An Implementation of Argument Based Discussion *

Patrizio Barbini
University of Luxembourg
6, rue Richard
Coudenhove-Kalergi
L-1359 Luxembourg
patrizio.barbini@uni.lu

Yining Wu
University of Luxembourg
6, rue Richard
Coudenhove-Kalergi
L-1359 Luxembourg
yining.wu@uni.lu

Martin Caminada
University of Luxembourg
6, rue Richard
Coudenhove-Kalergi
L-1359 Luxembourg
martin.caminada@uni.lu

ABSTRACT

With the current demonstrator, we present an implementation of formal argumentation that is not only able to evaluate an argument according to standard argumentation semantics, but is also able to engage in a discussion to defend its answer. This discussion is formal yet natural enough to be applicable in agent-to-agent as well as in agent-to-human settings.

1. INTRODUCTION

Dung's notion of abstract argumentation [5] has become one of the leading approaches in formal argumentation and nonmonotonic reasoning. Some of the possible argumentation semantics (like grounded, preferred, ideal and stable) have a dialectical interpretation. That is, whether an argument is considered justified can be determined by means of a discussion game in which arguments are exchanged. Such a discussion game can serve as the basis for agent communication protocols, in which agents discuss among each other about the validity of a claim or argument [7].

Although implementations of formal argumentation (like [8]) have been made, these are usually limited to merely answering queries regarding the validity of a claim or argument, based on a given knowledge base. What is missing is software that is also able to enter into a discussion aimed at explaining and defending it's answers. Ideally, such a discussion should follow a rigid and formalized protocol, yet at the same time be simple and intuitive enough to be understood by human observers. Users will be more willing to outsource their tasks to a multi-agent system if this system is able to provide intelligent feedback, and is actually able to explain on what grounds it acted. It is this explaining capability that is the topic of the current demonstrator.

2. BACKGROUND THEORY

The current demonstrator essentially implements complete semantics. That is, given an argumentation framework, it is able to answer the question of whether an argument is in at least one complete extension (credulous complete) and whether it is in all complete extensions (skeptical complete). Since an argument is in at least one complete extension iff it is in at least one preferred extension, and since an argument is in all complete extensions iff it is in the grounded extension, one could also say that the demonstrator implements credulous preferred and grounded.

In [3] Caminada proves that the notion of complete semantics actually coincides with that of (complete) argument labellings. Given an argumentation framework (Ar, def), a complete labelling is a function that assigns to each argument exactly one label (*accepted*, *rejected* or *undecided*)[1] such that for each argument

- the argment is labelled *accepted* iff all its defeaters are labelled *rejected*, and
- the argument is labelled *rejected* iff it has at least one defeater that is labelled *accepted*.

Basically, we have taken the discussion game for credulous preferred [9] and for grounded [6, 2] and have reinterpreted them in terms of argument labellings, like is done in [1]. The advantage of stating the games in terms of labellings is that it becomes easier to gain understanding of their fundamental design decisions. In essence, each complete labelling should be seen as a reasonable position one can take in the presence of conflicting information. An argument being labelled *accepted* means that one has a position in which the argument is explicitly accepted. An Argument being labelled *rejected* means that one has a position in which the argument is explicitly rejected. An argument being labelled *undecided* means that one has insufficient grounds for accepting the argument, and insufficient grounds for rejecting the argument.

Since the demonstrator supports credulous complete as well as skeptical complete (which is the same as credulous preferred and grounded) it essentially implements two types of discussion games. As for the credulous preferred game, it should be mentioned that an argument is in at least one preferred extension iff it is in at least one admissible set. In terms of labellings, an admissible set can be expressed as follows [4]. An admissible labelling is a function that assigns each argument exactly one label (*accepted*, *rejected* or *undecided*) such that for each argument

*(Produces the copyright information for AAMAS 2009). For use with aamas2009.CLS and aamas2009-extendedabs.cls

Cite as: An Implementation of Argument Based Discussion, Patrizio Barbini, Yining Wu, Martin Caminada, *Proc. of 8th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2009),* Decker, Sichman, Sierra and Castelfranchi (eds.), May, 10–15, 2009, Budapest, Hungary, pp. 1425 – 1426

[1]We use the labels *accepted*, *rejected* and *undecided* instead of the labels `in`, `out` and `undec` from [3] because for the purpose of the demonstrator, we would like to be closer to natural language.

- if the argument is labelled *accepted* then all its defeaters are labelled *rejected*, and
- if the argument is labelled *rejected* then it has a defeater that is labelled *accepted*.

It can be observed that every complete labelling is an admissible labelling (but not vice versa), just like every complete extension is an admissible set (but not vice versa).

Basically, the credulous game is aimed at testing whether the proponent indeed has an admissible labelling that labels the argument in question *accepted*. Hence, the proponent starts by claiming that it has a labelling in which the main argument (say A) is labelled *accepted*. Then, the opponent confronts the proponent with the consequences of it's own statement: "If you think that A is accepted, then you must also hold that A's defeater B is rejected. Based on which grounds do you reject B?" Then, the proponent could reply something like: "I reject B because I accept B's defeater C", after which the opponent could then ask for the reasons why the proponent rejects the defeaters of C, etc... In essence, the moves of the proponent are statements, whereas the moves of the opponent are questions. This also explains why the proponent is allowed to repeat itself (some questions may have the same answer) and why the opponent is not allowed to repeat itself (it would be useless to ask a question that has already been answered). More formal background and proofs of correctness can be found in [9, 1].

As for the skeptical game, the situation is slightly different. Here, the idea is to have a discussion whether the main argument has to be labelled *accepted* in *every* complete labelling. Hence, the proponent's moves are of the form "Argument A must be accepted, no matter what." The opponent's moves are of the form: "But perhaps A's defeater B doesn't have to be rejected", at which the proponent replies: "B has to be rejected because B's defeater C has to be accepted." The game ends when the proponent provides an argument that has no defeaters. Since the aim of the game is for the proponent to take away the doubt of the opponent, it is not productive for the proponent to repeat its own moves, because by doing so, one would merely go around in circles. More formal background and proofs of correctness can be found in [6, 2].

3. THE DEMONSTRATOR

The demonstrator implements three commands: *question*, *discuss* and *generate*. With *question* one asks for the status of an argument. There are three possible answers:

1. "the argument has to be accepted", meaning that the argument is labelled `in` in every complete labelling, meaning that the argument is labelled `in` in the grounded labelling (and is therefore an element of the grounded extension)
2. "the argument can be accepted but doesn't have to", meaning that the argument is labelled `in` in some but not all complete labellings (which implies that the argument is in at least one preferred extension, but not in the grounded extension)
3. "the argument cannot be accepted", meaning that the argument is not labelled `in` in any complete labelling (implying that it is not contained in any preferred extension)

With the *discuss* command, the software is willing to defend its position using the credulous and skeptical discussion games. The software is intelligent enough only to take positions that can be defended. Therefore, it will win any discussion the user wishes to engage in.

As an additional piece of functionality, we have also implemented the *generate* command, with which the demonstrator is able to provide the grounded, stable, semi-stable and preferred extensions, using the algorithm specified in [4].

4. ROUNDUP

The current demonstrator not only implements algorithms that can be used by an agent for its own internal reasoning (as demonstrated by the *question* command) but is also able to enter into a discussion to defend its position. This discussion is formal yet natural, which means that the agent can defends its position in both agent-to-agent and agent-to-human settings.

5. REFERENCES

[1] Martin Caminada and Yining Wu. An argument game of stable semantics. *Logic Journal of IGPL*, in print.

[2] M.W.A. Caminada. *For the sake of the Argument. Explorations into argument-based reasoning*. Doctoral dissertation Free University Amsterdam, 2004.

[3] M.W.A. Caminada. On the issue of reinstatement in argumentation. In M. Fischer, W. van der Hoek, B. Konev, and A. Lisitsa, editors, *Logics in Artificial Intelligence; 10th European Conference, JELIA 2006*, pages 111–123. Springer, 2006. LNAI 4160.

[4] M.W.A. Caminada. An algorithm for computing semi-stable semantics. In *Proceedings of the 9th European Conference on Symbolic and Quantitalive Approaches to Reasoning with Uncertainty (ECSQARU 2007)*, number 4724 in Springer Lecture Notes in AI, pages 222–234, Berlin, 2007. Springer Verlag.

[5] P. M. Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial Intelligence*, 77:321–357, 1995.

[6] H. Prakken and G. Sartor. Argument-based extended logic programming with defeasible priorities. *Journal of Applied Non-Classical Logics*, 7:25–75, 1997.

[7] Iyad Rahwan, Simon Parsons, and Chris Reed, editors. *Argumentation in Multi-Agent Systems, 4th International Workshop, ArgMAS 2007, Honolulu, HI, USA, May 15, 2007, Revised Selected and Invited Papers*, volume 4946 of *Lecture Notes in Computer Science*. Springer, 2008.

[8] Matthew South, Gerard Vreeswijk, and John Fox. Dungine: A java dung reasoner. In Philippe Besnard, Sylvie Doutre, and Anthony Hunter, editors, *COMMA*, volume 172 of *Frontiers in Artificial Intelligence and Applications*, pages 360–368. IOS Press, 2008.

[9] G. A. W. Vreeswijk and H. Prakken. Credulous and sceptical argument games for preferred semantics. In *Proceedings of the 7th European Workshop on Logic for Artificial Intelligence (JELIA-00)*, number 1919 in Springer Lecture Notes in AI, pages 239–253, Berlin, 2000. Springer Verlag.