

An Investigation of Concept Learning by a Monte-Carlo Tree Search of Argumentations

Régis Riveret¹, Pasquale Caianiello², Stefania Costantini² and Moez Draief¹

¹ Imperial College, London, United Kingdom

`{r.riveret,m.draief}@imperial.ac.uk`

² Università degli Studi dell'Aquila, L'Aquila, Italy

`{pasquale.caianiello,stefania.costantini}@univaq.it`

Abstract. Monte-Carlo Tree Search (MCTS) is a heuristic to search in large trees. We employ it for concept learning in argumentation: MCTS pursues the best argumentation meant to account for a possibly inconsistent set of training examples. We provide experimental results of our approach with the search variant Upper Confidence Bounds on Tree (UCT).

1 Introduction

Concept learning regards the problem of searching through a space of hypotheses for the hypotheses fitting training examples. The problem has attracted considerable attention in machine learning and the approaches range from logic induction to neural networks or genetic algorithms (see [9]). In particular logics are often proposed to hold a richer representation language of the domain knowledge than the basic pairs attribute-value of bare learning systems. However, formal argumentation has been barely considered for concept learning from inconsistent dataset of training examples (c.f. [1,7,12]).

In this paper, we approach concept learning with argumentation where the space of the hypotheses is constituted from possible argumentation graphs and such that the grounded extension of a hypothetical argumentation is meant to account for a possibly inconsistent set of training examples.

The possible exponential explosion of hypothetical argumentations calls for efficient heuristics that have not been investigated so far. In this regard, when the domain can be modelled as a tree and in particular if the search tree has a large size and no strategic or tactical knowledge about the given domain exist, Monte-Carlo Tree Search (MCTS) is a heuristic that may give interesting results (see [4] for a recent survey). So, it has notably gained success in the hard problem of playing the game Go, and has also shown effective in other applications with no knowledge beyond the constituting rules. The relative simplicity of MCTS

Proceedings of the 21st RCRA workshop on *Experimental Evaluation of Algorithms for Solving Problems with Combinatorial Explosion* (RCRA 2014).

Vienna, Austria, July 17–18, 2014.

is congruent to the heuristics of human players: both exploit repetitively and incrementally promising lines of strategies while occasionally exploring a priori weaker options.

We investigate concept learning and argumentation with a MCTS heuristic by structuring the search space as a tree. As both argumentation and MCTS feature natural phenomena (human argumentation for formal argumentation, reinforcement learning and swarm intelligence for MCTS), our integration is meant to preserve these biologic and cognitive flavours by exploiting and exploring hypothesis through (single agent internal) dialogues.

This paper is organised as follows. The problem is set in Section 2. We introduce our argumentation framework in Section 3. An evaluation of hypothetical argumentations is proposed in 4. Monte-Carlo Tree Search is presented in 5, and its integration with argumentation in Section 6. Experimental results are given in Section 7. Our approach is discussed with respect to related work in Section 8, before concluding.

2 Problem setting

Concept learning regards the problem of searching through a space of hypotheses for the hypotheses best fitting (possibly inconsistent) training examples. In our case, a hypothesis is an argumentation, and a training example relates an instance, i.e. a set of positive literals labelled either “in”, “out” or “un”, with a target concept labelled either “in”, “out” or “un”. A literal is an atom or its negation, a positive literal is an atom. Informally, a literal labelled “in” means that the referent is observed as ‘true’, a label “out” means that the referent is observed as ‘false’ while “un” means that the referent is observed as ‘undecided’. A dataset is a set of training examples, conveniently represented in a table (see e.g. Table 1) where each line is a training example relating an instance with a target concept.

There is no formal definition of the problem of concept learning that is commonly accepted, so we adopt the following formalisation. Given:

1. A set Φ of observation features, each feature is modelled as a positive literal ϕ_i (e.g. $\Phi = \{a, b, c, d\}$);
2. A space $\mathfrak{B}$ of instances. An instance B is a labelling of all literals in Φ , the labelling of a literal is modelled as a total function $\Phi \rightarrow \{\text{in}, \text{out}, \text{un}\}$;
3. A target concept modelled as a pair (ϕ, C^ϕ) where ϕ is a positive literal and C^ϕ is a total function $C^\phi : \mathfrak{B} \rightarrow \{\text{in}, \text{out}, \text{un}\}$ associating to any instance B a label in the set $\{\text{in}, \text{out}, \text{un}\}$. In the following we will address to a target concept (ϕ, C^ϕ) simply as C^ϕ .
4. A space $\mathfrak{H}$ of hypothetical argumentations; and a labelling of any hypothetical argumentation, modelled as a total function $H^\phi : \mathfrak{B} \times \mathfrak{H} \rightarrow \{\text{in}, \text{out}, \text{un}\}$;
5. A set D of training examples where a training example is a pair $(B, C^\phi(B))$ associating an instance B with the target concept value $C^\phi(B)$.

The goal is to find a hypothetical argumentation $\mathcal{H} \in \mathfrak{H}$ such that a fitness measure $\mu(H^\phi(B, \mathcal{H}), C^\phi(B))$ is maximal for each instance B in the set of training

examples D . Typically, $\mu(H^\phi(B, \mathcal{H}), C^\phi(B))$ shall be 1 if $H^\phi(B, \mathcal{H}) = C^\phi(B)$ and 0 otherwise. However this goal may not be adequate when a set of examples is inconsistent, i.e. it contains two examples with the same instances but with different target concept labelling. Inconsistencies may appear in a noisy, non-monotonic or temporal setting. Inconsistencies can also appear because of unobserved features/variables. So, a humbler goal shall be to find a hypothesis maximising $\mu(H^\phi(B, \mathcal{H}), C^\phi(B))$ over the training examples.

	a	b	d	c
F ₁	in	in	out	out
F ₂	in	out	out	in
F ₃	in	out	un	un
.	.	.	.	.
F _i	.	.	.	.

Table 1. A dataset, i.e. a set of training examples, represented as a table. Each line is an example. c is the literal associated to the concept target.

As for notation in the remainder, $\text{in}(S)$ will denote the set of literals labelled in in a set of labelled literals S , $\text{out}(S)$ the set of literals labelled out, and $\text{un}(S)$ the set of literals labelled un.

3 Argumentation framework

We aim at integrating MCTS with argumentation for concept learning: MCTS shall pursue the best argumentations with respect to a set of training examples. In this section, we present our argumentation setting. We begin with the definition of an argument.

Definition 1 (Argument). *An argument is set of premises (a set of literals) supporting a conclusion (a literal).*

The premises of an argument A is denoted $\text{prem}(A)$, and its conclusion $\text{conc}(A)$, and we shall represent this argument as a ‘jumping’ directed association of the form $A : \phi_1, \dots, \phi_n \Rightarrow \phi$ where $\text{prem}(A) = \{\phi_1, \dots, \phi_n\}$ and $\text{conc}(A) = \phi$.

Arguments may be incompatible. An argument is incompatible with another if (but not only if) their conclusions are incompatible. Arguments may be incompatible even if their conclusions are compatible (i.e. not incompatible), for example when an argument undercuts another. Let ψ be an atom, and “ $-$ ” denote a complement function such that $-\psi = \neg\psi$ and $-\neg\psi = \psi$. We assume that two arguments A and B are incompatible if $\text{conc}(A) = -\text{conc}(B)$. Incompatibilities shall be resolved on the basis of preferences over arguments: if an argument A is preferred over another argument B , then it is denoted as $A \succ B$ (we cannot have $A \succ B$ and $B \succ A$).

Definition 2 (Attack). *An argument B attacks an argument A if, and only if, they are incompatible and $A \not\sim B$.*

Note that if there is an attack between two arguments we can deduce that they are incompatible. Thus the direction of any attack inform about the preference over these arguments.

Example 1. Let's assume the following arguments:

$$B : a, b \Rightarrow x \quad C : a \Rightarrow c \quad D : d \Rightarrow x$$

The argument B attacks C and C does not attack B if, and only if, these two arguments are incompatible and $B \succ C$. The arguments C and D attack each other if, and only if, they are incompatible and $D \not\sim C$ and $C \not\sim D$.

A set of arguments and attacks amongst them constitute an argumentation graph that we define following Dung [6].

Definition 3 (Argumentation graph). *An argumentation graph is a pair $\langle \mathcal{A}, \rightsquigarrow \rangle$ where $\mathcal{A}$ is a set of arguments, and $\rightsquigarrow \subseteq \mathcal{A} \times \mathcal{A}$ is a binary relation of attack. In particular we assume that for any argument A and B , $B \rightsquigarrow A$ if, and only if, B attacks A .*

As for notation, given an argumentation graph $\mathcal{G} = \langle \mathcal{A}, \rightsquigarrow \rangle$, we have $\text{arg}(\mathcal{G}) = \mathcal{A}$ and $\rightsquigarrow(\mathcal{G}) = \rightsquigarrow$.



Fig. 1. An argumentation graph. The argument B attacks the argument C . The arguments C and D attack each other. Since B and C have compatible conclusions, B undercuts C . For a similar reason, C and D undercut each other.

We will generate a set of hypothetical arguments, we will search for the possible attacks amongst these arguments. By doing so, we shall consider multiple hypothetical graphs, so that the set of attacks of each graph will imply a set of incompatibilities and preferences amongst these arguments.

Once we have an argumentation graph, we can compute the set of arguments that are justified or rejected, that is, those arguments that shall survive or not to the possible attacks. Many semantics exist and, for our purposes, we shall only consider the Dung's grounded semantics [6] by labelling arguments as in [10]. Each argument is associated with one label, which is either *in*, *out* or *un*. A label “in” means the argument is justified while a label “out” indicates that it is discarded. The label “un” marks the status of the argument as undecided.

Definition 4 (Labelling Argument). *Let $\langle \mathcal{A}, \rightsquigarrow \rangle$ be an argumentation graph. A labelling is a total function $L : \mathcal{A} \rightarrow \{\text{in}, \text{out}, \text{un}\}$.*

Roughly speaking, an argument labelled **in** means that it is justified (we may say “true”) as well as its conclusion. We write $\text{in}(L)$ for $\{A \mid L(A) = \text{in}\}$, $\text{out}(L)$ for $\{A \mid L(A) = \text{out}\}$ and $\text{un}(L)$ for $\{A \mid L(A) = \text{un}\}$. In the remainder, a labelling L shall be represented as a triple $(\text{in}(L), \text{out}(L), \text{un}(L))$.

Definition 5 (Complete labelling). *Let $\langle \mathcal{A}, \rightsquigarrow \rangle$ be an argumentation graph. A complete labelling is a labelling such that for every $A \in \mathcal{A}$ it holds that:*

- A is labelled **in** if, and only if, all attackers of A are labelled **out**,
- A is labelled **out** if, and only if, A has an attacker that is labelled **in**.

The grounded labelling is characterized as the complete labelling maximising **un** [10].

Definition 6 (Grounded labelling). *Let $\mathcal{G} = \langle \mathcal{A}, \rightsquigarrow \rangle$ be an argumentation graph. L is the grounded labelling if, and only if L is a complete labelling where $\text{un}(L)$ is maximal (w.r.t. set inclusion) among all complete labellings of $\mathcal{G}$.*

An argumentation graph has a unique grounded labelling[6,10]. An algorithm for generating the grounded labelling of an argumentation graph as proposed in [6] is given in Algorithm 1 .

Algorithm 1 Computation of the Grounded Labelling

```

 $L_0 = (\emptyset, \emptyset, \emptyset),$ 
repeat
   $\text{in}(L_{i+1}) = \text{in}(L_i) \cup \{A \mid A \text{ is not labelled in } L_i, \text{ and } \forall B : \text{if } B \text{ attacks } A \text{ then } B \in \text{out}(L_i)\}$ 
   $\text{out}(L_{i+1}) = \text{out}(L_i) \cup \{A \mid A \text{ is not labelled in } L_i, \text{ and } \exists B : B \text{ attacks } A \text{ and } B \in \text{in}(L_{i+1})\}.$ 
until  $L_i = L_{i+1}$ 
Return  $(\text{in}(L_i), \text{out}(L_i), A - (\text{in}(L_i) \cup \text{out}(L_i)))$ 

```

The grounded extension of an argumentation graph is the set of arguments labelled **in** according to the grounded labelling. We say:

- a literal ϕ is labelled **in** with respect to an argumentation graph $\mathcal{G}$, denoted $\text{in}(\phi, \mathcal{G})$, if and only if, ϕ is the conclusion of an argument labelled **in**, or $-\phi$ is labelled **out**,
- a literal ϕ is labelled **out** with respect to an argumentation graph $\mathcal{G}$, denoted $\text{out}(\phi, \mathcal{G})$, if and only if, $-\phi$ is the conclusion of an argument labelled **in**, or all the arguments with the conclusion ϕ are labelled **out**,
- a literal ϕ is labelled **un** with respect to an argumentation graph $\mathcal{G}$, denoted $\text{un}(\phi, \mathcal{G})$, if and only if, it is neither labelled **in** nor labelled **out**.

In the remainder we adopt the following notation for the sets of labelled literals:

- $\text{in}(\mathcal{G}) = \{\phi \mid \text{in}(\phi, \mathcal{G})\}$: the set of literals labelled **in**,

- $\text{un}(\mathcal{G}) = \{\phi | \text{un}(\phi, \mathcal{G})\}$: the set of literals labelled **un**,
- $\text{out}(\mathcal{G}) = \{\phi | \text{out}(\phi, \mathcal{G})\}$: the set of literals labelled **out**.

Example 2 (continued). The arguments **B** and **D** are labelled **in**, the argument **C** is labelled **out**. Hence, the literal **x** is labelled **in**, and the literal **c** is **out**.

To measure the similarity of the grounded labellings of two argumentation graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ with respect to a set of literals Φ , we define a Jaccard flavoured similarity coefficient denoted $s(\Phi, \mathcal{G}_1, \mathcal{G}_2)$:

$$s(\Phi, \mathcal{G}_1, \mathcal{G}_2) = \frac{\mu_{XX}}{\mu_{XY} + \mu_{YX} + \mu_{XX}} \quad (1)$$

where

$$\mu_{XX} = |\Phi \cap \text{in}(\mathcal{G}_1) \cap \text{in}(\mathcal{G}_2)| + |\Phi \cap \text{out}(\mathcal{G}_1) \cap \text{out}(\mathcal{G}_2)| + |\Phi \cap \text{un}(\mathcal{G}_1) \cap \text{un}(\mathcal{G}_2)| \quad (2)$$

$$\mu_{XY} = |\Phi \cap \text{in}(\mathcal{G}_1) \cap \text{out}(\mathcal{G}_2)| + |\Phi \cap \text{in}(\mathcal{G}_1) \cap \text{un}(\mathcal{G}_2)| + |\Phi \cap \text{out}(\mathcal{G}_1) \cap \text{un}(\mathcal{G}_2)| \quad (3)$$

$$\mu_{YX} = |\Phi \cap \text{out}(\mathcal{G}_1) \cap \text{in}(\mathcal{G}_2)| + |\Phi \cap \text{out}(\mathcal{G}_1) \cap \text{un}(\mathcal{G}_2)| + |\Phi \cap \text{un}(\mathcal{G}_1) \cap \text{out}(\mathcal{G}_2)| \quad (4)$$

and $s(\Phi \cap \mathcal{G}_1, \mathcal{G}_2) = 1$ when the denominator is zero. Hence $s(\Phi \cap \mathcal{G}_1, \mathcal{G}_2) \in [0, 1]$: it is 0 when the graphs do not share any label with respect to Φ , it is 1 when all the labels match.

4 Instantiating and evaluating hypotheses

Recall that a training example F is a pair relating an instance B with a label $C^\phi = C^\phi(B)$ of the target concept ϕ , so $F = (B, C^\phi(B))$. Since the constituents of an example play no role in our argumentation setting, we reify them into arguments. So we associate an example F with an argumentation pair of an instance argumentation $\mathcal{B}$ accounting for the instance and a target concept argumentation $\mathcal{C}$ catering for the target concept, so $\mathcal{F} = (\mathcal{B}, \mathcal{C}^\phi)$ such that $\mathcal{B} = \mathcal{B}^{\text{in}} \cup \mathcal{B}^{\text{out}} \cup \mathcal{B}^{\text{un}}$ and $\mathcal{C}^\phi = \mathcal{C}^{\text{in}} \cup \mathcal{C}^{\text{out}} \cup \mathcal{C}^{\text{un}}$ where:

$$\mathcal{B}^{\text{in}} = \langle \{X : \Rightarrow \phi | \phi \in \text{in}(B)\}, \emptyset \rangle \quad \mathcal{C}^{\text{in}} = \langle \{ \Rightarrow \phi | C^\phi(B) = \text{in} \}, \emptyset \rangle$$

$$\mathcal{B}^{\text{out}} = \langle \{X : \Rightarrow -\phi | \phi \in \text{out}(B)\}, \emptyset \rangle \quad \mathcal{C}^{\text{out}} = \langle \{ \Rightarrow -\phi | C^\phi(B) = \text{out} \}, \emptyset \rangle$$

$$\mathcal{B}^{\text{un}} = \langle \{X : \Rightarrow \phi, Y : \Rightarrow -\phi | \phi \in \text{un}(B)\}, \{X \rightsquigarrow Y, Y \rightsquigarrow X\} \rangle$$

$$\mathcal{C}^{\text{un}} = \langle \{X : \Rightarrow \phi, Y : \Rightarrow -\phi | C^\phi(B) = \text{un}\}, \{X \rightsquigarrow Y, Y \rightsquigarrow X\} \rangle$$

so that the labelling of an instance matches the grounded labelling of the instance and target concept argumentation:

- $C^\phi(B_i) = \text{in}$ if and only if $\phi \in \text{in}(\mathcal{C}_i^\phi)$, and $\text{in}(B_i) = \text{in}(\mathcal{B}_i)$,
- $C^\phi(B_i) = \text{out}$ if and only if $\phi \in \text{out}(\mathcal{C}_i^\phi)$, and $\text{out}(B_i) = \text{in}(\mathcal{B}_i)$,
- $C^\phi(B_i) = \text{un}$ if and only if $\phi \in \text{un}(\mathcal{C}_i^\phi)$, and $\text{un}(B_i) = \text{un}(\mathcal{B}_i)$.

We consider the argumentation graph of a hypothetical argument $\mathcal{H}$ instantiated by B , denoted $\mathcal{H} \cdot B$:

$$\mathcal{H} \cdot B = \langle \arg(\mathcal{H}) \cup \arg(\mathcal{B}^{\text{in}}) \cup \arg(\mathcal{B}^{\text{out}}) \cup \arg(\mathcal{B}^{\text{un}}), \rightsquigarrow (\mathcal{H}) \cup \rightsquigarrow^{\text{out}} \cup \rightsquigarrow^{\text{un}} \rangle$$

where:

$$\rightsquigarrow^{\text{out}} = \{X \rightsquigarrow A \mid X \in \mathcal{B}^{\text{out}}, A \in \mathcal{A}, -\text{conc}(X) \in \text{prem}(A)\}$$

$$\rightsquigarrow^{\text{un}} = \{X \rightsquigarrow A \mid X \in \mathcal{B}^{\text{un}}, A \in \mathcal{A}, -\text{conc}(X) \in \text{prem}(A)\}$$

The labelling H^ϕ of an argumentative hypothesis (see Section 2) is defined by the grounded labelling of the instantiation $\mathcal{H} \cdot B$ of the associated hypothetical argumentation graph $\mathcal{H}$ with the instance B :

- $H^\phi(B, \mathcal{H}) = \text{in}$ if and only if $\phi \in \text{in}(\mathcal{H} \cdot B)$,
- $H^\phi(B, \mathcal{H}) = \text{out}$ if and only if $\phi \in \text{out}(\mathcal{H} \cdot B)$,
- $H^\phi(B, \mathcal{H}) = \text{un}$ if and only if $\phi \in \text{un}(\mathcal{H} \cdot B)$.

We evaluate a hypothetical argumentation graph $\mathcal{H}$ with respect to a training example $(B, C^\phi(B))$ by measuring the similarity of the grounded labelling of argumentation graphs $\mathcal{H} \cdot B$ and C^ϕ . Thus the measure $\mu(H^\phi(B, \mathcal{H}), C^\phi(B))$ is the similarity coefficient $s(\{\phi, -\phi\}, \mathcal{H} \cdot B, C^\phi)$.

Example 3 (continued). Consider the hypothetical argumentation graph $\mathcal{H} = \langle \{\mathbf{a} \Rightarrow \mathbf{c}\}, \emptyset \rangle$ and let's evaluate it with respect to the second training example F_2 (second line) given in Table 1. We have $\text{in}(\mathcal{H} \cdot B_2) = \{\mathbf{c}\}$, $\text{out}(\mathcal{H} \cdot B_2) = \{\neg \mathbf{c}\}$ and $\text{un}(\mathcal{H} \cdot B_2) = \emptyset$, while $\text{in}(C_2) = \{\mathbf{c}\}$, $\text{out}(C_2) = \{\neg \mathbf{c}\}$ and $\text{un}(C_2) = \emptyset$. The resulting similarity coefficient is maximal: $s(\{\mathbf{c}, \neg \mathbf{c}\}, \mathcal{H} \cdot B_2, C_2) = 1$.

When considering a finite dataset $D = \{F_1, \dots, F_N\}$, the overall similarity coefficient of a hypothetical argumentation with respect to this dataset is the average over the training examples:

$$S(\Phi, \mathcal{H}, D) = \frac{\sum s(\Phi, \mathcal{H} \cdot B_i, C_i)}{N} \quad (5)$$

When evaluating an argumentative hypothesis, we could consider other features, for example their complexity to avoid over-fitting and promote parsimony. These potential improvements will be studied in future work.

5 Monte-Carlo Tree Search (MCTS)

MCTS combines tree search with Monte-Carlo sampling. A search tree of sequential state-actions is iteratively grown according to the results and the estimation of the values of decisions are increasingly improved. In the search tree, a node represents a state, and links to child nodes represent actions leading to subsequent states. The basic MCTS iteration has four phases, repeated until a computational budget (e.g. a time or a memory budget) is reached:

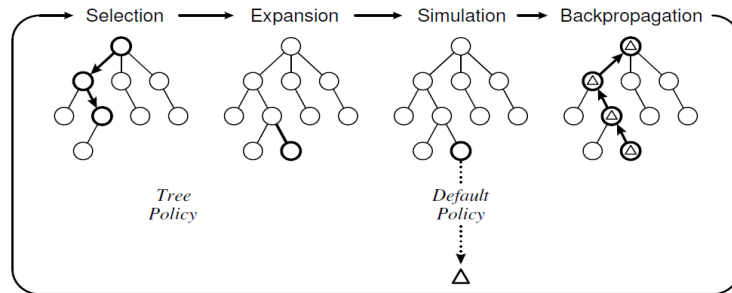


Fig. 2. One iteration of the general MCTS approach [4].

1. Selection: the tree is traversed following a child selection policy from the root node to a node with unvisited children,
2. Expansion: if the selected leaf node is expandable (i.e. it does not represent a terminal state) then one (or more) node is added to expand the tree,
3. Simulation: the nodes selection is completed following a default policy from the newly added node to a terminal node, obtaining thus an outcome,
4. Backpropagation: the outcome is backpropagated through the tree, i.e., in each node traversed the average outcome of the simulations is computed.

An illustration of one MCTS iteration is given in Figure 2. Nodes are recursively selected following a tree policy (typically based on some valuation of nodes) from the root to a leaf node which is terminal or a non-expanded node. If the node is expandable, then an action is selected and thus a new node is visited. The simulation is another selection phase from the last visited node to a terminal node where the nodes are selected following a default policy resulting in a value Δ . This value is back-propagated to the nodes visited so far to update their statistics. Typically, an average value is propagated backwards and a visit counter is incremented. The back-propagation following every iteration of the algorithm ensures that nodes' statistics are always up-to-date. Hence, this makes MCTS an any-time algorithm and any additional iteration shall improve the result.

A common tree policy is to select nodes with the highest optimistic estimated value. A simple implementation of this policy is to maximise an upper confidence bounds (UCB) [2]:

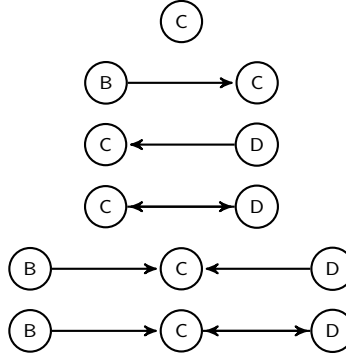
$$v_i + C \sqrt{\frac{\ln(N)}{n_i}} \quad (6)$$

where $v_i = V_i/n_i$ is the estimated value of the node, V_i is the sum of back-propagated value on the node, n_i is the number of the times the node has been visited, N is the total number of times that its parent has been visited, and C is a tunable bias parameter balancing the exploration and the exploitation of states. So, the more a node has been visited (and the more certain is its estimated value), the less it shall be visited. When UCB is applied to MCTS, we obtain the Upper Confidence Bounds on Trees (UCT) algorithm [8], which

has the property of being consistent, that is, given enough time, it will find the optimal node with a probability tending to one.

6 A Monte-Carlo tree search of hypothetical argumentations

Argumentation frameworks involve argumentation graphs, while MCTS is tree-based. To reconcile them, it is natural to consider the standard and well-established notion of dialogue trees in argumentation. Dialogue trees between a proponent and an opponent are commonly proposed to compute whether an argument is justified, i.e. belongs to the grounded extension (see e.g. [13]). Other protocols (e.g. [16]) are more flexible to deal with withdrawals or unplayed arguments for example. These protocols are not convenient for our purposes because, instead of arguing about the justification of an argument, we are interested by the generation and the selection of hypothetical argumentations.



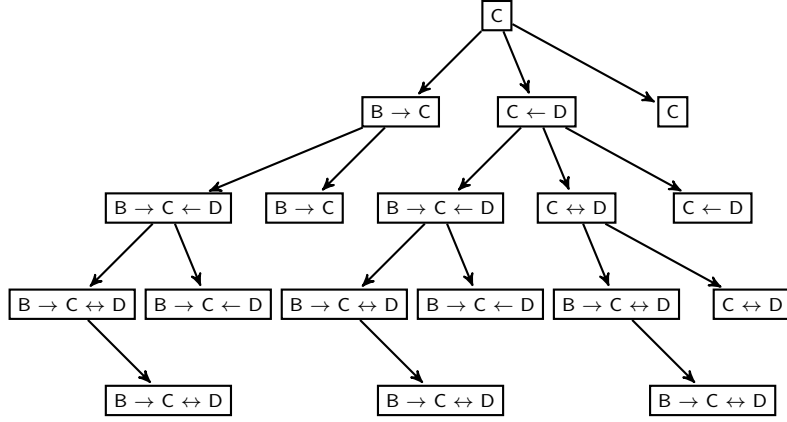


Fig. 4. A fully expanded tree of hypothetical argumentations based on the hypothetical argumentation framework of Figure 1. The explanandum is c and the unique argumentum is C . Each link is a move representing either an attack or a withdraw. Any node reached by a path induces a hypothetical argumentation graph.

a hypothesis is a sub-graph (of this hypothetical framework) induced by the argumentative procedure (see Figure 3 for an illustration of possible hypotheses).

The procedure generates hypotheses by constructing a *tree of hypothetical argumentations* where each node of this tree is a hypothesis. A branch of a tree of hypothetical argumentations is a finite non-empty sequence of moves $\text{move}_1, \dots, \text{move}_n$, where any move move_i is either an argumentum never previously moved, a withdraw or a valid attack. An attack of a move $\text{move}_i, i > 1$ is valid if and only if it counter-attacks any previous attack and it has not been previously moved.

A tree of hypothetical argumentations is all possible sequences of moves. Each node of an argumentation tree induces a hypothetical argumentation graph. The space of possible hypothetical argumentations within a tree built from an argumentation framework is the space of hypothesis $\mathfrak{H}$ (see Section 2).

We employ MCTS to pursue the best hypothetical argumentations accounting for a dataset within the tree generated from the argumentation procedure, a hypothetical argumentation framework and the target explanandum. Accordingly, the back-propagated value of a terminal hypothetical argumentation $\mathcal{H}$ with respect to a dataset D is the similarity coefficient $S(\{\phi, -\phi\}, \mathcal{H}, D)$.

The argumentation procedure possibly implies a large branching factor for the tree of hypothetical argumentations so that the performance of MCTS shall be fully appreciated. However, since the construction of the MCTS tree as an argumentation tree implies that several paths shall lead to the same argumentation, we deal with these path redundancies by employing a transposition table (see e.g. [5]) so that each node of an argumentation tree has the stored value of the associated hypothetical argumentation. Note that for common MCTS set-

tings, if moves are reversible then there is the danger of cycles to previous states impeding learning but we assumed directional moves in the sense that attacks could no be retracted. Furthermore, since moved attacks cannot be repeated, cycles are eliminated.

7 Experimental results

In order to get experimental insights, we generated datasets with different degrees of inconsistency and we run an implementation of our integration in the pursue of an argumentation accounting for each dataset. The dataset with the minimal degree of inconsistency, i.e. the consistent dataset, was generated from a source argumentation graph $\mathcal{S} = \langle \{S_1, S_2, S_3\}, \{S_1 \rightsquigarrow S_2, S_2 \rightsquigarrow S_3, S_3 \rightsquigarrow S_2\} \rangle$ (similar to the graph in Fig. 1) where:

$$S_1 : a, b \Rightarrow \neg c \quad S_2 : a \Rightarrow c \quad S_3 : d \Rightarrow \neg c$$

A training example was constructed by instantiating $\mathcal{S}$ with an instance regarding two observations a , b and d . An instance was randomly generated by assuming that the instance arguments $\Rightarrow a$, $\Rightarrow b$ and $\Rightarrow d$ appear independently with probability 0.9 for the first argument and 0.5 for the last ones. Let $\mathcal{B}_i$ denote the corresponding instance argumentation (see Section 4) resulting from a lottery. An example $F_i = (B_i, C_i^c)$ was generated such that:

$$\begin{aligned} \text{in}(B_i) &= \text{in}(\mathcal{B}_i) & \text{in}(C_i^c) &= \text{in}(\mathcal{S} \cdot B_i) \cap \{c\} \\ \text{out}(B_i) &= \text{in}(\mathcal{B}_i) & \text{out}(C_i^c) &= \text{out}(\mathcal{S} \cdot B_i) \cap \{c\} \\ \text{un}(B_i) &= \text{un}(\mathcal{B}_i) & \text{un}(C_i^c) &= \text{un}(\mathcal{S} \cdot B_i) \cap \{c\} \end{aligned}$$

Noisy datasets were generated with a degree of inconsistency p such that p is the probability to shuffle the labels of the target concept for each training example. When shuffled, if the target concept was originality labelling **in** then it was labelled **out**, and vice versa. If the target concept was originally labelled **un**, then it was equiprobably labelled **in** or **out**. A dataset is consistent if its degree of inconsistency is zero, $p = 0$.

To account for the generated datasets, we assumed a hypothetical argumentation framework $\mathfrak{H}$ where the possible arguments were:

$$\begin{array}{ll} A_1 : a, b, d \Rightarrow c & X_1 : a, b, d \Rightarrow x \\ A_2 : a, b \Rightarrow c & X_2 : a, b \Rightarrow x \\ A_3 : a, d \Rightarrow c & X_3 : a, d \Rightarrow x \\ A_4 : d, b \Rightarrow c & X_4 : d, b \Rightarrow x \\ A_5 : a \Rightarrow c & X_5 : a \Rightarrow x \\ A_6 : b \Rightarrow c & X_6 : b \Rightarrow x \\ A_7 : d \Rightarrow c & X_7 : d \Rightarrow x \end{array}$$

such that any argument A_i or X_i could attack each other. For example, X_1 may attack A_1 without attacking A_2 or A_3 . Hence, self-attacking arguments and loops were permitted. The arguments A_i constitute the set of argumenta.

We run our integration of UCT and argumentation over datasets generated with different degrees of inconsistency. The UCT computational parameter C was set at two, $C = 2$. Averaged results regarding the similarity coefficient (see Section 4) of the best argumentation found over time are shown in Fig. 5. The results show the logarithmic convergence of the search towards the best argumentation. They also show the relative robustness of the approach with respect to the degree of inconsistency. Finally, the results illustrate the “any time” character of MCTS as a hypothesis can be returned at any time. This suggests the use of the approach for concept learning on the fly.

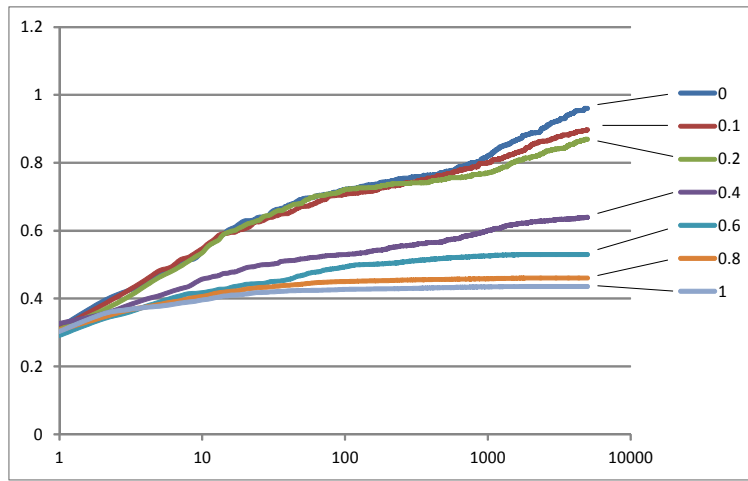


Fig. 5. Time steps versus the similarity coefficient of the best argumentation accounting for datasets. Each line corresponds to a dataset with a different degree of inconsistency. Each dataset has 100 training examples. The results for each degree of inconsistency are averaged over 100 runs.

8 Related work

Logic, and in particular first-order logic, is often associated with machine learning to hold a richer representation language of the domain knowledge than the basic pairs attribute-value of bare learning systems. The induction of hypothesised theories accounting for given examples like cases or precedents has attracted considerable attention in machine learning and the approaches range from logic induction to neural networks or genetic algorithms (see [9]).

Argumentation has already been investigated for machine learning and induction. S. Ontanón et al.[12] investigate the properties of a consequence relation characterising which argumentative hypothesis can be induced from given sets of

examples. B. Johnston and G. Governatori explore in [7] the induction of compact theories from datasets in the legal domain. L. Amgoud and M. Serrurier propose in [1] to construct a version space as the grounded extension of an argumentation graph induced by a set of training examples. In these works [12,7,1], datasets are assumed consistent whereas we relax this assumption by accepting inconsistent set of examples. Notice that all these approaches, arguments and their relations are meant to be the output of the inductive reasoning. M. Mozina et al. [11] propose to add arguments as input data to learning examples to ease the induction of rule-based theories.

Case-based reasoning (CBR) is the most common ground to approach argumentation and machine learning. While CBR is often meant to be not as eager than many inductive learning systems (e.g. IPL systems) to induce theories, many CBR systems do in practice induce theories accounting for a set of cases, most often with respect to an explananda. When CBR shows dialectical features, for example as in legal reasoning, argumentation is a natural mean to model parties arguing about cases. For example, H. Prakken and G. Sartor proposed in [14] an early investigation of rule-based argumentation with a semantics à la Dung for reasoning with precedents represented by a set of rules. In [3], F. Bex led a work on the logical inference to the best explanation: an explanation about an explananda is a story about some observations, and an explanation can be argued about. In these work employing argumentation, no machine learning is involved, but whenever procedures with interacting arguers are modelled with a formal argumentation setting, we believe that MCTS can be used as a heuristic for arguers.

The present work is an application of the abstract framework sketched in [15] where MCTS and formal argumentation have been integrated. To the best of our knowledge, there is no previous proposal of such an integration to search for the best hypothetical argumentations with regard to set of training examples. Interestingly, MCTS can be easily reinterpreted as a form of reinforcement learning, and indeed the present work has been triggered by the work in reinforcement learning and rule-based argumentation investigated in [17]. It is a first attempt using basic MCTS to by-pass the curse of dimensionality for learning large theories encoding multi-agent systems.

9 Conclusion and future directions

We investigated a heuristic, namely MCTS, for concept learning with argumentation. This integration was meant to pursue the best argumentations in large spaces of hypothetical argumentations accounting for a (possibly inconsistent) set of training examples. We hope the reader found inspiration in this line of research regarding an intuitive integration of formal argumentation and machine learning that allows theoretical insights and practical applications, e.g. from learning argumentations accounting for cases of natural phenomenon to the construction of learning argumentative agents.

Future work can be multiple, they range from the use of MCTS variants (see [4] to refinement with regard to the generation and the selection of argumentative hypothesis to prune the search tree. As the argumentation setting in this paper is partially abstract in the sense that the internal structure of arguments is left unspecified, we have the possibility for further investigations with more specific argumentative frameworks. Finally, the anytime character of MCTS also suggests future investigations for learning on the fly, particularly useful for complex event processing.

Acknowledgments. Part of this work is supported by the Marie Curie Intra-European Fellowships PIEF-GA-2012-331472 and has been carried out in the scope of the EC co-funded project SMART (FP7-287583).

References

1. Leila Amgoud and Mathieu Serrurier. An argumentation framework for concept learning. In *11th Intl. Workshop on Non-Monotonic Reasoning (NMR06)*, 2006.
2. Peter Auer, Nicolò Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47(2-3):235–256, 2002.
3. Floris J. Bex, Peter J. van Koppen, Henry Prakken, and Bart Verheij. A hybrid formal theory of arguments, stories and criminal evidence. *Artif. Intell. Law*, 18(2):123–152, 2010.
4. Cameron Browne, Edward J. Powley, Daniel Whitehouse, Simon M. Lucas, Peter I. Cowling, Philipp Rohlfshagen, Stephen Tavener, Diego Perez, Spyridon Samothrakis, and Simon Colton. A survey of monte carlo tree search methods. *IEEE Trans. Comput. Intellig. and AI in Games*, (1):1–43.
5. Benjamin E. Childs, James H. Brodeur, and Levente Kocsis. Transpositions and move groups in monte carlo tree search. In Philip Hingston and Luigi Barone, editors, *CIG*, pages 389–395. IEEE, 2008.
6. Phan Minh Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial Intelligence*, 77(2):321–358, 1995.
7. Benjamin Johnston and Guido Governatori. Induction of defeasible logic theories in the legal domain. In *Proceedings of the 9th International Conference on Artificial Intelligence and Law, ICAIL '03*, pages 204–213, New York, NY, USA, 2003. ACM.
8. Levente Kocsis and Csaba Szepesvri. Bandit based monte-carlo planning. In *In: ECML-06. Number 4212 in LNCS*, pages 282–293. Springer, 2006.
9. Thomas M. Mitchell. *Machine Learning*. McGraw-Hill, Inc., New York, NY, USA, 1 edition, 1997.
10. Sanjay Modgil and Martin Caminada. Proof theories and algorithms for abstract argumentation frameworks. In *Argumentation in artificial intelligence*, pages 105–129. Springer, 2009.
11. Martin Mozina, Jure Zabkar, and Ivan Bratko. Argument based machine learning. *Artif. Intell.*, 171(10-15):922–937, 2007.
12. Santiago Onta nón, Pilar Dellunde, Lluís Godo, and Enric Plaza. A defeasible reasoning model of inductive concept learning from examples and communication. *Artificial Intelligence*, 193:129–148, 12/2012 2012.

13. Henry Prakken and Giovanni Sartor. Argument-based extended logic programming with defeasible priorities. *Journal of Applied Non-Classical Logics*, 7(1), 1997.
14. Henry Prakken and Giovanni Sartor. Reasoning with precedents in a dialogue game. In *ICAIL*, pages 1–9, 1997.
15. Régis Riveret, Cameron Brown, Didac Busquet, and Jeremy Pitt. A monte-carlo tree search in argumentation. In *Proceedings of ArgMAS 2014*, page to appear.
16. Régis Riveret, Antonino Rotolo, Henry Prakken, and Giovanni Sartor. Heuristics in argumentation: a game-theoretical investigation. In *Proceedings of the 2nd International Conference on Computational Models of Argument*. IOS Press, 2008.
17. Régis Riveret, Antonino Rotolo, and Giovanni Sartor. Probabilistic rule-based argumentation for norm-governed learning agents. *Artif. Intell. Law*, 20(4):383–420, 2012.