 - 5: If stuck, then randomize the order of the columns
 - 6: Repeat from step 3
-

Finally, we note that we can generate solutions which are symmetric across the main diagonal. Since spatial balance is defined as the sum of distances within rows, swapping intact

order	6	8	9	11	12	14	15	17	18	20	21	23	24	26	27	29	30	32-35
CP	0.06	16	241															
SS	0.05	0.88	0.91	9.84	531	5K												
CS	0.02				14.4				107K									
CCCP	4e-5	3e-4	1e-4	1e-3	1e-3	0.02	0.01	0.25	2.3	16	16	104	281	609	4K	43k	≈160K	≈1.2M

Table 1: Solution times are given in seconds. CP, SS, and CS are the times to find the first solution via standard constraint programming, symmetric streamlining, and composition streamlining as reported by Gomes and Sellmann [2004] as run on a 550 MHz Pentium III machine. CCCP is the cyclic construction column permutation method presented here. Times given are the mean time to find a solution averaged across several independent runs on a 1.0 GHz Pentium III machine. Note that although our processor was only twice as fast, our performance is orders of magnitude better. Note that the largest solution from previous methods was only order 18, while our method can find solutions up to order 35. Additionally, no previous method had found solutions for orders 15 and 17.

1 2 3 4 5 6	1 3 4 2 6 5	1 3 4 2 6 5
2 3 4 5 6 1	2 4 5 3 1 6	3 5 6 4 2 1
3 4 5 6 1 2	3 5 6 4 2 1	4 6 1 5 3 2
4 5 6 1 2 3	4 6 1 5 3 2	2 4 5 3 1 6
5 6 1 2 3 4	5 1 2 6 4 3	6 2 3 1 5 4
6 1 2 3 4 5	6 2 3 1 5 4	5 1 2 6 4 3
(A)	(B)	(C)

Figure 3: (A) A cyclic Latin square (order 6). (B) The same cyclic Latin square with the columns permuted to form a SBLS. (C) The same SBLS as in (B), but with the rows reordered the same as the columns to make a symmetric SBLS.

rows will not affect the balancing. If we reorder the rows in the same way as the columns, the solution will become symmetric across the main diagonal and spatially balanced in terms of both the intra-row and intra-column distances (see Figure 3).

4 Empirical Results

The best performance on finding SBLS was reported by Gomes and Sellmann [2004], so we compare our results to theirs. Our streamlining is much simpler than theirs and performs dramatically better. They could only generate solutions up to order 18 (and were unable to find solutions for orders 15 and 17). We have generated SBLS for all possible orders up through 35. Times in seconds are given in Table 1.

We also tried using our streamlining with complete back-track search. While it improves performance over previous methods, the local search method presented here performs considerably better. In addition, we tried composing solutions from smaller cyclic Latin squares using local search, but the performance was slightly worse than the simple streamliner presented here.

Although we have only discussed permuting the columns of the cyclic construction, in principle, we could permute the columns of any Latin square in an attempt to find SBLS. However, most arbitrary random Latin squares (such as those generated by Jacobson and Matthews [1996]) *cannot* be permuted to be spatially balanced. This observation is remarkable, given that cyclic constructions are much more imbalanced than an average random Latin square. For instance, the imbalance of an order 30 cyclic Latin square is 46920 while the imbalance of a typical order 30 random Latin square is

only around 12000.

Through exhaustive enumeration, we analyzed the solution space of SBLS for small orders. Interestingly, we found that for orders 6 and 8, all the SBLS can be generated by row permuting and/or one-to-one symbol remapping (that is change all instances of symbol i to symbol j , symbol k to symbol l , and so on) the SBLS obtained as column permutations of the cyclic construction. However, for higher orders, there exist some solutions which cannot be derived in that way. We are currently studying the topology of the solution space for higher orders trying to identify relationships among solutions. We are also studying the matrix of distances (as in Figure 2). We believe that the characterization of solution relationships and the distance matrix provide valuable clues for the design of a construction for generating SBLS of arbitrary size.

5 Conclusions

We describe a competitive streamlined local search approach for generating SBLS. We believe that streamlined local search is a general technique, effective for finding objects with intricate combinatorial properties. We hope that this work will inspire other researchers interested in solving hard combinatorial design problems.

References

- [Gomes and Sellmann, 2004] C. Gomes and M. Sellmann. Streamlined constraint reasoning. In *CP 2004*, 2004.
- [Gomes *et al.*, 2004] C. Gomes, M. Sellmann, C. van Es, and H. van Es. The challenge of generating spatially balanced scientific experiment designs. In *CPAIOR 2004*, 2004.
- [Jacobson and Matthews, 1996] Mark Jacobson and Peter Matthews. Generating uniformly distributed random latin squares. *J. of Combinatorial Designs*, 4(6):405–437, 1996.
- [van Es and van Es, 1993] Harold van Es and Cindy van Es. The spatial nature of randomization and its effects on the outcome of field experiments. *Agronomy Journal*, 85:420–428, 1993.

POSTER PAPERS

KNOWLEDGE REPRESENTATION AND REASONING

Abduction with Hypotheses Confirmation*

Marco Alberti¹, Marco Gavanelli¹, Evelina Lamma¹, Paola Mello², and Paolo Torroni²

¹ ENDIF - University of Ferrara - Via Saragat, 1 - 44100 Ferrara, Italy.

{malberti|m gavanelli|elamma}@ing.unife.it

² DEIS - University of Bologna - Viale Risorgimento, 2 - 40136 Bologna, Italy.

{pmello|ptorroni}@deis.unibo.it

1 Introduction

Abductive Logic Programming (ALP) is an extension of Logic Programming to formalise hypothetical reasoning. It typically distinguishes between facts, defined within a static theory and known to be true or false, and potentially true, abducible atoms (hypotheses).

However, in most applications, this distinction is not adequate to capture the dynamics of knowledge, as reasoning is confronted with an environment in evolution. It may turn out that some hypotheses gain or lose strength, as events happen in the world. For example, in the medical domain, where a kind of hypothetical reasoning is the diagnostic inference, we can model symptoms as observations and diseases as (not directly measurable) hypotheses: test results and new possibly upcoming symptoms, instead, are none of these, and should be interpreted as hypotheses confirmable by events.

An abductive derivation verifies a goal by using deduction as in logic programming, but also by possibly assuming that some abducibles are true. In order to have this process converge to a meaningful explanation, an abductive theory normally comes together with a set of *integrity constraints* IC , and it is required that hypotheses be consistent with IC .

In our extended framework, we distinguish between two classes of abducible literals: *hypotheses*, as classically understood, and *expectations* about events. Expectations can be “positive” (to be confirmed by certain events occurring), or “negative” (to be confirmed by certain events not occurring).

We propose a new language to define abductive logic programs with expectations, inspired to the IFF proof procedure [Fung and Kowalski, 1997], and whose semantics extends those of classical ALP. The language permits to express abducible hypotheses and expectations with *variables* and *constraints*. Within this new framework we can model and reason about a number of concrete application scenarios. Our framework permits, e.g., to reason about deadlines, and to express and correctly handle expectations with universal quantification: this typically happens with negative expectations (“The patient is expected *not* to show symptom Q at all times”).

*This work is partially funded by the Information Society Technologies programme of the European Commission under the IST-2001-32530 SOCS Project, and by the MIUR COFIN 2003 projects *Sviluppo e verifica di sistemi multiagente basati sulla logica*, and *La Gestione e la negoziazione automatica dei diritti sulle opere dell'ingegno digitali: aspetti giuridici e informatici*.

The operational semantics is an abductive proof-procedure which builds on the IFF and is proven sound for a relevant class of programs. It has been implemented using *Constraint Handling Rules* and integrated in a Java-based system for hypothetical reasoning [Alberti *et al.*, 2005].

2 Dynamic ALP

An ALP [Kakas *et al.*, 1992] is a tuple $\langle KB, IC, \mathcal{A} \rangle$ where KB is a logic program, $\mathcal{A}$ is a set of predicates called *abducibles* not defined in KB , and IC is a set of formulae called *Integrity Constraints*. An abductive explanation for a goal G is a set $\Delta \subseteq \mathcal{A}$ s.t. $KB \cup \Delta \models G$ and $KB \cup \Delta \models IC$.

New dynamically upcoming events are encoded into atoms $\mathbf{H}(\text{Descr}[Time])$ where Descr is a ground term representing the event and $Time$ is an integer representing the time at which the event happened. Such events are recorded into a set (called history, or **HAP**) containing **H** atoms. A Dynamic Abductive Logic Program (DALP) is a sequence of ALPs, each grounded on a given history. We will write $DALP_{\mathbf{HAP}}$ to indicate the abductive logic program obtained by grounding the DALP with the history **HAP**. The history dynamically grows during the computation, as new events happen.¹

An instance $DALP_{\mathbf{HAP}}$ of this framework can be queried with a goal G , that may contain both predicates defined in KB and abducibles. The abductive computation produces a set Δ of hypotheses, partitioned in two sets: general hypotheses (ΔA) and *expectations* (**EXP**), containing *positive* expectations (in the form of $\mathbf{E}(\text{Descr}[Time])$ atoms), and *negative* expectations ($\mathbf{EN}((\text{Descr}[Time]))$ atoms).

Typically, expectations will contain variables, over which CLP constraints can be imposed. Variable quantification is *existential* in **E** expectations, and *universal* in **EN** expectations (unless the same variable is used outside of such expectation). Explicit negation can also be applied to expectations.² Constraints on universally quantified variables will be considered as *quantifier restrictions*. For instance, $\mathbf{EN}(p(X))$, $X > 0$ has the semantics $\forall_{X>0} \mathbf{EN}(p(X))$.

The declarative semantics for $DALP_{\mathbf{HAP}}$ is based on its ground version, and considers CLP-like constraints as defined

¹The source of events is not modelled, but can be imagined as a queue.

²For each abducible predicate $A \in \{\mathbf{E}, \mathbf{EN}\}$, the abducible predicate $\neg A$ is implicitly defined, to represent the negation of A , together with the integrity constraint $(\forall X) \neg A(X), A(X) \rightarrow \text{false}$.

predicates. First, an abductive explanation should entail the goal and satisfy the integrity constraints:

$$Comp(KB \cup \Delta A \cup \mathbf{EXP} \cup \mathbf{HAP}) \models G \quad (1)$$

$$Comp(KB \cup \Delta A \cup \mathbf{EXP} \cup \mathbf{HAP}) \models IC \quad (2)$$

where, as in the IFF proof procedure, the symbol $\models$ stands for three valued entailment and $Comp$ stands for completion.

Among the sets of expectations of an instance $DALP_{\mathbf{HAP}}$, we select the ones that are consistent with respect to expectations (i.e., the same event should not be both expected to happen and not to happen), and that are *confirmed*:

Definition 1 A set $\mathbf{EXP}$ is E-consistent iff for each (ground) term p : $\{\mathbf{E}(p), \mathbf{EN}(p)\} \not\subseteq \mathbf{EXP}$. (3)

Given a history $\mathbf{HAP}$, a set of expectations $\mathbf{EXP}$ is confirmed if and only if for each (ground) term p :

$$Comp(\mathbf{HAP} \cup \mathbf{EXP}) \cup \{\mathbf{E}(p) \rightarrow \mathbf{H}(p)\} \cup \{\mathbf{EN}(p) \rightarrow \neg \mathbf{H}(p)\} \not\models false \quad (4)$$

We write $DALP_{\mathbf{HAP}} \models_{\Delta A \cup \mathbf{EXP}} G$ if equations (1-4) hold.

The operational semantics is an extension of the IFF. Each state is defined by a tuple defining confirmed, disconfirmed, and pending expectations, along with the resolvent, the set of abduced literals that are not expectations, the constraint store, a set of *partially solved integrity constraints*, and $\mathbf{HAP}$.

A derivation D is a sequence of nodes T_j , where the initial node T_0 contains the goal G as the initial resolvent, and the other nodes $T_j, j > 0$, are obtained by applying one among a set of transitions, until quiescence.

Definition 2 Starting with an instance $DALP_{\mathbf{HAP}^i}$ there exists a successful derivation for a goal G iff the proof tree with root node T_0 has at least one consistent leaf node T_n (i.e., there exists for T_n a ground variable assignment such that all the constraints are satisfied). In that case, we write:

$$DALP_{\mathbf{HAP}^i} \vdash_{\Delta A \cup \mathbf{EXP}}^{\mathbf{HAP}^f} G$$

The transitions are those of the IFF, enlarged with those of CLP, and with specific transitions accommodating the concepts of hypotheses confirmation and evolving history.

The CLP(FD) solver has been extended for dealing with universally quantified variables and *quantifier restrictions*. For instance, given two expectations $\forall_{X < 10} \mathbf{EN}(p(X))$ and $\exists_{Y > 5} \mathbf{E}(p(Y))$, the solver is able to infer $\exists_{Y \geq 10} \mathbf{E}(p(Y))$. To the best of our knowledge, this is the only proof-procedure able to abduce atoms containing universally quantified variables; moreover, it also handles constraints *à la* CLP on universally quantified variables.

We proved soundness for *allowed* DALPs:³

Theorem 1 Given $DALP_{\mathbf{HAP}^i}$ and a ground goal G , if $DALP_{\mathbf{HAP}^i} \vdash_{\Delta A \cup \mathbf{EXP}}^{\mathbf{HAP}^f} G$ then $ALP_{\mathbf{HAP}^i} \models_{\Delta A} G$.

3 Case study

We conclude with an example from the medical domain to show two main contributions of our work: the dynamic detection of new facts, and the confirmation of hypotheses by events. Suppose a symptom s can be caused by one of three types of diseases. Let disease d_1 be an acceptable explanation for s if the patient is not also affected by d_3 , and in such a case

³The proof is based on a correspondence drawn between SCIFF and IFF transitions, and exploits the soundness results of the IFF.

the patient's temperature cannot go below 37°C. s may alternatively be caused by disease d_2 , and in this case red spots are expected to appear on the patient's skin within 4 days. Finally, d_3 may be the cause of s , provided that an exam r gives a positive result:

$$symptom(s, T_1) \leftarrow \mathbf{disease}(d_1, T_1) \wedge \mathbf{not} \mathbf{disease}(d_3, T_1) \wedge \mathbf{EN}(tem(T), T_1) \wedge T < 37.$$

$$symptom(s, T_1) \leftarrow \mathbf{disease}(d_2, T_1) \wedge \mathbf{E}(red_spots, T_2) \wedge T_1 < T_2 \leq T_1 + 4.$$

$$symptom(s, T_1) \leftarrow \mathbf{disease}(d_3, T_1) \wedge \mathbf{E}(exam(r, +), T_1).$$

The initial goal can be the observation $symptom(s, 1)$. We model $\mathbf{disease}$ as a classical abducible, whereas *expectations* are used to corroborate the explanations.

Notice the twofold use of expectations: both in the second and third clause, the expectation defines a further event that can support the diagnosis. But while $\mathbf{E}(red_spots, T_2)$ simply defines the expected course of illness (in order for the diagnosis to be corroborated), $\mathbf{E}(exam(r, +), T_1)$ can also be intended as a *suggestion* to the physician for a further exam to be done, or as a *request* of further information.

The combinations of abducible literals can be refined by means of ICs. For example, if the result of some exam r is positive, then we can assume that the patient is not affected by disease d_1 : $\mathbf{H}(exam(r, +), T_1) \rightarrow \mathbf{not} \mathbf{disease}(d_1, T_1)$. The dynamic occurrence of events can drive the generation and selection of abductive explanations. If the query is, e.g., $symptom(s, 1)$, there can be three alternative explanations:

$$\{\mathbf{disease}(d_1, 1), \forall_{T > 37} \mathbf{EN}(tem(T), 1)\}, \\ \{\mathbf{disease}(d_2, 1), \exists_{1 < T_2 \leq 5} \mathbf{E}(red_spots, T_2)\}, \text{ and} \\ \{\mathbf{disease}(d_3, 1), \mathbf{E}(exam(r, +), 1)\}.$$

If event $\mathbf{H}(tem(36), 1)$ happens, the first set contains a disconfirmed expectation: $\forall_{T > 37} \mathbf{EN}(tem(T), 1)$, so it can be ruled out. If, within the deadline $T_2 \leq 5$, the event red_spots does not happen, the second set is excluded as well, and only the third remains acceptable.

Finally, integrity constraints could suggest possible cures, or warn about consequences of not taking certain drugs:

$$\mathbf{disease}(d_3, T_1) \rightarrow \mathbf{E}(aspirin, T_1) \\ \vee \mathbf{E}(tem(T), T_2) \wedge T > 40 \wedge T_2 < T_1 + 2.$$

4 Conclusions

We presented SCIFF, an abductive proof-procedure able, beside proposing explanations, to infer *expectations* about the happening of events. Expectations are abducibles, but more expressive: they can contain universally quantified variables, possibly with CLP constraints. They can represent requests for information, or the expected evolution of a system. SCIFF is able to process dynamically incoming events to confirm the expectations, providing corroboration to abduced hypotheses.

References

- [Alberti et al., 2005] M. Alberti, F. Chesani, M. Gavanelli, E. Lamma, P. Mello, and P. Torroni. Compliance verification of agent interaction: a logic-based software tool. *Applied Artificial Intelligence*, 2005. To appear.
- [Fung and Kowalski, 1997] T. H. Fung and R. A. Kowalski. The IFF proof procedure for abductive logic programming. *Journal of Logic Programming*, 33(2):151–165, November 1997.
- [Kakas et al., 1992] A.C. Kakas, R.A. Kowalski, and F. Toni. Abductive logic programming. *J. Log. Comput.*, 2(6):719–770, 1992.

A language for functional interpretation of model based simulation

Jonathan Bell, Neal Snooke and Chris Price

Department of Computer Science, University of Wales Aberystwyth
Penglais, Aberystwyth, Ceredigion, SY23 3DB, U. K.
jpb, nns, cjp@aber.ac.uk

Abstract

Functional modeling is in use for the interpretation of the results of model based simulation of engineered systems for design analysis, enabling the automatic generation of a textual design analysis report that expresses the results of the simulation in terms of the system's purpose. We present a novel functional description language that increases the expressiveness of this approach, allowing a system function to be decomposed in terms of subsidiary functions as well as required effects, increasing the range both of systems and design analysis tasks for which the approach can be used.

1 Introduction

The automation of the design analysis of engineered systems requires both simulation of the behavior of the system and interpretation of the results, to enable the automatic generation of a draft report outlining the results of the design analysis. This report is cast in terms of the system's purpose and as system function can be regarded as relating its behavior to purpose, this interpretation of simulation for automatic report generation is a useful rôle of functional description.

Functional modeling has been used for deriving the behavior of a system from knowledge of its structure and component functions [Sticklen and Chandrasekaran, 1989], supporting design by functional refinement [Iwasaki *et al.*, 1993], and interpretation of simulation for automatic report generation [Price, 1998] where system functions are associated with significant behaviors (outputs or goal states). Here we present a novel language for the description of system function that increases the expressive power of that approach by allowing partial fulfilment of a system's purpose to be described. Novel features of the language also increase both the range of systems that can be described and the range of tasks for which the language is useful.

2 Describing and decomposing function

The definition of function that underlies the language presented here might be given informally as "how a device achieves its purpose" or more formally as

Function: An object *O* has a function *F* if it achieves an intended goal by virtue of some external trigger *T* resulting in the achievement of an external effect *E*.

This is distinct from notions of behavior and purpose, which is a problem with some definitions of function used in model based reasoning. It follows that a representation of function has three elements; a description of purpose and the function's trigger and effect. The trigger and effect are Boolean expressions and act as the recognizer of achievement of the function. A function can fail in two ways. Either the trigger can fail to result in the expected effect or the effect can occur without the trigger, as summarized in Table 1. The four

Trigger	Effect	Function
false	false	Inoperative
true	false	Failed
false	true	Unexpected
true	true	Achieved

Table 1: Achievement of function using trigger and effect.

states of a function are defined in terms of the truth of the trigger and effect, so if a function is triggered but the effect is not present it is said to be failed, for example. Agreement in value between the trigger and effect is consistent with correct behavior of the system. We cannot, however, pair off these cases as the consequences of the two inconsistent behaviors will differ and the resulting report must reflect this.

The description of purpose is separate from the functional description itself. This is consistent with the idea that function is concerned with how a purpose is fulfilled rather than the purpose itself and also encourages model reuse as similar purposes might be fulfilled by systems with different functional requirements, such as cars' and motorbikes' lighting systems.

A function might depend on more than one trigger and effect so logical operators can be used to describe the required combinations. Alternatively, a function can be composed of subfunctions, each with its own trigger, effect and purpose. This distinction allows cases where the achievement of one subfunction mitigates failure of the top level function, such as where a warning system has visual and audible signals and the presence of either one means at least some warning is given.

Decomposition in terms of functions means that the top level function's states must be derived from the four possible states of each of the subfunctions. This is shown in Table 2. The

Child 1	Child 2	AND	OR	XOR
I	I	I	I	I
I	A	I	A	A
I	Fa	(I)	Fa	Fa
I	U	(I)	U	U
A	A	A	A	I
A	Fa	Fa	(A)	(U)
A	U	U	(A)	Fa
Fa	Fa	Fa	Fa	(I)
Fa	U	(I)	(A)	(A)
U	U	U	U	(I)

Table 2: States of functions and sub-functions.

rule is that triggering of the top level function is derived from the triggering of the children and achievement of its effect from those of the children. So, for example, in the sixth line of the table, where Child 1 is achieved (both trigger and effect are true) and Child 2 failed (trigger true but effect false) a top level function that depends on Child 1 AND Child 2 has failed as it is triggered (Child 1 trigger AND Child 2 trigger is true) but the effect is not achieved as Child 1 effect AND Child 2 effect resolves to false. Where the state of the function is in brackets, the report will ignore the top level function and include the associated failure of the child function. In addition to the logical operators, a functional description might use the sequential operators described in [Bell and Snooke, 2004] to allow systems whose function depends on intermittent or sequential behaviour to be described.

As there are three elements in a complete functional representation, there are three possible pairs that can be used as incomplete subfunctions to simplify the functional decomposition. As pairing trigger and purpose (with no effect) is unnecessary, this leaves two classes of incomplete function that can usefully be incorporated into the language. These are

Purposive incomplete function (PIF) maps effect to purpose. This can be used when several subfunctions share a trigger, as in the warning example mentioned above.

Operational incomplete function (OIF) maps trigger to effect with no distinct purpose. It ensures a trigger is associated with a specific effect where alternative combinations of trigger and effect can achieve some purpose.

These are only used as subsidiary (child) functions.

3 Example of functional description

The following is (part of) the functional description of a car's seat belt reminder system that warns the driver if either front seat is occupied with its seat belt unbuckled.

```
FUNCTION belt_warning
  ACHIEVES unbuckled_warning
  BY
    vehicle moving
```

```
AND (driver_unbuckled OR
     (passenger_present AND
      passenger_unbuckled))
TRIGGERS
  PIF warning_lamp AND PIF chimer
```

```
PURPOSE unbuckled_warning
DESCRIPTION "Warn seat belt is unbuckled"
FAILURE_CONSEQUENCE
  "No warning given of dangerous state"
```

The PIFs and their associated purpose descriptions are, like the description of purpose shown, separate entities. The decomposition in this example allows the language to distinguish the case where one of the PIFs fails from the more serious case where both fail so no warning is given. In addition to this increased expressiveness the explicit inclusion of the trigger differs from the language in [Price, 1998] and allows unambiguous description of functions that are triggered by the state of some other system function, increasing the range of systems that can be modeled. The trigger and the use of labels for trigger and effect allow a more precise specification of required system behavior than the earlier language used for functional labeling. They also enable the functional description to be built independently of a target system, so the language for can be used for functional refinement of design in addition to interpretation of model based simulation.

4 Conclusion

The functional description language introduced here increases both the expressiveness and range of the language used for functional labeling in [Price, 1998]. It shares the advantages of simplicity, reusability and capability claimed for that language and in addition allows the functional model to be constructed independently of the system, so supporting the use of the language in functional refinement of a design as well as for interpretation of simulation.

References

- [Bell and Snooke, 2004] Jonathan Bell and Neal A. Snooke. Describing system functions that depend on intermittent and sequential behavior. In *Proceedings 18th International Workshop on Qualitative Reasoning, QR2004*, 2004.
- [Iwasaki et al., 1993] Yumi Iwasaki, R. Fikes, M. Vescovi, and B. Chandrasekaran. How things are intended to work: Capturing functional knowledge in device design. In *Proceedings of 13th International Joint Conference on Artificial Intelligence*, pages 1516–1522, 1993.
- [Price, 1998] Christopher J. Price. Function-directed electrical design analysis. *Artificial Intelligence in Engineering*, 12(4):445–456, 1998.
- [Sticklen and Chandrasekaran, 1989] Jon Sticklen and B. Chandrasekaran. Integrating classification-based compiled level reasoning with function-based deep level reasoning. In Werner Horn, editor, *Causal AI Models, Steps Toward Applications*, pages 191–220. Hemisphere Publishing Corporation, 1989.

A Non-monotonic Logic for Specifying and Querying Preferences

Guido Boella

Dipartimento di Informatica
Università di Torino
Italy
guido@di.unito.it

Leendert van der Torre

CWI Amsterdam
and Delft University of Technology
The Netherlands
torre@cwi.nl

1 Introduction

Preferences are becoming of greater interest in many areas of artificial intelligence, such as knowledge representation, multi-agent systems, constraint satisfaction, decision making, and decision-theoretic planning. In the logic of preference there is a debate when a set of preferences should be consistent. For example, Bacchus and Grove [1996] criticize *ceteris paribus* preferences, because $\{p > \neg p, \neg p \wedge q > p \wedge q\}$ should be consistent, and they criticize most existing logics of preference, because $\{p > \neg p, q > \neg q, \neg(p \wedge q) > p \wedge q\}$ should be consistent. In order not to restrict the use of the logic of preference, we propose a minimal logic of preference in which any set of specified preferences is consistent. To make it useful for practical applications, we extend this logic to specify preferences with a logic to query preferences, and with a non-monotonic reasoning mechanism.

2 New semantics for the logic of preference

We introduce a logic of preference that distinguishes between specifying and querying, inspired by preference-based deontic logic [van der Torre and Tan, 2000]. We define two types of preference statements, where $p >_! q$ can be read as “it is specified that p is preferred to q ” and $p >_? q$ can be read as “the result of the query states that p is preferred to q .”

Definition 1 (Language) Given a set $A = \{a_1, \dots, a_n\}$ of propositional atoms, we define the set L_0 of propositional formulas and the set L of preference formulas as follows.

$L_0 \ni p, q: a_i \mid \neg p \mid (p \wedge q)$

$L \ni \phi, \psi: (p >_! q) \mid (p >_? q) \mid \neg \phi \mid (\phi \wedge \psi)$

Moreover, disjunction $\vee$, material implication $\rightarrow$ and equivalence $\leftrightarrow$ are defined as usual. We abbreviate formulas of the preference logic using the following order on logical connectives: $\neg \mid \vee, \wedge \mid >_!, >_? \mid \rightarrow, \leftrightarrow$. For example, $\neg p >_! q \wedge r$ is short for $(\neg p >_! (q \wedge r))$.

Typically many conflicts arise during the specification of preferences. Our logic of preference represents any set of preferences in a consistent way, which is achieved in two steps. First, we do not use a total pre-order on worlds in the semantics, but a partial pre-order on worlds, i.e., a reflexive and transitive relation. Incomparable worlds indicate some kind of conflict among these worlds. Secondly, and more

originally, we formalize a preference $p >_! q$ as the absence of a q world that is preferred over a p world.

Definition 2 (Specifying) Let W be the set of propositional interpretations of L_0 , and $\succeq$ a partial pre-order on W . We write $w \succ w'$ for $w \succeq w'$ without $w' \succeq w$, and $|\alpha|$ for $\{w \in W \mid w \models \alpha\}$.

- $\succeq \models p >_! q$ iff $\forall w \in |p|$ and $\forall w' \in |q|$ we do not have $w' \succ w$.

Semantic entailment for more complex formulas, logical entailment and other logical notions are defined as usual. For example, $\phi_1, \dots, \phi_n \models \phi$ if and only if for all $\succeq$ such that $\succeq \models \phi_i$ for $i = 1 \dots n$, we have $\succeq \models \phi$.

We define a logic for querying preferences by optimization in a specified preference relation. We use the same semantic structures, but to simplify the definitions we assume that there are no infinitely ascending chains (see, e.g., [Boutilier, 1994] for a discussion). Moreover, p is preferred to q if the optimal $p \vee q$ worlds are p worlds.

Definition 3 (Querying) We assume $\succeq$ does not contain any infinite ascending chains. We write $\max(\succeq, p)$ for $\{w \in W \mid w \models p \text{ and } \forall w' \in W : w' \succ w \Rightarrow w' \not\models p\}$.

- $\succeq \models p >_? q$ iff $\max(\succeq, p \vee q) \subseteq \max(\succeq, p)$.

We distinguish models called most connected models.

Definition 4 (Distinguished models) A model $\succeq_1$ is at least as connected as another model $\succeq_2$, written as $\succeq_1 \sqsubseteq \succeq_2$, if $\succeq_2 \subseteq \succeq_1$, that is, if $\forall w_1, w_2 \in W : w_1 \succeq_2 w_2 \Rightarrow w_1 \succeq_1 w_2$. A model $\succeq_1$ is most connected if there is not another model $\succeq_2$ such that $\succeq_2 \sqsubset \succeq_1$, that is, such that $\succeq_2 \sqsubseteq \succeq_1$ without $\succeq_1 \sqsubseteq \succeq_2$.

Non-monotonic reasoning is based on distinguished models in the usual way. These distinguished models have sometimes been called preferred models, and non-monotonic entailment has been called preferential entailment. However, to avoid confusion with the preferences formalized in our logical language, we do not use this terminology in this paper.

Definition 5 (Non-monotonic entailment) A preference specification is a set of specification preferences $PS = \{p_1 >_! q_1, \dots, p_n >_! q_n\}$. A distinguished model of a preference specification PS is a most connected model of PS . A preference specification non-monotonically entails ϕ , written as $PS \models_{\sqsubseteq} \phi$, if for all distinguished models $\succeq$ of PS we have $\succeq \models \phi$.

3 Logical properties

Specifying and querying preferences are complementary in the sense that the logic of the former satisfies left and right strengthening, and the logic of the latter satisfies transitivity.

3.1 Ordering

We first show that our desideratum holds. A preference specification which only contains preferences which are individually consistent, is itself consistent.

Proposition 1 *If there is a model for each preference in PS , then there is a model of PS .*

Proof. The identity relation $\{w \succeq w \mid w \in W\}$ is such a model of PS .

The logical relations among preference specifications are characterized by left and right additivity.

LA $\models (p \vee q >_! r) \leftrightarrow (p >_! r) \wedge (q >_! r)$

RA $\models (p >_! q \vee r) \leftrightarrow (p >_! q) \wedge (p >_! r)$

Consequently, reading the additivity formulas from left to right, specification preferences are strong preferences, in the sense that they satisfy the following properties of left and right strengthening.

LS $\models (p >_! q) \rightarrow (p \wedge r >_! q)$

RS $\models (p >_! q) \rightarrow (p >_! q \wedge r)$.

Moreover, reading the additivity formulas from right to left, specification preferences satisfy left and right disjunction.

LOR $\models (p >_! r) \wedge (q >_! r) \rightarrow (p \vee q >_! r)$

ROR $\models (p >_! q) \wedge (p >_! r) \rightarrow (p >_! q \vee r)$

In case of a finite set of atoms, the combination of LS, RS, LOR and ROR give a simple way to derive all implied preferences from a preference specification. Call a formula a complete sentence when it implies p or $\neg p$ for each proposition $p \in A$. Given a preference specification PS , first use LS and RS to derive preferences among complete sentences. Secondly, use LOR and ROR to derive all preferences among formulas.

Finally we consider some borderline cases. As usual, we cannot have that a proposition is preferred to itself. This is due to the fact that $\succeq$ is reflexive.

Id $\models \neg(p >_! p)$

Moreover, an important distinction among preference logics is whether they satisfy transitivity. Our logic to specify preferences does not, because we may have r worlds preferred to p worlds, and q worlds incomparable to both p and r worlds.

T $\not\models (p >_! q) \wedge (q >_! r) \rightarrow (p >_! r)$

3.2 Optimizing

Optimizing preferences have been studied in the context of total pre-orders. For example, Weydert [1991] defines elementary qualitative magnitude logic as follows, in a propositional preference logic extended with actual worlds:

QM1 $(p > q) \wedge (q > r) \rightarrow (p > r)$ (transitivity)

QM2 $(p > q) \wedge (p > r) \leftrightarrow (p > q \vee r)$ (additivity)

QM3 $\neg(q > p) \wedge (q > r) \rightarrow (p > r)$ (maximality)

QM4 $\perp > \perp \wedge ((\perp > p) \leftrightarrow (p > p))$ (quasi-reflexivity)

QM5 $\perp > p \rightarrow \neg p$ (correctness)

QM6 $(p > q) \leftrightarrow (r > s)$ if $\vdash (p \leftrightarrow r) \wedge (q \leftrightarrow s)$ (extensionality)

Of these formulas QM5 cannot be expressed in our logic, and the relevant properties to check are QM1, QM2 and QM3. It can be verified that QM1 and QM2 also hold for our logic based on partial pre-orders, but QM3 does not.

In some cases, it may be useful to define a stronger notion of specification preferences, as a combination of $>_!$ and $>_?$.

Definition 6 $p >_0 q = p >_! q \wedge p >_? q$

Consequently, if we have $\succeq \models p >_0 q$ then we also have $\succeq \models p >_? q$, a property that does not hold for $>_!$. That is, if we have $\succeq \models p >_! q$, then we do not necessarily have $\succeq \models p >_? q$.

3.3 Modal characterization

It is well known that the combination of two logics in a single logical system may be problematic, and that the axiomatization of such a combined logic is a non-trivial task. To obtain an axiomatization of our logic, we follow [Boutilier, 1994] by providing a modal characterization of the logic. We define (or *simulate*) the preference logic in a normal bimodal system, where $\Box$ is a universal modal operator and $\Box'$ is a normal modal operator. See, e.g., [Boutilier, 1994] for an axiomatization of the bimodal logic.

Definition 7 Let L' be a modal logic with two modal operators $\Box$ and $\Box'$. Moreover, let a model of L' be a tuple $M = \langle W, \succeq, V \rangle$, with W a set of worlds, $\succeq$ a partial pre-order on W , and V a valuation function of propositions at the worlds. The satisfiability relation for modal formulas is defined as follows.

- $M, w \models \Box p$ when $\forall w' \in W$ we have $M, w' \models p$.
- $M, w \models \Box' p$ when $\forall w' \in W : w' \succeq w \Rightarrow M, w' \models p$.

Moreover, we add the following definitions to L' :

- $\Diamond p = \neg \Box \neg p$, $\Diamond' p = \neg \Box' \neg p$
- $p >_! q = \Box(p \rightarrow \Box' \neg q)$
- $p >_? q = \Box(q \rightarrow \Diamond'(p \wedge \neg q \wedge \Box'(p \vee q \rightarrow p \wedge \neg q)))$

References

- [Bacchus and Grove, 1996] F. Bacchus and A. Grove. Utility independence in a qualitative decision theory. In *Proceedings of the KR-96*, pages 542–552, 1996.
- [Boutilier, 1994] C. Boutilier. Conditional logics of normality: a modal approach. *Artificial Intelligence*, 68:87–154, 1994.
- [van der Torre and Tan, 2000] L. van der Torre and Y. Tan. Two-phase deontic logic. *Logique et Analyse*, 171-172:411–456, 2000.
- [Weydert, 1991] E. Weydert. Qualitative magnitude reasoning. In *Nonmonotonic and Inductive Logic*, volume 543 of *Lecture Notes in Computer Science*, pages 138–160. Springer, 1991.

A Multidimensional Semantic Framework for Adaptive Hypermedia Systems

Francesca Carmagnola, Federica Cena, Cristina Gena, Ilaria Torre

Department of Computer Science
University of Turin
carmagnola,cena,cgena,torre@di.unito.it

Abstract

This paper introduces a multidimensional semantic framework for adaptive systems. Different planes allow us to represent ontologies of user, her actions, context, device, domain, while the intersection between planes allow us to represent the semantic rules for inferring new user features or adaptation strategies. The adoption of ontology-based framework aims at creating a server for user modeling and adaptation strategy.

1 Introduction

The Semantic Web aims at representing information in the WWW such that machines can use it for automation, integration and reuse of knowledge across applications. The advantage of such an approach can be particularly useful in the field of adaptive hypermedia systems. These systems typically reflect some features of the user in the user model and apply this model to adapt various aspects of the system (content, interface, navigation, etc) to the user [Brusilovsky, 1996]. Current adaptive systems may also take into account other features, besides the user model, such as the context of interaction, the device, etc. Usually the corpus of the documents and services the system can adapt is already known at the design time and can be defined as a *closed corpus of adaptation* [Dolog et. al, 2004]. The application of Semantic Web technologies to those systems and the use of shared ontologies and metadata to describe resources can contribute to extend the closed corpus to an *open corpus of adaptation*. Thus, external documents and resources, which are semantically annotated, can be considered during the adaptation to the users. Furthermore, representing the user model with a semantic formalism and shared ontologies can be the base for building a user model server: a server that enables the reuse of user models across applications [Kay et al., 2002]. Different adaptive systems can query the same user model server, be primed with the user model that has already built up and share the common knowledge.

This paper describes an ontology-based framework for adaptive hypermedia systems which is aimed at providing a methodological approach for the semantic definition of two kinds of relevant knowledge for adaptation: (i) knowledge

regarding what features of the system have to be adapted and which dimensions (of the user, context, etc.) have to be taken into account to perform adaptation; (ii) knowledge regarding adaptation strategies and rules for inferring new knowledge. Following the 'equation' $ontology = (i) taxonomy + (ii) axioms$ proposed by the RuleML Initiative [Boley et al, 2001], we represent (i) the declarative descriptions of user models, domain knowledge, etc., with taxonomies expressed in the standard semantic markup language for the Semantic Web, OWL¹, and (ii) the adaptation rules with RuleML², a candidate rule markup language.

2. Goals of the project

While many works exploit ontologies to describe application domains and some recent ones adopt them to represent user models, devices features, context of interaction, etc. (e.g., UbiWorld, <http://www.u2m.org/>), the semantic representation of reasoning strategies is still little addressed in mature projects. Nevertheless, besides the discussed advantages of using standard markup languages for the ontological representation of knowledge, a semantic representation of rules could extend this goal. Indeed, it could allow applications (and designers) to share reasoning strategies, to detect incompatibilities, to validate or eventually refuse them.

Furthermore, the conjunction of *taxonomies* with a semantic representation of *rules* brings benefits to both of them. For example, in adaptive systems, it allows to provide explanations about the inference of new user features, the system behaviour and the forms of adaptation by exploiting the ontological description of terms which are involved in the rules. Moreover, since the semantic representation of rules provides a proof of the reasoning of the system, it is also relevant for the so called *proof layer* of the Semantic Web, which "involves the actual deductive process as well as the representation of proofs in Web Languages and proof validation"[Antoniou et al., 2004].

3. Framework description

¹ <http://www.w3.org/TR/owl-features/>

² <http://www.ruleml.org/>

The framework aims at supporting the visual design, the semantic representation and the implementation of rules in adaptive hypermedia systems based on symbolic reasoning. These rules include *user modeling rules* (which can be considered *derivation rules*) that add new knowledge about user, starting from her current environment, usage data and context of interaction, and *adaptation rules* (which can be considered *reaction rules*), that define adaptation strategies (content, presentation, modality).

The choice of using a semantic formalism in order to define the framework arises from the evidence that user modeling dimensions are common to different applications and, if semantically described, they can be shared among them (e.g. the feature *user familiarity with the system* is used by almost all adaptive systems). Defining these dimensions once for all represents an interesting opportunity in terms of reducing costs of design and optimization of results. Similar considerations can be made for modality and contexts of interaction. While for the domain and service features, the reason to exploit an ontological representation especially deals with the diffusion of this kind of ontologies on the web, and the possibility to link such ontologies and integrating with Web Services [Mizoguchi et al, 1997].

For the definition of this semantic framework we developed a multidimensional matrix [Torre, 01] in which *each plane* contains the *ontological representation of a specific kind of knowledge* (user model, user actions, device, domain, context and adaptation ontology).

Being a framework, the ontologies on the planes are application independent, modular and layered in a first level ontology, containing the definition of general concepts, and successive levels with specialized concepts. In this way, the framework can be used by different applications, selecting a sub-part of the most generic ontology, in the considered planes, and instantiating only the concepts they are interested in.

The basic idea of the matrix is that derivation rules and reaction rules derive from the intersection of planes and that the matrix representation helps the visual design of such rules. For example, for the definition of derivation rules, on the X_1 -plane we put the ontology which describes the *user actions* on adaptive system (selection, bookmark, print, etc.); on the X_2 -plane the ontology which describes the possible *domain features* (business, tourist, e-learning etc); and on the X_3 -plane, the ontology of the *user model dimensions* (demographic features, preferences, interests, etc).

From the intersection of dimensions on these planes we can define user modeling rules (derivation rules), in the form of:

If ((X_1 Plane user actions= a) **AND** (X_2 Plane domain_feature= b)
AND(X_3 Plane explicit_user_features= c,d))
Then (inferred_user_feature= i)

in which the *Left Hand Side* specifies the dimensions that contribute to define the value of the inferred feature and the *Right Hand Side* represents the assignment of this value.

The same methodology can be applied to infer other dimensions, such as, user's goals and plans, or to define the adaptation rules, clearly changing the planes to take into account. For instance, to identify adaptation rules, on the Z_1 -plane we put the ontology that describes the *device* that can be used by the user (PDA, PC, mobile phone, on-board system etc.); on the Z_2 -plane we put the ontology of the *adaptation* (adaptation of content, interface, etc.); on the Z_3 -plane, the ontology of context conditions (e.g. driving, walking, night, etc.) and on the Z_4 -plane, the ontology of the *user model dimensions*, integrated/updated with the user's dimensions inferred by the previous user modeling rules.

The ontologies on the planes are written in OWL, while rules, at the intersection of planes, are written in RuleML. Both of them can be translated into popular rule engines such as CLIPS and Jess.

Conclusion and Future Work

In future we plan to use the proposed methodology with different adaptive applications (e.g., [Amendola et al., 2004]) we developed in the past in order to use these applications as test bench to evaluate the correctness of our approach.

References

- [Amendola, 2004] Amendola I., Cena F., Console L., Crevola A., Gena C., Goy A., Modeo S., Perrero M., Torre I., Toso A.: "UbiquiTO: A Multi-device Adaptive Guide", In *Proc. of Mobile HCI 2004*, LNCS 3160, pp.409-414, 2004.
- [Antoniou et al., 2004] Antoniou, G., Van Harmelen F. A Semantic Web Primer. The MIT Press, Cambridge, MA, 2004.
- [Boley et al, 2001] Boley, H., S. Tabet, and G. Wagner., Design Rationale of RuleML: A Markup Language for Semantic Web Rules, in *Proc. of the First Semantic Web Working Symposium*, SWWS'01, pp. 381-402, 2001.
- [Brusilovsky, 1996] Brusilovsky P.L.: Methods and Techniques of Adaptive Hypermedia. In *User Modelling and User-Adapted Interaction* 6 pp. 87-129, 1996.
- [Dolog et. Al., 2004] P. Dolog, N. Henze, W. Nejdl, M. Sintek. The Personal Reader: Personalizing and Enriching Learning Resources using Semantic Web Technologies. In *Proc. of the AH 2004* , pp. 85-94, LNCS, Springer Verlag, 2004.
- [Kay et al, 2002] J. Kay, R. Kummerfeld, P. Lauder. Personis: A Server for User Models. In *Proc. of the AH*, pp. 203-212, 2002, LNCS, Springer Verlag, 2002.
- [Mizoguchi et al, 1997] R. Mizoguchi, M. Ikeda, K. Sinita: Roles of shared ontology in AI-ED Research. In *Proc. of AI-ED*, pp. 537-544, 1997.
- [Torre, 2001] Torre I.: "Goals, tasks and Application domains as the guidelines for defining a framework for User modelling", in *User Modelling 2001*, LNCS, Springer Verlag, pp. 260-262, 2001.

Explaining Search Results*

Maurice Coyle and Barry Smyth

Smart Media Institute, Department of Computer Science,
University College Dublin, Ireland
{maurice.coyle, barry.smyth}@ucd.ie

Abstract

In this paper we argue that it may be possible to help searchers to better understand the relevance of search results by generating explanations that highlight how other users have interacted with such results under similar search conditions in the past. We propose the use of the search histories of a community of online users as a source of these explanations. We describe the results of a recent study to examine the use of such explanation-based techniques to help Web searchers better appreciate the relevancy of search results. We highlight shortcomings of this approach in its current form and offer suggestions as to how it may be improved in future work.

1 Introduction

Scale, document diversity, and limited searcher expertise all combine to make the Web a very challenging information retrieval environment. Consequently, Web search engines have had to evolve beyond their term-based information retrieval roots; see [Brin and Page, 1998; Lawrence, 2000] for approaches that leverage connectivity and context information to improve search. In this paper we adopt a very different perspective on how search engines might be improved. We ask whether there is an opportunity to present search results to users more effectively. This is related to the use of clustering methods to impose order on search results at presentation time (e.g., [Leuski, 2001]). However, we try to *explain* results to users, to help users gain a better understanding of the appropriateness of these results to their needs.

In related work, [Hearst, 1995] introduces a visualisation paradigm that offers an explanation not just of how strong the match between the user's query and a given result page is, but also how frequent each term is, how each term is distributed in the text and where the terms overlap within the document. This approach provides the user with additional explanatory information which better aids them in their decision as to whether or not each page is relevant to their information need.

*The support of the Informatics Research Initiative of Enterprise Ireland is gratefully acknowledged.

For related work on the issue of explanation in recommender systems see [Cunningham and McSherry, 2004].

Of course, Web search engines do attempt to provide *some* explanation with their result-lists. Typically, for example, each result is accompanied by a *snippet* of text from the result page in question to give some insight into the content of the page. Both these query-sensitive snippets and the approach to explanation in information retrieval introduced in the TileBars system ([Hearst, 1995]) are content-based techniques for aiding the user in result selection. We believe that other forms of explanatory information can be usefully provided.

In [Wexelblat and Maes, 1999], *interaction histories* are introduced, whereby a user's interactions with their Web browser (e.g. selection of a link in a Web page, typing a URL into the location bar, etc.) are recorded as they browse, and this information is later used to help other users to locate information items. The concept of interaction histories is applied to Web search in [Coyle and Smyth, 2005]; the pages in a user's navigation sequence beyond the initial results page are recorded and used to enhance future search result lists when similar queries are submitted. We propose the use of search history information (specifically query-submission and result-selection information) for the generation of search result explanations. In this paper we describe a recent study to evaluate different approaches to generating these explanations based on 3 types of user interaction information.

2 Explaining Search Results

Result snippets provide one type of explanation. What other types of explanation might be generated? In this paper we consider 3 additional types of explanation information that can be derived from the search histories of communities of users. We will describe these types of information in a moment but first it is worthwhile considering the source of these search histories.

2.1 Collaborative Web Search

Collaborative Web search (CWS) is a form of meta-search, relying on the search services of a set of underlying search engines and manipulating their results in response to the learned preferences of a given community of users ([Smyth *et al.*, 2003; In Press]). Very briefly, a central data structure in CWS is the *hit-matrix*, *H*. It encodes the search behaviour of a given community of users; each time a community member

selects a result p_j in response to some query q_i , the entry in cell H_{ij} is incremented. In turn, the *relevance* of a page p_j to q_i can be estimated as the relative number of selections p_j has received in the past for q_i . For example, if the page www.jaguarcars.com has a relevance of 0.65 for the query *jaguar*, this means that 65% of the selections for this query have been for this result. More generally, the relevance of a page p_j to some target query q_T can be calculated as the weighted sum of its relevance to a set of queries ($q_1, \dots, q_n$) that are similar to q_T , with each individual relevance value discounted by the similarity between q_T and the query in question; see [Smyth *et al.*, In Press] for further information.

The type of interaction data generated by CWS, which can be stored and later used as a source of search histories includes, but is not limited to, search result selection, query submission and query modification (i.e., the same user submitting a number of queries one after the other which are similar to each other regarding their component terms). The point to note is that this stored interaction data can be leveraged in future search sessions to provide explanation text alongside search results which can be used to determine a result's popularity, by analysing how users within the same searching community have previously interacted with it. With this added information, the user is better positioned to judge whether or not each result is relevant to their information need.

2.2 Explanation Types

In this paper we propose to use the interaction data captured by CWS as a source of 3 different explanation types: 1) relevance scores; 2) related queries; 3) timing information. From these we have generated the following 6 explanation types.

Type 1 - Relevance. The relevance of a result to some query is simply the percentage of times that other searchers have selected the page for the query. Thus each result is accompanied by an explanation of the form: "*r% of searchers for this query have selected this page*".

Type 2 - Related Queries. Related queries refer to other queries that have also led to the selection of the page in question. They provide users with an alternative understanding of the circumstances in which other users have found a page to be worth selecting; e.g., "*This page has also been selected for queries such as 'q1' and 'q2'*".

Type 3 - Timing. Timing information helps the searcher to understand the recency of any selection activity for this page and may give an indication of the currency of its content; e.g., "*This page was last selected t minutes ago*".

Type 4 - Relevance + Related Queries. We can combine relevance information with related queries to generate explanations of the form: "*r1% of searchers for this query have selected this page and r2% of searchers for similar queries such as 'q1' and 'q2' have also selected it*".

Type 5 - Relevance + Timing. Similarly, we can combine relevance and timing information to generate explanations of the form: "*r% of searchers for this query have selected this page as recently as t minutes ago*".

Type 6 - Relevance + Related Queries + Timing. And finally, we can combine all 3 types of data to produce explanations of the form: "*As recently as t minutes ago, r1% of searchers for this query have selected this page and r2% of*

IJCAI-05

... Call for Tutorial Proposals. Call for IJCAI Award Nominations ...
co-host. co-host. sponsored by. IJCAI ...
<http://ijcai05.csd.abdn.ac.uk>

Figure 1: Search result for the query 'ijcai' with only a snippet text explanation.

IJCAI-05

... Call for Tutorial Proposals. Call for IJCAI Award Nominations ...
co-host. co-host. sponsored by. IJCAI ...
<http://ijcai05.csd.abdn.ac.uk>

? 54% of searchers for the current query have selected this result (most recently 4 days ago) and 82% of all searchers for the related query 'ijcai 2005' have also selected it ?

Figure 2: Search result for the query 'ijcai' with interaction history-based explanation (type 6) added below the snippet text explanation.

searchers for similar queries such as 'q1' and 'q2' have also selected it".

By way of an example, consider the search results presented in Figures 1 and 2. Figure 1 shows a standard Web search result returned for the query 'ijcai' with the page's title and a contextualised snippet provided as a means for the user to determine if the result is relevant to their query. In Figure 2, we can see the same search result but this time with explanation text added below it (explanation type 6, see above) which was generated using the interaction histories of previous users. This explanation contains information about how many times this result has been selected in the past for the current query and also for other, related queries. Further, the user can see how recently this result has been selected, which should give them some idea of how 'fresh' the result is for the current query.

3 Evaluating Search Result Explanations

To evaluate the utility of these different explanation types, we surveyed the opinions of 57 postgraduate students. Each survey was composed of 6 pairs of search engine result-lists. In each pair there was a *standard* result-list, that contained the sort of result-list presented by Google and other major search engines; each result contained a title, a snippet, and a URL (see Figure 1). Each pair also contained an *explanation* result-list in which the *same* results were presented with one of the 6 explanation types described above (see Figure 2). The 6 pairs of each survey allowed users to compare each explanation type to a standard result-list; for each pair the user was asked to indicate whether they felt the explanations helped them to understand the value of the results in relation to the target query. In each survey the type of results and queries used to generate them were randomised so no two pairs of result-lists contained the same query-result combination for any user.

Summary results are presented in Figure 3 as the percentage of users who found the explanations to be *more informative* (better) than the standard result-list against each result type; we also show the percentage of users who found each

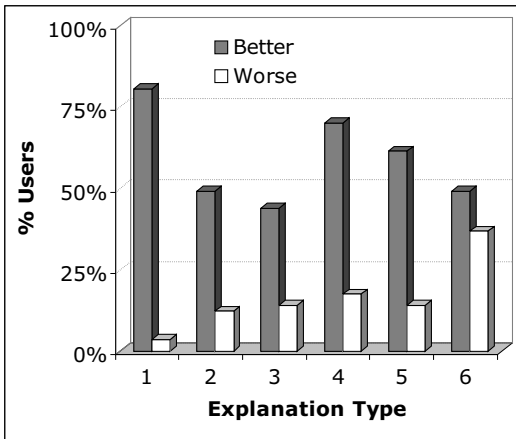


Figure 3: Summary of results.

explanation type to be *less informative* (worse) than the default. The results show a clear preference for relevance information (type 1 explanations) with more than 80% of users indicating that this type of explanation helped them to better understand the usefulness of results; only 3% of users rated this explanation type as worse than the default.

In general, the users appear to be less convinced about the value of the other explanation types with just under 50% viewing the related queries (type 2) and timing (type 3) as better than the default; although only about 12%-14% of users found these explanations to be worse than the default. When these explanation types are combined with relevance information (in types 4, 5 and 6) the user ratings are seen to increase, although never above the 80% found for relevance information on its own. This increase appears to be due to the inclusion of the relevance data, however.

4 Conclusions

In this paper we have suggested that modern Web search engines might be usefully improved if they included some form of explanation information alongside their search results. This could potentially provide searchers with a clearer understanding of how each result relates to their information need, if at all. We have tested a number of different explanation types using a combination of information sources.

The results of this experiment indicate that there is potential for Web searchers to benefit from the presentation of additional explanatory information alongside standard result-list information. Concrete relevance information in particular was found to be valued by more than 80% of users, although other forms of explanation were found to be less compelling. One of the problems with these other forms of explanation data appears to be linked to their increased space requirements. For example, the use of related queries and timing information in particular adds considerably to the size of explanations, which takes away from valuable screen-space that could otherwise be used to present additional search results.

It is likely that this presentation issue could be resolved however, either through the use of more economical tex-

tual presentation techniques such as mouse-over tool-tips or else through the use of a visualisation approach whereby the user interaction information for a search result is summarised using graphical icons in a manner similar to that found in [Hearst, 1995]. It is worth noting that only a small percentage of users viewed the additional explanations as having a negative impact on the search results. This allows us to be optimistic that a more economical, visual presentation metaphor might reduce this percentage and lead to an increase in the perceived worth of the various explanation types.

References

- [Brin and Page, 1998] Sergey Brin and Lawrence Page. The Anatomy of a Large-Scale Hypertextual Web Search Engine. *Computer Networks and ISDN Systems*, 30(1-7):107-117, 1998.
- [Coyle and Smyth, 2005] Maurice Coyle and Barry Smyth. Enhancing web search result lists using interaction histories. In *Proceedings of the 27th European Conference on Information Retrieval, (ECIR-05)*, pages 543-545, Santiago De Compostela, Spain, 2005. Springer-Verlag.
- [Cunningham and McSherry, 2004] Padraig Cunningham and David McSherry. *Workshop on Explanation in CBR*. 7th European Conference on Case-Based Reasoning, Madrid, Spain, 2004.
- [Hearst, 1995] Marti A. Hearst. Tilebars: Visualisation of term distribution information in full text information access. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI'95*, pages 59-66, Denver, Colorado, USA, May 1995. ACM Press.
- [Lawrence, 2000] Steve Lawrence. Context in Web Search. *IEEE Data Engineering Bulletin*, 23(3):25-32, 2000.
- [Leuski, 2001] Anton Leuski. Evaluating document clustering for interactive information retrieval. In Ling Liu Henrique Paques and David Grossman, editors, *Proceedings of Tenth International Conference on Information and Knowledge Management (CIKM'01)*, pages 41-48, Atlanta, Georgia, USA, 2001. ACM Press.
- [Smyth et al., 2003] Barry Smyth, Evelyn Balfe, Peter Briggs, Maurice Coyle, and Jill Freyne. Collaborative Web Search. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence, (IJCAI-03)*, pages 1417-1419. Morgan Kaufmann, 2003. Acapulco, Mexico.
- [Smyth et al., In Press] Barry Smyth, Evelyn Balfe, Jill Freyne, Peter Briggs, Maurice Coyle, and Oisín Boyde. Exploiting Query Repetition and Regularity in an Adaptive Community-based Web Search Engine. *User Modeling and User-Adapted Interaction: The Journal of Personalization Research*, (In Press).
- [Wexelblat and Maes, 1999] Alan Wexelblat and Pattie Maes. Footprints: history-rich tools for information foraging. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI'99*, pages 270-277, Pittsburgh, Pennsylvania, USA, 1999. ACM Press.

A Cognitive Model of Visual Analogical Problem- SolvingTransfer

Jim Davies

Queen's University, School of Computing
Kingston, Ontario, Canada K7L 3N6, jim@jimdavies.org

Ashok K. Goel, Nancy J. Nersessian

Georgia Institute of Technology, College of Computing
801 Atlantic Drive, Atlanta, Georgia, USA, 30332, {goel, nancyn}@cc.gatech.edu

Abstract

Complex problem solving typically involves the generation of a procedure consisting of an ordered sequence of steps. Analogical reasoning is one strategy for solving complex problems, and visual reasoning is another. Visual analogies pertain to analogies based only on visual knowledge. In this paper, we describe the use of Galatea, a computational model of visual analogies in problem solving, to model the problem solving of a human subject (L14). L14 was given the task of solving a complex problem using analogy in a domain that contained both visual and non-visual knowledge, and was encouraged to use visual analogy. We describe how Galatea models L14's use of visual analogy in problem solving.

1 Introduction

Complex problem solving typically involves the generation of a procedure consisting of a strongly-ordered sequence of steps. These procedures have two important properties: 1) the procedure contains two or more steps, and 2) some steps cannot be executed before some other steps have already been executed. Analogical reasoning is one strategy humans often use to solve complex problems. Analogical problem solving takes a solution from a source analog and applies some version of that solution to a target problem. Visual reasoning is another strategy humans sometimes use to address complex problems.

We have used Galatea to model data on problem solving by several human subjects. This data was collected in a psychological experiment run by Dr. David Craig. In this experiment, the human participants were given the task of solving a complex problem using analogy in a domain that contained both visual and non-visual knowledge, and were encouraged to use visual analogy in their problem solving. In this paper, we describe how Galatea models one participant's (L14's) use of visual analogy in problem solving. In the interest of space, we refer you to [Davies and Goel, 2001] for information about the Galatea modelling language.

2 The Galatea Model of L14

Galatea is intended to be a partial cognitive model of visual analogical transfer in human beings. To support Galatea with respect to its psychological plausibility we modelled some of the visual aspects of four experimental participants' drawings, one of which we will describe in detail in this paper.

Dr. David Craig ran 34 participants in an analogical transfer experiment [Craig *et al.*, 2002]. Participants were shown a problem-solving solution about a laboratory, presented with text and a diagram. They were asked to solve an analogous problem with a weed-trimmer, presented with text only. Of these, 17 participants (in three conditions) correctly described the analogous solution. All participants were asked to draw a diagram to illustrate their suggested solutions.

The source given was a laboratory clean room problem. A single door lets in dirty air, so a vestibule is added, with two doors where one door stayed shut while the other was open. The target problem is a weed trimmer arm attached to a truck that must be able to pass through street signs. The analogous solution is to design an arm with two latching doors, so that while one is open to let the sign pass, the other stays closed to support the arm and trimmer. Participants produced diagrams describing their solutions to the problems. We modelled four of these experimental participants in Galatea, one of which, L14, we will describe in detail.

We represented the source analog as a series of *s-images* connected with *transformations*. See the top of Figure 2 for an abstract diagram of our Galatea model of the source analog.

The first transformation is *replicate*. The second transformation is *add-connections* which places the door sets in the correct position in relation to the top and bottom walls. The third and fourth transformations are *add-component*, which add the top and bottom containment walls. The fifth transformation, another *add-connections*, places these containment walls in the correct positions in relation to the door sets and the top and bottom walls.

2.1 Evaluation

We can now examine what made L14 differ from the stimulus drawing: long vestibule, rotation, line to double line, sliding doors, added objects, numeric dimensions added, and mechanisms added. Of these seven differences, our model success-

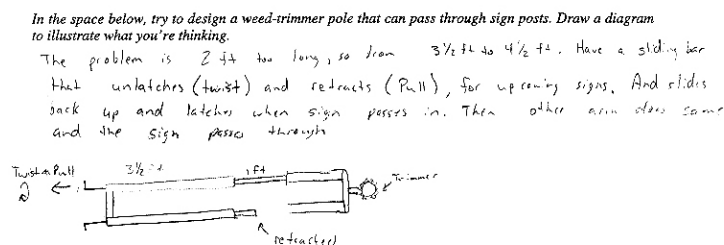


Figure 1: The source data for L14. The drawing above and handwritten text are what participant L14 produced on the experiment sheet.

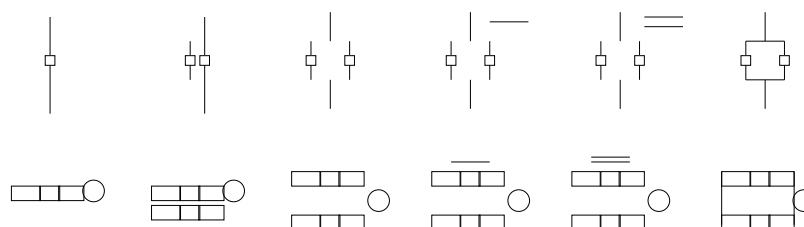


Figure 2: The implementation of L14. The top series of S-images represents the source analog (the lab problem) and the bottom series the target. There are six S-images for the five transformations.

fully re-creates four of them, all through manipulation of the input knowledge and by adding transformations. That is, the source Galatea code did not need to be changed.

Unaccounted for are the two bent lines emerging from the vestibule on the left side, the numeric dimensions and words describing the mechanism. Also, L14 shows one of the doors retracting, and the model does not. The model also fails to capture the double line used to connect the door sections, because the single line is transferred without adaptation from the source. This could be fixed, perhaps, by representing the argument to the add-component as a function referring to whatever element is used to represent another wall, rather than as a line.

3 Conclusion

Our initial view was that visual knowledge facilitates transfer even when non-visual knowledge might be available. In conclusion, the evaluation supported this and resulted in one unexpected discovery, for a total of two claims: **Claim one: Visual knowledge facilitates transfer even when non-visual knowledge might be available.** L14's data is an example of a cross-domain analogical problem solving that could have been represented both non-visually and visually. Our implemented model of L14 uses only visual knowledge. The level of abstraction of the visual symbols in Covlan allows the cross-domain transfer of problem-solving steps to occur, supporting our hypothesis. Further it shows that Galatea can account for human participant data.

Claim two: The successful transfer of strongly-ordered procedures in which new objects are created requires the reasoner to generate intermediate knowledge states and mappings between the intermediate knowledge states of the source and target analogs. In the course of creating this model, we discovered something about analogical transfer in general: A characteristic of strongly-ordered procedures is that components of the problem are *created* by the operations, and these components are acted on by later operations.

Our model shows how this might work for human cognition: The doorway is replicated, then moved, then sealed with containing walls. For the transfer of multi-step, strongly-ordered procedures it was necessary for Galatea to generate intermediate knowledge states and mappings.

References

- [Craig *et al.*, 2002] David L. Craig, Nancy J. Nersessian, and Richard Catrambone. *Model-Based Reasoning: Science, Technology, & Values*, chapter Perceptual simulation in analogical problem solving. Kluwer Academic: Plenum Publishers, 2002.
- [Davies and Goel, 2001] Jim Davies and Ashok K. Goel. Visual analogy in problem solving. In *Proceedings of the International Joint Conference for Artificial Intelligence 2001*, 2001.

An Architecture for Proof Planning Systems

Louise A. Dennis

School of Computer Science and Information Technology,
University of Nottingham, Nottingham, NG8 1BB
lad@cs.nott.ac.uk

Abstract

This paper presents a generic architecture for proof planning systems in terms of an interaction between a customisable proof module and search module. These refer to both global and local information contained in reasoning states.

1 Introduction

Proof planning systems attempt to automate the process of mathematical proof by searching for a high-level representation of a proof rather than one at the level of logical inference rules. This makes the resulting proofs more human-like; aiding comprehension, cognitive modeling and higher-level reasoning about failed proof attempts. Proof planning represents an important methodology in automated reasoning and has been proposed as a generic framework for understanding reasoning in general [Bundy, 1991]. Unfortunately over 15 years of research and exploration of the concept has so altered the initial descriptions that it is no longer trivial to describe exactly what proof planning *is*. This paper presents a generic architecture for proof planning systems as a first step towards a unified description.

2 Proof Planning

Proof planning was first introduced in [Bundy, 1988] where the term *proof plan* was used to describe both a way of going about a particular sort of proof and a high-level representation of the proof itself. In order to avoid confusion, this paper refers to these two concepts as the *proof strategy* and the *proof representation* respectively. A common example of a proof strategy is mathematical induction. Here the strategy is to split the problem into a base case and a step case and the step case can be further broken down into a sequence of rewrites followed by appeal to the induction hypothesis.

The proof representation is a graph/tree structure. The nodes are labelled by proof goals and a justification associated with a *proof method*. The intended semantics is that a goal formula can be derived from its children and that this derivation is justified by the method. These

proof representations are formed using the proof methods as plan operators within a framework inspired by STRIPS [Fikes and Nilsson, 1971]. The edges in a proof representation can be expanded to a proof at the level of logical inference rules using planning operators based on LCF-style tactics. These tactics are associated with the method that justifies the parent node.

Methods are generally represented as a frame structure with slots for preconditions which must be satisfied before the method can apply. A proof strategy is either represented by some form of *method hierarchy* which restricts consideration of particular methods to particular points in a proof or by *control rules* [Siekman *et al.*, 2003] which analyse the current state of the plan representation and the history of the planning attempt to decide on the methods to be considered at any point. The proof strategy is an important technique for limiting search and has been represented in a variety of ways.

The proof planning methodology also includes failure-triggered plan operators (eg. [Ireland and Bundy, 1996]) which propose major changes to the current proof representation (as opposed to methods which simply extend the graph). Selection and application of these operators is determined by preconditions and the proof strategy.

At present there is no general description of proof planning which addresses the varied ideas of proof strategy implemented in the proof planning systems of the last decade. This paper seeks to answer the question of what proof planning is by presenting a generic architecture that can encompass existing implementations.

3 A General Architecture for Proof Planning Systems

Figure 1 shows a general architecture for proof planning systems as a data/control flow diagram. In this diagram circles indicate functions or operations and rectangles stores of data. Solid arrows into operators represent information used by the operator and arrows from it show outputs or side-effects of its operation. Dashed arrows show the flow of control between operations.

Most descriptions of proof planning have described the proof planner as operating on proof representations however an important insight in [Dixon and Fleuriot, 2003]

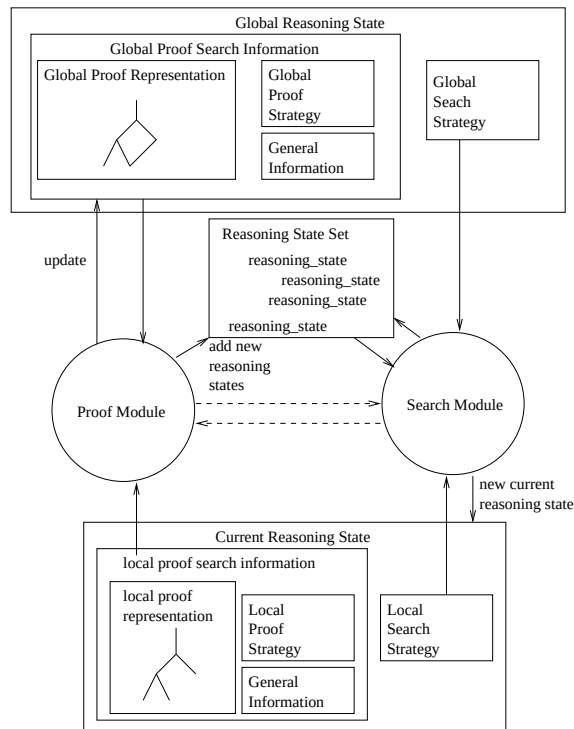


Figure 1: A General Architecture for Proof Planning Systems

is to notice that proof planners operate on *reasoning states*. Reasoning states contain a search strategy (such as depth-first search) and proof search information which includes general information (such as constraints) as well as a proof representation and a proof strategy.

There are two reasoning states consulted by the system: a global reasoning state which is in existence for the lifetime of the program's execution and which can be dynamically updated and a current reasoning state which is generated by the proof module at one point in time, placed in a global set and later selected by the search module to become current after which it is deleted. This reasoning state is not altered once it is generated.

Global information is used in all systems for pragmatic reasons. However there is also a need to store a *history* (eg. [Siekman et al., 2003]) of the planning attempt, including abandoned branches of the search space. This needs to be global and dynamically updatable.

At the start of a proof attempt a user presents a global reasoning state and a set (usually a singleton) of local reasoning states to the system. The search module then selects one of the local reasoning states to become current and hands control to the proof module.

Control passes back and forth between the proof module and the search module. The proof module uses proof search information to generate new reasoning states and to update the global reasoning state. Control is then handed to the search module which uses the search strategies to propose a new current reasoning state and

to update the reasoning state set.

A key idea is that the proof module should be customisable by the user via the proof strategy. In this way a proof planner can be adapted for individual proof families without the user needing a firm grip on the internal code of the system. User support for expressing a proof strategy varies widely and appears to be an area ripe for further research.

4 Conclusion and Further Work

I have presented here a general architecture for proof planning as an interaction between a proof module and a search module operating on two reasoning states (one global and one local) and a reasoning state set.

Dixon et al. [2004] have recently started work on a comparison of the major proof planning systems. We are in the process of combining our work and hope this will generate further insights into the definition of proof planning and the construction of proof planning systems.

Acknowledgements

This work was supported by EPSRC Platform Grant GR/S01771/01. Thanks are due to Lucas Dixon and Martin Pollet for valuable discussions.

References

- [Bundy, 1988] Alan Bundy. The use of explicit plans to guide inductive proofs. In R. Lusk and R. Overbeek, editors, *CADE 9*, volume 310 of *LNCS*, pages 111–120. Springer, 1988.
- [Bundy, 1991] A. Bundy. A science of reasoning. In J.-L. Lassez and G. Plotkin, editors, *Computational Logic: Essays in Honor of Alan Robinson*, pages 178–198. MIT Press, 1991.
- [Dixon and Fleuriot, 2003] L. Dixon and J. D. Fleuriot. IsaPlanner: A prototype proof planner in Isabelle. In F. Baader, editor, *CADE 19*, volume 2741 of *LNCS*, pages 279–283. Springer, 2003.
- [Dixon et al., 2004] L. Dixon, M. Jamnik, and M. Pollet. Proof planning: Comparing Ω MEGA, λ Clam and ISA-PLANNER. In B. Bennett, editor, *ARW 11*, pages 50–52. University of Leeds, School of Computing, 2004. Held in Association with the AISB'04 Convention.
- [Fikes and Nilsson, 1971] R. E. Fikes and N. J. Nilsson. STRIPS: A new approach to the application of theorem proving to problem solving. *AI*, (2):189–208, 1971.
- [Ireland and Bundy, 1996] A. Ireland and A. Bundy. Productive use of failure in inductive proof. *JAR*, 16(1–2):79–111, 1996.
- [Siekman et al., 2003] J. Siekman, C. Benzmler, A. Fiedler, A. Meier, I. Normann, and M. Pollet. Proof development in OMEGA: The irrationality of square root of 2. In F. Kamareddine, editor, *Thirty Five Years of Automating Mathematics*, Kluwer Applied Logic series. Kluwer Academic Publishers, 2003.

Explaining preferences with argument positions *

Sylvie Doutre Trevor Bench-Capon Paul E. Dunne

Department of Computer Science

University of Liverpool

Liverpool L69 7ZF

United Kingdom

{sd,tbc,ped}@csc.liv.ac.uk

Abstract

When deciding what to do agents must choose among alternative actions and different agents may make different choices according to what they wish to achieve in the light of their preferences and values. It cannot be assumed, however, that agents have a conscious understanding of their value preferences independent of the reasoning situations in which they engage. In this paper we consider an extension to a generic framework for reasoning about arguments justifying actions in terms of values in which the preferences amongst values emerge from the reasoning process.

1 Introduction

We are concerned with *practical reasoning* - reasoning about what to do in a given situation. Such reasoning is quite different from reasoning about beliefs, and has a number of characteristic features which any account must respect.

First, arguments justifying actions must be considered in the context of other related arguments: arguments can only be accepted if due consideration to arguments attacking and defending them is given. In a set of arguments relating to an issue - which we call a *debate* - the acceptability of an argument relies on it forming part of a coherent subset of such arguments able to defend itself against all attacking arguments. We call such a coherent subset a *position*. The construction of a position and related problems of acceptability have been explored in AI through the use of argumentation frameworks, e.g. [Dung, 1995; Bench-Capon, 2003]. Such reasoning can be naturally explored through the use of a dialogue in which an argument is attacked and defended [Cayrol *et al.*, 2003; Dunne and Bench-Capon, 2003].

Second, debates about which action is best to perform must permit rational disagreement. Whereas the truth of facts may be demonstrated and compel rational acceptance, with regard to actions there is an element of choice: we cannot choose what is the case, but we can choose what we attempt to bring about, and different people may rationally make different choices. Such differences in values and interests mean that

arguments will have different *audiences*, and what is acceptable to one audience may be unacceptable to another. Disagreements are represented in [Dung, 1995] by the presence of multiple acceptable positions. In [Bench-Capon, 2003], the disagreements are explained within an extended argumentation framework which explicitly relates arguments to values and explicitly represents audiences in terms of their preferences over values.

While a framework such as that of [Bench-Capon, 2003] can be used to explain disagreements between different audiences in terms of their different ranking of values, it does not explain how these value rankings are formed. A third feature of practical reasoning (as indicated in [Searle, 2001]) is that we cannot presuppose that people bring to a debate a knowledge of their value preferences. It means that the *value preferences should emerge from the construction of a position* instead of being taken as an input.

Finally, practical reasoners may not be equally open to all arguments: they may have certain arguments that they wish to include in their position (some *desired arguments*), certain arguments that they wish to exclude (*rejected arguments*), and they may be indifferent to the status of the remainder (the *optional arguments*). For example, a politician forming a political programme may recognise that raising taxation is electorally inexpedient and so must reject any arguments with the conclusion that taxes should be raised from the manifesto, while desiring that arguments justifying actions bringing about core objectives are present: other arguments are optional, and acceptable if they enable this. Such a distinction between arguments has been taken into account in the construction of positions for Dung's framework by [Cayrol *et al.*, 2002], but not for that of [Bench-Capon, 2003].

It is an account of these two last phenomena in the [Bench-Capon, 2003] framework which is the objective of our work.

2 Definition of a position

To accommodate the last feature of practical reasoning identified above, we define an extension of the [Bench-Capon, 2003] framework, called a **DOR-partitioned value-based argument framework** (DOR-VAF), as a tuple $\langle \mathcal{X}, \mathcal{A}, \mathcal{V}, \eta \rangle$, where: $\mathcal{X} = D \cup O \cup R$ for three disjoint sets D (the *desired arguments*), O (the *optional arguments*) and R (the *rejected arguments*); $\mathcal{A} \subseteq \mathcal{X} \times \mathcal{X}$ is an *attack* relation between arguments; $\mathcal{V} = \{v_1, v_2, \dots, v_k\}$ is a set of k values, and

*Sylvie Doutre is supported by the Aspic project of the European Commission (IST-FP6-002307).

$\eta : \mathcal{X} \rightarrow \mathcal{V}$ is a mapping that associates a value $\eta(x) \in \mathcal{V}$ with each argument $x \in \mathcal{X}$.

As in [Bench-Capon, 2003], preferences between values are considered through the notion of an *audience*, which is a transitive, irreflexive relation $\vartheta \subset \mathcal{V} \times \mathcal{V}$. A pair $\langle v_i, v_j \rangle$ is referred to as ' v_i is preferred to v_j ' w.r.t. ϑ . Given an audience ϑ , an argument x *defeats* an argument y if x attacks y and $\langle \eta(y), \eta(x) \rangle \notin \vartheta$; an argument x is a *defender* of an argument y if and only if there is a finite sequence $a_0, \dots, a_{2n}$ such that $y = a_0$, $x = a_{2n}$, and $\forall i, 0 \leq i \leq (2n - 1)$, a_{i+1} defeats a_i w.r.t. ϑ .

A **position in a DOR- VAF** is defined as a set of arguments $P = D \cup Y$ with $Y \subseteq O$, such that there exists at least one audience ϑ with respect to which: (i) no argument in P defeats w.r.t. ϑ another argument in P ; (ii) any defeated argument in P w.r.t. ϑ has a defender w.r.t. ϑ in P ; (iii) each optional argument of Y is a defender w.r.t. ϑ of a desired argument of D . An audience for which P is a position is said to be a *corresponding audience* of P .

This new notion of a position accommodates the third feature of practical reasoning: the preferences between values are not given as an input on the basis of which the position is constructed, but are a result of constructing the position.

3 Development of a position

In order to build a position in a DOR-VAF, one may start by considering the set of desired arguments. This set must be first tested to demonstrate that there is at least one audience w.r.t. which no desired argument defeats another desired argument. It may be that this condition can only be satisfied by imposing some value preferences. If we can satisfy this test we must next ensure that any defeated argument in the set has a defender in the set w.r.t. at least one audience. To this end, some optional arguments may be added to the set as defenders of defeated arguments and/or some additional constraints on the ordering of values may be imposed. If the process succeeds, then the set developed is a position and the set of constraints determined by the construction can be extended into a corresponding audience of this position, by taking its transitive closure. Otherwise, the user has to reconsider the partition of the set of arguments.

This construction can be presented in the form of a *dialogue* between two players. One, the *opponent*, outlines why the set under development is not yet a position, by identifying arguments which defeat members of the set. The other, the *proponent*, tries to make the set under development a position by extending it with some optional arguments and/or some constraints between values. If the proponent terminates the dialogue, then the set of arguments played is a position, and the set of constraints advanced can be extended into a corresponding audience. Otherwise, the set of desired arguments cannot be extended into a position.

This presentation in a dialogue form has the main advantage of making clear why some constraints between values must be imposed, and why some optional arguments must belong to the position. Moreover, it captures well the first feature of practical reasoning. In [Doutre et al., 2005], an original formal dialogue framework, extending previous dialogue

frameworks for argumentation, is introduced and instantiated in order to capture the construction above. The instantiation takes into account some heuristics aiming at keeping the extensions of the set under development to a minimum. The idea is that the proponent will wish to meet an attack in a way which retains as much room for subsequent manoeuvre as possible. There are four ways of meeting an attack from an argument x . In ascending order of commitment they are: include an optional argument requiring no new value preference; add a new value preference; include an optional argument requiring an additional value preference, but does not conflict with any existing position argument; and add an optional argument which does conflict with some existing position argument so requiring additional value preferences to be imposed to resolve the conflict. The first three responses are computationally cheap: the complexity in constructing positions arises when the need to use the fourth response arises.

4 Conclusion

We believe that this approach will have significant practical application in the analysis and modelling of argumentation about which action should be chosen, for instance in areas such as case law and political debate, both of which are receiving increasing attention as the notion of e-democracy becomes widespread. Interesting possibilities also arise if we consider extending a debate in which a position has already been constructed. Some additions should radically change the best position, giving insight into phenomena such as paradigm shift in science, and landmark cases which require extensive theory revision in the development of case law.

References

- [Bench-Capon, 2003] T. J. M. Bench-Capon. Persuasion in Practical Argument Using Value-based Argumentation Frameworks. *Journal of Logic and Computation*, 13(3):429–448, 2003.
- [Cayrol et al., 2002] C. Cayrol, S. Doutre, M.-Ch. Lagasque-Shiex, and J. Mengin. “Minimal defence”: a refinement of the preferred semantics for argumentation frameworks. In *Proc. NMR’2002*, pages 408–415, 2002.
- [Cayrol et al., 2003] C. Cayrol, S. Doutre, and J. Mengin. On Decision Problems related to the preferred semantics for argumentation frameworks. *Journal of Logic and Computation*, 13(3):377–403, 2003.
- [Doutre et al., 2005] S. Doutre, T. J. M. Bench-Capon, and P. E. Dunne. On the development of argument positions. Technical Report ULCS-05-002, Univ. of Liverpool, 2005.
- [Dung, 1995] P. M. Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n -person games. *Artificial Intelligence*, 77:321–357, 1995.
- [Dunne and Bench-Capon, 2003] P. E. Dunne and T. J. M. Bench-Capon. Two party immediate response disputes: properties and efficiency. *Artif. Intell.*, 149:221–250, 2003.
- [Searle, 2001] J. R. Searle. *Rationality in Action*. MIT Press, 2001.

Heuristics for Hard ASP Programs *

Wolfgang Faber[†] and Nicola Leone and Francesco Ricca

Department of Mathematics, University of Calabria, I-87030 Rende (CS), Italy

Email: {faber,leone,ricca}@mat.unical.it

Abstract

We define a new heuristic h_{DS} for ASP, and implement it in the (disjunctive) ASP system DLV. The new heuristic improves the evaluation of Σ_2^P/Π_2^P -hard ASP programs while maintaining the benign behaviour of the well-assessed heuristic of DLV on NP problems. We experiment with the new heuristic on QBFs. h_{DS} significantly outperforms the heuristic of DLV on hard 2QBF problems. We compare also the DLV system (with the new heuristic h_{DS}) to three prominent QBF solvers. The results of the comparison, performed on instances used in the last QBF competition, indicate that ASP systems can be faster than QBF systems on Σ_2^P/Π_2^P -hard problems.

1 Introduction

Answer set programming (ASP) is a novel programming paradigm, which has been recently proposed in the area of nonmonotonic reasoning and logic programming. The idea of answer set programming is to represent a given computational problem by a logic program whose answer sets correspond to solutions, and then use an answer set solver to find such a solution [Lifschitz, 1999]. The knowledge representation language of ASP is very expressive in a precise mathematical sense; in its general form, allowing for disjunction in rule heads and nonmonotonic negation in rule bodies, ASP can represent *every* problem in the complexity class Σ_2^P and Π_2^P (under brave and cautious reasoning, respectively) [Eiter *et al.*, 1997]. Thus, ASP is strictly more powerful than SAT-based programming, as it allows us to solve even problems which cannot be translated to SAT in polynomial time. The high expressive power of ASP can be profitably exploited in AI, which often has to deal with problems of high complexity. For instance, problems in diagnosis and planning under incomplete knowledge are complete for the complexity class Σ_2^P or Π_2^P , and can be naturally encoded in ASP [Baral, 2002; Leone *et al.*, 2001].

*This work was partially supported by the European Commission under projects IST-2002-33570 (INFOMIX) and IST-2001-37004 (WASP), and by FWF (Austrian Science Funds) under projects P16536-N04 and P17212-N04.

[†]Funded by APART of the Austrian Academy of Sciences.

Most of the optimization work on ASP systems has focused on the efficient evaluation of non-disjunctive programs (whose power is limited to NP/co-NP), whereas the optimization of full (disjunctive) ASP programs has been treated in fewer works (e.g., in [Janhunen *et al.*, 2000; Koch *et al.*, 2003]). In particular, we are not aware of any work concerning heuristics for Σ_2^P/Π_2^P -hard ASP programs.

In this paper, we address the following two questions:

- Can the heuristics of ASP systems be refined to deal more efficiently with Σ_2^P/Π_2^P -hard ASP programs?
- On hard Σ_2^P/Π_2^P problems, can ASP systems compete with other AI systems, like QBF solvers?

We define a new heuristic h_{DS} for the (disjunctive) ASP system DLV, aiming at improving the evaluation of Σ_2^P/Π_2^P -hard ASP programs, but maintaining the benign behaviour of the heuristic of DLV on NP problems. We experimentally compare h_{DS} against the DLV heuristic on hard 2QBF instances, showing a clear benefit. We also experiment the competitiveness of ASP w.r.t. QBF solvers on hard problems, indicating that ASP systems are very competitive with QBF systems on Σ_2^P/Π_2^P -hard problems.

2 Answer Set Computation and Heuristics

We first recall the main steps of the computational process performed by ASP systems, in particular the DLV system, which will be used for the experiments.

An answer set program $\mathcal{P}$ in general contains variables. The first step of a computation of an ASP system eliminates these variables, then the following algorithm is invoked:

Function ModelGenerator(I : Interpretation): Boolean;
begin

```
I := DetCons(I);  
if I =  $\mathcal{L}$  then return False; (* inconsistency *)  
if no atom is undefined in I then return IsAnswerSet(I);  
Select an undefined ground atom  $A$  according to a heuristic;  
if ModelGenerator( $I \cup \{A\}$ ) then return True;  
else return ModelGenerator( $I \cup \{\text{not } A\}$ );
```

end;

Roughly, the Model Generator produces some “candidate” answer sets. The stability of each of them is subsequently verified by the function *IsAnswerSet(I)*, which verifies whether the given “candidate” I is a minimal model of the GL-transformed program and outputs the model, if so. *IsAnswer-*

Set(I) returns True if the computation should be stopped and False otherwise.

The function *DetCons()* computes an extension of *I* with the literals that can be deterministically inferred (or the set of all literals $\mathcal{L}$ upon inconsistency). If *DetCons* does not detect any inconsistency, an atom *A* is selected according to a heuristic criterion and *ModelGenerator* is called on $I \cup \{A\}$ and on $I \cup \{\text{not } A\}$. The atom *A* plays the role of a branching variable of a SAT solver.

The heuristic h_{UT} , proposed in [Faber *et al.*, 2001] is currently employed in DLV. It is mostly based on the number of *UnsupportedTrue* (UT) atoms (called MBTs in [Faber *et al.*, 2001]), i.e., atoms which are true in the current interpretation but miss a supporting rule, trying to minimize UT atoms and hence more likely arrive at supported models.

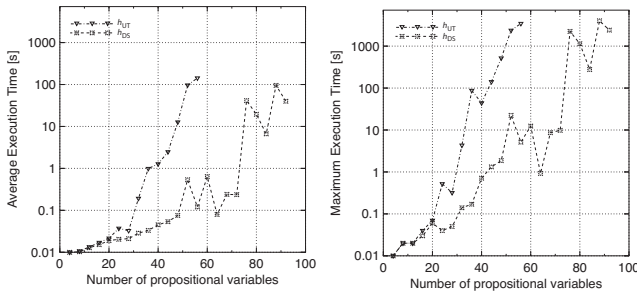
For hard ASP programs (i.e., non-HCF programs [Ben-Eliyahu and Dechter, 1994] – they express Σ_2^P -complete problems under brave reasoning), supported models are often not answer sets. Moreover, answer-set checking is computationally expensive (co-NP), and may consume a large portion of the resources needed for computing an answer set.

We therefore propose the new heuristic h_{DS} , which tries in addition to maximize the *degree of supportedness*, the average number of supporting rules for non-HCF true atoms. Intuitively, if all true atoms have many supporting rules in a model *M*, then the elimination of an atom from the model would violate many rules, and it becomes less likely to find a subset of *M* which is a model of the reduct $\mathcal{P}^M$, disproving that *M* is an answer set. We define h_{DS} as a refinement of the heuristic h_{UT} (i.e., $A <_{h_{UT}} B \Rightarrow A <_{h_{DS}} B$). In this way, h_{DS} keeps the behaviour of the well-assessed h_{UT} on NP problems while, as we will see in Section 3, it sensibly improves over h_{UT} on hard 2QBF problems (Σ_2^P -complete).

3 Comparing h_{UT} vs h_{DS} : Experiments

We generated randomly a data set of 100 2QBF formulas, following [Gent and Walsh, 1999], and used the ASP encoding described in [Leone *et al.*, 2005].

Experiments were performed on a PentiumIV 1500 MHz machine with 256MB RAM running SuSe Linux 9.0. For every instance, we allowed a maximum running time of 7200 seconds (two hours). The results of our experiments are displayed in the following graphs, in which a line stops whenever some instance was not solved within the time limit.



It is clear that the new heuristic h_{DS} outperforms the heuristic h_{UT} in these experiments, advancing the “maximum solvable-size” from 56 up to size 92, and reducing the average execution times of the smaller instances.

4 ASP vs QBF Solvers

The main goal of this paper is to improve the performance of ASP systems for problems located at the second level of the polynomial hierarchy. One may wonder whether, on such Σ_2^P/Π_2^P -hard problems, ASP systems are competitive with other AI systems, like the QBF solvers. In order to give a first answer to this question, we have also performed a comparison with QBF solvers **Quantor** [Biere, 2004], **Semprop** [Letz, 2002], and **yQuaffle** [Zhang and Malik, 2002] on the set of all Σ_2^P - and Π_2^P -complete QBF formulas of the last QBF competition. The results below report the number of instances solved within 660s and show that DLV (with heuristic h_{DS}) generally outperformed the QBF solvers.

	<i>DLV</i> (h_{DS})	<i>Quantor</i>	<i>Semprop</i>	<i>yQuaffle</i>
<i>Robot</i>	27 (84%)	10 (31%)	13 (41%)	17 (53%)
<i>Random</i>	108 (100%)	14 (12%)	90 (83%)	53 (49%)
<i>Tree</i>	2 (100%)	2 (100%)	2 (100%)	2 (100%)
<i>KPH</i>	1 (100%)	1 (100%)	1 (100%)	1 (100%)
<i>Total</i>	137 (96%)	26 (18%)	105 (73%)	72 (50%)

References

- [Baral, 2002] C. Baral. *Knowledge Representation, Reasoning and Declarative Problem Solving*. CUP, 2002.
- [Ben-Eliyahu and Dechter, 1994] R. Ben-Eliyahu and R. Dechter. Propositional Semantics for Disjunctive Logic Programs. *AMAI*, 12:53–87, 1994.
- [Biere, 2004] A. Biere. Resolve and Expand. 2004. SAT’04.
- [Eiter *et al.*, 1997] T. Eiter, G. Gottlob, and H. Mannila. Disjunctive Datalog. *ACM TODS*, 22(3):364–418, 1997.
- [Faber *et al.*, 2001] W. Faber, N. Leone, and G. Pfeifer. Experimenting with Heuristics for Answer Set Programming. In *IJCAI 2001*, pp. 635–640.
- [Gent and Walsh, 1999] I. Gent and T. Walsh. The QSAT Phase Transition. In *AAAI*, 1999.
- [Janhunen *et al.*, 2000] T. Janhunen, I. Niemelä, P. Simons, and J.-H. You. Partiality and Disjunctions in Stable Model Semantics. In *KR 2000*, 12-15, pp. 411–419.
- [Koch *et al.*, 2003] C. Koch, N. Leone, and G. Pfeifer. Enhancing Disjunctive Logic Programming Systems by SAT Checkers. *Artificial Intelligence*, 15(1–2):177–212, 2003.
- [Leone *et al.*, 2001] N. Leone, R. Rosati, and F. Scarcello. Enhancing Answer Set Planning. In *IJCAI-01 Workshop on Planning under Uncertainty and Incomplete Information*, pp. 33–42, 2001.
- [Leone *et al.*, 2005] N. Leone, G. Pfeifer, W. Faber, T. Eiter, G. Gottlob, S. Perri, and F. Scarcello. The DLV System for Knowledge Representation and Reasoning. *ACM TOCL*, 2005. To appear.
- [Letz, 2002] R. Letz. Lemma and Model Caching in Decision Procedures for Quantified Boolean Formulas. In *TABLEAUX 2002*, pp. 160–175, Denmark, 2002.
- [Lifschitz, 1999] V. Lifschitz. Answer Set Planning. In *ICLP’99*, pp. 23–37.
- [Zhang and Malik, 2002] L. Zhang and S. Malik. Towards a Symmetric Treatment of Satisfaction and Conflicts in Quantified Boolean Formula Evaluation. In *CP 2002*, pp. 200–215, NY, USA, 2002.

Incremental Diagnosis of Discrete- Event Systems

Alban Grastien
Irisa – Université Rennes 1
Rennes – France
agrastie@irisa.fr

Marie-Odile Cordier
Irisa – Université Rennes 1
Rennes – France
cordier@irisa.fr

Christine Largouët
University of New Caledonia
Noumea – New Caledonia
largouet@univ-nc.nc

1 Introduction

It is established that diagnosing dynamical systems, represented as discrete-event systems amounts to finding what happened to the system from existing observations. In this context, the diagnostic task consists in determining the trajectories (a sequence of states and events) compatible with the observations. The diagnosis is generally defined as resulting from the synchronization of the automaton modelling the behavior of the system with the automaton that represents the observations sent by the system during the diagnosis period.

In this article, we are interested in avoiding the global computation by slicing the automaton of observations and building the diagnosis on successive slices of observations.

We introduce the concept of *automata chain* to represent an automaton by a sequence of automata slices. We then provide the properties such an automata chain has to satisfy to be a *correct slicing* and define a *reconstruction* operation to get the global automaton back. We demonstrate that, given a correct slicing of the observations, we can compute a global and correct diagnosis from the reconstruction of a diagnosis automata chain.

2 Automata chain

In this section we introduce the concept of *automata chain* whose goal is to enable us to slice an automaton into pieces. We use the well-known definitions of automata (with I the set of initial states and F the set of final states), path and trajectories (paths between an initial and a final state). The main property of an automata chain (first bullet) is that a state is not allowed to appear in two distinct automata of the chain, except if it is a frontier state between two successive automata, i.e it is a final state of the former and an initial state of the later. More generally, if a state belongs to the i th automata and also to the j th automata, with $j > i$, it appears also in all the automata between the i th and the j th as a frontier state.

Definition 1 (Automata chain) A sequence of automata $(A^1, \dots, A^n)$ with $A^i = (Q^i, E, T^i, I^i, F^i)$ is called automata chain, and denoted $\mathcal{E}_A$, if:

- $\forall i, j, j > i, \forall q, q \in Q^i \cap Q^j \Rightarrow q \in F^i \wedge q \in I^{i+1}$,
- $\forall i, j, \forall q, q', \text{ if } \{q, q'\} \subseteq Q^i \cap Q^j \text{ then } \forall p, \text{ path of } A^i \text{ between } q \text{ and } q', p \text{ is also a path of } A^j$.

An automata chain is given in Figure 1.

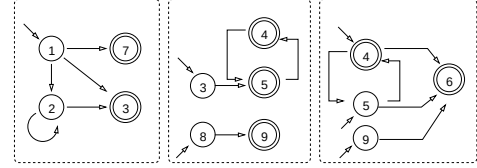


Figure 1: Chain of three automata

Let $\mathcal{E}_A$ be an automata chain $(A^1, \dots, A^n)$. A trajectory of $\mathcal{E}_A$ is defined as being the ordered (from 1 to n) concatenation of n trajectories, one for each automaton. For instance, the path going from state 1 to state 6 through the states 3 and 5 is a trajectory of the automata chain of Figure 1. Conversely, the path going from state 8 to 6 through 9 is not a trajectory.

Definition 2 (Correct slicing) Let A be an automaton and $\mathcal{E}_A = (A^1, \dots, A^n)$ an automata chain. $\mathcal{E}_A$ is a correct slicing of A , denoted $\mathcal{E}_A = \text{Sli}(A)$, iff the set of trajectories of $\mathcal{E}_A$ is equal to the set of trajectories of A .

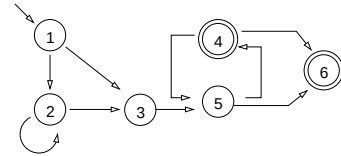


Figure 2: The chain in Figure 1 is a correct slicing of this automaton obtained by reconstruction (see Def. 3) of the chain

Definition 3 (Automaton reconstruction) Let

$\mathcal{E}_A = (A^1, \dots, A^n)$ be an automata chain with $A^i = (Q^i, E, T^i, I^i, F^i)$. We call reconstruction of the chain $\mathcal{E}_A$, the simplified automaton obtained from $(Q^1 \cup \dots \cup Q^n, E, T^1 \cup \dots \cup T^n, I^1, F^n)$.

Theorem 1 Let A be an automaton and $\mathcal{E}_A$ an automata chain. If $\mathcal{E}_A$ is a correct slicing of A , then A is obtained by the reconstruction of $\mathcal{E}_A$. The proof is not given here.

The reconstruction of $\mathcal{E}_A$ is denoted $\text{Sli}^{-1}(\mathcal{E}_A)$. If $\mathcal{E}_A$ is a correct slicing of A , then $A = \text{Sli}^{-1}(\mathcal{E}_A)$.

We call *prefix-closed automaton* of A (resp. *suffix-closed automaton* of A) denoted A^+ (resp. A^-) the automaton A whose all states are final (resp. initial). We denote $A^\#$, the automaton which is both prefix-closed and suffix-closed.

Definition 4 (Automata chain synchronization) We call synchronization of an automata chain $\mathcal{E}_A = (A^1, \dots, A^n)$ with an automaton M the sequence denoted $\mathcal{E}_A \otimes M$ defined by: $\mathcal{E}_A \otimes M = (A^1 \otimes M^+, A^2 \otimes M^\#, \dots, A^{n-1} \otimes M^\#, A^n \otimes M^-)$.

Theorem 2 Let $\mathcal{E}_A$ be an automata chain and M an automaton, then $\mathcal{E}_A \otimes M$ is an automata chain and $\mathcal{E}_A \otimes M$ is a correct slicing of $Sli^{-1}(\mathcal{E}_A) \otimes M$. The proof is not given.

3 Diagnosis by slices

Let us first recall the definitions used in the domain of discrete-event systems diagnosis where the model of the system is traditionally represented by an automaton. The model of the system describes its behaviour and the trajectories of Mod represent the evolutions of the system.

Definition 5 (Model) The model of the system, denoted Mod , is an automaton $(Q^{Mod}, E^{Mod}, T^{Mod}, I^{Mod}, F^{Mod})$. I^{Mod} is the set of possible states at t_0 . All the states of the system may be final, thus $F^{Mod} = Q^{Mod}$, $Mod^+ = Mod$ and $Mod^\# = Mod^-$.

Let us turn to observations and diagnosis definitions. Generally, we don't know the total order on the observations emitted by the system. Consequently, the observations are represented by an automaton, each trajectory of which represents a possible order of emission of the observations.

Definition 6 (Observations) The observations, denoted Obs_n , is an automaton describing the observations emitted by the system during the period $[t_0, t_n]$.

Definition 7 (Diagnosis) The diagnosis, denoted Δ_n , is an automaton describing the possible trajectories on the model of the system compatible with the observations sent by the system during the period $[t_0, t_n]$.

The diagnosis is defined (see [Sampath et al., 1996]) as resulting from the synchronization of the automata representing the system model and the observations: $\Delta_n = Mod \otimes Obs_n$. Using Theo. 2, it is possible to compute the diagnosis by slices. The idea is to compute diagnosis slices, corresponding to observations slices. The global diagnosis can then be reconstructed from the diagnosis automata chain which is obtained.

Definition 8 (Diagnosis by slices - Diagnosis slice) Let Mod be the system model. Let $\mathcal{E}_{Obs_n} = (Obs^1, \dots, Obs^n)$, be a correct slicing of Obs_n , the observations emitted during the period $[t_0, t_n]$. The synchronization of $\mathcal{E}_{Obs_n}$ with Mod , i.e. $\mathcal{E}_{Obs_n} \otimes Mod = (Obs^1 \otimes Mod, Obs^2 \otimes Mod^\#, \dots, Obs^n \otimes Mod^\#)$ is the diagnosis by slices of the system. It can be denoted by the diagnosis automata chain $(\Delta^1, \dots, \Delta^n)$, where Δ^i is called the i th diagnosis slice of the system.

It can be proved (using Theo. 2) that the diagnosis by slices of a system, here $\mathcal{E}_{Obs_n} \otimes Mod$, correctly represents the diagnosis computed on the global observations since the reconstruction of $\mathcal{E}_{Obs_n} \otimes Mod$ equals the global diagnosis:

Result 1 $\Delta_n = Mod \otimes Obs_n = Sli^{-1}(\mathcal{E}_{Obs_n} \otimes Mod)$

4 Incremental diagnosis

In the diagnosis by slices as presented above, the i th diagnosis slice, Δ^i , is computed independently from the others, by synchronizing the i th observation slice from the chain $\mathcal{E}_{Obs_n}$, Obs^i , with the system model $Mod^\#$. In the incremental synchronization, noted $\odot$, (see [Grastien et al., 2005] for more details), the set of initial states of an automaton of the chain is restricted by the set of final states of its predecessor.

Theorem 3 Let $\mathcal{E}_A$ be an automata chain and M be an automaton. We have $Sli^{-1}(\mathcal{E}_A \odot M) = Sli^{-1}(\mathcal{E}_A \otimes M)$. The proof is not given.

Provided that $\mathcal{E}_{Obs_n} = (Obs^1, \dots, Obs^n)$ is a correct slicing of Obs_n we have: $\Delta_n = Sli^{-1}(\mathcal{E}_{Obs_n} \odot Mod)$.

We note $\forall i$, $\mathcal{E}_{Obs_i} = (Obs^1, \dots, Obs^i)$, the automata chain of the first i observations automata. Let $i < n$, and $\mathcal{E}_{\Delta_i} = (\Delta^1, \dots, \Delta^i)$ the automata chain resulting from the incremental synchronisation of $\mathcal{E}_{Obs_i}$ with the system model Mod . We can incrementally compute $\mathcal{E}_{\Delta_{i+1}} = \mathcal{E}_{Obs_{i+1}} \odot Mod$ as follows:

Result 2 $\mathcal{E}_{\Delta_{i+1}} = (\Delta^1, \dots, \Delta^i, \Delta^{i+1})$ where Δ^{i+1} is the automaton $(Obs^{i+1} \otimes Mod^\#)$ whose initial states are restricted by the set of final states of Δ^i .

Let Obs_i be the automaton provided by the reconstruction operation on $\mathcal{E}_{Obs_i}$, and let Δ_i be the reconstruction of $\mathcal{E}_{\Delta_i}$.

Result 3 $\Delta_i = Obs_i \otimes Mod$.

5 Conclusion

In this paper, we formalized the computation by slices of diagnosis for discrete-event systems. We introduced and defined the concept of automata chain that enables us to handle slices of observations and slices of diagnosis rather than global observations and global diagnosis. In the diagnosis by slices, the i th diagnosis slice, Δ^i , is computed independently from the others. In [Grastien et al., 2005], we show that this result can be instantiated to the case where the observation automaton is sliced according to time, according to temporal windows.

Our study exhibits the (non trivial) correctness properties that the observation slicing, in an automata chain, has to satisfy in order to guarantee the completeness of the diagnosis computation. This first step is then essential before considering the incrementality of on-line diagnosis computation.

References

- [Grastien et al., 2005] A. Grastien, M.-O. Cordier, and C. Largouët. Incremental diagnosis of discrete event systems. In *16th International Workshop on Principles of Diagnosis (DX-2005)*, 2005.
- [Sampath et al., 1996] M. Sampath, R. Sengupta, S. Lafortune, K. Sinnamohideen, and D.C. Teneketzis. Failure diagnosis using discrete-event models. In *IEEE Trans. on Control Systems Technology (CST-96)*, pages 105–124, 1996.

Reflection Patterns for Interactive Knowledge Capture*

Jihie Kim

Information Sciences Institute, University of Southern California
4676 Admiralty Way, Marina del Rey, CA 90292
jihie@isi.edu

Abstract

Current knowledge acquisition tools have limited understanding of how users enter knowledge and how acquired knowledge is used, and provide limited assistance in organizing various knowledge authoring tasks. In this paper, we present a novel extension to existing knowledge acquisition tools where the system 1) captures the episodes of knowledge acquisition and knowledge use through a set of declarative *reflection patterns* 2) performs assessment on how to improve the future knowledge acquisition and knowledge use based on captured episodes, and 3) provides assistance to the users by combining the assessment results.

1 Introduction

Knowledge acquisition (KA) remains a key challenge to knowledge-based AI applications. There have been increasing interests in supporting end users (i.e., ordinary users who do not have computer science background) directly enter complex problem solving knowledge on how to perform tasks. Although these techniques have been applied in building sizable knowledge bases in some cases, detailed analyses of the user interactions reveal that existing KA tools provide limited assistance in organizing and executing various knowledge authoring tasks [Kim and Gil 2000]. Systems do not reflect on how users perform KA tasks, how acquired knowledge (called *k-items*) is tested and used, and how future k-items should be built based on past experience. Users themselves have to keep track of the past mistakes, current status, potential new problems, and progresses made over time in order to decide the best options among possible courses of actions. The challenges in supporting these reflective capabilities include:

- The system should be *self-aware*, accessing and reasoning on interesting aspects of knowledge acquisition and knowledge use in assisting users.
- The system should relate the current situation to similar situations in the past and assess the levels of confidence in pursuing alternative options based on the relations.

- The system should recognize dynamic changes in the problem solving and decide how to guide users in modifying and using relevant k-items.

2 ECHO: Reflection Patterns for Interactive Knowledge Acquisition

In this paper, we present a novel extension to existing KA tools where the system adds an additional layer to existing tools and explicitly keeps track of knowledge acquisition and knowledge use episodes through a set of declarative *reflection patterns*. Reflection patterns allow the system to *be aware of* interesting knowledge acquisition and knowledge use episodes (called *meta-events*) that the system can make use of in assisting users. Each meta-event is a sequence of basic knowledge acquisition and knowledge use events such as the user ignored a suggestion and then the problem solving failed. The system *assesses the levels of confidence* in providing a suggestion based on its supporting and opposing meta-events captured in the reflection patterns. Any changes that are noticed (e.g. confident k-item became inconsistent with problem solving results) and associated k-item modifications are explicitly captured in the reflection patterns and are used in guiding the user.

We have built a system called Echo (mEta-Cognitive History analysis and Organization) that provides these capabilities. Echo dynamically captures a set of meta-events by matching the reflection patterns against the episodes of how different types of k-items are acquired, modified and used over time. The captured meta-events affect how the suggestions to the user are provided (i.e. how confident it is on the suggestion) as well as the content of the suggestions.

A prototype system has been developed for a domain of interactive scheduling where the user incrementally builds scheduling constraints and the user entered constraints assist users during scheduling. Since scheduling constraints can change over time, the system should be able to assist users in making associated modifications. The details of the system are described in (Kim 2005).

In developing reflection patterns and determining which meta-events the system needs to keep track of we have analyzed typical ways in which failures can occur in interactive knowledge capture and use. The reflection patterns are built help users avoid those failures. Figure 1 shows the paths

* The support of IJCAI, Inc. is acknowledged.

that Echo exploit currently. Depending on a particular need of a given KA system and its undesirable paths, we can include other meta-events in the reflection patterns and adjust the assistance provided by the system.

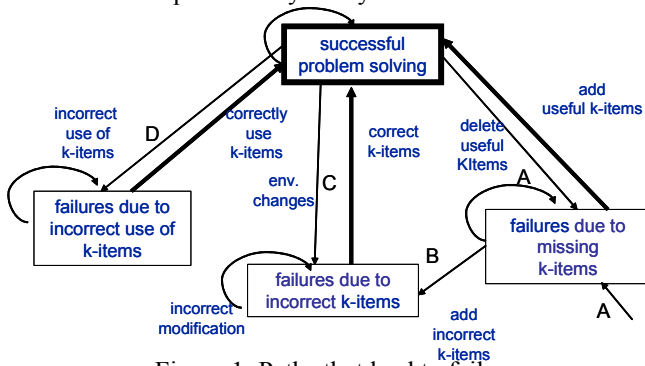


Figure 1: Paths that lead to failures.

Existing KA systems make use of these implicitly. That is, consideration of the above failures not explicit in the design and they are reflected in the implementation of the tool. Example-based validation approaches refine k-items collectively, and cannot tell ‘when’ and ‘how’ k-items can be improved and ‘how’ they should be used. The above steps can be enhanced by exploiting related meta-events. In particular, we make use of predictive assessment of similar situations and assessment on dynamic changes including how k-items can be improved.

Algorithm

- A.** assess likeliness of potential failures due to missing k-items by collecting and relating meta-events of
 - steps failed without k-item assistance
 - how similar steps failed without k-item assistance
 - how similar steps were assisted by k-items
 and produce suggestions to create new k-items based on
 - assessments on how to create k-items (see E)
- B.** assess likeliness of potential failures due to incorrect k-item definitions by collecting and relating meta-events of
 - inconsistencies with the current problem solving result
 - the degree of inconsistencies with valid past results
 - inconsistencies with confident k-items
 and produce suggestions to modify k-items based on
 - assessments on how to modify k-items (see E)
- C.** assess likeliness of potential failures due to changes by collecting and relating the meta-events of
 - how similar k-items were modified in similar situations
 - modification in similar situations led to successes
 - the user agreed to modify k-items in similar situations
 - the user specified when to modify k-items
 and produce suggestions to modify k-items based on
 - assessments on how to modify k-items (see E)
- D.** assess likeliness of potential failures when k-items are ignored by collecting and relating the meta-events of
 - consistencies between k-items and current result
 - consistencies between k-items with past valid results
 - inconsistent decisions resulting in failures
 - consistent decisions resulting in successes
 and produce suggestions to use k-items based on the assessment
- E.** assess how to create or modify k-items by collecting and relating the meta-events of

- incorrect definitions or modifications of similar k-items resulting in further failures
 - k-item definitions and modifications resulting in successes
- The above algorithm is represented by a set of declarative reflection patterns described in [Kim 2005].

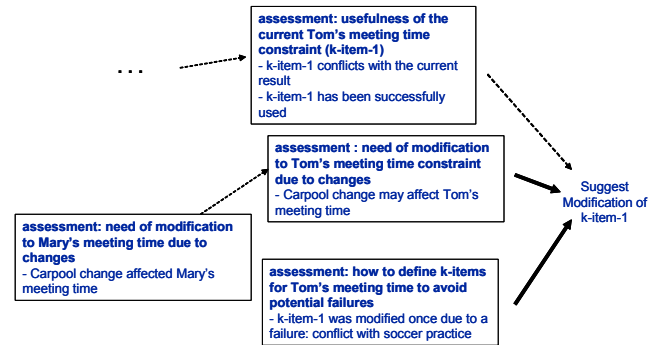


Figure 2: relations between assessments

Figure 2 shows how different assessments assist a k-item modification. In this case, Echo predicts that k-item-1 needs to be modified because it is less confident about the usefulness of the k-item (due to the current conflict) and more confident on potential changes needed. The suggestions include how the system predicts the k-item can be modified based on relevant past modifications (i.e. consideration of soccer practice time).

3. Preliminary Results

We performed a preliminary evaluation of Echo with a set of synthetic scenarios with varying user interactions and mistakes. We compare two KA systems where both of them use the same episodes of problem solving and problem changes but one of them was enhanced with Echo's reflection patterns. The results show that with Echo, the KA system can reduce the number of incorrect suggestions and the number of problem solving failures.

Average per scenario	With reflection patterns	Without reflection patterns
Avg # decisions assisted with k-items(# total decisions)	24.0 (84.0)	22.03 (82.03)
Avg # wrong user decisions	8	8
Avg # additional user mistakes	2.73	5.97
Avg # k-items creation or modification	4.87	4.34
Avg # sugg (Avg # confident sugg)	13.83 (10.27)	13.0 (0)
Avg # wrong sugg	0.90	3.07
Avg # failures	4.10	6.73

Table 1. With and without reflection patterns

References

- [Kim 2005] Reflection Patterns for Interactive Knowledge Capture, <http://www.isi.edu/~jihie/papers/echo-RP.pdf>.
- [Kim & Gil 2000] Acquiring Problem-Solving Knowledge from End Users: Putting Interdependency Models to the Test. *Proceedings of AAAI-2000*.

A Unified Framework of Propositional Knowledge Base Revision and Update Based on State Transition Models

Yasuo Kudo

Department of Computer Science
and Systems Engineering
Muroran Institute of Technology
Mizumoto 27-1, Muroran 050-8585, Japan
kudo@csse.muroran-it.ac.jp

Tetsuya Murai

Graduate School of Information
Science and Technology
Hokkaido University
Kita 14, Nishi 9, Sapporo 060-0814, Japan
muraiko@main.ist.hokudai.ac.jp

Abstract

Belief revision and belief update are two of the most basic types of belief change operations. We need to select either revision or update when we accept new information into the current belief, however, such decision making has not been considered. In this paper, we propose a unified framework of revision and update based on state transition models that enable us to do such decision making. This framework provides a hybrid operation of revision and update, called acceptance.

1 Introduction

Belief revision [Alchourfón *et al.*, 1985] and belief update [Katsuno and Mendelzon, 1992] are two of the most basic types of belief change operations. When we accept new information into the current belief, we need to estimate whether the new information represents more reliable information about a static world, or it reports some (unspecified) change occurred in a dynamic world. This estimation causes decision making to select either revision to fix some errors in the current belief, or update to reflect some change into the belief.

In this paper, we introduce a *state transition model* as a unified framework of belief revision and belief update. The state transition model illustrates *prior knowledge* for estimation about the background of the new information, and also provides a *selection mechanism* for the decision making. Using the state transition model, we also propose a hybrid operation, called *acceptance*, of revision and update.

2 Knowledge Base Revision and Update

Katsuno and Mendelzon [Katsuno and Mendelzon, 1991] have rephrased the AGM postulates for revision [Alchourfón *et al.*, 1985], and have provided a possible worlds characterization of revision. For a given propositional sentence KB that represents the current knowledge base, and a propositional sentence α that represents new information about a static world, $KB \circ \alpha$ denotes a *revision* of KB by α . Revision operators are characterized by postulates (R1) – (R6).

(R1) $KB \circ \alpha \models \alpha$.

(R2) If $KB \wedge \alpha$ is satisfiable, then $KB \circ \alpha \equiv KB \wedge \alpha$.

(R3) If α is satisfiable, then $KB \circ \alpha$ is also satisfiable.

(R4) If $\models \alpha \leftrightarrow \beta$, then $KB \circ \alpha \equiv KB \circ \beta$.

(R5) $(KB \circ \alpha) \wedge \beta \models KB \circ (\alpha \wedge \beta)$.

(R6) If $(KB \circ \alpha) \wedge \beta$ is satisfiable, then $KB \circ (\alpha \wedge \beta) \models (KB \circ \alpha) \wedge \beta$.

Katsuno and Mendelzon [Katsuno and Mendelzon, 1992] have proposed a general characterization of belief update. For a given knowledge base KB and a sentence α that represents new information by some (unspecified) change in a dynamic world, $KB \diamond \alpha$ denotes an *update* of KB by α . Update operators are characterized by postulates (U1) – (U8).

(U1) $KB \diamond \alpha \models \alpha$.

(U2) If $KB \models \alpha$, then $KB \diamond \alpha \equiv KB$.

(U3) If both KB and α are satisfiable, then $KB \diamond \alpha$ is also satisfiable.

(U4) If $\models \alpha \leftrightarrow \beta$, then $KB \diamond \alpha \equiv KB \diamond \beta$.

(U5) $(KB \diamond \alpha) \wedge \beta \models KB \diamond (\alpha \wedge \beta)$.

(U6) If $KB \diamond \alpha \models \beta$ and $KB \diamond \beta \models \alpha$, then $KB \diamond \alpha \equiv KB \diamond \beta$.

(U7) If KB is complete, then $(KB \diamond \alpha) \wedge (KB \diamond \beta) \models KB \diamond (\alpha \vee \beta)$.

(U8) $(KB_1 \vee KB_2) \diamond \alpha \equiv (KB_1 \diamond \alpha) \vee (KB_2 \diamond \alpha)$.

3 State Transition Models

We introduce a *state transition model* as prior knowledge for estimation about the background of the new information. State transition models are closely related to *event models* for abductive characterization of belief update [Boutilier, 1996].

Definition 1 A *state transition model* is a triple $\langle W, T, \preceq \rangle$, where W is a non-empty set of possible worlds, $T \subseteq W \times W$ is a non-empty set of state transitions, and $\preceq$ is a total preorder on T .

$(x, y) \in T$ is a state transition from the *starting point* x to the *terminal* y . The intuitive meaning of $(x, y) \in T$ is “we know that some (unspecified) change at x may cause the situation y ”. The total preorder $\preceq$ illustrates relative plausibility of state transitions. If we have $(x, y) \preceq (u, v)$, we interpret that (x, y) is at least as plausible as is (u, v) . Using the state transition model, we can represent the following two criteria about relative plausibility of possible worlds:

- Comparing (x, y) and (u, v) such that $x \neq u$: the relative plausibility of x and u as the actual world.
- Comparing (x, y) and (x, v) with the same starting point x : the relative plausibility of y and v as the result of some change at x .

Definition 2 For any $R \subseteq T$, we define $Sp(R) \subseteq W$ and $Ter(R) \subseteq W$ as follows, respectively:

$$Sp(R) = \{x \in W \mid \exists(x, y) \in R\}, \quad (1)$$

$$Ter(R) = \{y \in W \mid \exists(x, y) \in R\}. \quad (2)$$

If $R = \emptyset$, we define $Sp(R) = Ter(R) = \emptyset$.

For any $X \subseteq W$ and $Y \subseteq W$, we define $(X, Y) \subseteq T$ by:

$$(X, Y) = \{(x, y) \in T \mid x \in X, y \in Y\}. \quad (3)$$

If either $X = \emptyset$ or $Y = \emptyset$, we define $(X, Y) = \emptyset$. For any singleton $\{w\} \subseteq W$, we abbreviate $(\{w\}, X)$ and $(X, \{w\})$ as (w, X) and (X, w) , respectively.

Definition 3 Let STM be a state transition model. STM is called centered iff the following two conditions hold:

1. For each world $w \in W$, $(w, w) \in T$.
2. For any non-empty subset $X \subseteq W$ and any $w \in X$, the loop (w, w) is the minimum element in (w, X) , that is, if $(w, x) \in (w, X)$ and $x \neq w$, then $(w, w) \prec (w, x)$.

Using the given state transition model, the current knowledge base KB is semantically characterized by starting points of the most plausible state transitions in T .

Definition 4 Let STM be a state transition model. A knowledge base KB induced by STM is a propositional sentence such that

$$\|KB\| = Sp\left(\min_{\preceq} T\right). \quad (4)$$

4 Acceptance: A Hybrid Operation of Revision and Update

Using the given state transition model that illustrates prior knowledge for estimation, we provide a selection mechanism to decide we use either revision or update when we accept new information. *Explainability* of the new information we define below is the key concept of such decision making.

Definition 5 Let STM be a state transition model, and KB is a knowledge base induced by STM . A sentence α is explainable by STM iff $(w, \|\alpha\|) \neq \emptyset$ for all $w \in \|KB\|$.

We have the following simple selection strategy by explainability of the new information: Let $CSTM = \langle W, T, \preceq \rangle$ be a centered state transition model, and KB be the current knowledge base induced by $CSTM$. For any observation α ,

1. If α is explainable by $CSTM$, we regard α as the new information by some change, and update KB by α .
2. Otherwise, we regard α as more reliable information about a static world, and revise KB by α .

According to the selection strategy, we introduce a hybrid operation of revision and update, called *acceptance*, based on the given centered state transition model $CSTM$. We use

$KB \triangleleft \alpha$ to denote the result of acceptance of α into KB . The symbol $\triangleleft$ is called an *acceptance operator*. We intend to have either $KB \triangleleft \alpha \equiv KB \circ \alpha$ or $KB \triangleleft \alpha \equiv KB \diamond \alpha$ based on explainability of α by $CSTM$.

Theorem 1 Let $CSTM$ be a centered state transition model, KB be a knowledge base induced by $CSTM$, and $\triangleleft$ be an acceptance operator defined by the following equation:

$$\|KB \triangleleft \alpha\| = \begin{cases} \bigcup_{w \in \|KB\|} \left\{ Ter\left(\min_{\preceq}(w, \|\alpha\|)\right) \right\} & \text{if } \alpha \text{ is explainable by } CSTM, \\ Sp\left(\min_{\preceq}(\|\alpha\|, W)\right) & \text{otherwise.} \end{cases} \quad (5)$$

Then, for any sentence α ,

1. If α is explainable by $CSTM$, then $\triangleleft$ satisfies postulates (U1) – (U4), the following weakened (U5):
(U5w) If $\alpha \wedge \beta$ is explainable, then $(KB \diamond \alpha) \wedge \beta \models KB \diamond (\alpha \wedge \beta)$,
postulates (U6), (U7), and a postulate (U9) proposed by Boutilier [Boutilier, 1996]:
(U9) If KB is complete, $(KB \diamond \alpha) \not\models \neg\beta$ and $KB \diamond \alpha \models \gamma$, then $KB \diamond (\alpha \wedge \beta) \models \gamma$.
2. If α is not explainable by $CSTM$, then $\triangleleft$ satisfies postulates (R1) – (R6).

The acceptance operator $\triangleleft$ is well-defined as a revision operator $\circ$ when α is not explainable by $CSTM$. On the other hand, $\triangleleft$ does not satisfy (U8) when α is explainable. However, in equation (5), α is independently evaluated in each possible world $w \in \|KB\|$, therefore the idea of semantic characterization of KM update [Katsuno and Mendelzon, 1992] is illustrated in this framework.

Acknowledgement

We would like to thank the anonymous reviewers for helpful comments and suggestions.

References

- [Alchourrón et al., 1985] C. Alchourrón, P. Gärdenfors, and D. Makinson. On the logic of theory change: Partial meet contraction and revision functions. *Journal of Symbolic Logic*, 50:510–530, 1985.
- [Boutilier, 1996] C. Boutilier. Abduction to plausible causes : an event - based model of belief update. *Artificial Intelligence*, 83:143–166, 1996.
- [Katsuno and Mendelzon, 1991] H. Katsuno and A. O. Mendelzon. Propositional knowledge base revision and minimal change. *Artificial Intelligence*, 52(3):263–294, 1991.
- [Katsuno and Mendelzon, 1992] H. Katsuno and A. O. Mendelzon. On the difference between updating a knowledge base and revising it. In P. Gärdenfors, editor, *Belief Revision*, pages 183–203. Cambridge University Press, 1992.

Redesign Support Framework based on Hierarchical Multiple Models

I. López-Arévalo¹, A. Rodríguez-Martínez², A. Aldea³, R. Bañares-Alcántara⁴, and L. Jiménez⁵

¹ Department of Computer Engineering and Mathematics,

² Department of Chemical Engineering, University Rovira i Virgili, Spain.

³ Department of Computing, Oxford Brookes University, United Kingdom.

⁴ Department of Engineering Science, University of Oxford, United Kingdom.

⁵ Department of Chemical Engineering and Metallurgy, University of Barcelona, Spain.

Abstract

A redesign support framework for complex technical processes is described in this paper. This framework employs a multi-model hierarchical representation of the process to be redesigned together with a case-based reasoning engine that helps us to decide the elements of the process that should be modified. This framework has been tested in the chemical engineering domain.

1 Introduction

In this paper a redesign support framework is proposed. This framework integrates model-based reasoning and case-based reasoning techniques. The original process is modelled hierarchically exploiting means-end and part-whole concepts of MFM [Larsson, 1996] and Multi-modelling [Chittaro *et al.*, 1993] approaches. A CBR system is used to obtain alternative process sections, which can be adapted into the original process. Therefore, this framework allows us to model the process; to identify process components suitable for redesign; to obtain alternative components and; finally to adapt these components into the original process. This procedure can be seen as a reverse engineering activity where abstract models at different levels are generated from a detailed description of the existing process to reduce the complexity of the process. The framework has been applied on the chemical engineering domain, by means of three prototypes: HEAD, AHA!, and RETRO.

2 The Redesign Framework

The proposed framework (see Figure 1) consists of four main stages, which can be summarised as follows:

1. **Design-description acquisition.** This stage consists of:
 - *Data acquisition.* Knowledge about the structure and behaviour of all components of the process is automatically extracted from a numerical simulator. Therefore, human intervention is avoided and the simulator ensures that data is consistent.
 - *Functional identification.* This is the modelling stage. The data extracted from the simulator is used to auto-

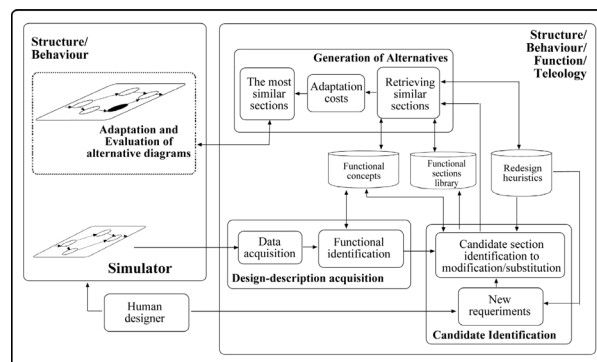


Figure 1. The redesign framework.

matically generate the hierarchical models. Based on the data extracted from the simulator, the functional and teleological models can be inferred.

A *unit* that encapsulates these four types of knowledge is generated for each physical equipment. Furthermore, an ontology of functions and equipments (not necessarily with formal specification) have to be specified. This ontology is used to define a priority order of functions. Given the priority of functions of *units* and on their goals, it is possible to identify incrementally the functional sections of the process. The grouping of functions generates *meta-units* and identifies a functional section. *Units/meta-units* with lower priority functions are “absorbed” by *units/meta-units* with higher priority functions. This forms a tree of functions of the process. The connections between such functions denote the causal relations between the process variables and the goals.

2. **Candidate identification.** The aim of this stage is to identify the *units* or *meta-units* that have to be modified to fulfil the new redesign objectives. Therefore, the design description of the process and the new set of specifications that the process must satisfy are required. The process variable to focus on, according to the new requirements of the process, must be set. Then a diagnosis algorithm is used to identify the *units/meta-units* that affect such process variables.

The diagnosis algorithm [Larsson, 1996] applies causal reasoning to identify the *units/meta-units* where the process variables do not achieve the needed values. No simulation is required; the algorithm uses the ontological assumptions and the values of such variables. As result, a small list of the most possible “faulty” *units/meta-units* is obtained. Based on any identified *unit/meta-unit*, their corresponding “cause” and “consequence” *units* are identified. “Cause” *unit/meta-unit* is the *unit/meta-unit* that provides the actual operational conditions to the involved process variables in the function of the *unit/meta-unit* of interest. “Consequence” *unit/meta-unit* is the *unit/meta-unit* affected by the operational conditions given by the *unit/meta-unit* of interest. The “causes” and “consequences” *units/meta-units* are important in the adaptation and evaluation stage.

3. **Generation of alternatives.** From the identified *unit/meta-unit* in the previous stage, similar *units/meta-units* can be suggested. New *units/meta-units* are generated from others similar processes. This can be achieved by a CBR system. The retrieving stage of the CBR system corresponds to this stage, and the rest of stages refer to the two last stages of the redesign framework (the adaptation and evaluation stages).

We apply a hierarchical case-based reasoning approach. Then each *unit* and *meta-unit* is considered a ground case and an abstract case respectively. A ground case is a case located at the lowest level in the hierarchy and represents a specific *unit*. An abstract case is a case represented at a higher level of abstraction and corresponds to a *meta-unit*. Thus the *unit* or *meta-unit* identified in the previous stage is the target case. Similar *units/meta-units* (source cases) are retrieved from a case library according to its similarity respect to the target case. Numerical, symbolic and hierarchical similarity measures are used. As result, a set of the most similar *units/meta-units* is obtained. The human designer may test them in the adaptation and evaluation stage.

4. **Adaptation and evaluation.** The reuse, revision, and retention stages of the CBR cycle correspond to the adaptation and evaluation stages in the redesign framework. Retention is not considered an explicit stage in the framework but is carried out.

The most similar *units/meta-units* proposed in the previous stage must be adapted and evaluated in the process of interest until obtain an appropriate alternative process design. Both stages are not systematised in the redesign framework. Since the aim of the redesign framework is to deal with complex technical process, the adaptation and revision of the most similar cases require complex simulations. The human designer must carry out them manually by means of the simulator used in the data acquisition

To facilitate the adaptation, an adaptation cost is computed to suggest the “adaptability” of the chosen *unit/meta-unit*. This adaptation cost is based on the differences between the chosen *unit* (source case) and the cause and consequence *units/meta-units* identified with the diagnosis algorithm. Thus, the cost is a normalised value denoting the difference on the values of the process variables involved in the performance of the *unit* and the values of the process variables involved in the performance of the neighbour *units/meta-units*. The adaptation cost will have a value between 0 and 1. Values close to 0 means that the adaptation is difficult.

Note that this framework does not redesign processes either automatically or autonomously. The aim is to support human designers to understand a process and facilitate the redesign activities. Also any assumption on some specific domain has been considered, but the domain must allow a well-defined structure on functions.

3 Evaluation and conclusions

This framework has been implemented and applied to the chemical engineering process domain over 20 chemical processes (consequently the number of ground and abstract cases is higher). Interesting results have been. The functional ontology developed includes concepts from the well-known chemical process design methodologies developed by Douglas [Douglas, 1988] and Turton [Turton *et al.*, 1998]. The numerical simulators employed were Hysys and Aspen.

The central point of this framework is the hierarchical multi-model representations used in all redesign activities. The framework implements a hybrid approach of problem solution: means-end and part-whole model-based methodologies. This framework integrates model-based reasoning and case-based reasoning. The framework aims to support human designers to understand the process and to guide them in the redesign process by suggesting components or sections to be modified or substituted.

References

- [Chittaro *et al.*, 1993] Chittaro, L., Guida, G., Tasso, C. and Toppano, E. Functional and Teleological Knowledge in the Multimodeling Approach for Reasoning About Physical Systems: A case study in diagnosis. IEEE Trans. on Sys, Man & Cyb. 23(6), pp. 1718-1751, 1993.
- [Douglas, 1988] Douglas J.M. Conceptual design of chemical process, Mc Graw Hill, New York, 1988.
- [Larsson, 1996] Larsson, J.E. Diagnosis based on explicit means-end models, Artificial Intelligence 80, 29-93. 1996.
- [Turton *et al.*, 1998] Turton R., R.C. Bailie, W.B. Whiting and J.A. Shaeiwitz, Analysis, synthesis and design of chemical process, Prentice-Hall, New Jersey, 1998.

Knowledge Based Approach for Mechanically Verifying Security Protocols

Xiaoqi Ma, Xiaochun Cheng and Rachel McCrindle

Department of Computer Science
The University of Reading
Whiteknights, Reading RG6 6AY, UK
xiaoqi.ma@reading.ac.uk

Abstract

A new knowledge-based security protocol verification approach is proposed in this paper. A number of predicates, functions, assumptions and rules are used to infer the knowledge of participating principals. These items are implemented with Isabelle, which enables mechanical proving. This approach can prove protocols concerning interleaving protocol sessions and can prove the correctness of a medium-sized security protocol in a couple of seconds. The mechanical proofs of a number of important secure properties and then of the correctness of the Needham-Schroeder-Lowe protocol are given as examples to show the effectiveness of this method.

1 Introduction

To evaluate and verify security protocols in a systematic way, significant research work has been conducted in the area of designing formal methods for analysis of cryptographic protocols, and many good models have been proposed. Generally speaking, all these methods can be broadly divided into two categories, namely state based methods and rule based methods.

State based methods model security protocols using finite state machines. They search the state space exhaustively to see whether all the reachable states are safe [Paulson, 1997]. If some reachable state in a security protocol is proved to be unsafe, a flaw may be reported; otherwise, the protocol will be reported correct and safe. State based methods are usually complete and can find most flaws in protocols.

Rule based methods formally express what principals can infer from messages received [Paulson, 1997]. In these approaches, the protocols, the necessary assumptions and the goals of the protocols are formulated in formal logic. Some specific properties of the protocols can be proved by using the axioms and rules of the logic. Since rule based methods do not have to search large state space, they can normally converge quickly.

To gain effectiveness from state based methods and efficiency from rule based methods, we propose in this paper a new security protocol verification method, which is based on a knowledge-based framework. The method analyses the knowledge of participating principals and infers what they

can know and can never know. It takes protocols concerning multiple interleaving sessions into consideration and can find flaws which are often overlooked by many rule based methods. Unlike state based method, our method avoids searching large state space. By implementing the method in a mechanical reasoning platform, Isabelle [Nipkow *et al.*, 2003], it can be used to mechanically verify cryptographic protocols and can converge very quickly. The Needham-Schroeder public key authentication protocol [Needham and Schroeder, 1978] with Lowe's fix [Lowe, 1995] is analysed in this paper as an example to show the effectiveness and efficiency of our method.

2 The Knowledge Based Method

Our method focuses on the knowledge of all participants in the protocol. We describe their initial knowledge and infer what they can and cannot know with the progress of the protocol. In other words, it concerns the knowledge analysis of all participants. This method can be implemented in Isabelle to enable mechanical verification.

All *principals* taking part in network communications can be divided into three categories: the *server*, the *friends*, and the *spy*.

Random numbers chosen by principals serve as *nonces* to identify protocol runs uniquely and to avoid replay attack. Every principal has some *keys*. All the nonces and keys are represented by natural numbers.

A message is a piece of information sent from one principal to another. A message consists of principals' names, nonces, keys, encrypted messages, signed messages, hashed messages, or a combination of these. It can be recursively defined in Isabelle.

We define a number of *functions* and *predicates*. Key functions map principals to their certain keys. Function *Nonce_of* maps principals to their nonces. To determine whether a message is a part of another one, we introduce the function *msg_part*.

The predicate *Know* describes a principal's knowledge about a certain message. Similarly, the predicate *Auth* describes one principal's authentication state about another principal on a certain message, i.e., whether the message is sent by the expected principal and is unmodified.

To describe cryptographic protocols, two *action functions* need to be introduced. One is *Send*, representing that one principal sends a message to another principal. Correspondingly, the other function is *Rcv*, meaning that a principal receives a certain message from others.

Our method is based on a number of assumptions, which are widely accepted by most researchers in this field.

We introduce a group of inference rules into the method to infer new knowledge from the old. All these rules can be divided into four categories: *encryption/decryption* rules, *message combination/separation* rules, *sending/receiving* rules and *authentication* rules.

3 Verifying Needham-Schroeder-Lowe Protocol Mechanically

To verify the Needham-Schroeder-Lowe protocol, we first model it in our framework, then prove a number of important properties, and finally prove the security guarantees.

Normally, the protocol can be formalised into three steps for all honest principals. The first step is an application of the *Send* action function for sending *A*'s nonce and name to *B*. The second step states that if the first step has been successfully carried out, principal *B* will correspondingly send a compound message consisting of *A*'s nonce, its own nonce and name to principal *A*. Similarly, the third step describes that if the second step has been successfully carried out, principal *A* will correspondingly send *B*'s nonce back to principal *B*.

These steps are sufficient for honest principals, but not for the spy who does not necessarily obey the rules. Therefore, we introduce four extra formulae. The first one is *Fake*, stating that the spy may send messages it knows to others. Another two formulae describe how honest principals respond to faked messages. The honest principals take the forged messages as protocol steps, and then respond them according to the protocol, as if they are legal messages.

To avoid interleaving attack, Lowe introduced the receiving principal's name into step 2 of the protocol [Lowe, 1995]. Accordingly, we introduce the *decline* rule: if the name in the message is not the name of the sender, the receiving principal should decline the message and terminate the communicating session.

With above modelling, we first prove some properties before we prove the final guarantees for the two participating principals.

One of these properties is that if a principal knows a message *M*, and *M*₁ is a part of *M*, then it should know the message *M*₁ as well. Due to the inductive definition of the data type *message*, we need to prove this lemma by induction. Several subgoals have been produced after we apply the induction command. The subgoals concerning principal, nonce, key and encrypted, signed and hashed messages can be proved by using the implication introduction, conjunction elimination and conjunction introduction rules of higher order logic. For subgoals concerning compound messages, we design a rule to decompose and resolve them automatically.

The correctness of the Needham-Schroeder-Lowe protocol relies greatly on the secrecy of the nonces used, and therefore the key point for proving the Needham-Schroeder-Lowe protocol is to prove the secrecy of nonces. We have two lemmas, one stating that the spy will never see *B*'s nonce, and the other describing that the spy will never see the content of the second protocol message, which is a compound message consisting of *A* and *B*'s nonces and *B*'s name.

We have to point out that the spy may intercept and know *A*'s nonce. But it does not affect the secrecy of the second protocol message sent from *B* to *A*.

We now can prove the guarantees. The guarantee for *B* after step 3 is that *B* authenticates that *A* has sent *B*'s nonce (encrypted by *B*'s public key) to *B* and this message has not been modified by the spy. The first authentication rule is applied and three subgoals are produced. With above lemmas, these subgoals can be easily solved by Isabelle. The guarantee for *A* can be similarly proved.

4 Conclusions

In this paper, we have presented a new framework to prove the correctness of security protocols. We model the protocols and infer them by analysing principals' knowledge.

To improve the efficiency of security protocol verification, we implement our framework using Isabelle. We have implemented all the data structures, functions, predicates, assumptions and inference rules in Isabelle. With this implementation, we are able to prove the correctness of protocols mechanically. All the proving details are generated by Isabelle, thereby saving users a significant amount of time. In addition, our framework also takes the cases concerning multiple interleaving sessions into consideration, making the method more powerful.

To show the effectiveness of our framework, we have used the Needham-Schroeder-Lowe public key authentication protocol as an example. The example shows how to use the framework and its implementation. Our implementation can prove the correctness of this protocol in a couple of seconds. Additionally, most of the codes can be reused for similar applications.

References

- [Lowe, 1995] Gavin Lowe. An attack on the Needham-Schroeder public-key authentication protocol. *Information Processing Letters*, 56(3):131–133, 1995.
- [Needham and Schroeder, 1978] Roger Needham and Michael Schroeder. Using encryption for authentication in large networks of computers. *Communications of the ACM*, 21(12):993–999, 1978.
- [Nipkow et al., 2003] Tobias Nipkow, Lawrence C. Paulson, and Markus Wenzel. *Isabelle/HOL: A Proof Assistant for Higher-Order Logic*. Springer Verlag, Heidelberg, 2003.
- [Paulson, 1997] Lawrence C. Paulson. Proving properties of security protocols by induction. In *Proceedings of the 10th Computer Security Foundations Workshop*, pages 70–83, Rockport, Massachusetts, June 1997.

Capturing and Reusing Case-Based Context for Image Retrieval*

Dympna O'Sullivan¹, Eoin McLoughlin¹, Michela Bertolotto¹ & David C. Wilson²

¹University College Dublin

Belfield, Dublin 4, Ireland

²University of North Carolina at Charlotte

Charlotte, NC 28223-0001 USA

dymphna.osullivan@ucd.ie, eoin.a.mcloughlin, michela.bertolotto.ucd.ie, davils@uncc.edu

Abstract

Like many other application areas, task-based domains that employ digital imagery are faced with the problem of information overload. Modeling the relationship between images and the tasks being performed is an important step in addressing this problem. We have developed an interactive approach for the capture and reuse of image context information that leverages a measure of a user's intentions with regard to tasks that they address. We analyze aspects of human-computer interaction information that enables us to infer why image contents are important in a particular context and how specific images have been used to address particular domain goals.

1 Introduction

Recent advances in digital image capture and storage technologies have increased the problem of information overload in imagery task domains. As a consequence, intelligent application support is needed to help manage imagery tasks. The majority of current retrieval techniques search for images based on similarity of appearance, using low-level features such as color and shape or by natural language textual querying where similarity is determined by comparing words in a query against words in semantic image metadata tags. The biggest problem arising from these techniques is the so-called semantic gap [Hollink *et al.*, 2004] — the mismatch between the capabilities of the retrieval system and user needs.

In order to address this problem we have developed a decision support system that can integrate information about underlying visual data with more high-level concepts provided by users as they complete specified tasks. Capturing human expertise and proficiency allows us to understand why relevant information was selected and also how it was employed in the context of a specific user task [Leake *et al.*, 1999]. From a case-based standpoint our research focuses on case knowledge acquisition and case knowledge reuse, where a case is represented by a complete user task including all system interactions in the course of carrying out the task. The approach allows us to capture and reuse best practice techniques

by automatically constructing a knowledge base of previous user experiences. This knowledge base can then be employed to improve future context-based query processing. Similar tasks are no longer interpreted as simply a collection of natural language terms: rather image retrieval requests take into account the context of the specific user domain goals. This approach allows for a dramatic reduction in both the time and effort required to carry out new tasks as amassed contextual knowledge is reused in support of the similar tasks. Furthermore, there are benefits from a knowledge management standpoint as contextual knowledge pertaining to particular tasks may now be stored and reused as an additional resource for support, training and preserving organizational knowledge assets.

2 System Overview

Our system enables a user to search directly for imagery corresponding to their current task needs. A typical task-based query to our image repository may consist of any combination of specified metadata, semantic task information and a sketched configuration of image-objects. The user may search an image database for imagery corresponding to their current goals or a knowledge base of previous user experiences (called sessions) for other user's tasks that may bear resemblance to their own. In either case the imagery/sessions are ranked according to a percentage matching score. The user may then interact with the returned information as part of their task by annotating and highlighting relevant image aspects.

3 Capturing Case-Based Context

Our research draws on work in capturing task knowledge with the ultimate goal of performing more effective image retrieval using annotations [Lieberman and Liu, 2002]. To this end we have developed tools for direct image manipulation to assist the user in organizing information about relevant imagery and their task to capture important contextual information about specific user goals. These insights are then distilled into a form the system may use to perform similarity matching. The tools for direct image manipulation include filters, transformation, highlighting, sketching, post-it type and multimedia annotations. They allow the user to identify regions of interest that can be linked to clarifications and rationale. All con-

*The support of the Proof of Concept Fund of Enterprise Ireland is gratefully acknowledged



Figure 1: Capturing User Context through Image Annotation

textual knowledge is gathered by the system using implicit analysis so that users are shielded from the burden of knowledge engineering [Claypool *et al.*, 2001]. We unobtrusively monitor, interpret and respond to user actions as they interact with the imagery via the annotational tools and record this as user context.

Figure 1 shows how a user can make use of the image annotation tools as part of a geo-spatial image task. The particular task that this user had in mind was the construction of an airport on the outskirts of an urban area. The user has uploaded some multimedia and textual annotations; these textual annotations are represented by the icons on the image. The notebook icon replaces the text box where the user has typed comments and prevents the image from becoming cluttered. The camera icon replaces the video uploaded by the user, and when clicked on will play back the recording. The dark rectangles around the icons are the areas associated with these annotations and are emphasized once the icons are moused over. The area emphasized by the rectangle surrounding the notebook icon represents an undeveloped area outlying a residential part of the urban region. The user comment states that this is an area of low elevation, away from residential areas and drained naturally by a large river (i.e. suitable to proceed with the new development). The user has also highlighted a group of buildings by sketching an oval shape around them (they may have concerns that these buildings are quite close to the area selected to build the airport). Once the user has finished interacting with the imagery their entire task process is stored as an encapsulated session case in the knowledge base.

4 Reusing Similar User Context for Retrieval

The application allows for the reuse of similar user context for retrieval. More specifically, if a user working in a similar task domain to a previous user queries the system, the application reuses the previously captured context to compute similarity between the new task and the goals of previous users in order to recommend images and user sessions that may be of help to the current user. A more complete description of how

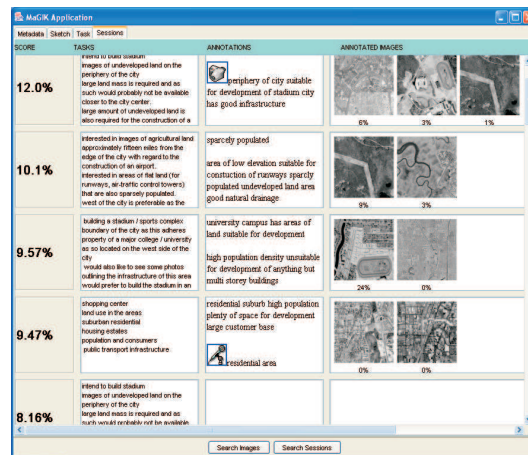


Figure 2: Session Retrieval

this similarity is calculated is described in [O'Sullivan *et al.*, 2004]. The current user can compare their task to previous work both to find relevant imagery and also to examine the decisions and rationale involved in addressing the previous task. Figure 2 shows the interface for displaying retrieved sessions.

Each row in the interface represents a similar user session and is summarized to include the percentage matching score, the most similar queries, the most important annotations added by the similar user, media buttons to play multimedia annotations and thumbnails of any annotated imagery. The user can reuse these similar sessions in carrying out their task by incorporating all or part of a relevant session into their own task context. Because the task-based retrieval system is tightly coupled with the activities that the user is performing, the system has the capacity to make proactive recommendations in an unobtrusive manner by monitoring the current task context. Information requests, increments in the information accessed and annotations provided are no longer taking place in an isolated environment. Rather they can be grounded in the context of the activities that the user is performing so the system can correspondingly anticipate and update what previous knowledge would be relevant, making it available to the user. This knowledge is provided unobtrusively, and need only be accessed when required. Thus the process does not distract from the task at hand, yet makes relevant knowledge available just-in-time.

5 Experimental Evaluation

The system was originally designed for the retrieval of geo-spatial imagery and an earlier evaluation of session retrieval indicated that sessions were useful as a basis for task-based retrieval [O'Sullivan *et al.*, 2004]. In this paper we show that task-based annotation context has advantages for individual image retrieval by employing the application as an aid for travel planning. We evaluated the system using images from the Corel Image Dataset [Wang and Li, 2003]. We began by analyzing a subset of 500 images (corresponding to 5 countries - Australia, England, France, Ireland and Thailand) and

manually annotating each of them with four/five keywords describing image objects and scenes depicted in the images. In order to extract contextual information for each image, the set of keywords for each image was entered as a query to web sites offering information on holidays in the outlined countries, for example <http://www.visiteurope.com/france.html>.

The resulting web pages were analyzed and the most relevant paragraphs from these pages were applied as annotations to the images. These annotations were between 2 and 5 sentences in length. An example of an annotation for a photograph featuring a part of the English Lake District was *"The Lake District National Park in the north-west of England is the largest of Englands National Parks. Its 2,292 square kilometres cover high fells, lush green dales, still lakes, vibrant villages and quiet hamlets"*. The purpose of these annotations was to act a baseline for comparison with task-based queries relating to holidays in the outlined countries.

In order to demonstrate the task-based retrieval capabilities of the system, several tasks were outlined relating to holidays in the countries and user behavior was simulated by annotating relevant imagery with task-specific information. An example of a task description for a holiday in England was *"Planning a trip to England, spend a few days in London, visit the markets, followed by a few days in the countryside"*. Once the images had been annotated according to this first set of tasks another set of tasks corresponding to more specific types of holidays in the countries were outlined. An example of such a task for planning a holiday in England was *"Interested in an adventure holiday in the English countryside - I am particularly interested in hill-walking and water sports"*. The purpose of this was to see if the contextual annotations added to the images for the first set of tasks could be re-used to find imagery relevant to the second set of tasks. The 500 images were then categorized by hand as either relevant or not relevant to each outlined task in the second category of tasks. Each task was entered as a query to the system and the returned images were analyzed. Figure 3 demonstrates the average scores associated with the top five most relevant images (as judged by a real user) returned for the task-based queries. The left/right columns give the similarity respectively for keyword-based retrieval and annotation-based retrieval. For the most part the annotation-based retrieval vastly outperforms the keyword-based retrieval, demonstrating that task-based annotations can provide useful additional context that can improve image retrieval over a dataset. We plan to expand the library both in terms of the number of images and in terms of the number of applied task-based queries. This experiment is a starting point for automating the construction of datasets from where information can be retrieved based on current context rather than just on keyword or content-based search.

6 Conclusions

We have introduced our approach to developing a context aware system with a task-based focus for retrieving imagery and knowledge. Our intention is to use the presented experimental results as a baseline for future evaluation frameworks where we will examine the relative contributions of differ-

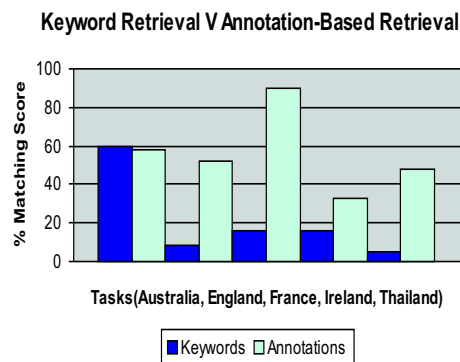


Figure 3: Keyword Retrieval V Annotation-Based Retrieval

ent levels of user, task and annotation contexts. We hope to perform a comprehensive user trial in the near future. We plan to extend our implicit knowledge acquisition techniques by extracting information from a greater variety of user actions. We also intend to supplement our primarily text-based retrieval system by including multimedia annotational information for retrieval.

References

- [Claypool *et al.*, 2001] Mark Claypool, Phong Le, Makoto Waseda, and David Brown. Implicit interest indicators. In *Proceedings of ACM Intelligent User Interfaces Conference, IUI-01*, pages 33–40. ACM Press, 2001.
- [Hollink *et al.*, 2004] Laura Hollink, Guus Schreiber, Bob Wielinga, and Marcel Worring. Classification of user image descriptions. *International Journal of Human Computer Studies*, 66(5):601–626, 2004.
- [Leake *et al.*, 1999] David B. Leake, Larry Birnbaum, Cameron Marlow, and Hao Yang. Task-based knowledge management. In *Exploring Synergies of Knowledge Management and Case-Based Reasoning, Proceedings of The American Association of Artificial Intelligence (AAAI-99) Workshop*, pages 35–39. AAAI Press, 1999.
- [Lieberman and Liu, 2002] Henry Lieberman and Hugo Liu. Adaptive linking between text and photos using common sense reasoning. In *Proceedings of the 2nd International Conference on Adaptive Hypermedia and Adaptive Web Based Systems*, pages 2–11. Springer-Verlag, 2002.
- [O’Sullivan *et al.*, 2004] Dymrna O’Sullivan, Eoin McLoughlin, Michela Bertolotto, and David C. Wilson. A case-based approach to managing geo-spatial imagery tasks. In *Proceedings of ECCBR 2004, Seventh European Conference on Case-Based Retrieval*, pages 702–716. Springer-Verlag, 2004.
- [Wang and Li, 2003] James Wang and Jia Li. Automatic linguistic indexing of pictures by a statistical modeling approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(9):1075–1088, 2003.

Dependency Calculus Reasoning in a General Point Relation Algebra

Marco Ragni and Alexander Scivos

Institut für Informatik, Albert-Ludwigs-Universität Freiburg
Georges-Köhler-Allee 52, 79110 Freiburg, Germany
{ragni, scivos}@informatik.uni-freiburg.de

Abstract

The point algebra is a fundamental formal calculus for spatial and temporal reasoning. We present a new generalization that meets all requirements to describe dependencies on networks. Applications range from traffic networks to medical diagnostics. We investigate satisfaction problems, tractable subclasses, embeddings into other relation algebras, and the associated interval algebra.

1 The Dependency Calculus

Reasoning about complex dependencies between events is a crucial task. However, qualitative reasoning has so far concentrated on spatial and temporal issues. In contrast, we present a calculus [Ragni and Scivos, 2005], a proper generalization of the nonlinear relation algebra, created for specific questions of reasoning about consequences.

This algebra, called dependency calculus (DC), meets all requirements to describe dependencies in networks. There are two aspects: dependencies of points are described by the point algebra PA_{dc} , and of intervals by the associated interval algebra IA_{dc} . For these we analyze questions concerning the satisfaction problems, and show the correspondence to other relation algebras. For this, we use an isomorphism preserving the tractability of subclasses. This method promises to structure the field of relation algebras and to transfer algebraic aspects and complexity results from one algebra to another.

If we observe pollution in an ecosystem of flowing water, we can draw conclusions about pollution at other points (cf. Fig. 1). If pollution is found at D , F is polluted as well. It

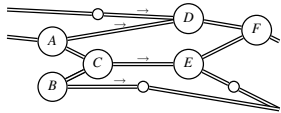


Figure 1: A pipe network. Flow occurs along the “pipes” from left to right. Is there a difference between the pairs (A, B) and (D, E) ?

might be caused from a source at A , but not B , C , or E . If C is polluted, D probably is also polluted. Does the same hold true for B and D ? No: they have no common point upstream. DC directly represents such differences. For pairs like (C, D) or (D, E) , we have the new “fork” relation ($\wedge$). The relations

“fork”, “before”, “equal”, and “after” are called “dependent”. The other case, like (A, B) , is called “independent” ($\asymp$). A medical problem is given in Fig. 2.

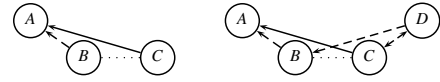


Figure 2: A virus transmittance scheme. Arrows indicate direction of assured, dashed of unassured donorship, dotted lines mean that both persons carry the same virus. No lines means that we do not have prior knowledge. The situation on the left must be incomplete. The existence of a fourth person D accounts for this. PA_{dc} concludes that indeed there was indirect transmittance from D to A .

If x, y are points in a partial order $\langle T, \leq \rangle$, then we define these relations in terms of the partial order as follows:

$$\begin{aligned} x < y & \text{ iff } x \leq y \text{ and not } y \leq x. \\ x = y & \text{ iff } x \leq y \text{ and } y \leq x. \\ x > y & \text{ iff } y \leq x \text{ and not } x \leq y. \\ x \wedge y & \text{ iff } \exists z z \leq y \wedge z \leq x \text{ and neither } x \leq y \text{ nor } y \leq x. \\ x \asymp y & \text{ iff } \text{neither } \exists z z \leq y \wedge z \leq x \text{ nor } x \leq y \text{ nor } y \leq x. \end{aligned}$$

All relations between nodes in Fig. 1 can be described by these five basic relations.

2 Computational Complexity

Assume that a set of constraints between some points is given. One question is whether this set is consistent. Is it possible to construct a network in which all constraints are satisfied? This problem is called PA_{dc} -SAT. What is the computational effort to construct such a network?

Definition 1. For two relation algebras Γ, Γ' a homomorphism is a function γ from Γ to Γ' such that γ preserves all operations of the boolean algebra and for all relations R, S :

- (converse): $\gamma(R^{-1}) = \gamma(R)^{-1}$
- (composition): $\gamma(R \circ S) = \gamma(R) \circ \gamma(S)$

Definition 2. For two relation algebras Γ, Γ' a tractability-preserving-homomorphism (tph) is a homomorphism γ from Γ to Γ' such that each subset $\beta \subseteq \Gamma$ is tractable iff $\gamma(\beta) \subseteq \Gamma'$ is tractable. An isomorphic tph is called tpi.

Lemma 1. A tpi γ from PA_{dc} to $RCC-5$ (Fig. 3) is given by:

$$\begin{aligned} \gamma: \quad < \mapsto PP & \quad = \mapsto EQ & \quad > \mapsto PP^{-1} \\ \wedge \mapsto PO & \quad \asymp \mapsto DR \end{aligned}$$

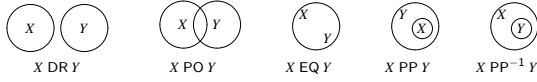


Figure 3: The RCC-5 Relations

Table 1: The tractable subclasses of PA_{dc} .

	τ_{28}	τ_{20}	τ_{17}	τ_{14}
$\perp$	•	•	•	•
$\{=\}$	•	•	•	•
$\{\lambda\}$	•	•	•	•
$\{\prec, \lambda\}$	•	•	•	•
$\{\prec\}$	•	•	•	•
$\{\prec, \prec\}$	•	•	•	•
$\{\lambda, \prec\}$	•	•	•	•
$\{\prec, \lambda, \prec\}$	•	•	•	•
$\{\succ\}$	•	•	•	•
$\{\succ, \succ\}$	•	•	•	•
$\{\lambda, \succ\}$	•	•	•	•
$\{\succ, \lambda, \succ\}$	•	•	•	•
$\{\prec, \succ\}$	•	•	•	•
$\{\prec, \prec, \succ\}$	•	•	•	•
$\{\lambda, \prec, \succ\}$	•	•	•	•
$\{\prec, \lambda, \prec, \succ\}$	•	•	•	•

The relations including $\{=\}$ are contained in $\tau_{28}, \tau_{20}, \tau_{14}$ iff $R \setminus \{=\}$ is in $\tau_{28} (\tau_{20}, \tau_{14})$. All relations including $\{=\}$ are in τ_{17} .

With [Renz and Nebel, 1999] and Lemma 1, we get:

Theorem 1. PA_{dc} -SAT is NP-complete.

Theorem 2. The four classes $\tau_{28}, \tau_{20}, \tau_{17}, \tau_{14}$ (cf. Tab. 1) are the only maximal subclasses of PA_{dc} .

3 The Associated Interval Algebra

There are applications in which it is not sufficient to compare single points in a network. For instance, pollution in a pipe network is not restricted to single points but extends to whole sections, and automated planning and project management deal with tasks that span over time intervals.

Definition 3. An interval $I = [s_I, e_I]$ is a pair of points satisfying $s_I \prec e_I$. The interval algebra IA_{dc} is the relation algebra generated by quadruples of relations as basic relations

$$\mathcal{B} = \left\{ \begin{pmatrix} R_{ss} & R_{se} \\ R_{es} & R_{ee} \end{pmatrix} \mid R_{ss}, R_{se}, R_{es}, R_{ee} \in \{\prec, =, \succ, \asymp, \lambda\} \right\}$$

closed under $\cap, \cup, -, \circ, ^{-1}$. For $I = [s, e]$ and $I' = [s', e']$, being in relation $I R I'$ means $s R_{ss} s', s R_{se} e', e R_{es} s',$ and $e R_{ee} e'$.

IA_{po} is analogically defined, based on $\{\prec, =, \succ, \asymp, \lambda\}$.

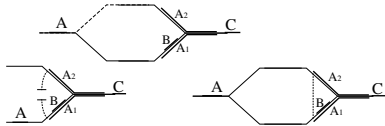


Figure 4: Reasoning with uncertainty. The dashed line indicates that it is unknown if A is before A_2 . In contrast to IA_{po} , IA_{dc} concludes such knowledge: Depending on the IA_{dc} relation between A_1 and A_2 , the obstruction B can or cannot be bypassed using A_2 .

If $R_{ss} = R_{se} = R_{es} = R_{ee} =: R'$, we write (R') . Fig. 4 and Fig. 5 show advantages of the new, finer interval calculus: More situations can be distinguished and more conclusions

are possible. The situation given in Fig. 4 is described by:

$$A (\prec) A_1, A_1 (\prec) C, A_2 (\prec) C, B (\prec) C,$$

$$B \left(\begin{pmatrix} \succ, \prec \\ \prec, \prec \end{pmatrix} \prec \right) A_1, \quad B \left(\begin{pmatrix} \asymp, \lambda \\ \asymp, \lambda \end{pmatrix} \prec \right) A_2, \quad A_1 \left(\begin{pmatrix} \asymp, \lambda \\ \asymp, \lambda \end{pmatrix} = \right) A_2$$

By specifying the latter relation, e.g. to $A_1 \left(\begin{pmatrix} \prec & \prec \\ \prec & \prec \end{pmatrix} \right) A_2$, new conclusions can be drawn, in this case, $A (\prec) A_2$ becomes impossible. This means, if B is an obstacle on the path from A to C via A_1 , then there is no alternative route via A_2 .

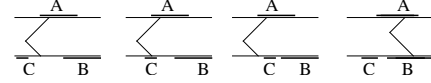


Figure 5: In contrast to IA_{po} , IA_{dc} discerns these cases which are specifications of the composition $A(\{\asymp, \lambda\})B \circ B(\succ)C$.

For a set of IA_{dc} constraints, we define IA_{dc} -SAT analogically to PA_{dc} -SAT. How hard is it to decide the satisfiability?

Theorem 3. IA_{dc} -SAT is NP-hard.

Definition 4. For a subset $S \subseteq PA_{dc}$, a relation R is called S -pointizable if it belongs to the class

$$\mathcal{P}_S = \left\{ \begin{pmatrix} R_{ss} & R_{se} \\ R_{es} & R_{ee} \end{pmatrix} \mid R_{es}, R_{se}, R_{es}, R_{ee} \in S \right\}$$

A gadget for an IA_{dc} relation R is a set of points $p_1, \dots, p_m$ ($m \geq 4$) with PA_{dc} relations that are satisfiable iff $[p_1, p_2]R[p_3, p_4]$ is satisfiable. For $S \subseteq PA_{dc}$, an IA_{dc} relation R is called S -gadgetable ($R \in \mathcal{G}_S$) if all PA_{dc} relations between the endpoints extended with additional points are relations of S . $\mathcal{G}$ is an abbreviation for $\mathcal{G}_{PA_{dc}}$ and $\mathcal{P}$ for $\mathcal{P}_{PA_{dc}}$.

Theorem 4. $\mathcal{G}, \mathcal{P}$ are intractable subclasses of IA_{dc} .

Not all gadgetable relations are pointizable. For instance, $(\prec) \cup (\succ)$ is gadgetable. Hence, $\mathcal{B} \subsetneq \mathcal{P} \subsetneq \mathcal{G} \subsetneq IA_{dc}$. If S is tractable (in PA_{dc}), then $\mathcal{P}_S$ is tractable in IA_{dc} . What about the larger class $\mathcal{G}_S$? Is it still tractable?

Theorem 5. If $\mathcal{R}$ is a class of S -gadgetable IA_{dc} relations and the satisfiability problem over S is tractable, then the satisfiability problem over $\mathcal{R}$ is tractable.

Corollary 1. $\mathcal{B}, \mathcal{G}_{\tau_{28}}, \mathcal{G}_{\tau_{20}}, \mathcal{G}_{\tau_{17}}, \mathcal{G}_{\tau_{14}} \subsetneq IA_{dc}$ are tractable.

Corollary 2. IA_{dc} -SAT is NP-complete.

4 Conclusion

We presented and investigated a new algebra for reasoning about causal relations. Both PA_{dc} and IA_{dc} are NP-complete, and we identified tractable subclasses. Promising ideas for future work are introducing probabilities to this calculus to model reasoning in Bayesian networks and the temporalization of DC for modeling dependencies that vary over time.

References

- [Ragni and Scivos, 2005] M. Ragni and A. Scivos. The Dependency Calculus. Technical report, Institut für Informatik, Universität Freiburg, Germany, 2005.
- [Renz and Nebel, 1999] J. Renz and B. Nebel. On the complexity of qualitative spatial reasoning: A maximal tractable fragment of the Region Connection Calculus. *AIJ*, 108(1-2):69–123, 1999.

Compact Propositional Encodings of First- Order Theories

Deepak Ramachandran and Eyal Amir

Computer Science Department

University of Illinois at Urbana-Champaign

Urbana, IL 61801, USA

{dramacha,eyal}@cs.uiuc.edu

1 Introduction

A *propositionalization* of a theory in First-Order Logic (FOL) is a set of propositional sentences that is satisfiable iff the original theory is satisfiable. We cannot translate arbitrary FOL theories to propositional logic because FOL is only semi-decidable. However, when possible, it is often advantageous to do so because we can use optimized, efficient SAT solvers (e.g. [Moskewicz *et al.*, 2001]) to solve the resulting SAT problem. Propositionalization is used frequently in Planning [Kautz and Selman, 1996], Relational Data Mining [Krogel *et al.*, 2003], and Formal Verification [Kropf, 1999].

Current propositional encodings (*naive prop.*), based on [Gilmore, 1960], result in prohibitively large propositional encodings even for moderate applications and assume a known finite domain for the theory. However despite their drawbacks, they are the most efficient solutions known so far. Examples of naive prop. are given in Table 1.

We briefly describe a novel, systematic approach to translating two important subsets of FOL into propositional logic. Our approach leverages structure in the FOL formulation to provide significantly more compact propositional encodings without requiring a finite fixed domain.

Specifically, we present algorithms for translating two important decidable subsets of FOL: (1) function-free *monadic*, and (2) the *Bernays-Schöenfinkel-Ramsey* class (see [Börger *et al.*, 1996]) in which all existential quantifiers must occur before all universal ones (all arity is allowed for predicates, with equality, but no functions). These subsets cover important problems in AI and computer science, such as expressive planning, data mining, constraint satisfaction, propositional modal logic, and quantified boolean formulae (QBF).

Our algorithms generate propositional encodings of these subsets of FOL as follows. They start by grouping axiom sets into a tree of partitions following the approach of [Amir and McIlraith, 2004]. Then, they translate each partition separately using only a restricted set of constants that depend on

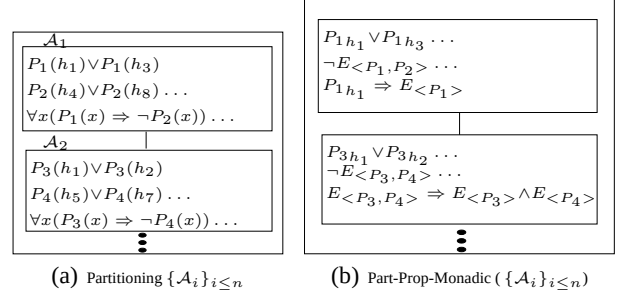


Figure 1: The partitioning of the theory $\mathcal{A}$.

the structure of the partitioning. Finally, they combine the translated parts into a single propositional theory.

In the following, we briefly describe our methods with examples, and then present experimental results from applying these methods to an encoding of the Pigeonhole problem.

2 Partitioned Prop. for the Monadic Class

Figure 1 shows an example of the partitioned prop. of a theory in the monadic class (all predicates have arity 1). The theory $\mathcal{A}$ on the left is partitioned into $\mathcal{A}_1, \mathcal{A}_2 \dots \mathcal{A}_n$. There is an edge between every two partitions that share symbols. The partitioning is done in such a way that *message-passing* [Amir and McIlraith, 2004], a reasoning procedure that involves sending messages between partitions when a formula is in the intersection language, is sound and complete.

Figure 1 b) shows the propositional encoding of the partitioned theory. There are two kinds of propositional symbols, those of the form P_a , which stand for $P(a)$, and $E_{\langle P, Q \dots \rangle}$ which stand for $\exists x[P(x) \wedge Q(x) \dots]$. It can be shown that every monadic FOL formula can be converted into an equivalent propositional sentence using only these symbols. To ensure that the propositional symbols have the right semantics we need to add some consistency axioms to the propositionalization. For eg. $P_{1h_1} \Rightarrow E_{\langle P_1 \rangle}$ ensures that if there is some constant h_1 for which $P_1(h_1)$ holds, then $\exists x P_1(x)$ holds as well and $E_{\langle P_3, P_4 \rangle} \Rightarrow E_{\langle P_3 \rangle} \wedge E_{\langle P_4 \rangle}$ ensures that if $\exists x[P_3(x) \wedge P_4(x)]$ is true, then $\exists x P_3(x)$ and $\exists x P_4(x)$ is true as well.

The union of these partitioned propositionalizations can be

FOL Theory	Naive Prop.
$\neg(P(a) \wedge Q(b))$	$\neg(P_a \wedge Q_b)$
$\forall x P(x) \Rightarrow \forall y \neg Q(y)$	$(P_a \wedge P_b \wedge P_c \Rightarrow (\neg Q_a \wedge \neg Q_b \wedge \neg Q_c))$
$\exists z(R(z, c) \wedge \neg Q(a))$	$(R(a, c) \wedge \neg Q(a)) \vee (R(c, c) \wedge \neg Q(a))$

Table 1: Naive propositionalization

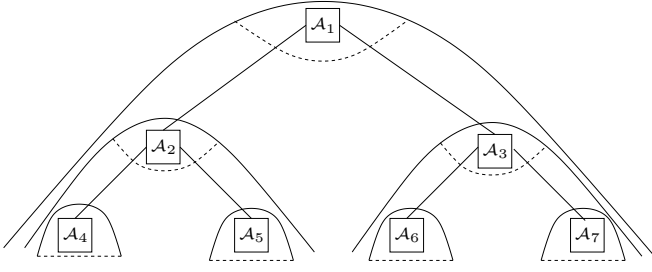


Figure 2: The solid boundaries represents the constants and the broken ones represent the predicates in each separate propositionalization.

shown to be equisatisfiable with $\mathcal{A}$. The proof relies on the completeness of the message passing procedure. Briefly, it can be shown that if the intersection graph of the partitioning is a tree, for every formula α s.t. $\mathcal{A} \vdash \alpha$, there is a proof s.t. each step uses either axioms that are completely within the same partition $\mathcal{A}_i$ or messages transmitted to $\mathcal{A}_i$ from other partitions. The proof of this relies on the following theorem:

Theorem 1 (Craig’s Interpolation Theorem). *If α and β are first order formula s.t. $\alpha \vdash \beta$, then there is a formula $\gamma \in \mathcal{L}(L(\alpha) \cap L(\beta))$ such that $\alpha \vdash \gamma$ and $\gamma \vdash \beta$.*

3 Partitioned Prop. for the Ramsey Class

It is known that the naive partitioning is correct for the Ramsey class without any additional semantic assumptions [Börger *et al.*, 1996]. The number of propositional symbols used is $|P||C|^k$ where k is the maximum arity of any predicate. This can be significantly reduced by partitioning. But it is not enough to propositionalize each partition individually. For example, consider the case where $P(c)$ and $\forall x[P(x) \Rightarrow Q(a)]$ are in different partitions. Because we propositionalize these separately, $P(c) \Rightarrow Q(a)$ will not be in the propositional theory and thus $Q(a)$ cannot be deduced.

However, it can be shown that there exists an ordering $\prec$ of the vertices in the intersection graph of the partitioning such that a correct propositionalization results if we propositionalize the predicates of each partition $\mathcal{A}_i$ w.r.t. every constant that occurs in some partition $\mathcal{A}_j$, $\mathcal{A}_j \prec \mathcal{A}_i$. For example, in a prop. of the theory in figure 1 a), P_{3h_3} will be in the language, but P_{1h_2} will not. There are n such orderings, one for each choice of a partition as the root of the tree. In the case of a binary tree with n vertices (figure 2) each prop. will have $\frac{\log n}{n}|P||C|^k$ variables. Thus the problem of satisfiability of $\mathcal{A}$ reduces to n independent SAT instance of size $\frac{\log n}{n}|P||C|^k$. The running time will be $O(n \cdot 2^{\frac{\log n}{n}|P||C|^k})$ compared to the naive prop. which takes $O(2^{|P||C|^k})$.

4 Experimental Results

In figure 3 we present the results of some experiments with our algorithms on two different problems. The first is a first-order encoding of the pigeonhole problem that belongs to the Ramsey class. The upper two graphs compare our Ramsey prop. with the naive prop. for this problem, on the basis of

the number of variables created and the running time for the SAT solver. Our algorithm does substantially better, enabling the solution of problems that are outside the scope of the naive encoding. The size of the encoding for the partitioned Ramsey prop. grows as $O(p \log p)$ versus $O(p^2)$ for the naive prop, where p is the number of pigeons. The running time for our SAT solver shows orders of magnitude speedup.

The bottom two graphs present a similar comparison for a *partitioned pigeonhole* problem (i.e. where each pigeon can be assigned only to a fixed subset of the holes) which is in both the Ramsey and Monadic classes. We compare both the Monadic and Ramsey partitioned prop. with naive prop. For the naive prop., the size of the encoding grows as $O(p^2)$ versus $O(p)$ for the partitioned monadic prop.

The experiments were done on a 3.2 GHz Linux machine with Zchaff [Moskewicz *et al.*, 2001] as the SAT solver.

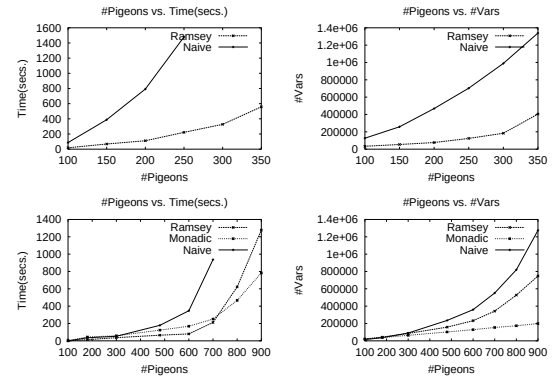


Figure 3: Experimental Results

References

- [Amir and McIlraith, 2004] E. Amir and S. McIlraith. Partition-based logical reasoning for first-order and propositional theories. *Artificial Intelligence*, 2004. Accepted for publication.
- [Amir, 2001] E. Amir. Efficient approximation for triangulation of minimum treewidth. In *UAI’01*, pages 7–15. Morgan Kaufmann, 2001.
- [Börger *et al.*, 1996] E. Börger, E. Grädel, and Y. Gurevich. *The Classical Decision Problem*. Springer-Verlag, 1996.
- [Gilmore, 1960] P.C. Gilmore. A proof method for quantification theory: It’s justification and realization. *IBM Journal of Research and Development*, 4(1):28–35, January 1960.
- [Kautz and Selman, 1996] H. Kautz and B. Selman. Pushing the envelope: Planning, propositional logic, and stochastic search. In *AAAI’96*, 1996.
- [Krogel *et al.*, 2003] M. Krogel, N. Luvrac, and S. Wrobel. Comparative evaluation of approaches to propositionalization. In *ILP*, pages 197–214, 2003.
- [Kropf, 1999] T. Kropf, editor. *Introduction to Formal Hardware Verification*. Springer, 1999.
- [Moskewicz *et al.*, 2001] M. W. Moskewicz, C. F. Madigan, Y. Zhao, L. Zhang, and S. Malik. Chaff: Engineering an efficient SAT solver. In *Proceedings of the 38th Design Automation Conference (DAC’01)*, 2001.

Computationally Grounded Model of BDI-Agents*

Kaile Su¹, Abdul Sattar¹, Kewen Wang¹ and Guido Governatori²

¹Institute for Integrated and Intelligent Systems, Griffith University, Brisbane, Australia

²School of ITEE, The University of Queensland, Brisbane, Australia

1 Introduction

There are two main semantic approaches to formalizing agent systems via modal logics, the *possible worlds semantics* [Hintikka, 1962] and the *interpreted system* model [Fagin *et al.*, 1995]. The first approach includes the well-known theory of intension [Cohen and Levesque, 1990] and the formalism of the belief-desire-intension paradigm [Rao and Georgeff, 1998]. The second approach, offers a natural interpretation, in terms of the states of computer processes, to S5 epistemic logic. The significance of the second approach is that we are able to associate the system with a computer program, and formulas can be understood as properties of program computations. In this sense, the interpreted system model is *computationally grounded* [Wooldridge, 2000].

There are few computationally grounded models for formalizing general BDI-agents. A number of researchers have attempted to develop executable agent languages such as AgentSpeak(L) [Rao, 1996], but these agent languages have comparatively simple semantics [van der Hoek and Wooldridge, 2003]. The aim of this paper is to present a computationally grounded model of general BDI-agents.

The main idea of our BDI-model is that we characterize an agent's belief, desire and intention as a set of runs (computing paths), which is exactly a *system* in the interpreted system model. Let $\mathcal{B}_i$, $\mathcal{D}_i$ and $\mathcal{I}_i$ be sets of runs related to the beliefs, desires and intentions of agent i , respectively. Then, runs in $\mathcal{B}_i$ are possible computing paths from the viewpoint of the agent; runs in $\mathcal{D}_i$ are those computing paths that the agent desires; and runs in $\mathcal{I}_i$ are those computing paths with the agent's choices of the possible actions. Clearly, it is reasonable to assume that $\mathcal{D}_i \subseteq \mathcal{B}_i$, that is, every desired computing path is a possible one, but we need not to assume that $\mathcal{I}_i \subseteq \mathcal{D}_i$ or even $\mathcal{I}_i \subseteq \mathcal{B}_i$ because an agent's intention may fail to achieve its goal and the real computing path may be beyond the agent's belief even though the agent has chosen and completed an intentional series of actions.

The salient point of our work is that we present a general form of BDI agent programs, from which BDI-models can be generated and specifications in full BDI logics can be verified by symbolic model checking techniques.

*This work was supported by the Australian Research Council grant DP0452628, and partially by the NSFC grants 60496327, 10410638 and 60473004.

2 Computationally Grounded BDI Logic

We introduce a multimodal logic of belief, desire and intention, called OBDI logic, where the changes and computation of agents' beliefs, desires, and desires are based on agents' *observations* (i.e. local states). We assume the reader is familiar with the notion of interpreted system model, and we will follow the terminology of [Fagin *et al.*, 1995].

2.1 Syntax

Given a set Φ of propositional atoms, the language of OBDI logic is defined by the following BNF notations:

$$\langle wff \rangle ::= \text{any element of } \Phi \mid \neg \langle wff \rangle \mid \langle wff \rangle \wedge \langle wff \rangle \mid \bigcirc \langle wff \rangle \mid \langle wff \rangle \mathbf{U} \langle wff \rangle \mid B_i \langle wff \rangle \mid D_i \langle wff \rangle \mid I_i \langle wff \rangle$$

Informally, $B_i \varphi$ and $D_i \varphi$ means that agent i believes and desires φ , respectively. While $I_i \varphi$ denotes that φ holds under the assumption that agent i acts based on his intention.

2.2 The BDI-system Model

Given a set G of global states and a system $\mathcal{K}$ over G , an agent's *mental state* over system $\mathcal{K}$ is a tuple $\langle \mathcal{B}, \mathcal{D}, \mathcal{I} \rangle$, where $\mathcal{B}$, $\mathcal{D}$ and $\mathcal{I}$ are systems (sets of runs over G) such that $\mathcal{I} \subseteq \mathcal{K}$ and $\mathcal{D} \subseteq \mathcal{B} \subseteq \mathcal{K}$. A *BDI-system* is a structure $\langle \mathcal{K}, \mathcal{M}_1, \dots, \mathcal{M}_n \rangle$, where $\mathcal{K}$ is a system and for every i , $\mathcal{M}_i$ is agent i 's mental state over $\mathcal{K}$.

Assume that we have a set Φ of primitive propositions. An *interpreted BDI-system* I consists of a pair $(\mathcal{S}, \pi)$, where $\mathcal{S}$ is a BDI-system and π is a valuation function, which gives the set of primitive propositions true at each point in G .

2.3 Semantics

In what follows, we inductively define the satisfaction relation $\models_{\text{OBDI}}$ between a formula φ and a pair of interpreted BDI-system and a point. Given an interpreted BDI-system $I = (\mathcal{S}, \pi)$, suppose that $\mathcal{S} = \langle \mathcal{K}, \mathcal{M}_1, \dots, \mathcal{M}_n \rangle$ and for every i , $\mathcal{M}_i = \langle \mathcal{B}_i, \mathcal{D}_i, \mathcal{I}_i \rangle$. Let r be a run in $\mathcal{K}$ and u a natural number, then we have that:

- $(I, r, u) \models_{\text{OBDI}} B_i \varphi$ iff $(I, r', v) \models_{\text{OBDI}} \varphi$ for those $(r', v) \in \mathcal{B}_i$ such that $(r, u) \sim_i (r', v)$;
- $(I, r, u) \models_{\text{OBDI}} D_i \varphi$ iff $(I, r', v) \models_{\text{OBDI}} \varphi$ for those $(r', v) \in \mathcal{D}_i$ such that $(r, u) \sim_i (r', v)$;
- $(I, r, u) \models_{\text{OBDI}} I_i \varphi$ iff $(I, r', v) \models_{\text{OBDI}} \varphi$ for those $(r', v) \in \mathcal{I}_i$ such that $(r, u) \sim_i (r', v)$;

Other cases are trivial and omitted here.

Proposition 1 *The following formulas are valid:*

- $X(\varphi \Rightarrow \psi) \Rightarrow (X\varphi \Rightarrow X\psi)$
 $X\varphi \Rightarrow YX\varphi$
 $\neg X\varphi \Rightarrow Y\neg X\varphi$
where X and Y stand for B_i , D_i or I_i (for the same i).
- *Relationship between belief and desire*
 $B_i\varphi \Rightarrow D_i\varphi$
- *Temporal operators*
 $\bigcirc(\varphi \Rightarrow \psi) \Rightarrow (\bigcirc\varphi \Rightarrow \bigcirc\psi)$
 $\bigcirc(\neg\varphi) \Rightarrow \neg\bigcirc\varphi$
 $\phi U\psi \Leftrightarrow \psi \vee (\varphi \wedge \bigcirc(\varphi U\psi))$

2.4 Axiomatization

We give a proof system, called OBDI proof system, for those BDI-agents with perfect recall and a global clock, which contains the axioms of propositional calculus plus those formulas in Propositions 1. The proof system is closed under the propositional inference rules plus: $\frac{\vdash\varphi}{\vdash D_i\varphi}$ and $\frac{\vdash\varphi}{\vdash I_i\varphi}$ for every agent i .

Theorem 2 *The OBDI proof system is sound and complete with respect to interpreted BDI-systems with satisfaction relation $\models_{\text{OBDI}}$.*

3 Model Checking BDI-Agents

In order to make our model checking algorithm practically useful, we must consider where our model, an interpreted BDI-system comes from. To make the things simpler, we may consider some abstract programs such as *finite-state programs*, which are expressive enough from the standpoint of theoretical computer science. Moreover, to make our model checking system practically efficient, we use symbolic model checking techniques. Thus, a finite-state program in our approach is represented in a symbolic way.

3.1 Symbolic representation of interpreted BDI-agents

We formally define a (symbolic) *finite-state program with n agents* as a tuple $\mathcal{P} = (\mathbf{x}, \theta(\mathbf{x}), \tau(\mathbf{x}, \mathbf{x}'), O_1, \dots, O_n)$, where $\mathbf{x}$ is a set of system variables; θ is a boolean formula over $\mathbf{x}$, called the *initial condition*; τ is a boolean formula over $\mathbf{x} \cup \mathbf{x}'$, called the *transition relation*; and for each i , $O_i \subseteq \mathbf{x}$, containing agent i 's *local variables*, or *observable variables*. Given a state s , we define agent i 's local state at state s to be $s \cap O_i$. We may associate with $\mathcal{P}$ an interpreted system $I_{\mathcal{P}} = (\mathcal{R}, \pi)$ called *the generated interpreted system of $\mathcal{P}$* .

For convenience, we may use $\mathcal{P}(\theta, \tau)$ to denote a finite-state program with n agents $(\mathbf{x}, \theta(\mathbf{x}), \tau(\mathbf{x}, \mathbf{x}'), O_1, \dots, O_n)$, if $\mathbf{x}$ and $O_1, \dots, O_n$ are clear from the context. Given a finite-state program $\mathcal{P}(\theta, \tau)$ with n agents, we define an *agent's internal program* (over $\mathcal{P}(\theta, \tau)$) as a tuple $\langle \mathcal{P}(\theta_1, \tau_1), \mathcal{P}(\theta_2, \tau_2), \mathcal{P}(\theta_3, \tau_3) \rangle$, where $(\theta_j \Rightarrow \theta) \wedge \tau_j \Rightarrow \tau$, for $j = 1, 2, 3$, and $(\theta_2 \Rightarrow \theta_1) \wedge \tau_2 \Rightarrow \tau_1$ are valid. Clearly, an agent's internal program is exactly related with an agent's mental state. Thus, we define a (symbolic) *BDI-program with n agents* as a tuple $P_A = (\mathcal{P}_K, P_1, \dots, P_n)$, where $\mathcal{P}_K$

is a finite-state program with n agents and for each agent i , P_i is agent i 's internal program over $\mathcal{P}$. We use I_{P_A} to denote the corresponding interpreted BDI-system.

3.2 Model checking OBDI logic

Theorem 3 *Given a BDI-program with n agents $P_A = (\mathcal{P}_K, P_1, \dots, P_n)$, suppose that $\mathcal{P}_K = (\mathbf{x}, \theta(\mathbf{x}), \tau(\mathbf{x}, \mathbf{x}'), O_1, \dots, O_n)$, and for every i , $P_i = \langle \mathcal{P}(\theta_1^i, \tau_1^i), \mathcal{P}(\theta_2^i, \tau_2^i), \mathcal{P}(\theta_3^i, \tau_3^i) \rangle$. Then, for every LTL formula φ and agent i , the following are valid in I_{P_A} ,*

1. $B_i\varphi \Leftrightarrow \forall(\mathbf{x} - O_i)(G(\mathcal{P}(\theta_1^i, \tau_1^i)) \Rightarrow \Gamma(\varphi, \theta_1^i, \tau_1^i))$
2. $D_i\varphi \Leftrightarrow \forall(\mathbf{x} - O_i)(G(\mathcal{P}(\theta_2^i, \tau_2^i)) \Rightarrow \Gamma(\varphi, \theta_2^i, \tau_2^i))$
3. $I_i\varphi \Leftrightarrow \forall(\mathbf{x} - O_i)(G(\mathcal{P}(\theta_3^i, \tau_3^i)) \Rightarrow \Gamma(\varphi, \theta_3^i, \tau_3^i))$

where $\Gamma(\varphi, \theta, \tau)$ is a boolean formula built from φ, θ, τ by using quantifications and fixed-point operations **lft** and **gft**.

Remark that Theorem 3 provides a reduction of OBDI to LTL, while Proposition 9 in [Su, 2004] gives an OBDD-based method of model checking LTL formulas. The complexity of our reduction of logic OBDI to LTL is *PSPACE*-complete. However, because quantifications of boolean functions and fixed-point operators can be dealt with in any OBDD package, the reduction can be based on OBDDs. In fact, we implemented a prototype of the OBDI model checker using CUDD, a very influential OBDD package developed by Fabio Somenzi, and achieved some preliminary experimental results.

4 Concluding Remarks

In this work, we have explored computationally grounded modal logics that characterize the internal attitudes of an agent—its beliefs, desires, etc, beyond S5 axioms and carried out a methodology on symbolic model checking for general BDI-agents.

References

- [Cohen and Levesque, 1990] P.R. Cohen and H.J. Levesque. Intension is choice with commitment. *Artificial Intelligence*, 42:23–261, 1990.
- [Fagin et al., 1995] R. Fagin, J. Halpern, Y. Moses, and M. Vardi. *Reasoning about knowledge*. MIT Press, Cambridge, MA, 1995.
- [Hintikka, 1962] J. Hintikka. *Knowledge and Belief*. Cornell University Press, Ithaca, NY, 1962.
- [Rao and Georgeff, 1998] A.S. Rao and M.P. Georgeff. Decision procedures for BDI logics. *Journal of Logic and Computation*, 8(3):293–344, 1998.
- [Rao, 1996] A.S. Rao. BDI agent speak out in a logical computable language. In *LNAI*, volume 1038, pages 42–55, 1996.
- [Su, 2004] K. Su. Model checking temporal logics of knowledge in distributed systems. In *AAAI-04*, pages 98–103. AAAI, 2004.
- [van der Hoek and Wooldridge, 2003] W. van der Hoek and M. Wooldridge. Towards a logic of rational agency. *L.J. of IGPL*, 11(2):135–159, 2003.
- [Wooldridge, 2000] M. Wooldridge. Computationally grounded theories of agency. In E. Durfee, editor, *Proceedings of ICMAS-00*, pages 13–22. IEEE Press, 2000.

The Ontology Revision*

Yu Sun and Yuefei Sui

Key Laboratory of Intelligent Information Processing
Institute of Computing Technology, Chinese Academy of Sciences, China
{sunyu,yfsui}@ict.ac.cn

Abstract

An ontology consists of a set of concepts, a set of constraints imposing on instances of concepts, and the subsumption relation. It is assumed that an ontology is a tree under the subsumption relation between concepts. To preserve structural properties of ontologies, the ontology revision is not only contracting ontologies by discarding statements inconsistent with a revising statement, but also extracting statements consistent with the revising statement and adding some other statements. In the ontology revision, the consistency of a revising statement with the theory of the logical closure of the ontology under the closed world assumption is discussed. The basic postulates of the ontology revision are proposed and a concrete ontology revision is given based on the consistence or inconsistency of an ontology and a revising statement.

1 Introduction

A general approach for studying belief revision is to provide a set of postulates such as the AGM axiom [Alchourr *et al.*, 1985] and the DP axiom [Darwiche and Pearl, 1997] for belief revision functions.

In the process of building and maintaining ontologies, new statements which may be inconsistent with ontologies are added to the ontologies constantly, and it is necessary for the ontologies to be revised to accommodate new statements. For convenience, we assume an ontology O is a tree under the subsumption relation between concepts and can infer what are not explicitly stated by the ontology.

The ontology revision should have the following features:

- Similar to belief revision, there are two kinds of ontology revision: ontology-set revision and ontology-base revision. For an ontology O , its ontology-set is $Th(O)$, a set of statements deduced from O by a set of inference rules; and its ontology-base is O . In this paper, we only discuss ontology-base revision. Moreover, like belief revision, the ontology

revision satisfies some basic principles: success, consistency and minimal change.

- In belief revision, if a knowledge base K and a revising statement α is inconsistent then K is to be contracted. In ontology revision, let $O \circ \theta$ be the ontology results from revising O by a revising statement θ ; S be the smallest set of statements extracted from O to ensure $(O \cup \{\theta\}) - S$ is consistent; Δ be the smallest set of statements extracted from O other than S , and T be the smallest set of statements added to $O \circ \theta$ other than θ , which is consistent with $(O \cup \{\theta\}) - S$. To preserve structural properties of the revised ontology, let $O \circ \theta = ((O \cup \{\theta\}) - (S \cup \Delta)) \cup T$.

- An ontology O is inconsistent iff there is a statement δ such that $\delta, \neg\delta \in O^{CWA}$, where O^{CWA} is an extended theory of $Th(O)$ in terms of the closed world assumption. An ontology revision should consider two cases: (1) $O \cup \{\theta\}$ is consistent; and (2) $O \cup \{\theta\}$ is inconsistent. When $O \cup \{\theta\}$ is inconsistent, if θ is positive then either $\neg\theta \in O$, $\neg\theta \in Th(O)$ or $\neg\theta \in O^{CWA}$; otherwise, either $\neg\theta \in O$ or $\neg\theta \in Th(O)$.

In terms of the postulates the ontology revision should satisfy, to preserve structural properties of ontologies and infer implicit statements ontologies have inherently during revision processes, we propose a Z axiom system for the ontology revision and a concrete operator satisfying the Z axiom system.

2 Ontologies

Definition 1 An ontology O consists of

- (1) a set of concepts and properties;
- (2) four binary relations: the subsumption relation $\sqsubseteq$ between concepts; the inheritance relation $\Rightarrow$ and the default one $\Rightarrow_d$ between concepts and properties; the implication relation $\mapsto$ between properties; and
- (3) a set of positive statements of the form

$$C \sqsubseteq D | C \Rightarrow \varphi | C \Rightarrow_d \varphi | C | \varphi \mapsto \psi$$

and their negations of the form

$$C \not\sqsubseteq D | C \not\Rightarrow \varphi | C \not\Rightarrow_d \varphi | C | \varphi \not\mapsto \psi,$$

where C, D are concepts and φ, ψ are properties.

$C \sqsubseteq D$ means that C is a sub-concept of D ; $C \Rightarrow \varphi$ means that φ is a property of C instantiated by every instance of C ; $C \Rightarrow_d \varphi$ means that φ is a default property of C , which means that normally, every instance of C satisfies φ ; $\varphi \mapsto \psi$

*The work is supported by the National NSF of China (60373042, 60273019 and 60073017), the National 973 Project of China (G1999032701) and Ministry of Science and Technology (2001CCA03000).

means that φ implies ψ logically; $\varphi \Rightarrow C$ means that any instance satisfying φ is an instance of C . We assume that if a statement δ is positive, then $\neg\delta$ is its negation; otherwise, $\neg\delta$ is its positive form.

An ontology O is assumed to be complete about the properties in the following sense: (1) every property is in O ; (2) for any properties $\varphi, \psi \in O$, if φ logically implies ψ then $\varphi \mapsto \psi \in O$; and (3) for any concepts $C, D \in O$, if $C \neq D$ then there is a property φ such that either $C \Rightarrow_d \varphi$ and $D \not\Rightarrow_d \varphi$, or $C \not\Rightarrow_d \varphi$ and $D \Rightarrow_d \varphi$.

Deduction rules in ontologies include transitivity rules, (default) inheritance rules and the closed world assumptions. For the inheritance rules,

$$\frac{C \sqsubseteq D, D \Rightarrow \varphi}{C \Rightarrow \varphi} \quad \text{and} \quad \frac{C \sqsubseteq D, D \Rightarrow_d \varphi : C \Rightarrow_d \varphi}{C \Rightarrow_d \varphi},$$

there is a priority order among the default rules so that sub-concepts override super-concepts. For the transitivity, besides the transitivity of $\sqsubseteq$ among concepts, there are the following three kinds of transitivity rules:

$$\frac{C \Rightarrow \varphi, \varphi \mapsto \psi}{C \Rightarrow \psi}, \quad \frac{C \Rightarrow_d \varphi, \varphi \mapsto \psi}{C \Rightarrow_d \psi}, \quad \frac{\varphi \mapsto \psi, \psi \Rightarrow C}{\varphi \Rightarrow C}.$$

With the above rules, we define $O \vdash \delta$ if there is a deduction of δ from O . Then, $Th(O) = \{\delta : O \vdash \delta\}$ and $O^{CWA} = Th(O) \cup \{\neg\delta : O \not\vdash \delta, \delta \text{ is positive}\}$.

3 The ontology revision

For an ontology O to be revised and a revising statement θ , we have the following presuppositions about $\circ$:

1. O is consistent and $O \circ \theta$ is an ontology.
2. θ is of the form: $C \sqsubseteq D, C \not\sqsubseteq D, C \Rightarrow \varphi$ and $C \not\Rightarrow \varphi$.
3. The ontology revision satisfies the principles of success ($\theta \in O \circ \theta$), consistency ($O \circ \theta$ is consistent) and minimal change (the symmetric difference between the set of statements in O and the set of statements in $O \circ \theta$ is minimal.)
4. Statements other than θ are added to $O \circ \theta$ to infer implicit statements.

To keep the structure of $O \circ \theta$, we may add to $O \circ \theta$ new statements if $\theta = C \not\sqsubseteq D$; and extract statements from O if $\theta = C \sqsubseteq D$. Not like in the belief revision, in the ontology revision, revising statement θ is atomic.

Based on the above discussion, we propose a Z axiom system for the ontology revision:

- Z0. $O \circ \theta$ is an ontology.
- Z1. $O \circ \theta$ is consistent if θ is not contradictory.
- Z2. $\theta \in O \circ \theta$.
- Z3. If $O \cup \{\theta\}$ is consistent, then $O \circ \theta = (O \cup \{\theta\}) - \Delta$.
- Z4. If $O \cup \{\theta\}$ is inconsistent, then

$$O \circ \theta = ((O \cup \{\theta\}) - (S \cup \Delta)) \cup T.$$

- Z5. If $O \circ \theta \vdash \delta$ then $(O \circ \delta) \circ \theta \equiv O \circ \theta$.
- Z6. If $O \circ \theta \vdash \neg\delta$ then $(O \circ \delta) \circ \theta \equiv O \circ \theta$.

The Z axiom system is the combination of the AGM axiom and the DP axiom for the iterated belief revision. For the ontology revision, such a combination is appropriate, because of δ being atomic.

Theorem 1 *The Z axiom system satisfies the principles of success, consistency and minimal change.*

Theorem 1 holds for the ontology revision, because we use atomic statements to revise ontologies, and in the belief revision, the revising statements may not be atomic.

4 A concrete ontology revision $\circ$

We shall give a concrete ontology revision $\circ$ satisfying the Z axiom system. Assume that θ is not contradictory (otherwise, let $O \circ \theta = \emptyset$). By presupposition 2, θ is of one of the following forms: $C \sqsubseteq D$; $C \not\sqsubseteq D$; $C \Rightarrow \varphi$; $C \not\Rightarrow \varphi$.

In the ontology revision, $O \cup \{\theta\}$ is either consistent or inconsistent. When $O \cup \{\theta\}$ is consistent, if θ is positive then either $\theta \in O, \theta \in Th(O)$ or $\neg\theta \in O^{CWA}$; otherwise $\neg\theta \notin Th(O)$. When $O \cup \{\theta\}$ is inconsistent, if θ is positive then either $\neg\theta \in O, \neg\theta \in Th(O)$ or $\neg\theta \in O^{CWA}$; otherwise either $\neg\theta \in O$, or $\neg\theta \in Th(O)$.

The concrete ontology revision $\circ$ is given according to the above 18 cases. For the page limit, we only discuss two cases:

Case 1. $O \cup \{\theta\}$ is consistent and $\theta = C \sqsubseteq D, \theta \notin O, Th(O), \neg\theta \in O^{CWA}$. Let $O \circ \theta = (O \cup \{\theta\}) - \Delta$, where

$$\Delta = \{C \sqsubseteq E \in O : D \sqsubset E \notin Th(O)\}.$$

Case 2. $O \cup \{\theta\}$ is inconsistent and $\theta = C \sqsubseteq D, \neg\theta \in O$. Let $O \circ \theta = ((O \cup \{\theta\}) - (S \cup \Delta)) \cup T$; where

$$\begin{aligned} S &= \{C \not\sqsubseteq D\} \cup \{C \not\Rightarrow \varphi \in O : D \Rightarrow \varphi \in Th(O)\} \cup \\ &\quad \{C \not\Rightarrow_d \varphi \in O : D \Rightarrow_d \varphi \in Th(O)\}; \\ \Delta &= \{C \sqsubseteq E \in O : D \sqsubset E \notin Th(O)\}; \\ T &= \{C \Rightarrow \varphi \in Th(O) : D \Rightarrow \varphi \notin Th(O)\} \cup \\ &\quad \{C \Rightarrow_d \varphi \in Th(O) : D \Rightarrow_d \varphi \notin Th(O)\}. \end{aligned}$$

By verifying that $\circ$ defined in every case satisfies the Z axiom, we have the following theorem:

Theorem 2 *The defined $\circ$ satisfies the Z axiom system.*

5 Further works

The future work will introduce to the ontology revision the logical properties between statements (such as $C \Rightarrow \varphi$ implies $C \Rightarrow_d \varphi$, etc.), the structure of concepts (complex concepts as $\neg C, C \sqcap D$, etc.) and properties in description logics and first order logic, which are omitted for the simplicity in this paper.

References

- [Alchourr et al., 1985] C.E. Alchourrón, P. Gärdenfors and D. Makinson. On the logic of theory change: partial meet contraction and revision functions. *Journal of Symbolic Logic*, 50:510-530, 1985.
- [Darwiche and Pearl, 1997] A. Darwiche and J. Pearl. On the logic of iterated belief revision. *Artificial Intelligence*, 89(1-2):1-29, 1997.
- [Gruber, 1993] T.R. Gruber. A translation approach to portable ontology specifications. *Knowledge Acquisition*, 5(2):199-220, 1993.
- [Guarino and Welty, 2000] N. Guarino and C. Welty. Ontological analysis of taxonomic relationship. In *Proc. of International Conference on Conceptual Modelling*, pages 210-224, 2000.
- [Touretzky, 1986] D.S. Touretzky. *The Mathematics of Inheritance Systems*. Morgan Kaufmann, 1986

A CLP-Based, Diagnosticity-Driven System for Concept Combinations

Georgios Tagalakis

University College Dublin
Department of Computer Science
Dublin 4, Ireland
georgios.tagalakis@ucd.ie

Daniela Ferrari

University of Siena
Department of Philosophy
and Social Sciences
via Roma, Siena 53100, Italy
daniela.ferrari@ucd.ie

Mark T. Keane

University College Dublin
Department of Computer Science
Dublin 4, Ireland
mark.keane@ucd.ie

Abstract

Diagnosticity operates as an important selection criterion for several computational models of concept combination. Unfortunately, it has not been clear how the diagnosticity of property and relational predicates of the concepts combined can be formalized and quantified. Using an information retrieval method we compute, in a uniform manner, diagnosticity values of concepts predicates. We go on to present a reasoning system that attempts to create meaningful interpretations of novel noun-noun combinations. The system is based solely on diagnostic predicates values and a set of constraint satisfaction rules. We show the effectiveness and plausibility of our methods and discuss their potential.

1 Introduction

Noun-noun combinations play a central role in language and cognition. People produce two main types of interpretation to noun-noun composite concepts: *relational* interpretations that hinge on a relation being found to connect the two concepts (e.g., *finger cup*: “a cup to clean fingers in”) and *property* interpretations that hinge on transferring a property from one concept to the other (e.g., *finger cup*: “a thin cup”) (see [Levi, 1978]). In this process, the *diagnostic* predicates of the concepts combined have been found to play a key role. Furthermore, several computational models use the *diagnosticity* constraint as an important selection criterion (e.g., [Costello and Keane, 2000]). However, the diagnosticity mechanisms have so far only been formalized for property predicates, partly because it has not been clear how one can operationally define diagnosticity for relational predicates. Property predicates typically represent perceptual or physical aspects of objects. Relational predicates usually put the concept in connection with other concepts or refer to functions.

In this study, we use a new method for characterizing the diagnosticity of property and relational predicates, which is based on van Rijsbergen’s [1979] *tf-idf* information retrieval schema. We applied this method to compute the diagnosticity values of the predicates of a set of concepts. The data are input to a reasoning system that attempts to provide meaningful interpretation(s) for a given noun-noun combination based on

its interpretation type, the diagnosticity values of the predicates of its constituent concepts and a small set of rules.

2 Formalizing Diagnosticity

The idea underlying the formalization of diagnosticity is that a feature predicate $\mathcal{F}$ is important for a concept C within a set of concepts, if it occurs often in descriptions of concept C , and rarely in descriptions of the rest of the concepts in the set [Ferrari and Keane, 2005]. Given a set of descriptions of concepts, we can thus compute diagnosticity values for each feature relative to a concept $D_{C,\mathcal{F}}$ by adapting the information retrieval *tf-idf* formula [van Rijsbergen, 1979] as:

$$D_{C,\mathcal{F}} = f_{C,\mathcal{F}} \times \frac{\log_2 N_{CS}}{n_{\mathcal{F}}} \quad (1)$$

where $f_{C,\mathcal{F}}$ is the frequency of the feature in a certain concept, and $\log_2 N_{CS}/n_{\mathcal{F}}$ is a non-linear measure derived from the ratio between the number of concepts N_{CS} in the set and the number of concepts $n_{\mathcal{F}}$ described by the feature at issue.

Once the diagnosticity values for concept predicates are derived, we can class every concept as property or relational depending on whether its total diagnosticity is given mainly by relations or properties. When we make a combination from two single concepts each of them bears diagnosticity of property features (*DPF*) and diagnosticity of relational features (*DRF*). These values can in turn be summed as to characterize the combination with a total property diagnosticity (*TPD*) and a total relational diagnosticity (*TRD*).

A list of predicates was formed by asking subjects to describe a set of concepts and any associated knowledge. Formula 1 was used to compute diagnosticity values for their diagnostic predicates. Predicates with one standard deviation or more above the mean score were defined as highly diagnostic (*HD*). A combination was classified as relational, when $TRD > TPD$ or property, when $TPD > TRD$.

3 Diagnosticity-Driven Interpretation Generation

To test the potential of diagnosticity as a driving force in concept combination, we developed a reasoning system (in Prolog) using constraint programming techniques. The techniques were developed after analysis of the data of a previous

Algorithm 1 Diagnosticity-Driven Combination Rules for Relational Noun-Noun Composite Concepts

```
get all MN and HN's property and relational diagnostic features ( $DPF_{MN}$ ,  $DRF_{MN}$ ,  $DPF_{HN}$ ,  $DRF_{HN}$ )
if 1 or more  $DPF_{MN} \geq HD$  then
  if all  $DRF_{MN} < HD$  then
    HN that [ $DPF_{MN}$ ], s.t.  $DPF_{MN} \geq HD$ 
    HN [ $DRF_{MN}$ ] (MN), s.t.  $DRF_{MN} = DRF_{max_{MN}}$ 
    MN [ $DRF_{HN}$ ] (HN), s.t.  $DRF_{HN} = DRF_{max_{HN}}$ 
  else
    HN [ $DRF_{MN}$ ] (MN), s.t.  $DRF_{MN} \geq HD$ 
    MN [ $DRF_{HN}$ ] (HN), s.t.  $DRF_{HN} = DRF_{max_{HN}}$ 
    HN that [ $DPF_{MN}$ ], s.t.  $DPF_{MN} \geq HD$ 
  end if
else if
  if all  $DRF_{MN} < HD$  then
    HN [ $DRF_{MN}$ ] (MN), s.t.  $DRF_{MN} = DRF_{max_{MN}}$ 
    MN [ $DRF_{HN}$ ] (HN), s.t.  $DRF_{HN} = DRF_{max_{HN}}$ 
  else
    HN [ $DRF_{MN}$ ] (MN), s.t.  $DRF_{MN} = DRF \geq HD$ 
    MN [ $DRF_{HN}$ ] (HN), s.t.  $DRF_{HN} = DRF_{max_{HN}}$ 
  end if
end if
```

empirical study, where 30 people were asked to provide interpretations to 60 novel noun-noun combinations with predicates of varied diagnosticity scores and distributions.

3.1 Architecture and Implementation

The system has an interface that asks the user to enter the modifier noun (MN) and the head noun (HN) of a composite concept. This input is parsed and checked against its knowledge base. The knowledge base is in clausal form and consists of a declarative definition of concepts with their predicates and their corresponding diagnostic values as computed above. If both nouns are known to the system, it proceeds to apply the reasoning procedures. The core procedure for the interpretation of relational combinations is sketched in Algorithm 1 and that of the property ones in Algorithm 2. Action constraints are represented as binary functions and enforce valid transitions between states.

3.2 Evaluation

Evaluation of the interpretations produced was conducted with 30 noun-noun concepts, half of which were property and half relational, with constituent concepts ($N = 50$) of varying diagnosticity scores. All reasoning sub-mechanisms were tested. The results were compared with existing people's data. Sixty two interpretations were produced. 81% ($N = 50$) of the interpretations were deemed to be sensible by two judges. Half of the meanings generated were among those people had produced. For the property combinations in particular, it was found that the interpretations generated were those people produced more often.

4 Discussion

Currently, the application of the new method for computing diagnosticity to "diagnostically mark" concept predicates of

Algorithm 2 Diagnosticity-Driven Combination Rules for Property Noun-Noun Composite Concepts

```
get all MN and HN's property and relational diagnostic features ( $DPF_{MN}$ ,  $DRF_{MN}$ ,  $DPF_{HN}$ ,  $DRF_{HN}$ )
if 1 or more  $DPF_{MN} \geq HD$  then
  if all  $DRF_{MN} < HD$  then
    HN that [ $DPF_{MN}$ ], s.t.  $DPF_{MN} \geq HD$ 
  else
    HN that [ $DPF_{MN}$ ], s.t.  $DPF_{MN} \geq HD$ 
    HN [ $DRF_{MN}$ ] (MN), s.t.  $DRF_{MN} \geq HD$ 
  end if
else
  if all  $DRF_{MN} < HD$  then
    HN that [ $DPF_{MN}$ ], s.t.  $DPF_{MN} = DPF_{max_{MN}}$ 
    HN [ $DRF_{MN}$ ] (MN), s.t.  $DRF_{MN} = DRF_{max_{MN}}$ 
  else
    HN [ $DRF_{MN}$ ] (MN), s.t.  $DRF_{MN} \geq HD$ 
    HN that [ $DPF_{MN}$ ], s.t.  $DPF_{MN} = DPF_{max_{MN}}$ 
  end if
end if
```

concept glosses of large, standardized knowledge bases (e.g., WordNet) is being examined. The benefits of exploiting such a rich information for concept combination has been demonstrated in [Tagalakakis and Keane, 2004]. Diagnosticity marking can be an important enhancement to the knowledge representation of ontology systems in general.

As regards the reasoning model, the percentage of correct interpretations it produces based on diagnostic values is very encouraging; especially when one considers that the system is self-contained and does not take into account other key aspects of concept understanding (e.g., plausibility, informativeness, previous knowledge). Future work will provide additional model validation and programming methodology details.

References

- [Costello and Keane, 2000] F. J. Costello and M. T. Keane. Efficient creativity: Constraint-guided conceptual combination. *Cognitive Science*, 24:299–349, 2000.
- [Ferarri and Keane, 2005] D. Ferarri and M. T. Keane. Diagnosing conceptual combinations: A methodology for operationalizing diagnosticity in conceptualization. Unpublished manuscript, 2005.
- [Levi, 1978] J. Levi. *The Syntax and Semantics of Complex Nominals*. Academic Press, New York, 1978.
- [Tagalakakis and Keane, 2004] G. Tagalakakis and M. T. Keane. Automating the interpretation of novel, noun-noun compounds using WordNet. In P. Gervás and K. M. Gupta, editors, *Proceedings of the European Conference on Case-Based Reasoning 2004 Workshops*, pages 225–236, Madrid, 2004. Complutense University of Madrid.
- [van Rijsbergen, 1979] C. J. van Rijsbergen. *Information Retrieval*. Butterworths, London, 1979.

Proof General / Eclipse: A Generic Interface for Interactive Proof

Daniel Winterstein¹, David Aspinall¹ and Christoph Lüth²

¹University of Edinburgh, U.K. and ²Universität Bremen, Germany

1 Introduction

In spite of the considerable achievements of the formal proof programme, take-up of these systems by mathematicians and programmers remains poor. At least one reason for this is the lack of good development tools. The *Proof General* (PG) project is an ongoing attempt to redress this issue. Here we present PG/ECLIPSE, an interface/development environment that marks a new phase in the project.

2 The PG/ECLIPSE Interface

PG/ECLIPSE provides a rich range of features, many of which will be familiar to users of modern programming IDEs.

- Proof scripts (and projects) are presented to the user via a specialised text-editor and associated viewers (e.g. the document outline view) which support various actions.
- The central feature of PG/ECLIPSE is *script management* (c.f. [Bertot, 1998]), which is a form of step debugging tailored to linear scripts.
- *Symbol support* can make a huge difference to how readable a proof script is. This is possibly an area where TP interfaces have something to offer to programming interfaces. PG/ECLIPSE provides support for using mathematical symbols – including the use of typing shortcuts to enter symbols, and a symbol table editor, allowing users to adapt and extend the use of symbols to fit their own needs.
- *Theory navigation*: As a theory grows, finding specific definitions and proofs can become increasingly difficult and time consuming. This alone can make a large difference to the usefulness of a theory. PG/ECLIPSE provides support for theory navigation. As proof script files are read, their structure is analysed and the theories, theorems and lemmas they contain are indexed. This index is then used to provide several ways to navigate files.
- A *Content Assistant* which suggests completions for keywords/phrases.
- Good documentation and help is key to both the uptake of theorem provers and the reuse of proof scripts – but is often neglected. PG/ECLIPSE provides tools for documenting prover commands, prover settings and proof scripts, with integrated help displays. Help for theory

elements (e.g. theorems) is created simply by writing a preceding comment (an idea taken from the Java approach to documenting code (c.f. [Friendly, 1995])).

- One of the great potentials for provers is in mathematical/scientific education. Although there are excellent computer algebra systems used in education, these neither perform nor teach proofs, which are central to mathematical work. PG/ECLIPSE introduces a teaching tool (based around an embedded web-browser), designed for delivering teaching material that interacts with a prover.
- *Interface scripting* Often a script will run better under certain settings. PG/ECLIPSE allows a theory developer to encode these settings in the proof script file. It defines a small set of commands that can be used to script the interface itself.

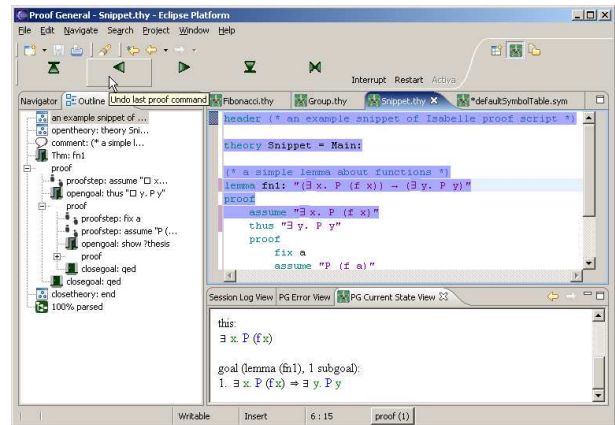


Figure 1: The PG/ECLIPSE display. The main window shows the proof script; view windows below show the prover output. A Problems view (not shown here) lists outstanding problems, such as syntax errors or unfinished proof-goals. To the left of the editor window is an outline view of the proof's structure. Above the editor, a toolbar triggers proof or undo steps by sending instructions to the prover (left-to-right, the buttons are: *undo all*, *undo*, *interrupt*, *step forward*, *process all*, and – most useful of all – *go to cursor location*).

3 Design principles for prover interfaces

Here we describe two design principles specific to theorem prover interfaces.

3.1 Proof scripting is a form of programming

It has often been noted that theorem proving has a great many similarities with programming. However this insight has not really been used in prover interface design. Since programming interfaces serve a much larger community, they have been much more actively developed. We therefore borrow ideas from programming interfaces where possible. One consequence of this was the decision to use Eclipse as a base. Eclipse is an open-source state-of-the-art IDE, originally intended for Java programming [Eclipse Foundation, 2003].

3.2 Separation of proof engine and interface

This is a form of modularisation. Modularisation is especially important in this domain, where – in spite of the relatively small size of the community – there are a range of target applications and a diverse wealth of systems. Each of these needs a good interface. Hence separating prover and interface should facilitate the development of TPs, by giving TP developers easy access to sophisticated interfaces. It would also benefit prover interface and application designers, who need not be tied to one proof system.

The Proof General Kit

We enforce this clear separation by specifying an API (the PG/KIT) for all prover interactions *and* all knowledge of prover behaviour. This API consists of two linked parts: An abstract model for prover behaviour, and a protocol for communication within proof sessions.

The model is based on abstracting the common behaviour of many interactive proof systems. It acts as a clearly specified “virtual layer” that must be emulated in each prover to cooperate properly with PG/ECLIPSE.

The communication protocol is called *PGIP*, for *Proof General Interactive Proof*. The basic principle for representing proof scripts in PGIP is to keep the prover’s native language, and mark up the script with PGIP tags (which give the information needed by an interface). Designing PGIP as a wrapper for native languages has two main consequences:

Firstly, the user employs the prover’s native language (PGIP mark-up is only used internally). This is necessary, since the wide variety of logics supported by modern provers make it very hard to come up with one language which efficiently supports all of these. It is also pragmatic, as users can continue to work in the proof language they are familiar with, and old proof scripts can still be used.

Secondly, the PGIP-annotation of the proof script (and subsequent user input) is done by the prover (or a component coming with the prover), and not by the interface, which knows nothing about the prover’s language. Parsing requests and responses form part of the PGIP protocol.

4 Related work

We can only give a brief overview here of the closest related projects.

The Theorema theorem prover and the TeXMacS systems both have excellent support for entering and viewing mathematical expressions [Buchberger, 2001][Audebaud, 2003]. Various systems already implement script management (e.g. the previous PG system). However, these offer less functionality than PG/ECLIPSE, and do not provide a satisfactory solution to the problem of handling different provers. The most closely related work is that done within the PG/KIT framework: a PGIP Emacs mode, and a PGIP-based ‘proof desktop’, where theories and proofs are built up using graphical actions such as drag-and-drop. There is also a *broker* component in that can act as a middle-man between a collection of PGIP-equipped provers and interfaces. This should lead to increasingly flexible ways to develop proofs.

5 Future work

There are many possible lines for future development. Firstly, we want to use the Eclipse framework to further explore the analogy between theory development and software engineering. We can also go beyond adapting program development tools. One promising line of work is on using interactive proof planning to construct proof scripts (c.f. [Fleuriot, 2003]). Interactive planning could also be a valuable idea to transfer to IDE based programming. The closest existing analogue is the idea of ‘programming templates’; fragments of code which can be used to help build up programs (e.g. a standard widget initialisation method). Proof planning can be considerably more powerful.

5.1 State of the project

This work is described in more depth at <http://proofgeneral.inf.ed.ac.uk/Kit>. An alpha-release of PG/ECLIPSE is now available (also from this website), and the latest version of Isabelle (available from www.cl.cam.ac.uk/Research/HVVG/Isabelle) supports PGIP. Developers interested in using Proof General for other provers should contact the authors.

References

- [Audebaud, 2003] P.Audebaud & L.Rideau. Texmacs as authoring tool for formal developments. In *User Interfaces for Theorem Provers UITP’03*, 2003.
- [Bertot, 1998] Y.Bertot & L.Théry. A generic approach to building user interfaces for theorem provers. *Journal of Symbolic Computation*, 1998.
- [Buchberger, 2001] B.Buchberger. Theorema: A short introduction. *Mathematica Journal*, 8:247–252, 2001.
- [Eclipse Foundation, 2003] Eclipse Foundation. Eclipse platform technical overview. www.eclipse.org, 2003.
- [Fleuriot, 2003] L.Dixon & J.Fleuriot. Isaplanner: A prototype proof planner in isabelle. In *19th International Conference on Automated Deduction*, 2003.
- [Friendly, 1995] L.Friendly. The design of distributed hyperlinked programming documentation. In *International Workshop on Hypermedia Design ’95*, 1995.

Using Learned Browsing Behavior Models to Recommend Relevant Web Pages

Tingshao Zhu, Russ Greiner

Dept of Computing Science
University of Alberta
Edmonton, Alberta T6G 2E8
{ tszhu, greiner }@cs.ualberta.ca

Gerald Häubl

School of Business
University of Alberta
Edmonton, Alberta T6G 2R6
Gerald.Haeubl@ualberta.ca

Kevin Jewell, Bob Price

Dept of Computing Science
University of Alberta
Edmonton, Alberta T6G 2E8
{ kjewell, price }@cs.ualberta.ca

Abstract

We introduce our research on learning browsing behavior models for inferring a user's information need (corresponding to a set of words) based on the actions he has taken during his current web session. This information is then used to find relevant pages, from essentially anywhere on the web. The models, learned from over one hundred users during a five-week user study, are session-specific but independent of both the user and website. Our empirical results suggest that these models can identify and satisfy the current information needs of users, even if they browse previously unseen pages containing unfamiliar words.

Keywords: machine learning, web mining

1 Introduction

While the World Wide Web contains a vast quantity of information, it is often difficult for web users to find the relevant information they are seeking. While modern search engines have now made search over billions of web pages an everyday occurrence, this approach still requires the user to explicitly initiate a search and to formulate a specific search query. The alternative approach, which involves surfing the web by following appropriate links, is also challenging as users cannot always determine which links are most likely to lead to the relevant information. Our goal is a client-side Web recommender system that simply observes the user's browsing behavior, then uses this information to suggest relevant pages from anywhere on the Web, without requiring the user to provide any additional input.

Historically, information filtering approaches avoid the need for explicit queries, but they assume the user has stable long-term interests that can be used to pick out relevant material from an ongoing, dynamically changing flow of content [Lewis and Knowles, 1997]. In this case, a user's interests can be determined by statistical inference over extensive historical samples of the content that the user preferred. Alternatively, information assistants are designed to respond effectively to transient short-term information needs that vary from task to task using content stored in large stable repositories. In contrast, we present a method to determine the user's

current information need without either explicit queries nor a large sample of past content.

2 Model

For the purposes of our present work, we focus on users using the Web to obtain some information. Our goal is to automatically provide the user with additional relevant content beyond what would be immediately revealed by either links on their current page, or results from a search query they might initiate. We assume that this activity can be separated into distinct sessions in which the user is pursuing a single specific information need.

As the user browses the Web, he visits pages and takes actions on these pages, such as following a link, backing up to a previous page, or bookmarking a page. We view the user's web browsing actions as implicit judgments on the content of the pages (with respect to his current information need). Presently, we assume that the user is looking for text-based resources. In this case, the content of the pages is represented by words. We theorize that the actions the user takes while browsing provide evidence about the degree to which words on visited pages represent the user's current information need.

This intuition has appeared in previous work in special cases: for instance, the idea that words appearing in the anchors of hyperlinks followed by the user are probably more representative of user's needs than words that do not. Indeed, the Letizia [Lieberman, 1995] agent and the recommender created by Watson [Budzik and Hammond, 1999] use observations of user behavior to help them locate pages, but they are both based on a limited set of hand-coded heuristic rules. Alternatively, correlation-based recommenders [Agrawal and Srikant, 1995] tend to be tied to a specific site. Moreover, such systems typically suggest that the current user go to pages that other similar users have visited, without knowing whether previous users actually found the page helpful. Our "behavior-based inference" differs from content-based systems [Billsus and Pazzani, 1999], as we do not require any previous experience with the actual words in the hyperlink anchor; moreover, our models are explicitly trained to identify pages that satisfy any particular information need.

For example in our models, following any hyperlink provides evidence for the relevance of those words appearing in the hyperlink anchor, independent of what words are actually present in the hyperlink. Consequently the technique is not

restricted to a subset of indexed sites. Indeed, we attempt to predict the words that best capture the user's information need with a combination of basic text-based features (e.g., the overall frequency of the word in the user's current session, etc.), as well as a generalized set of behavior-based features (e.g., the presence of the word in anchor text, the use of the word on a page from which the user backed up, and the order in which a list of links using the word were selected, etc.).

Since the precise significance of a word appearing in a followed hypertext anchor, or being in some font, etc., is difficult to state a priori, we use machine learning methods to learn the weight of each of our features, towards identifying which word will be relevant. This is based on data collected from a pool of calibration subjects. Once the weights have been calculated, we can apply our user-independent models to new users without retraining. In fact, since the models are trained on behaviors (e.g., the value of appearing in a followed hyperlink, etc.) *instead* of the information viewed (i.e., "the user is interested in 'baseball'"), the models can also be applied to new domains without retraining, and can respond to new user needs as they arise during browsing.

To apply machine learning to this domain, we need a training set. The input to the machine learning process consists of the various text-based and behavior-based *features* associated with each word in the current context [WebIC, 2005], plus a target label describing whether or not the word is representative of the user's interests. We have developed three methods to produce this label using different sources of information, each based on the subjects in a calibration pool.

IC-wordModel: Each subject hand-labels pages that contain desired information content as IC-pages (otherwise the pages are considered -IC-pages by default). We then assume that all non-stopwords appearing on an "IC-page" are representative of the user's current information need, whereas words that do not are unrepresentative.

IC-RelevantModel: Each subject explicitly chooses words from the current browsing session that they felt best represented their current information need. All other words are assumed to be unrepresentative.

IC-QueryModel: Each subject hand-labels pages that contain desired information content as IC-pages (otherwise the pages are considered -IC-pages by default). But while the IC-word model viewed *all* words in the IC-page as important, this model is more selective, as it includes only the subset of non-stopwords that enable a search engine (e.g., Google) to find that page. In a separate sub-project, we estimate a search-engine-specific function that, given a page (here an IC-page, p) and a list of words W , estimates whether giving W as a query to the search engine will produce p . Using this function, we label only the k highest-scoring words from the session as representative.

Given a set of browsing sessions, we can form a matrix where each row corresponds to a word that appears, and each column to one of the browsing features. We then label each row using one of the models described above, and run a standard learning algorithm (e.g., C4.5) to construct 3 different classifiers. Each classifier predicts whether a word is representative of the current information need from the text-based and behavior-based features extracted from the user's current browsing session.

3 LILAC Study

The goals of the LILAC (Learn from the Internet: Log, Annotation, Content) study are to evaluate the browsing behavior models presented above (Section 2), and to gather data for future research, from people working on their day-to-day tasks.

3.1 Overview of WebIC

WebIC (Figure 1) is a client-side, Internet Explorer-based web browser and recommender system. It uses information from the current browsing session to recommend a page, from anywhere on the Web, that it predicts the user will find useful.

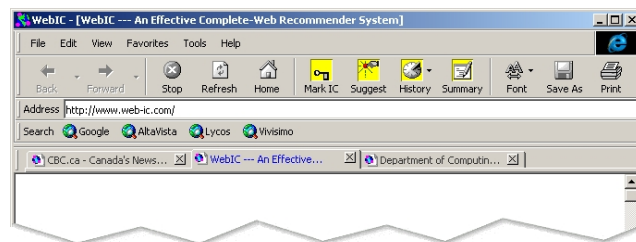


Figure 1: WebIC — An Effective Complete-Web Recommender System

To make a prediction, WebIC first computes the browsing features for all stemmed non-stopwords that appear in any page of the current browsing session. It then determines which of these words to submit as a query to a search engine, using one of the models learned previously. WebIC recommends the top page returned from the query, as the page it considers most likely to satisfy the user's current information need.

We modified WebIC for the LILAC study. Here, whenever the user requests a recommendation by clicking the "Suggest" button, WebIC will select one of its models randomly to generate a recommendation page. As one of the goals of the LILAC study is to evaluate our various models, this specialized version of WebIC will therefore ask the user to evaluate this proposed page.

In addition, we also instructed the participants to click "MarkIC" whenever they found an IC-page. After marking an IC-page, WebIC will recommend an alternative web page as before (excluding the IC-page), as if the user had clicked "Suggest" here. Once again, this specialized version of WebIC will then ask the user to evaluate this recommended page.

As part of this evaluation, the user is asked to "Tell us what you feel about the suggested page", to indicate whether the information provided on the suggested page was relevant to his/her search task. There are two categories of relevance evaluations: *related* and *not related at all*.

In addition, the user was also asked to select informative "Descriptive Keywords" from a short list of words that WebIC predicted as relevant. The information collected here will be used to train the IC-Relevant model described above.

3.2 Experiment Design

To participate in the LILAC study, the subjects were required to install WebIC on their own computer, and then use it when

browsing their own choice of non-personal English language web pages. They were told to use another browsing engine when dealing with private issues, such as banking, e-mail, or perhaps chat-rooms, etc.

WebIC kept track of all the interactions, including a record of the pages the user visited, as well as the evaluation data for the pages that WebIC recommends.

LILAC considered four models: the three described in Section 2 — IC-word, IC-Relevant, and IC-Query— and “Followed Hyperlink Word” (FHW), which is used as a baseline. FHW collects the words found in the anchor text of the followed hyperlinks in the page sequence. As such, there is no training involved in this model. This is similar to “Inferring User Need by Information Scent (IUNIS)” model [Chi *et al.*, 2001]. In all models, words are stemmed and stopwords are removed prior to prediction.

We used the data collected during the study to weekly re-train each of our IC-models. That is, the users initially used the IC-word₀, IC-Relevant₀ and IC-Query₀ models, which were based on a model obtained prior to the study. On the 2nd week, they used the IC-word₁, IC-Relevant₁ and IC-Query₁ models, based on the training data obtained from week 1, as well as the prior model. We repeated this process throughout the study.

3.3 Overall Results

A total of 104 subjects participated in the five-week LILAC study, from both Canada (97) and the US (7); 47% are female and 53% are male, over a range of ages (everyone was at least 18, and the majority were between 21-25). The subjects visited 93,443 Web pages, clicked “MarkIC” 2977 times and asked for recommendations by clicking the “Suggest” button 2531 times. As these two conditions are significantly different, we analyzed the evaluation results for “Suggest” and “MarkIC” separately.

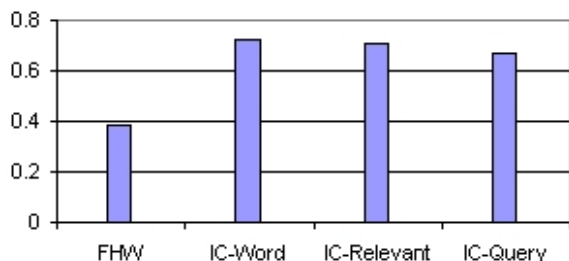


Figure 2: How often the User rated a Recommended Page as “Related”, after “Suggest”

Figure 2 indicates how often the user considered the “Suggest”ed page to be “Related”. We see that each of our 3 IC-models works much better than the baseline model — each was over 65%, versus the 38% for FHW.

Figure 3 shows the evaluation results for the recommended pages after the user clicked “MarkIC”. We again observe that our IC-models work better than FHW. The scores for IC-word and IC-Relevant remained at around 70%, roughly the same values they had for the “Suggest” case, while the IC-Query

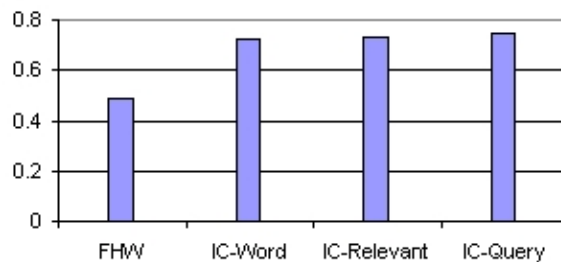


Figure 3: How often the User rated a Recommended Page as “Related”, after “MarkIC”

model increased from 66% to 74%. We also observed that FHW increased by almost 10%. As an explanation, we speculate the following: If the subject is able to find an IC-page, then the links followed in the current session appear to provide a very strong hint of what constitutes the relevant information content; and FHW benefits significantly from this hint.

4 Conclusion and Future Work

We have demonstrated a practical and extensible framework for a behavior-based web recommender system. The basic framework covered here was able to find relevant pages 70% of the time and consistently outperformed a common baseline method (FHW). Our method can easily be extended to include additional features (e.g., timing, multiple search engines), and to employ personalized training models for custom feature weights. In addition, we anticipate our results, achieved using C4.5, could be improved by more sophisticated learning techniques. Extensive results from our user study suggest that behavior-based recommendation is a promising approach for automatically finding the pages, from anywhere on the Web that address the user’s current information need, without requiring the user to provide any explicit input.

References

- [Agrawal and Srikant, 1995] R. Agrawal and R. Srikant. Mining sequential patterns. In *Int’l Conference on Data Engineering (ICDE)*, 1995.
- [Billsus and Pazzani, 1999] D. Billsus and M. Pazzani. A hybrid user model for news story classification. In *User Modeling*, 1999.
- [Budzik and Hammond, 1999] J. Budzik and K. Hammond. Watson: Anticipating and contextualizing information needs. In *American Society for Information Science*, 1999.
- [Chi *et al.*, 2001] E. Chi, P. Pirolli, K. Chen, and J. Pitkow. Using information scent to model user information needs and actions on the web. In *ACM CHI 2001 Conference on Human Factors in Computing Systems*, pages 490–497, Seattle WA, 2001.
- [Lewis and Knowles, 1997] D. Lewis and K. Knowles. Threading electronic mail: A preliminary study. *Information Processing and Management*, 33(2), 1997.
- [Lieberman, 1995] H. Lieberman. Letizia: An agent that assists web browsing. In *IJCAI*, 1995.
- [WebIC, 2005] 2005. <http://www.web-ic.com>.

NEXT SECTION