

Model Checking for Multivalued Logic of Knowledge and Time

Beata Konikowska and Wojciech Penczek *

Institute of Computer Science, PAS
Ordona 21, 01-237 Warsaw, Poland

email: {beatak,penczek}@ipipan.waw.pl

ABSTRACT

We present a multivalued μ K-calculus, an expressive logic to specify knowledge and time in multi-agent systems. We show that the general method of translation [22] from multivalued to two-valued De Morgan algebras can be extended to mv μ K-calculus model checking. This way can we reduce the model checking problem for mv μ K-calculus to several instances of the model checking problem for two-valued μ K-calculus. As a result, properties involving mv μ K-calculus or its subsets, like mv CTLK or mv CTL*K, can be verified using any of the available model checking algorithms. Three simple examples are shown to exemplify possible applications of multivalued logics of knowledge and time.

Categories and Subject Descriptors

F.3.1 [Specifying and Verifying and Reasoning about Programs]: Specification techniques; D.2.4 [Software/Program Verification]: Model checking; I.2.11 [Distributed Artificial Intelligence]: Multiagent systems

General Terms

Verification

Keywords

Model checking, interpreted systems, epistemic logic, multivalued modal logic.

1. INTRODUCTION

The field of multi-agent systems (MAS) theories is traditionally concerned with formal representation of mental attitudes of autonomous entities, known as *agents*, in a distributed system. Over the past 20 years, several modal log-

ics, involving operators referring to time, knowledge, beliefs, desires, goals, and intentions, have been developed with this motivation in mind. These logics are perceived as *specifications* of particular classes of MAS systems. Their aim is to offer a description of the *macroscopic* mental properties (such as knowledge, beliefs, etc.) that a MAS should exhibit in a specific class of scenarios. Model checking is a widespread, basic method used in automated verification of concurrent systems, like hardware circuits, communication protocols, and distributed programs [4]. It amounts to verifying that a finite state program P satisfies a property φ (denoted $P \models \varphi$). When P is represented by its model M_P and the property φ is given by a temporal logic formula, one checks whether $M_P \models \varphi$, whereby the process of generating M_P for P and checking that $M_P \models \varphi$ is automated.

Recently, the field of model checking has been extended to cover verification of multi-agent systems [30], which has become a topic of very active research. In particular, recent contributions [12, 13, 24] have focused on extending to such systems the model checking tools and techniques usually employed for verification of reactive systems. Essentially, these study the validity of a formula representing a property of interest in the model representing all the computations of the multiagent system (MAS) under consideration. Iterative approaches to model checking (i.e., those based on explicit enumeration of all the possible states of computation) are known to suffer from the state explosion problem. For this reason, methods based on symbolic representation are currently seen as the most promising ones. In particular, methods based on variants of the Binary Decision Diagrams (BDD's) [26], and satisfiability (SAT) checkers [25, 18] are a continuing focus of research. Verification via the BDD's involves translating the model checking problem into operations on boolean functions represented in a concise and canonical way, whereas SAT-checkers are used for testing satisfiability of the formulas which encode the model checking problems.

In addition to "classic", two-valued model checking within the framework of two-valued modal logics, increasing attention has recently been devoted to multi-valued model checking, based on various variants of multi-valued (mv) modal logics. This paradigm turns out to be very useful in a number of problems for which we need to reason under uncertainty or to employ models containing inconsistency. Uncertainty can occur either when complete information is not known or cannot be obtained, or when this information has been abstracted away [9]. On the other hand, models can be inconsistent as they combine conflicting points of view or

*Also affiliated with Institute of Informatics, Podlasie Academy, Siedlce. The author acknowledges support from Ministry of Science and Education (grant No. 3T11C01128).

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

AAMAS'06 May 8–12 2006, Hakodate, Hokkaido, Japan.
Copyright 2006 ACM 1-59593-303-4/06/0005 ...\$5.00.

involve components developed by many designers [5].

The aim of this paper is to report on recent progress in applying multi-valued logic to verification of not only temporal, but also epistemic properties of a MAS. Our approach is novel in offering a model checking method for a newly defined, multi-valued μ -calculus combined with knowledge modalities (named mv μ K-calculus). We continue here the line of research initiated in our former papers [22, 21], showing that the general method of translating the model checking problem for mv CTL* to the two-valued model checking problem for CTL* considered in those papers can be extended to modal mv μ K-calculus.

First, we show that an injection f of a De Morgan algebra $\mathcal{L} = (L, \leq, \neg)$ underlying the mv-logic in question into itself which preserves the bounds in the lattice and complement allows us to translate the model-checking problem for the mv μ K-calculus over $\mathcal{L}$ into the model-checking problem for the mv μ K-calculus over a simpler algebra $\mathcal{L}' = (L', \leq \cap(L' \times L'), \neg \cap L')$, where $L' = f(L)$. As before, a repeated application of this method reduces an mv model checking problem for the mv μ K-calculus to a set of problems we can already solve.

Second, we treat the case when the translation f mentioned above does not preserve complement. Then in order to compute the value of a formula in the target, $\mathcal{L}'$ -based model M' for the mv μ K-calculus, we need both the values of $f(x)$ and $f(\bar{x})$ for $x \in L$. For this purpose, we use negated propositions and a two-component transition relation, with the second component storing negation of the first.

The rest of the paper is organized as follows. In the next section we discuss the related work. The mv μ K-calculus is introduced in Section 3. An example of a coffee machine is presented in Section 4. Then, in Section 5 the general translation method is given. In the last section we give some final remarks.

2. RELATED WORK

Translations from multi-valued temporal logics to the standard model checking problem were defined by several authors. The main idea behind the reduction techniques appears in Fitting's work [7, 8] on the connection between 'multi-expert' and 'multi-valued' semantics for a propositional modal logic. Reduction methods for the special case of 3-valued logics were discussed in many papers [10, 14, 15]. In our former paper [22], we showed a translation from negation-free mv-CTL* model checking to CTL* model checking for models over finite quasi-boolean lattices. The above translation was generalized to the full multi-valued modal μ -calculus in [11, 2].

On the other hand, there have been recently several new approaches to model checking of temporal logics of knowledge: CTLK [12, 23, 25, 17], CTLKD [31, 28, 27], TCTLKD [32], and an epistemic extension of ATL [19, 20, 16].

3. LOGIC OF KNOWLEDGE AND TIME

In this section we define syntax and semantics of a multi-valued modal logic of knowledge, called mv μ K-calculus.

Syntax of mv μ K-calculus

Let PV be a set of propositional variables, and Var — a set of fixed point variables.

We consider a positive normal form of mv μ K-calculus

(called mv μK^+), where negation is only applied to propositional variables. It is important that mv μK^+ has the same expressive power as mv μK . The language of mv μK^+ is defined by:

$$\varphi ::= p \mid \neg p \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid EX\varphi \mid AX\varphi \mid Z \mid \mu Z.\varphi(Z) \mid \nu Z.\varphi(Z) \mid K_i\varphi \mid D_\Gamma\varphi \mid E_\Gamma\varphi \mid \bar{K}_i\varphi \mid \bar{D}_\Gamma\varphi \mid \bar{E}_\Gamma\varphi,$$

where p ranges over PV , Z — over Var , $i \in A$, and $\Gamma \subseteq A$, with A being a finite set of agents defined below. Here AX and EX represent the familiar temporal modalities "at the next step for all paths" and "at the next step for some path", while ν and μ are the greatest and the least fixed point modalities, respectively. When formulating properties in mv μK^+ we also use the following derived operators:

$$AG\varphi = \nu Z.(\varphi \wedge AX(Z)), \quad EF\varphi = \mu Z.(\varphi \vee EX(Z)).$$

In turn, K_i , E_Γ and D_Γ represent the well-known modalities of the logic of knowledge: K_i means "agent i knows", E_Γ — "everybody in group Γ knows" (everyone knows), and D_Γ — "somebody in group Γ knows" (distributed knowledge), with $\bar{K}_i$, $\bar{E}_\Gamma$ and $\bar{D}_\Gamma$ being their duals. What is more, with help of ν and μ we can also define the more sophisticated group knowledge modality C_Γ (common knowledge), together with its dual $\bar{C}_\Gamma$, as derived operators¹

$$C_\Gamma\varphi = \nu Z.(E_\Gamma\varphi \wedge E_\Gamma(Z)), \quad \bar{C}_\Gamma\varphi = \mu Z.(\bar{E}_\Gamma\varphi \vee \bar{E}_\Gamma(Z)).$$

Semantics for mv μ K-calculus

We start with defining quasi-boolean lattices, which are used as logical domains of interpretation for the formulas of our multi-valued logics.

DEFINITION 1. A lattice is a partial order $\mathcal{L} = (L, \leq)$, where, for any two elements $x, y \in L$, there exist their greatest lower bound (called meet and denoted by $x \cap y$) and their least upper bound (called join and denoted by $x \cup y$) with respect to the lattice order $\leq$.²

In what follows, we consider **finite** lattices only, for which $\mathbf{F}(\mathbf{T})$ denotes the least (greatest, resp.) element.

DEFINITION 2. A tuple $\mathcal{L} = (L, \leq, \neg)$ is a quasi-boolean algebra if $(L, \leq)$ is a lattice and $\neg$ is a unary operator (complement) having the following properties for any $x, y \in L$:

- $\overline{x \cap y} = \bar{x} \cup \bar{y}$,
- $\overline{x \cup y} = \bar{x} \cap \bar{y}$,
- $x \leq y$ iff $\bar{y} \leq \bar{x}$,
- $\bar{\bar{x}} = x$.

$\mathcal{L}$ is distributive if the following two conditions hold for any $x, y, z \in L$:

- $z \cup (x \cap y) = (z \cup x) \cap (z \cup y)$,
- $z \cap (x \cup y) = (z \cap x) \cup (z \cap y)$.

¹In the logic of knowledge, the usual practice is to introduce C_Γ as a primary operator, but the presence of the fixpoint operators in our logic allows us to derive it from the simpler knowledge modalities.

²Clearly, from antisymmetry of a partial order it follows that both the greatest lower bound and the least upper bound of x, y are uniquely determined elements of L .

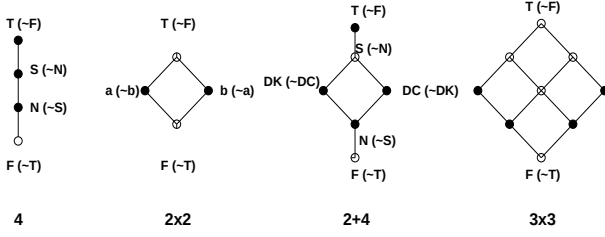


Figure 1: Examples of De Morgan algebras

A distributive, quasi-boolean algebra is called a De Morgan algebra.

EXAMPLE 1. In Figure 3 we show a finite total order 4 representing uncertainty, and three well-known finite lattices, the use of which is motivated by clear practical intuitions: the lattices 2×2 and 3×3 representing disagreement, and the lattice $2 + 4$ representing both uncertainty and disagreement.

The semantics of mv μK^+ is based on the well-known notion of interpreted systems semantics [6], which can be shortly defined as follows. Assume a set $A = \{1, \dots, n\}$ of agents, together with sets Loc_i and Act_i of local states and possible actions, respectively, for each agent $i \in A$, and sets Loc_e and Act_e of local states and actions for the environment. The set of global system states is defined as $S \subseteq Loc_1 \times \dots \times Loc_n \times Loc_e$, where each $(l_1, \dots, l_n, l_e) \in S$ represents a computational state of the whole system. Further assume a set of protocols $P_i : Loc_i \rightarrow 2^{Act_i}$, for $i = 1, \dots, n$, representing the functional behaviour of every agent, and a function $P_e : Loc_e \rightarrow 2^{Act_e}$ for the environment. We can model the computation taking place in the system by means of a transition function $t : S \times Act \times S \rightarrow L$, where $Act \subseteq Act_1 \times \dots \times Act_n \times Act_e$ is the set of joint actions. We assume³ $t(s, a, s') = t(s, a', s')$ for each $s, s' \in S$ and $a, a' \in Act$. Based on the above framework, we define the notion of a model underlying the semantics of mv μK^+ :

DEFINITION 3. Given a set of agents $A = \{1, \dots, n\}$, an mv modal epistemic model (or simply an mv model) is a tuple $M = (S, \mathcal{R}, \sim_1, \dots, \sim_n, V, \mathcal{L})$, where

- S is a finite set of global states of the system,
- $\mathcal{R} : S \times S \rightarrow L \times L$ is an mv transition relation on S ,
- $\sim_i \subseteq S \times S$ ($i \in A$) is an epistemic accessibility relation for each agent $i \in A$ defined by $s \sim_i s'$ iff $l_i(s') = l_i(s)$, where $l_i : S \rightarrow Loc_i$ extracts the local state of agent i from a global state s . Obviously, $\sim_i$ is an equivalence relation,
- $V : S \times PV \rightarrow L$ is a valuation function, for propositional variables PV ,
- $\mathcal{L} = (L, \leq, \neg)$ is a De Morgan algebra.

Our definition of $\mathcal{R}$ is a bit more general than that of [11] as we assign two values to each transition. In what follows, by $\mathcal{R}_j(s, t)$, for $j = 1, 2$, we mean the j -th component

³This restriction allows to simplify the definition of $\mathcal{R}$ in a model.

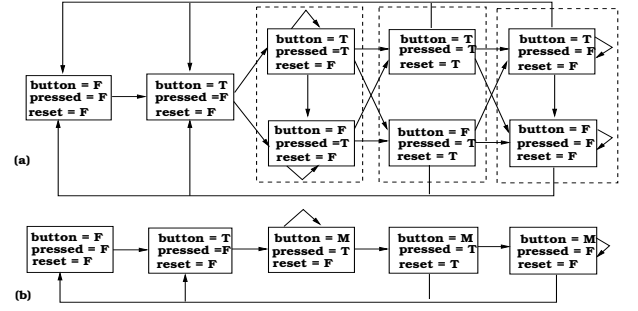


Figure 2: A model of the Button and the abstracted Button over the lattice 3

of $\mathcal{R}(s, t)$. In the model corresponding to an interpreted system (defined above), we take $\mathcal{R}_1(s, s') = t(s, a, s')$ for some $a \in Act$. $\mathcal{R}_2$ ⁴ is used for storing the complement of $\mathcal{R}_1$. Next, we show a simple example of an mv model, which results from the standard two-valued model by using abstraction that collapses concrete states into a single abstract one. Typically, abstraction is used to overcome the state-explosion problem in classical model-checking

EXAMPLE 2. The elevator model of Plath & Ryan consists of a single elevator which accepts requests made by agents (not shown in the figure) pressing buttons on the floor landings and from inside the elevator. The elevator moves up and down between floors and opens and closes its doors in response to these requests. The main module of the elevator is composed of several instances of the module Button (one per floor), which is shown in Figure 2 (a). Each of the states assigns values to the three propositional variables: **button**, **pressed**, and **reset** with the following meaning: **button** = T - button available for pressing, **pressed** = F - button not latched, and **reset** = F - request not fulfilled yet. The arrows denote transitions between states. The transition relation is two-valued (so the transitions are not labeled) and corresponds to the possible changes between the states. The states indicated by the dashed lines (having symmetric transition relations) can be abstracted. This requires the use of a three-valued abstracted model (see Figure 2 (b)) in order to assign the value M (see Figure 5 (3)) to the variable **button** in the abstracted states.

The semantics of mv μK^+ is given by the function $[\cdot]_\rho^M(s)$, which, for each formula φ of mv μK^+ , a model M , a state s in M , and a valuation of the fixed point variables $\rho : Var \rightarrow L^S$, returns the value of φ at the state s of the model M for the valuation ρ , defined as follows:

- $[p]_\rho^M(s) = V(s, p)$ for $p \in PV$,
- $[Z]_\rho^M(s) = \rho(Z)(s)$ for $Z \in Var$,
- $[\neg p]_\rho^M(s) = \overline{[p]_\rho^M(s)}$,
- $[\varphi \vee \psi]_\rho^M(s) = [\varphi]_\rho^M(s) \cup [\psi]_\rho^M(s)$,
- $[\varphi \wedge \psi]_\rho^M(s) = [\varphi]_\rho^M(s) \cap [\psi]_\rho^M(s)$,
- $[EX\varphi]_\rho^M(s) = \bigcup_{t \in S} (\mathcal{R}_1(s, t) \cap [\varphi]_\rho^M(t))$,

⁴We do not show $\mathcal{R}_2$ in the figures.

- $[AX\varphi]_\rho^M(s) = \bigcap_{t \in S} (\overline{\mathcal{R}_1(s,t)} \cup [\varphi]_\rho^M(t))$,
- $[\nu Z.\varphi(Z)]_\rho^M(s) = \bigcup \{ \mathcal{O} \in L^S \mid \mathcal{O} \leq' [\varphi]_{\rho[Z \rightarrow \mathcal{O}]}^M(s) \}$,
- $[\mu Z.\varphi(Z)]_\rho^M(s) = \bigcap \{ \mathcal{O} \in L^S \mid [\varphi]_{\rho[Z \rightarrow \mathcal{O}]}^M(s) \leq' \mathcal{O} \}$,
where $\rho[Z \rightarrow \mathcal{O}]$ coincides with ρ except for mapping Z to $\mathcal{O}$, and $\leq'$ is a pointwise extension of the corresponding operator of $\mathcal{L}$.
- $[K_i\varphi]_\rho^M(s) = \bigcap_{\{s \in S \mid s \sim_i s'\}} [\varphi]_\rho^M(s')$,
- $[E_\Gamma\varphi]_\rho^M(s) = \bigcap_{i \in \Gamma} [K_i\varphi]_\rho^M(s)$
- $[D_\Gamma\varphi]_\rho^M(s) = \bigcup_{i \in \Gamma} [K_i\varphi]_\rho^M(s)$,
- $[\overline{K}_i\varphi]_\rho^M(s) = \bigcup_{\{s \in S \mid s \sim_i s'\}} [\varphi]_\rho^M(s')$,
- $[\overline{E}_\Gamma\varphi]_\rho^M(s) = \bigcup_{i \in \Gamma} [\overline{K}_i\varphi]_\rho^M(s)$
- $[\overline{D}_\Gamma\varphi]_\rho^M(s) = \bigcap_{i \in \Gamma} [\overline{K}_i\varphi]_\rho^M(s)$,

where φ, ψ are formulas of $\text{mv } \mu K^+$, $i \in A$ and $\Gamma \subseteq A$. Note that in fact we can get rid of negation (complement) completely. Indeed, negated propositions can be replaced by fresh propositions, following e.g. [1]. That is, for $p \in PV$ we can define $\overline{p}$ such that $V(s, \overline{p}) = \overline{V(s, p)}$ ⁵; the complements of the values of $\mathcal{R}_1$ can be represented by the second component of $\mathcal{R}$, set at $\mathcal{R}_2(s, t) = \overline{\mathcal{R}_1(s, t)}$; and $\overline{\mathcal{R}_1(s, t)}$ can be replaced by $\mathcal{R}_2(s, t)$ in the semantics of $AX\varphi$. This way we can 'forget' about negation at the price of having to use two values for the transition relation. We show that the above 'trick' is superfluous not only when we do not translate formulas containing both universal and existential subformulas [11], but also when the translation preserves complement.

4. MOTIVATING EXAMPLES

Next, we show two examples. The first of them justifies the use of multiple logical values for expressing the knowledge possessed by agents, whereas the second one shows an mv model where the transition relation is also multivalued.

EXAMPLE 3. Consider an agent B representing the bank, whose task is to decide if a customer C applying for a loan is to be granted that loan to not. For this purpose, the agent sends a query regarding the reliability of customer to a number of databases which contain information about bad debtors. From each database, B may obtain the following information: **Yes**, customer C is a bad debtor; **No**, customer C is not a bad debtor; **Error** - access denied or impossible. What is more, different databases can surely provide different responses — depending on whether customer C occurs in them or not. Clearly, in such a situation the two classical logical values are not enough, and the agent has to take into consideration all basic combinations of responses from the databases: **F** — no information on the customer; **a** — unanimous "No", **b** — unanimous "Yes", and **T** — inconsistent information from different databases (both "No" and "Yes" occur) Thus the natural logic for the agent would be a four-valued one, featuring the above values ordered into the lattice 2×2 shown in Figure 4. Let p_C represent the proposition " C is a bad debtor", and s — the agent's decision state.

⁵We consider only propositions that are used by the formula to be verified.

Then, given suitable formal details about the model M^6 , a reasonable policy to be followed by the agent B is: Refuse the loan irrevocably if $[K_{Bp_C}]^M(s) = \mathbf{b}$, and grant it unconditionally if $[K_{Bp_C}]^M(s) = \mathbf{a}$. In turn, if $[\overline{K}_{Bp_C}]^M(s) = \mathbf{F}$, the agent could delay decision until getting some information from the databases, and if $[\overline{K}_{Bp_C}]^M(s) = \mathbf{T}$ (conflicting information about C 's reliability) — demand a security for the credit from the customer.

EXAMPLE 4. In Figure 3 we show an example of a standard multi-valued model representing our requirements for a simple coffee machine (a modified example from [3]). The coffee machine can be operated by several agents. However, here we show one agent only, for simplicity and clarity sake. The model uses multiple truth values to distinguish between transitions and propositions that **must**, **should**, **should not**, or **must not** be true. Two types of unknown values are used: **do not know** for values controlled by the system, but not yet decided, and **do not care** for values controlled by the environment. Labelling a state s with $p = x$ means that $V(s, p) = x$, where p is a proposition and x is a truth value. Labelling a transition with x means that its truth value is x . In the figure we omit all the transitions labelled **F**, i.e., "prohibited". The coffee machine starts in the state **OFF**, where it is irrelevant whether a cup is provided or not, which is modeled by $\text{cup} = \mathbf{DC}$. Since it has not been decided yet whether the machine should be able to go directly to **IDLE**, the corresponding transition is labelled **DK**. On the other hand, all the transitions labelled **T** represent the required changes of states. Note that the transitions from the states **Delivers ...** to **OFF** are labelled **N**. As they are dependent on the environment they cannot be prohibited, i.e., labelled with **F**. The value **DK** of the proposition power in the state **IDLE** represents the fact that it has not yet been decided whether the power should be **ON** or **OFF** on idle. Notice that the model abstracts away from the payments required from clients for simplicity sake.

The transitions labelled with actions $BC, C + S, WC, WC + S, \text{Ret}$ are shared between the coffee machine and the agent. More precisely, a transition labelled with an action of the coffee machine can be only executed synchronously with a transition labelled with the same action of the drinking agent. Namely, the agent can select a type of coffee to be delivered by the machine and the machine returns to **READY** only after the agent has finished drinking and moved back to the state **HAPPY**.

There are several properties that can be specified and then checked in the model:

1. The machine must eventually be able to deliver coffee,
2. Coffee is always delivered in a cup,
3. After coffee has been delivered, coffee cannot be immediately delivered again.
4. Always when coffee is delivered the agent knows that he is drinking,
5. Always when the agent is thirsty, he will eventually drink coffee,
6. Always when there is no coffee, the agent knows that he is not drinking.

The properties of the coffee machine model can be specified by the following formulas:

⁶We disregard ρ here for simplicity.

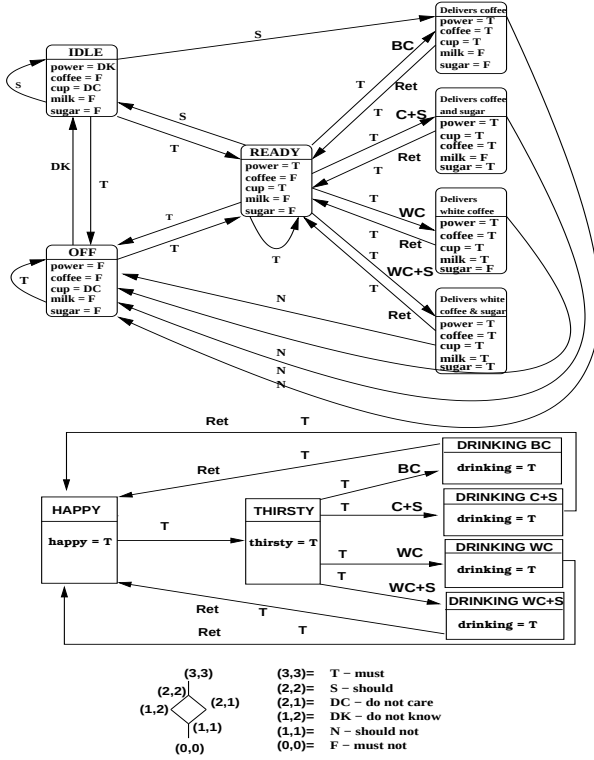


Figure 3: A coffee machine model with one drinking agent over the lattice $2 \times 2 + 2$

1. $EF \text{ coffee}$,
2. $AG(\neg \text{coffee} \vee \text{cup})$,
3. $AG(\neg \text{coffee} \vee AX(\neg \text{coffee}))$.

The properties of the drinking agent model can be specified by the following formulas:

4. $AG(\text{coffee} \Rightarrow K_1 \text{ drinking})$,
5. $AG(\text{thirsty} \Rightarrow EF \text{ drinking})$,
6. $AG(\neg \text{coffee} \Rightarrow K_1 \neg \text{drinking})$.

5. MODEL CHECKING FOR $mv \mu K^+$

Now we proceed with the basic results which will underlie all of the translations for the mv logic of knowledge and time with more than two logical values.

5.1 Translations preserving complement

THEOREM 1. Let $\mathcal{L} = (L, \leq, \neg)$ be an arbitrary quasi-boolean lattice⁷ and let $f : L \rightarrow L$ be a function with $f(\mathbf{F}) = \mathbf{F}$ preserving complement and arbitrary bounds in $\mathcal{L}$, i.e.,

$$\begin{aligned} f(\overline{x}) &= \overline{f(x)}. \\ f(\bigcap_{i \in I} x_i) &= \bigcap_{i \in I} f(x_i), \quad f(\bigcup_{i \in I} x_i) = \bigcup_{i \in I} f(x_i). \end{aligned} \quad (1)$$

⁷Actually, the theorem holds for any lattice, but if $\mathcal{L}$ does not satisfy De Morgan laws, then the positive normal form $mv \mu K^+$ of $mv \mu K$ we consider here is not equivalent to the full language $mv \mu K$ allowing arbitrary placement of negation.

where I is an arbitrary set of indices. Further, let $M = (S, \mathcal{R}, \sim_1, \dots, \sim_n, V, \mathcal{L})$ be an L -valued model of $mv \mu K^+$, and let $M' = (S, \mathcal{R}', \sim_1, \dots, \sim_n, V', f(\mathcal{L}))$ be a model⁸ with values in $f(L)$, obtained out of M by replacing each value $x \in L$ returned by V and $\mathcal{R}_1$ in M with $f(x)$, i.e., $\mathcal{R}'_1(s, s') = f(\mathcal{R}_1(s, s'))$ and $V'(s, p) = f(V(s, p))$, for $s, s' \in S$ and $p \in PV$. Then, for any formula φ of $mv \mu K^+$ over $\mathcal{L}$, any valuation ρ , any state s , and any $x \in f(L)$, we have⁹

$$[\varphi]_{f\rho}^{M'}(s) = x \text{ iff } [\varphi]_{\rho}^M(s) \in f^{-1}(x) \quad (2)$$

The proof of the theorem is based on a key result for the extended language which we will define now. Assume the set of atomic formulas of our language $mv \mu K^+$ is extended by adding expressions of the form $\underline{\mathcal{R}}_1(s, t)$ for $s, t \in S$, where, for any s , $\underline{s}$ is a constant representing the state s , and $\underline{\mathcal{R}}_1$ is a constant representing the first component of the mv-transition relation $\mathcal{R}$. Denote the resulting language by $mv \mu K_e$, and assume that the interpretation of $mv \mu K^+$ is accordingly extended to $mv \mu K_e$.

LEMMA 1. Let a formula φ be such that $[\varphi]_{\rho}^M(s) = \bigcup_{i \in I} [\varphi_i]_{\rho}^M(g_i(s))$ or $[\varphi]_{\rho}^M(s) = \bigcap_{i \in I} [\varphi_i]_{\rho}^M(g_i(s))$ for any model M , any valuation ρ and any state s of M , where I is an arbitrary set, $\varphi_i \in mv \mu K_e$, $g_i : S \rightarrow S$ for $i \in I$, and (2) holds for each φ_i . Then (2) holds for φ too.

PROOF. We consider the case when $[\varphi]_{\rho}^M(s) = \bigcap_{i \in I} [\varphi_i]_{\rho}^M(g_i(s))$; proof of the other case is similar. As f preserves the bounds, by the assumption on φ we have $f([\varphi]_{\rho}^M(s)) = \bigcap_{i \in I} f([\varphi_i]_{\rho}^M(g_i(s)))$. Since each φ_i satisfies (2), this yields $f([\varphi]_{\rho}^M(s)) = \bigcap_{i \in I} [\varphi_i]_{f\rho}^{M'}(g_i(s)) = [\varphi]_{f\rho}^{M'}(s)$, whence $[\varphi]_{f\rho}^{M'}(s) = x$ iff $[\varphi]_{\rho}^M(s) \in f^{-1}(x)$, and (2) holds for φ . $\square$

Now we can prove the theorem itself. To simplify notation, we drop the brackets in expressions denoting application of functions wherever possible. We use induction on the length of the formula. The hypothesis (2) clearly holds for propositional variables and their negations, as well as for $\underline{\mathcal{R}}_1(s, t)$ and $\neg \underline{\mathcal{R}}_1(s, t)$. It also holds for fixed point variables, since $f([Z]_{\rho}^M(s)) = f((\rho Z)(s)) = ((f\rho)(Z))(s) = [Z]_{f\rho}^{M'}(s)$. Next, we assume (2) holds for the formulas of length at most k , and prove it holds for any formula φ of length $k + 1$.

If $\psi = \varphi_1 \vee \varphi_2$, $\psi = \varphi_1 \wedge \varphi_2$, $\psi = EX\varphi$, $\psi = AX\varphi$, $\psi = K_i\varphi$ for $i \in A$, where $\varphi, \varphi_1, \varphi_2$ are formulas of length at most k , then by the definition of semantics for $\vee, \wedge, EX, AX, K_i, \overline{K}_i$ the formula ψ satisfies the assumptions of Lemma 1, whence (2) holds for ψ by that lemma. Further, if $\psi \in \{D_{\Gamma}\varphi, E_{\Gamma}\varphi, \overline{D}_{\Gamma}\varphi, \overline{E}_{\Gamma}\varphi\}$, then by the definition of semantics for $D_{\Gamma}, E_{\Gamma}, \overline{D}_{\Gamma}, \overline{E}_{\Gamma}$ and the result obtained above for $K_i\varphi, \overline{K}_i\varphi$ the formula ψ satisfies the assumptions of the lemma too.

It remains to consider the case when $\psi \in \{\mu Z.\varphi(Z), \nu Z.\varphi(Z)\}$, where φ is of length at most k . We shall only give the proof for the operator μ , since the case of ν is just dual, and the case of $C_{\Gamma}\varphi$ reduces to the latter. Let $[\varphi]_{\rho}^M$ denote the function which maps each $s \in S$ to $[\varphi]_{\rho}^M(s)$. First, note that (2) is equivalent to

$$f([\varphi]_{\rho}^M) = [\varphi]_{f\rho}^{M'} \quad (3)$$

⁸By $f(\mathcal{L})$ we denote the structure $(f(L), \leq \cap (f(L) \times f(L)), \neg \cap f(L))$.

⁹The composition of the functions f and ρ is denoted by $f\rho$ or by $f(\rho)$.

We shall assume the above holds for φ , and prove it also holds for $\psi = \mu Z. \varphi(Z)$.

By the fixed point properties of μ [29], we have

$$[\psi]_{\rho}^M = \bigcup_{i=0}^{\infty} [\varphi]_{\rho_i}^M \quad (4)$$

where $\rho_0 = \rho[Z \rightarrow \mathbf{FF}]^{10}$, $\rho_{i+1} = \rho_i[Z \rightarrow [\varphi]_{\rho_i}^M]$ for $i \geq 0$. Likewise,

$$[\psi]_{f\rho}^{M'} = \bigcup_{i=0}^{\infty} [\varphi]_{\rho'_i}^{M'} \quad (5)$$

where $\rho'_0 = (f\rho)[Z \rightarrow \mathbf{FF}]$, $\rho'_{i+1} = \rho'_i[Z \rightarrow [\varphi]_{\rho'_i}^{M'}]$ for $i \geq 0$. We shall now prove by induction on i that

$$f\rho_i = \rho'_i, \quad i \geq 0 \quad (6)$$

For $i = 0$, we have $f\rho_0 = f(\rho[Z \rightarrow \mathbf{FF}]) = (f\rho)[Z \rightarrow \mathbf{FF}] = \rho'_0$, since $f(\mathbf{F}) = \mathbf{F}$. Now assume the thesis holds for some $i \geq 0$. Then, $f\rho_{i+1} = f(\rho_i[Z \rightarrow [\varphi]_{\rho_i}^M]) = f\rho_i([Z \rightarrow [\varphi]_{f\rho_i}^{M'}]) = \rho'_i[Z \rightarrow [\varphi]_{\rho'_i}^{M'}] = \rho'_{i+1}$, where the second equality follows from the inductive assumption on φ , and the third - from the inductive assumption on ρ_i . This ends the proof of (6). As f preserves arbitrary bounds, from (4), the inductive assumption on φ , and finally (5) we get that $f([\psi]_{\rho}^M) =$

$$f\left(\bigcup_{i=0}^{\infty} ([\varphi]_{\rho_i}^M)\right) = \bigcup_{i=0}^{\infty} f([\varphi]_{\rho_i}^M) = \bigcup_{i=0}^{\infty} [\varphi]_{f\rho_i}^{M'} = \bigcup_{i=1}^{\infty} [\varphi]_{\rho'_i}^{M'} = [\psi]_{f\rho}^{M'}$$

Thus (3) holds for ψ too, which ends the proof.

First and foremost, the bounds and complement preservation conditions (1) imply clearly that $f(\mathcal{L}) = (f(L), \leq \cap (f(L) \times f(L)), \neg \cap f(L))$ is a De Morgan subalgebra of $\mathcal{L}$ - hence we can indeed consider a model M' taking values in $f(L)$. This theorem provides the basis for reducing the model-checking problem over an algebra $\mathcal{L}$ to the model-checking problems over its proper subalgebras of the form $f(\mathcal{L})$, where f is the mapping referred to in Theorem 1.

The general method defines k mappings (translations) $f_1, \dots, f_k$ such that:

- $f_1(\mathcal{L}), \dots, f_k(\mathcal{L})$ are algebras for which we already have model-checking algorithms,
- for any $x \in L$, there exist some $x_i \in f_i(L)$, $i = 1, \dots, k$, such that $\bigcap_{i=1}^k f_i^{-1}(x_i) = \{x\}^{11}$.

Indeed: in such a case, Theorem 1 implies that, for any formula φ of mv μK^+ , any model M over $\mathcal{L}$, any valuation ρ , and any state $s \in S$, $[\varphi]_{\rho}^M(s) = x$ iff $[\varphi]_{f_i\rho}^{M_i}(s) = x_i$ for all $i = 1, \dots, k$, where each M_i is the model over $f_i(L)$ obtained out of M using the mapping (translation) f_i . Hence the model-checking problem for $\mathcal{L}$ reduces to the model-checking problems over $f_1(\mathcal{L}), \dots, f_k(\mathcal{L})$.

Notice that so far we have not used the second component of $\mathcal{R}$, i.e., $\mathcal{R}_2$, which is not needed in the case where the translation preserves complement.

¹⁰ $\mathbf{FF}$ denotes the function $S \rightarrow \{\mathbf{F}\}$.

¹¹Note that the condition imposed on the x_i 's is satisfied by join-irreducible elements (see Definition 4). In that case, the condition is equivalent to Birkhoff's representation theorem for finite distributive lattices. To the authors' best knowledge, the above formulation is new.

5.2 Translations not preserving complement

When complement is not preserved by our translation f , we cannot directly reduce the validity over M to the validity over M' using single values for V and $\mathcal{R}$ only. The problem is that in order to compute $[\varphi]_{f\rho}^{M'}$ we need both the values $f(x)$ and $f(\bar{x})$ for $x = V(s, p)$ or $x = \mathcal{R}_1(s, s')$. Unfortunately, now $f(\bar{x})$ cannot be computed from $f(x)$. Therefore, to keep this information we need new names not only for the negated propositions, but also for the complement of the first component of the transition relation. The situation with the propositions is much easier to deal with as we can simply assign fresh propositions to the states. As far as the transition relation is concerned, there are several possibilities to deal with $\mathcal{R}_1(s, s')$. When translating to **2**, one could obviously get a standard 2-valued model by duplicating states (see [2] or [11] p. 275) of M , so that the values of $f(\mathcal{R}_1(s, s'))$ and $f(\overline{\mathcal{R}_1(s, s')})$ were available as corresponding to the transition between either the original states or their copies.

Here we suggest also another solution, which would not give us a standard model, but does not require duplicating states, and can be directly used by some model checkers which encode transition relation w.r.t. the formula to be verified. As we have already mentioned in the former section, the idea is to use the second component of $\mathcal{R}$ to store the complement of the first one in the model M , i.e., to have $\mathcal{R}_2(s, s') = \overline{\mathcal{R}_1(s, s')}$, and to replace $\overline{\mathcal{R}_1(s, s')}$ with $\mathcal{R}_2(s, s')$ in the semantics of $\text{AX}\varphi$. Then the function f is applied to both the values, returning $\mathcal{R}'_1(s, s') = f(\mathcal{R}_1(s, s'))$ and $\mathcal{R}'_2(s, s') = f(\mathcal{R}_2(s, s'))$ in M' . In other words, we translate the complement of $\mathcal{R}_1(s, s')$ directly. In case of two-valued models, the transitions would be labelled by pairs $(\mathbf{F}, \mathbf{F})$, $(\mathbf{F}, \mathbf{T})$, $(\mathbf{T}, \mathbf{F})$, and $(\mathbf{T}, \mathbf{T})$, and depending on whether the value of an existential or an universal formula is computed, either the first or the second element of the pair would be used.

5.3 Translation to 2-valued algebra

In this section we show that the translation method based on join irreducible elements [11, 2] can be seen as a special instance of our general translation.

DEFINITION 4. *An element $l \in L$ is called join-irreducible iff $l \neq \mathbf{F}$ and, for any $x, y \in L$, $l = x \cup y$ implies that either $l = x$ or $l = y$.*

The set of all the join-irreducible elements of a lattice $\mathcal{L}$ is denoted by $\mathcal{JI}(\mathcal{L})$.

In Figure 3, the join-irreducible elements in the lattices are marked with black dots.

It is well known that every element of a finite distributive lattice can be uniquely decomposed into the join of all the join-irreducible elements in its downward closure $\downarrow x = \{y \in L \mid y \leq x\}$ of x , i.e.,

$$x = \bigcup (\mathcal{JI}(\mathcal{L}) \cap \downarrow x) \quad (7)$$

Let S be a set and $\mathcal{L} = (L, \leq, \neg)$ a De Morgan algebra. Denote by L^S the set of all the functions from S to L . Then, $(L^S, \leq', \neg')$ is a De Morgan algebra, where $\leq'$ and $\neg'$ are pointwise extensions of the corresponding operators of $\mathcal{L}$.

For $x \in L$ let $\uparrow x = \{y \in L \mid x \leq y\}$ denote the upward closure of x .

THEOREM 2. Let $\mathcal{L}$ be a finite De Morgan algebra, and let $l \in \mathcal{JI}(\mathcal{L})$. Then, the function $f_l : L \rightarrow \{\mathbf{T}, \mathbf{F}\}$ defined by $f_l(\uparrow l) = \mathbf{T}$ and $f_l(L \setminus \uparrow l) = \mathbf{F}$ preserves arbitrary bounds.

PROOF. This proof uses two facts, namely that $l \in \mathcal{JI}(\mathcal{L})$ and that $\mathcal{L}$ is distributive. We should prove that

$$(a) f_l(\bigcup_{i \in I} x_i) = \bigcup_{i \in I} f_l(x_i), \quad (b) f_l(\bigcap_{i \in I} x_i) = \bigcap_{i \in I} f_l(x_i)$$

(a): Obviously, $f_l(\bigcup_{i \in I} x_i) = \mathbf{T}$ implies $(\bigcup_{i \in I} x_i) \in \uparrow l$, whence $l = l \cap \bigcup_{i \in I} x_i$. As $\mathcal{L}$ is distributive, the latter implies $l = \bigcup_{i \in I} (l \cap x_i)$. Since $l \in \mathcal{JI}(\mathcal{L})$, one can easily see that this in turn yields $l \cap x_i = l$, i.e., $x_i \in \uparrow l$, for some $i \in I$, whence $f_l(x_i) = \mathbf{T}$, and in consequence $\bigcup_{i \in I} f_l(x_i) = \mathbf{T}$.

Conversely, if $\bigcup_{i \in I} f_l(x_i) = \mathbf{T}$, then $f_l(x_i) = \mathbf{T}$ for some $i \in I$. Thus $x_i \in \uparrow l$ for some $i \in I$, whence $\bigcup_{i \in I} x_i \in \uparrow l$ by the monotonicity of join and $f_l(\bigcup_{i \in I} x_i) = \mathbf{T}$, which proves (a).

(b): Clearly, $f_l(\bigcap_{i \in I} x_i) = \mathbf{T}$ iff $\bigcap_{i \in I} x_i \in \uparrow l$ iff $x_i \in \uparrow l$ for each $i \in I$ by the anti-monotonicity of join. This in turn holds iff $f_l(x_i) = \mathbf{T}$ for each $i \in I$ iff $\bigcap_{i \in I} f_l(x_i) = \mathbf{T}$. Thus (b) holds too. $\square$

By (7), for each $x \in L$ we have $x = \bigcup(\mathcal{JI}(\mathcal{L}) \cap \downarrow x)$. Let M^l be the model obtained using the translation f_l . Therefore, according to Theorem 1 (provided f_l preserves complement)

$$[\varphi]_{f_l(\rho)}^{M_l}(s) = x \text{ iff } [\varphi]_{\rho}^M(s) \in f_l^{-1}(x)$$

whence

$$[\varphi]_{f_l(\rho)}^{M_l}(s) = \mathbf{T} \text{ iff } [\varphi]_{\rho}^M(s) \in \uparrow l$$

Thus,

$$[\varphi]_{\rho}^M(s) = \bigcup \{l \in \mathcal{JI}(\mathcal{L}) \mid [\varphi]_{f_l(\rho)}^{M_l}(s) = \mathbf{T}\} \quad (8)$$

Now, we give a sufficient condition for f_l to preserve complement.

THEOREM 3. Let f_l be the translation function defined as in Theorem 2 for $l \in \mathcal{JI}(\mathcal{L})$ satisfying the following condition: for each $x \in L$, $x \in \uparrow l$ iff $\bar{x} \in L \setminus \uparrow l$. Then, f_l preserves complement, i.e., $f_l(\bar{x}) = \overline{f_l(x)}$ for each $x \in L$.

PROOF. By the definition of f_l , we have $f_l(\bar{x}) = \mathbf{T}$ iff $\bar{x} \in \uparrow l$. By the condition assumed, the latter holds iff $\bar{x} \in L \setminus \uparrow l$. As $\bar{\bar{x}} = x$, this is true iff $f_l(x) = \mathbf{F}$, whence indeed $f_l(\bar{x}) = \overline{f_l(x)}$. $\square$

EXAMPLE 5. For three lattices and their join-irreducible elements l , we show which of the corresponding functions f_l preserve (Figure 4) and which do not preserve (Figure 5) complement. Notice that no translation from **3** preserves complement. This explains why all the existing translations require duplication of states in the translated models. For the lattice **4** only f_S preserves complement, for 2×2 both the functions f_a and f_b preserve complement, whereas for $2 + 4$ only f_{DK} and f_{DC} do so. The functions f_l for the remaining join-irreducible elements l of the above lattices do not preserve complement. To mark a translation graphically, we adopt the convention that the lattice elements lying above the extra line are superimposed on the lattice structure mapped to $\mathbf{T}$, whereas the elements below the line are mapped to $\mathbf{F}$. Notice that for the lattice $2 + 4$ two translations are shown in Figure 4.

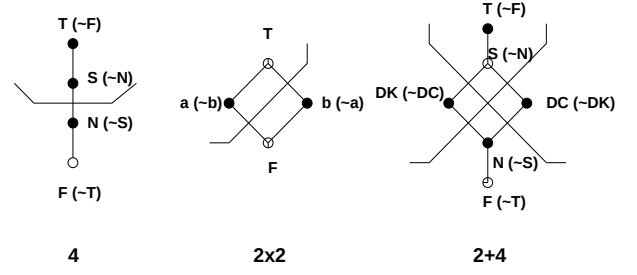


Figure 4: Translations preserving complement

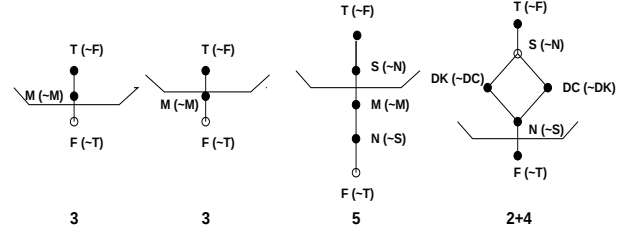


Figure 5: Translations not preserving complement

6. FINAL REMARKS

We have combined here three different logical frameworks: the logic of knowledge with interpreted system semantics, temporal logic and modal μ -calculus. Motivated by the cases of incomplete or inconsistent information occurring in real life applications, we have equipped the resulting cross-framework language in a logical domain being an arbitrary De Morgan lattice. We have obtained a new, unified mv logic of knowledge and time with a great expressive power, providing a universal framework for reasoning simultaneously about the phenomena related to knowledge and properties of concurrent systems in a multi-valued environment, and featuring additionally the fixpoint operators of μ -calculus. We have shown that the general method of model checking for mv temporal logic and mv μ -calculus via a translation to standard, two-valued models which preserves the lattice bounds, developed in our earlier papers, can be extended to the above unified logic, and is characterized by greater efficiency if the translation preserves complement in addition to the bounds. We have also given several examples illustrating the motivation for this type of framework and supporting practical applicability of our improvement.

7. REFERENCES

- [1] G. Bruns and P. Godefroid. Generalized model checking: Reasoning about partial state spaces. In *Proc. of the 11th Int. Conf. on Concurrency Theory (CONCUR'00)*, volume 1877 of *LNCS*, pages 168–182. Springer-Verlag, 2000.
- [2] G. Bruns and P. Godefroid. Model checking with multi-valued logics. In *ICALP*, pages 281–293, 2004.
- [3] M. Chechik, B. Devereux, and S. Easterbrook. Implementing a multi-valued symbolic model checker. In *Proc. of the 7th Int. Conf. on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'01)*, volume 2031 of *LNCS*, pages 404–419.

- Springer-Verlag, 2001.
- [4] E. M. Clarke, O. Grumberg, and D. Peled. *Model Checking*. MIT Press, 1999.
 - [5] S. Easterbrook and M. Chechik. A framework for multi-valued reasoning over inconsistent viewpoints. In *Proc. of the 23rd Int. Conf. on Software Engineering (ICSE'01)*, pages 411–420. IEEE Computer Society, 2001.
 - [6] R. Fagin, J. Y. Halpern, Y. Moses, and M. Vardi. *Reasoning about Knowledge*. MIT Press, Cambridge, 1995.
 - [7] M. Fitting. Many-valued modal logics. *Fundamenta Informaticae*, 15(3-4):335–350, 1991.
 - [8] M. Fitting. Many-valued modal logics II. *Fundamenta Informaticae*, 17:55–73, 1992.
 - [9] P. Godefroid, M. Huth, and R. Jagadeesan. Abstraction-based model checking using modal transition systems. In *Proc. of the 12th Int. Conf. on Concurrency Theory (CONCUR'01)*, volume 2154 of *LNCS*, pages 426–440. Springer-Verlag, 2001.
 - [10] P. Godefroid and R. Jagadeesan. On the expressiveness of 3-valued models. In *Proc. of the 4th Int. Conf. on Verification, Model Checkig, and Abstract Interpretation (VMCAI'03)*, volume 2575 of *LNCS*, pages 206–222. Springer-Verlag, 2003.
 - [11] A. Gurfinkel and M. Chechik. Multi-valued model checking via classical model checking. In *Proc. of the 14th Int. Conf. on Concurrency Theory (CONCUR'03)*, volume 2761 of *LNCS*, pages 266–280. Springer-Verlag, 2003.
 - [12] W. Hoek and M. Wooldridge. Model checking knowledge and time. In *Proc. of the 9th Int. SPIN Workshop (SPIN'02)*, volume 2318 of *LNCS*, pages 95–111. Springer-Verlag, 2002.
 - [13] W. Hoek and M. Wooldridge. Tractable multiagent planning for epistemic goals. In *Proc. of the 1st Int. Conf. on Autonomous Agents and Multi-Agent Systems (AAMAS'02)*, volume III, pages 1167–1174. ACM, July 2002.
 - [14] M. Huth, R. Jagadeesan, and D. A. Schmidt. A domain equation for refinement of partial systems. *Mathematical Structures in Computer Science*, 14:469–505, 2004.
 - [15] M. Huth and S. Pradhan. Consistent partial model checking. In *Proc. of the Workshop on Domains VI*, volume 73 of *ENTCS*, pages 45–85. Elsevier, 2004.
 - [16] W. Jamroga and W. Hoek. Agents that know how to play. *Fundamenta Informaticae*, 63(2-3):185–219, 2004.
 - [17] M. Kacprzak, A. Lomuscio, and W. Penczek. From bounded to unbounded model checking for temporal epistemic logic. *Fundamenta Informaticae*, 63(2-3):221–240, 2004.
 - [18] M. Kacprzak, A. Lomuscio, and W. Penczek. Verification of multi-agent systems via unbounded model checking. In *Proc. of the 3rd Int. Conf. on Autonomous Agents and Multi-Agent Systems (AAMAS'04)*, volume II, pages 638–645. IEEE Computer Society, July 2004.
 - [19] M. Kacprzak and W. Penczek. Unbounded model checking for alternating-time temporal logic. In *Proc. of the 3rd Int. Conf. on Autonomous Agents and Multi-Agent Systems (AAMAS'04)*, volume II, pages 646–653. IEEE Computer Society, July 2004.
 - [20] M. Kacprzak and W. Penczek. Fully symbolic unbounded model checking for alternating-time temporal logic. *Autonomous Agents and Multi-Agent Systems*, 11(1):69 – 89, 2005.
 - [21] B. Konikowska and W. Penczek. Model checking multi-valued modal μ -calculus revisited. In *Proc. of the Int. Workshop on Concurrency, Specification and Programming (CS&P'04)*, volume 170(2) of *Informatik-Berichte*, pages 307–318.
 - [22] B. Konikowska and W. Penczek. Model checking for multi-valued computation tree logics. In M. Fitting and E. Orłowska, editors, *Beyond Two: Theory and Applications of Multiple Valued Logic*, pages 193–210. Physica-Verlag, 2003.
 - [23] A. Lomuscio, T. Lasica, and W. Penczek. Bounded model checking for interpreted systems: Preliminary experimental results. In *Proc. of the 2nd NASA Workshop on Formal Approaches to Agent-Based Systems (FAABS'02)*, volume 2699 of *LNAI*, pages 115–125. Springer-Verlag, 2003.
 - [24] R. Meyden and N. V. Shilov. Model checking knowledge and time in systems with perfect recall. In *Proc. of the 19th Conf. on Foundations of Software Technology and Theoretical Computer Science (FSTTCS'99)*, volume 1738 of *LNCS*, pages 432–445. Springer-Verlag, 1999.
 - [25] W. Penczek and A. Lomuscio. Verifying epistemic properties of multi-agent systems via bounded model checking. *Fundamenta Informaticae*, 55(2):167–185, 2003.
 - [26] F. Raimondi and A. Lomuscio. A tool for specification and verification of epistemic properties of interpreted systems. In *Proc. of the Int. Workshop on Logic and Communication in Multi-Agent Systems (LCMAS'03)*, volume 85(2) of *ENTCS*. Elsevier, 2003.
 - [27] F. Raimondi and A. Lomuscio. Verification of multiagent systems via ordered binary decision diagrams: An algorithm and its implementation. In *Proc. of the 3rd Int. Conf. on Autonomous Agents and Multi-Agent Systems (AAMAS'04)*, volume II, pages 630–637. IEEE Computer Society, July 2004.
 - [28] F. Raimondi and A. Lomuscio. Symbolic model checking of multi-agent systems using OBDDs. In *Proc. of the 3rd NASA Workshop on Formal Approaches to Agent-Based Systems (FAABS III)*, volume 3228 of *LNCS*, pages 213–221. Springer-Verlag, 2005.
 - [29] A. Tarski. A lattice-theoretical fixpoint theorem and its applications. *Pacific Journal of Mathematics*, 5:285–309, 1955.
 - [30] M. Wooldridge. *An Introduction to Multiagent Systems*. John Wiley & Sons, 2002.
 - [31] B. Woźna, A. Lomuscio, and W. Penczek. Bounded model checking for deontic interpreted systems. *Electr. Notes Theor. Comput. Sci.*, 126:93–114, 2005.
 - [32] B. Woźna, A. Lomuscio, and W. Penczek. Bounded model checking for knowledge and real time. In *Proc. of the 4th Int. Conf. on Autonomous Agents and Multi-Agent Systems (AAMAS'05)*, pages 165–172. IEEE Computer Society, 2005.