Computational Models of Argument
S. Modgil et al. (Eds.)

doi:10.3233/978-1-61499-906-5-463

Reasoning with Metalevel Argumentation Frameworks in Aspartix

Nadin KOKCIYAN [a,1], Isabel SASSOON [a], Anthony P. YOUNG [a], Sanjay MODGIL [a] and Simon PARSONS [a]
[a] *Department of Informatics, King's College London*

Keywords. Computational argumentation, metalevel argumentation frameworks

1. Introduction

In this demo paper, we propose an encoding for *Metalevel Argumentation Frameworks* (MAFs) to be used in *Aspartix*, an *Answer Set Programming* (ASP) approach to find the justified arguments of an AF [2]. MAFs provide a uniform encoding of object level Dung Frameworks and extensions thereof that include values, preferences and attacks on attacks (EAFs). The justification status of arguments in the object level AF can then be evaluated and explained through evaluation of the arguments in the MAF. The demo includes multiple examples from the literature to show the applicability of our proposed encoding for translating various object level AFs to the uniform language of MAFs.

2. An Encoding to Reason with MAFs in Aspartix

In Table 1, we provide a subset of ASP rules that can be used for reasoning with MAFs. Each rule is of the form *head* :- *body*. In this work, *body* consists of a conjunction of predicates and *head* is a predicate. Every predicate in the metalevel is shown in *italics*, every predicate in the object level is shown in normal text, and each variable is in uppercase letter. At the object-level, a(X) and r(a(X), a(Y)) represent the argument X and the attack relation between the arguments X and Y, respectively. Moreover, an attack on an attack is represented with d(a(Z), a(X), a(Y)) where the argument Z attacks the attack between X and Y. Such an object-level description is then translated into a MAF [4]. At the metalevel, the meta-argument A and the meta-attack between the meta-arguments A and B are described by *arg*(A) and *att*(A, B), respectively. Each object-level argument has a *justified* or *rejected* status in the metalevel. The object-level attacks are translated into *defeat* or *ddefeat* meta-attack arguments. A preference argument is specified with the predicate *preferred*. The rules $r_1 - r_6$ are the core rules to map the object-level arguments and attacks to the corresponding MAF. P-MAF rules ($r_7 - r_9$) add preferences and E-MAF rules ($r_{10} - r_{12}$) add attacks on attacks into the metalevel.

[1]Corresponding Author: Department of Informatics, King's College London, Bush House, Strand Campus, 30 Aldwych, WC2B 4BG, London, United Kingdom.; E-mail: nadin.kokciyan@kcl.ac.uk

Dung MAF	
r_1: *arg*(*justified*(X))	:- a(X)
r_2: *arg*(*rejected*(X))	:- a(X)
r_3: *arg*(*defeat*(X, Y))	:- r(a(X), a(Y))
r_4: *att*(*defeat*(X, Y), *justified*(Y))	:- *arg*(*defeat*(X, Y)), *arg*(*justified*(Y))
r_5: *att*(*rejected*(X), *defeat*(X, Y))	:- *arg*(*defeat*(X, Y)), *arg*(*rejected*(X))
r_6: *att*(*justified*(X), *rejected*(X))	:- *arg*(*justified*(X)), *arg*(*rejected*(X))
P-MAF	
r_7: *att*(*preferred*(X, Y), *preferred*(Y, X))	:- *arg*(*preferred*(X, Y)), *arg*(*preferred*(Y, X))
r_8: *arg*(*preferred*(X, Y))	:- p(a(X), a(Y))
r_9: *att*(*preferred*(X, Y), *defeat*(Y, X))	:- *arg*(*preferred*(X, Y)), *arg*(*defeat*(Y, X))
E-MAF	
r_{10}: *arg*(*ddefeat*(Z, X, Y))	:- d(a(Z), a(X), a(Y))
r_{11}: *att*(*ddefeat*(Z, X, Y), *defeat*(X, Y))	:- *arg*(*defeat*(X, Y)), *arg*(*ddefeat*(Z, X, Y))
r_{12}: *att*(*rejected*(Z), *ddefeat*(Z, X, Y))	:- *arg*(*rejected*(Z)), *arg*(*ddefeat*(Z, X, Y))

Table 1. A Subset of ASP Rules to Reason with *Metalevel Argumentation Frameworks* (MAFs)

Demo Details. We use multiple scenarios from the literature to demonstrate our proposed encoding in various AFs. Our first scenario concerns finding a consistent firewall policy that contains a set of firewall rules collected from various gateways with different preferences [1]. In this scenario, a MAF is used to explain why conflicts occur between policies and help one to resolve such conflicts. In a variation of the same scenario, different parties associate values with their arguments and a preference relation between these values. Our second scenario concerns recommending hypertension treatments [3] where an EAF is used to recommend treatments given doctor and patient preferences. We also consider various abstract examples from [4]. We have a Python implementation that (1) uses DLV[2] as the ASP engine to map an AF into a MAF and computes the set of acceptable arguments according to Dung's semantics, (2) generates the corresponding MAF graphs as figures that explain the decision-making process. The X-MAF encodings, the input/output files for the scenarios and our Python implementation will be accessible online at `https://consult.kcl.ac.uk/comma18-demo/.`

Acknowledgments. This research was supported by the UK Engineering & Physical Sciences Research Council (EPSRC) under grant #EP/P010105/1.

References

[1] A. Applebaum, Z. Li, K. Levitt, S. Parsons, J. Rowe, and E. I. Sklar. Firewall configuration: An application of multiagent metalevel argumentation. *Argument & Computation*, 7(2-3):201–221, 2016.

[2] U. Egly, S. Gaggl, and S. Woltran. Aspartix: Implementing argumentation frameworks using answer-set programming. *Logic Programming*, pages 734–738, 2008.

[3] N. Kokciyan, I. Sassoon, A. Young, M. Chapman, T. Porat, M. Ashworth, V. Curcin, S. Modgil, S. Parsons, and E. Sklar. *Towards an Argumentation System for Supporting Patients in Self-Managing their Chronic Conditions*. 2018.

[4] S. Modgil and T. J. Bench-Capon. Metalevel argumentation. *Journal of Logic and Computation*, 21(6):959–1003, 2011.

[2] `http://www.dlvsystem.com/dlv/`