A Monte-Carlo Tree Search in Argumentation

Régis Riveret[1], Cameron Browne[2], Dídac Busquets[1], Jeremy Pitt[1]

[1] Department of Electrical and Electronic Engineering, Imperial College of Science, Technology and Medicine, London, United Kingdom
{r.riveret,didac.busquets,j.pitt}@imperial.ac.uk,
[2] Goldsmiths College, London, United Kingdom
camb@gold.ac.uk

Abstract. Monte-Carlo Tree Search (MCTS) is a heuristic to search in large trees. We apply it to argumentative puzzles where MCTS pursues the best argumentation with respect to a set of arguments to be argued. To make our ideas as widely applicable as possible, we integrate MCTS to an abstract setting for argumentation where the content of arguments is left unspecified. Experimental results show the pertinence of this integration for learning argumentations by comparing it with a basic reinforcement learning.

1 Introduction

Common questions in argumentation regard what conclusions are (defeasibly) justified, or how to proceed to obtain their justification in a sound and complete manner. Heuristics, understood as a guidance for the arguers, have received less attention. Work on heuristics have focused on game-theoretical aspects to determine optimal strategies in dialogue games for argumentation (see e.g. [12]) and machine learning techniques (e.g. [13]).

Unfortunately, the curse of dimensionality often impedes most common game-theoretical or learning techniques when the search spaces become too large. Efficient heuristics can thus be investigated. 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 is congruent to the heuristics of human players: both exploit repetitively and incrementally promising lines of strategies while occasionally exploring a priori weaker options.

In this paper, we investigate the use of MCTS with an abstract and formal setting of argumentation while preserving the intuitive concept of argumentation like (single agent internal) dialogues. We apply it to 'argumentative puzzles' where, in the pursuit of the best argumentative picture, one exploit and explore argumentations. To make the analogy: an argument is a piece of a puzzle, attacks amongst arguments are the connections between pieces, a set of arguments with

attacks is an argumentative picture, each picture is associated with a reward. The problem faced by an arguer regards the selection of an argument picture called an argumentation, the goal of the arguer is to maximize the sum of rewards earned through a sequence of choices of argumentations (one argumentation being chosen at each step of the sequence). The problem is quite similar to the problem of multi-armed bandits, but in this paper, each argumentation has a fixed reward. The setting is abstract because arguments have no particular structure, arguments attack some other arguments, and for our purposes, every argument is attached with an abstract numerical value.

The remainder is organised as follows. Next Section 2 sketches the basic MCTS approach and Section 3 introduces argumentation frameworks. The integration of MCTS with our argumentative setting is evaluated in Section 4. Section 5 discusses our integration and relates it to other work on machine learning and logic. The last section concludes.

2 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 a strategy becomes better at estimating the values of decisions. 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:

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 1. 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 back up 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

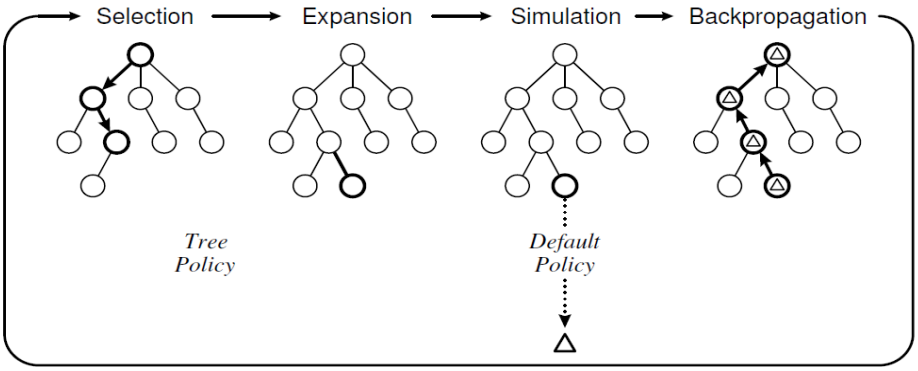

Fig. 1. One iteration of the general MCTS approach [4].

bounds (UCB) [2]:

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

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 probability tending to 1.

We will compare UCT to a variant of MCTS in which the tree policy selects the nodes according to a Gibbs-Boltzmann distribution:

$$\frac{e^{v_i/\tau}}{\sum_i e^{v_i/\tau}}$$

where τ is a parameter (analogous to a temperature) balancing the exploitation and the exploration.

3 Argumentation framework

To make our investigation on MCTS and argumentation as widely general as possible, we adopt an abstract argumentation framework. The framework is based on a directed graph where each node represent an argument and an arrow is an attack from one argument to another. A simple argumentation graph is illustrated in Figure 2.

Definition 1 (Argumentation graph). *An argumentation graph is a pair* $\langle \mathcal{A}, \triangleright \rangle$ *where* $\mathcal{A}$ *is a set of arguments, and* $\triangleright \subseteq \mathcal{A} \times \mathcal{A}$ *is a binary relation of attack. In particular we assume that for any argument* X *and* Y*,* $X \triangleright Y$ *if, and only if,* X *attacks* Y*.*

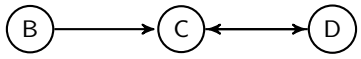

Fig. 2. An argumentation graph. Argument B attacks the argument C. The arguments C and D attack each other.

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 and for the sake of simplicity, we shall only consider the Dung's grounded semantics [6] though the approach is general enough to support other semantics.

Definition 2 (Argumentation framework).

- ***Conflict-free set****: A set* $\mathcal{S}$ *of arguments is* conflict-free *if, and only if, there is no argument* A *and* B *in* $\mathcal{S}$ *such that* B *attacks* A
- ***Acceptable argument****: An argument* A *is acceptable w.r.t. a set of arguments* $\mathcal{S}$ *if and only if any argument attacking* A *is attacked by an argument in* $\mathcal{S}$*.*
- ***Characteristic function****: The characteristic function of an argumentation graph* $\mathcal{G} = \langle \mathcal{A}, \triangleright \rangle$*, is defined as* $F_{\mathcal{G}} : 2^{\mathcal{A}} \rightarrow 2^{\mathcal{A}}$ *and* $F_{\mathcal{G}}(\mathcal{S}) = \{A |\ A$ *is acceptable w.r.t.* $\mathcal{S} \subseteq \mathcal{A}\}$*.*
- ***Admissible set****: A conflict-free set* $\mathcal{S}$ *of arguments is admissible if and only if* $\mathcal{S} \subseteq F(\mathcal{S})$*.*
- ***Grounded extension****: A grounded extension of an argumentation graph* $\mathcal{G}$ *is the least fixed-point of* $F_{\mathcal{G}}$*.*

Example 1. The grounded extension of the argumentation graph of Fig. 2 is $\{\mathsf{B}, \mathsf{D}\}$ (B is not attacked by any argument, C is attacked by B).

Dialogue games between a proponent and a opponent are often proposed to compute whether an argument is justified, i.e. belongs to the grounded extension (see e.g. [10]).Other protocols (e.g. [12]) 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 construction of possible argumentations relative to a set of arguments called here an *argumenta*. Given an argumentation graph, an argumentation for an argumenta is a sub-graph containing the argumenta (see Figure 3 for an illustration), and defined by construction using a procedure called the *argumentative procedure*.

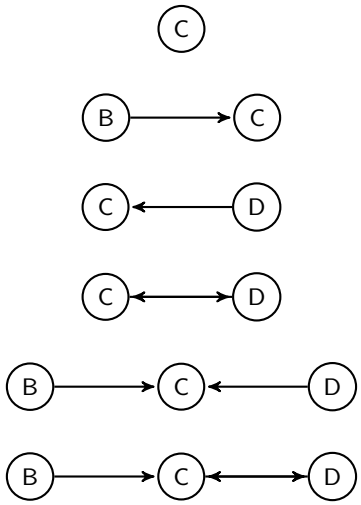

Fig. 3. Possible argumentations for the set of argumenta $\{\mathsf{C}\}$. Notice that the sub-graph $\langle\{\mathsf{C},\mathsf{D}\},\{\mathsf{C}\triangleright\mathsf{D}\}\rangle$ is not a valid argumentation with respect to the argumentum C because $\mathsf{C}\triangleright\mathsf{D}$ does not counter-attack any attack.

The argumentative procedure takes as input an argumentation graph and an argumenta, it outputs a tree of argumentations (each node induces an argumentation, and the set of argumentations is thus the set of nodes).

A branch is a finite non-empty sequence of moves $\text{move}_1,\ldots,\text{move}_n$, where the first move is an argumentum attacking a dummy root argument (that we usually leave implicit) and any move $\text{move}_i, i>1$ is 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.

An argumentation tree is all possible sequences of moves. Each node induces an argumentation graph called simply an argumentation. An argumentation tree of our running toy example is illustrated in Figure 4.

The procedure possibly involve a large branching factor for the argumentation tree so that the performance of MCTS can be fully appreciated. In the next section, MCTS is applied to argumentation trees to find out the best argumentation.

4 Integrating MCTS and argumentation

Argumentation frameworks involve argumentation graphs, while MCTS is tree-based. In the previous section, argumentation trees were introduced as the result of the argumentative procedure developing argumentations about a set of arguments called an argumenta. In the pursuit of the best argumentation, we use MCTS to exploit and explore argumentations within an argumentation tree.

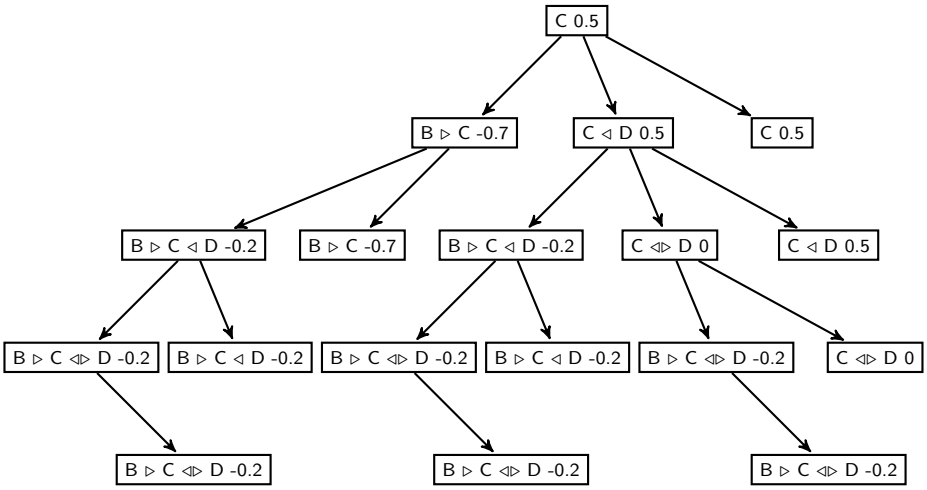

Fig. 4. A fully expanded argumentation tree based on the simple argumentation graph of the Figure 2. Each link is a move, i.e. a link represents an attack being moved or a withdraw. Any node reached by a path is an argumentation (graph). The number of a node is the value of the corresponding state when the values of arguments are $\mathsf{B} : -0.7$, $\mathsf{C} : 0.5$ and $\mathsf{D} : 0.5$.

We assume that the best argumentation is the argumentation with the highest value (whatever the value) and though we believe that the choice of any valuation shall depend on the concerned application, we have to consider values and argumentation frameworks (see e.g. [7]). For example, an epistemic value of an argumentation shall measure its potential to account for some observation, while a practical value shall indicate the utility of tasks involved by this argumentation within an environment. For our purposes, and since we are interested by an abstract setting, we want it to reflect the non-monotonic nature of argumentation, that is, the value of the grounded extension of an argumentation shall not grow with the size of this argumentation. So, we have a simple and abstract valuation: any argument is associated with a numeric value between -1 and 1 drawn from an uniform distribution. The value of an argumentation is the sum of the values of arguments in its grounded extension. Hence the valuation of arguments between -1 and +1 implies that the value of a grounded extension does not grow with its size. To obtain the best argumentation, this valuation involves the rejection of the maximum of negative arguments, and at the same time the justification of the maximum of positive arguments: a trade-off is often necessary. To play this abstract argumentative puzzle, an argumentation graph is given, and few arguments are labelled as argumentum. A dummy 'root' node of a tree is assumed from which any argumentum can be argued about. A convenient representation of the argumentation tree of our running toy example is provided in Figure 3.

Note that for common MCTS settings, if moves are reversible then there is the danger of cycles to previous states impeding learning. However, we assumed directional moves in the sense that attacks could no be retracted. Furthermore, since moved attacks cannot be repeated, cycles are eliminated. Nevertheless, the construction of the MCTS tree as an argumentation tree implies that several paths shall lead to the same argumentation. So, future work shall investigate improvements to deal with these path redundancies (referred to usually as transpositions) (see e.g. [5]).

To evaluate our integration of MCTS to argumentation, we considered a random argumentation graph. The argumentation graph is generated such that its structure obeys a certain degree distribution: the degree of a node in the graph is the number of connections that it has to other nodes and the degree distribution of a graph is the probability distribution of these degrees over the whole graph. Since an argumentation graph is an undirected graph, we have some edges coming into nodes and some edges going out from nodes. For the sake of simplicity, we assumed the independence between the in-degree distribution and the out-degree distribution. For both, we considered a Erdos-Renyi distribution so that we set an attack link between each pair of arguments with equal probability, independently of the other attacks.

Since MCTS in its UCT and Boltzmann-based variants belong to the family of reinforcement learning techniques, we compared UCT and Boltzmann-based MCTS with a basic reinforcement learning technique (RL), see Figure 5. In the RL technique, the search space is the set of possible argumentations $\mathfrak{G}$. At each time step t, an argumentation $\mathcal{G}$ is associated with a quality $Q^t(\mathcal{G})$, and one argumentation is evaluated by drawing it with the probability $P^t(\mathcal{G})$ defined by Gibbs-Boltzmann distribution:

$$P^t(\mathcal{G}) = e^{Q^t(\mathcal{G})/\tau} / \sum_{\mathcal{G} \in \mathfrak{G}} e^{Q^t(\mathcal{G})/\tau}$$

where τ is a parameter balancing exploration and exploitation of argumentations. The quality of one argumentation $\mathcal{G}$ is the moving discounted average of its value $V(\mathcal{G})$ updated at each selection:

$$Q^{t+1}(\mathcal{G}) = Q^t(\mathcal{G}) + \alpha.[V(\mathcal{G}) - Q^t(\mathcal{G})]$$

with $Q^0(\mathcal{G}) = 0$. The performance of the different approaches is evaluated by the averaged optimality measure V^t/V^* over episodes where V^* is the value of the best argumentation and V^t is the value of the terminal argumentation selected at time t.

We observe that UCT is the slowest learning approach to take off in the very first steps but turns out to have the best results after few hundreds steps (the time to grow the search tree). Besides, while RL implies to build and store for search the whole set of possible argumentations which can be immense, MCTS has the advantage of short-cutting this computational issue since the set of possible argumentation is progressively grown as an asymmetric tree. For this reason, the implementation of MCTS appeared to be faster than RL by few orders of magnitude (depending of the size of the set of possible argumentations).

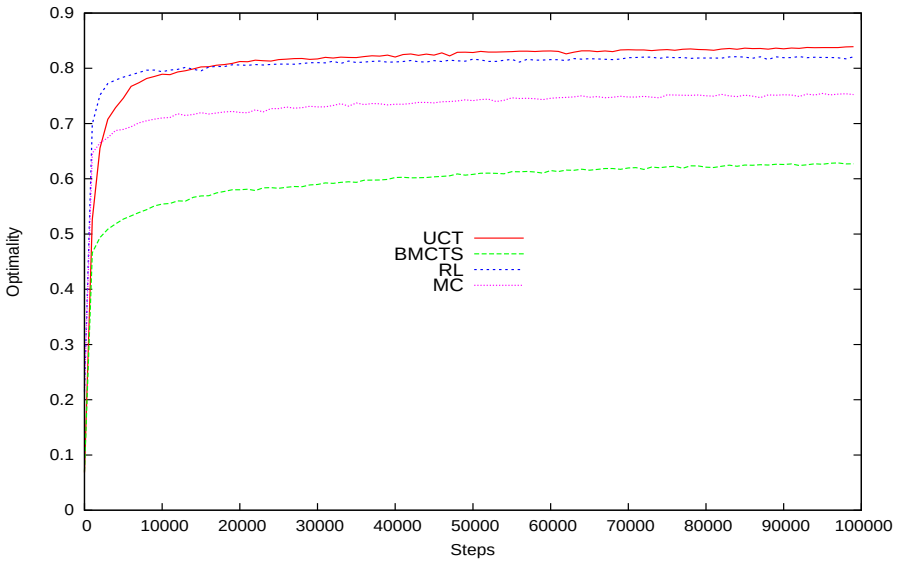

Fig. 5. Averaged optimality over time steps. The random argumentation graphs are generated with a number of nodes fixed at 20, and the probability of an attack relation between two nodes is 0.05. After 100,000 iterations, the top curve is UCT, the second one is RL with the discount $\alpha = 0.1$, the third is RL with $\alpha = 1$ (MC), and the bottom curve is Boltzmann MCTS. The rewards are averaged at each time step over 1000 episodes. The average number of possible argumentations was 12,901. The UCT computational parameter C is fixed at 2 to favour exploration. For Boltzmann MCTS and RL, the exploitation parameter τ is fixed at 0.1.

5 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. A large amount of work exists and the approaches range from inductive logic programming (IPL) to reinforcement learning, with different techniques from graphical models to neural networks.

In terms of inductive reasoning, a common challenge is the induction of hypothesised theories accounting for given examples like cases or precedents (see e.g. IPL). In our setting, we used MCTS to search for the best argumentation (that amounts to a theory) amongst other possible argumentations induced by

a given argumentation graph. Thus a further step of our approach with regard to theory induction is the use of MCTS to search for the best argumentation given a set of examples (instead of a given argumentation graph). Argumentation has also been investigated for concept learning: L. Amgoud and M. Serrurier proposed in [1] to construct a version space as the grounded extension of an argumentation graph induced by a set of training examples. In a parallel line of research, argumentation and inductive reasoning have already been approached by Mozina et al. [9] who propose to add arguments as data to learning examples to ease the induction of rule-based theories.

Case-based reasoning (CBR) is the most common ground with regard to 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 [11] an early investigation of rule-based argumentation with a semantics a 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 using 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 an heuristic for arguers.

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 [13], It is a first attempt using basic UCT to by-pass the curse of dimensionality for learning large theories encoding multi-agent systems.

6 Conclusion

In this paper, we investigated the use of MCTS in argumentation. The experiments show that the bare UCT approach competes well with reinforcement learning in terms of convergence while MCTS avoids the construction and the storage of the whole set of possible argumentations. We have here not only a technique to learn for the best argumentation but also the path reaching this argumentation. Hence, MCTS allows us to take into account procedural parameters like the cost of moves. This contrasts to existing work on reinforcement learning and argumentation which did not take advantage of the structure of argumentation procedures.

The use of more sophisticated development of MCTS (starting from simple transposition tables for example) shall be investigated to improve the overall performance. Furthermore, we have an intensive representation of the system's states through a formal and abstract argumentative logic. Hence, by considering

an instantiation of this abstract framework like in an rule-based argumentation framework, we may take advantage of the resulting intensive representation to improve learning.

Finally, as argumentation is easily framed as a game played by different arguers, the integration of game-theoretical considerations shall be investigated to better deal with scenarios of interacting agents.

Acknowledgments. Part of this work is supported by the Marie Curie Intra-European Fellowships PIEF-GA-2012-331472.

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. Paul E. Dunne, Anthony Hunter, Peter McBurney, Simon Parsons, and Michael Wooldridge. Weighted argument systems: Basic definitions, algorithms, and complexity results. *Artif. Intell.*, 175(2):457–486, 2011.
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. Martin Mozina, Jure Zabkar, and Ivan Bratko. Argument based machine learning. *Artif. Intell.*, 171(10-15):922–937, 2007.
10. Henry Prakken and Giovanni Sartor. Argument-based extended logic programming with defeasible priorities. *Journal of Applied Non-Classical Logics*, 7(1), 1997.
11. Henry Prakken and Giovanni Sartor. Reasoning with precedents in a dialogue game. In *ICAIL*, pages 1–9, 1997.
12. 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.
13. Régis Riveret, Antonino Rotolo, and Giovanni Sartor. Probabilistic rule-based argumentation for norm-governed learning agents. *Artif. Intell. Law*, 20(4):383–420, 2012.