An Implementation of Argument-Based Discussion using ASPIC-

Martin CAMINADA [a,1] Sören UEBIS [b]
[a] *Cardiff University, Cardiff, UK*
[b] *FernUniversität in Hagen, Hagen, Germany*

Keywords. argument-based discussion, chatbot, ASPIC-

An often mentioned advantage of argumentation theory (compared to other formalisms for non-monotonic reasoning) is that it is based on concepts of human reasoning. However, quite some of the argumentation semantics are defined in terms of fixpoints [1] which, although appealing to mathematicians, do not seem to coincide with how most humans tend to reason in everyday life. In order to bring argument-based entailment closer to human intuitions, we propose to use formal discussion as a bridge technology. For this, we are applying argument-based discussion theory [3] which reformulates argument-based reasoning as the ability to win a particular type of discussion.[2] More specifically, an argument is in the grounded extension iff a proponent of the argument has a winning strategy in the Grounded Discussion Game [3].

In the context of abstract argumentation theory, an implementation of the Grounded Discussion Game (as well as of the Preferred Discussion Game) is already available [2]. With the current demonstrator, however, we are going one step further by basing the discussion not on abstract arguments, but on rule-based arguments that are constructed from an underlying knowledge base. For this, we base ourselves on the ASPIC- framework, which is a variant of ASPIC+ where the definition of attack is more suitable for interactive applications [4].

Our demonstrator, called ABDA (Argument-Based Discussion using ASPIC-) is written in Python3, does not require any non-standard libraries, and has been tested to work under both Windows and Linux. The knowledge base is stored in a file called `aspic-rules.txt`. The file starts with a number of strict rules (such as `a, b, c -> d`), each on its own line. After that comes a blank line, followed by a number of defeasible rules (such as `a, b, c => d [r1]` where `r1` is the name of the rule, to be used for purposes of undercutting [4]), each on its own line. These defeasible rules come in blocks consisting of several lines, which are seperated by blank lines. Defeasible rules in the same block have the same strength, whereas those in later blocks have a higher strength than those in earlier blocks. For instance, if the file contains three defeasible rules, followed by a blank

[1] Corresponding Author. Email: CaminadaM@cardiff.ac.uk

[2] One of the advantages of [3] above previous approaches (e.g. [6,5]) is that it avoids an exponential blowup in the number of moves required. We refer to [3] for details.

line, and then two other defeasible rules, then the first three rules have strength 1 and the last two rules have strength 2.

The demonstrator can be started from the command line, and takes as parameters `-wl` (to implement the *weakest link* principle [4]) or `-ll` (to implement the *last link* principle [4]), as well as `-do` (to implement the *democratic order* [4]) or `-eo` (to implement the *elitist order* [4]).

Once the demonstrator has been started, it is possible to query the inference engine if a particular statement is justified (that is, if the statement is the conclusion of an argument in the grounded extension), e.g. `warranted car_safe`. The system would then reply with either `car_safe is warranted` or `car_safe is not warranted`. The user can then ask for explanation and start a discussion with the system, e.g. `discuss car_safe`. If the statement is justified, the system will assume the role of the proponent and the user the role of the opponent. If the statement is not justified, the user will assume the role of the proponent and the system the role of the opponent. As the discussion is sound and complete for grounded semantics [3], the system is able to play a winning strategy.

At the moment, the arguments played in the game are written in a nested, machine readable way, as specified by ASPIC- (a format that is very close to ASPIC+). However, in future work we aim to be able to convert between machine readable (structured) arguments and arguments in (controlled) natural language. The overall aim is to bring human-to-computer discussion as close as possible to human-to-human discussion. For instance, when applied to the medical domain, talking to the system should resemble as much as possible talking to a more senior colleague.

The source code of ABDA, together with examples of knowledge bases, can be downloaded from `http://users.cs.cf.ac.uk/CaminadaM/demonstrators.html`

References

[1] P. Baroni, M.W.A. Caminada, and M. Giacomin. An introduction to argumentation semantics. *Knowledge Engineering Review*, 26(4):365–410, 2011.

[2] R. Booth, M.W.A. Caminada, and B. Marshall. DISCO: A web-based implementation of discussion games for grounded and preferred semantics. In S. Modgil, K. Budzynska, and J. Lawrence, editors, *Proceedings of COMMA 2018*, pages 453–454. IOS Press, 2018.

[3] M.W.A. Caminada. A discussion game for grounded semantics. In E. Black, S. Modgil, and N. Oren, editors, *Theory and Applications of Formal Argumentation (proceedings TAFA 2015)*, pages 59–73. Springer, 2015.

[4] M.W.A. Caminada, S. Modgil, and N. Oren. Preferences and unrestricted rebut. In Simon Parsons, Nir Oren, Chris Reed, and Federico Cerutti, editors, *Computational Models of Argument; Proceedings of COMMA 2014*, pages 209–220. IOS Press, 2014.

[5] S. Modgil and M.W.A. Caminada. Proof theories and algorithms for abstract argumentation frameworks. In I. Rahwan and G.R. Simari, editors, *Argumentation in Artificial Intelligence*, pages 105–129. Springer, 2009.

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