Abstract Argumentation and Answer-Set Programming*

Sarah Alice Gaggl, Stefan Woltran

Vienna University of Technology, Austria

Abstract. In this article, we overview encodings for problems associated to abstract argumentation frameworks (AFs) in the language of Answer-Set Programming (ASP). Our encodings are formulated as fixed queries, such that the input is the only part depending on the actual AF to process. We illustrate the functioning of this approach, which is underlying the argumentation system ASPARTIX, briefly report on our experimental experiences, and give links to the relevant articles in the literature.

1 Motivation

In Artificial Intelligence (AI), the area of argumentation (the survey by [1] gives an excellent overview) has become one of the central issues during the last decade. Argumentation provides a formal treatment for reasoning problems arising in a number of application fields, including Multi-Agent Systems and Law Research. In a nutshell, so-called abstract argumentation frameworks (AFs) formalize statements together with a relation denoting rebuttals between them, such that the semantics gives an abstract handle to solve the inherent conflicts between statements by selecting acceptable subsets of them. The reasoning underlying such argumentation frameworks turned out to be a very general principle capturing many other important formalisms from the areas of AI and Knowledge Representation.

Argumentation problems are in general intractable, for instance deciding if an argument is contained in some preferred extension is known to be NP-complete. Therefore, developing dedicated algorithms for the different reasoning problems is non-trivial. A promising way to implement such systems is to use a reduction method, where the given problem is translated into another language, for which sophisticated systems already exist. It turned out that Answer-Set Programming (ASP) is especially well suited for this purpose.

Earlier work already proposed reductions from argumentation problems to certain target formalisms. Most notably are encodings in terms of (quantified) propositional logic [2, 6] and logic programs [10–12, 15] (see [14] for a survey). The main difference of this earlier work compared to our approach is the necessity to compile (at least, for some of the semantics) each problem instance into a different instance of the target formalism (e.g., into a different logic program). In our approach, *all* semantics are encoded

* This work has been funded by the Vienna Science and Technology Fund (WWTF) through project ICT08-028.

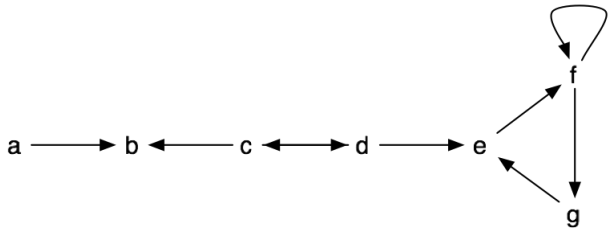

Fig. 1. The argumentation framework F from Example 1.

within a *fixed* query (independent from the concrete AF to process). Thus, we are more in the tradition of a classical implementation, because we construct an interpreter in ASP which takes an AF given as input.

2 ASP- Encodings for Abstract Argumentation Frameworks

Abstract argumentation frameworks have been first introduced by Dung [3] in 1995. It is a very simple but also very powerful formalism to reason over conflicting knowledge. The syntax only consists of a set of statements called *arguments* and a binary relation between them, the *attacks* denoting the conflicts between the arguments. As we are on the abstract level, we do not concentrate on the internal structure of the arguments but only on their relation to each other.

An *argumentation framework (AF)* is a pair $F = (A, R)$, where A is a finite set of arguments and $R \subseteq A \times A$. The pair $(a, b) \in R$ means that a attacks b. A set $S \subseteq A$ of arguments *attacks* b (in F), if there is an $a \in S$, such that $(a, b) \in R$. An argument $a \in A$ is *defended* by $S \subseteq A$ (in F) iff, for each $b \in A$, it holds that, if $(b, a) \in R$, then S attacks b (in F).

Such an AF can be represented as a directed graph as in the following example.

Example 1. Consider the AF $F = (A, R)$, consisting of the set or arguments $A = \{a, b, c, d, e, f, g\}$ and the set of attack relations $R = \{(a, b), (c, b), (c, d), (d, c), (d, e), (e, f), (f, f), (f, g), (g, e)\}$ as illustrated in Figure 1.

The inherent conflicts between the arguments are solved by selecting subsets of arguments, where a semantics σ assigns a collection of sets of arguments to an AF F. The basic requirement for all semantics is that none of the selected arguments attack each other; these sets are then called *conflict-free*. Then *admissible extensions* of an AF are those conflict-free sets which defend their arguments against all attacks.

In the following we present the ASP-encodings for admissible semantics as used in the system ASPARTIX (see [5] for a detailed description of most of the argumentation semantics and the corresponding encodings). First the input AF from $F = (A, R)$ is defined as,

$$\widehat{F} = \{\mathrm{arg}(a) \mid a \in A\} \cup \{\mathrm{att}(a, b) \mid (a, b) \in R\}. \quad (1)$$

Table 1. Complexity for decision problems in argumentation frameworks.

	adm	$pref$	$semi$	$stage$	$grd*$	$ground$	$cf2$	$stage2$
$Cred$	NP-c	NP-c	Σ_2^P-c	Σ_2^P-c	NP-c	in P	NP-c	Σ_2^P-c
$Skept$	(trivial)	Π_2^P-c	Π_2^P-c	Π_2^P-c	coNP-c	in P	coNP-c	Π_2^P-c

Then, the program π_{adm} computes admissible extensions by first guessing all subsets $S \subseteq A$ of arguments with $\mathrm{in}(.)$ (resp. $\mathrm{out}(.)$) denoting the arguments in (resp. not in) the set S. Then the constraints rule out those guesses which are not conflict free, or which do not defend their arguments.

$$\begin{aligned}\pi_{adm} = \{\ & \mathrm{in}(X) : -not\, \mathrm{out}(X), \mathrm{arg}(X);\\ & \mathrm{out}(X) : -not\, \mathrm{in}(X), \mathrm{arg}(X);\\ & : -\mathrm{in}(X), \mathrm{in}(Y), \mathrm{att}(X,Y);\\ & \mathrm{defeated}(X) : -\mathrm{in}(Y), \mathrm{att}(Y,X);\\ & : -\mathrm{in}(X), \mathrm{att}(Y,X), not\, \mathrm{defeated}(Y)\}.\end{aligned}$$

Typical reasoning tasks for an argumentation semantics σ are credulous and skeptical reasoning which are supported by most ASP solvers:

- $Cred_\sigma$: Given AF $F = (A, R)$ and $a \in A$. Is a contained in *some* $S \in \sigma(F)$?
- $Skept_\sigma$: Given AF $F = (A, R)$ and $a \in A$. Is a contained in *each* $S \in \sigma(F)$?

Depending on the computational complexity of the different semantics, ASPARTIX uses different techniques in the encodings.

- Stratified programs for grounded semantics [5];
- Normal programs for admissible, stable, complete (all in [5]) and $cf2$ [8] semantics;
- Disjunctive programs for preferred, semi-stable (both in [5]), stage [4] and $stage2$ semantics (on system-page);
- Manifold programs for ideal semantics [7];
- `metasp` optimization techniques (see [9]) for preferred, semi-stable, stage and resolution-based grounded semantics ($grd*$) [4].

3 Conclusion

An experimental evaluation of the encodings with different solvers like dlv, lparse, gringo, smodels, cmodels, clasp, claspD and gnt showed that AFs with up to 140 arguments can be used as input for most of the semantics [13]. For most of the encodings gringo/clasp or gringo/claspD outperformed the other solvers. Except for semi-stable semantics dlv performed better.

Furthermore, an evaluation of handcrafted saturation encodings versus the `metasp` optimization technique showed that the latter one not only makes the encodings easier but also performs surprisingly well [4].

All encodings incorporated in ASPARTIX are available at

http://www.dbai.tuwien.ac.at/research/project/argumentation/systempage/

and a web-application of ASPARTIX is provided under:

http://rull.dbai.tuwien.ac.at:8080/ASPARTIX

Bibliography

[1] Trevor J. M. Bench-Capon and Paul E. Dunne. Argumentation in artificial intelligence. *Artif. Intell.*, 171(10-15):619–641, 2007.

[2] Philippe Besnard and Sylvie Doutre. Checking the acceptability of a set of arguments. In James P. Delgrande and Torsten Schaub, editors, *Proceedings of the 10th International Workshop on Non-Monotonic Reasoning (NMR 2004)*, pages 59–64, 2004.

[3] Phan Minh Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artif. Intell.*, 77(2):321–358, 1995.

[4] Wolfgang Dvořák, Sarah Alice Gaggl, Johannes Peter Wallner, and Stefan Woltran. Making use of advances in answer-set programming for abstract argumentation systems. *CoRR*, abs/1108.4942, 2011.

[5] Uwe Egly, Sarah Gaggl, and Stefan Woltran. Answer-set programming encodings for argumentation frameworks. *In Argument and Computation*, 1(2):147–177, 2010.

[6] Uwe Egly and Stefan Woltran. Reasoning in argumentation frameworks using quantified boolean formulas. In Paul E. Dunne and Trevor J. M. Bench-Capon, editors, *Proceedings of the 1st Conference on Computational Models of Argument (COMMA 2006)*, volume 144 of *Frontiers in Artificial Intelligence and Applications*, pages 133–144. IOS Press, 2006.

[7] Wolfgang Faber and Stefan Woltran. Manifold answer-set programs for meta-reasoning. In *Gelfond Festschrift*, volume 6565 of *Lecture Notes in Artificial Intelligence*, pages 44–63. Springer, 2011.

[8] Sarah Alice Gaggl and Stefan Woltran. cf2 semantics revisited. In Pietro Baroni, Federico Cerutti, Massimiliano Giacomin, and Guillermo Ricardo Simari, editors, *Computational Models of Argument: Proceedings of COMMA 2010, Desenzano del Garda, Italy, September 8-10, 2010*, volume 216 of *Frontiers in Artificial Intelligence and Applications*, pages 243–254. IOS Press, 2010.

[9] Martin Gebser, Roland Kaminski, and Torsten Schaub. Complex optimization in answer set programming. *TPLP*, 11(4-5):821–839, 2011.

[10] Juan Carlos Nieves, Mauricio Osorio, and Ulises Cortés. Preferred extensions as stable models. *Theory and Practice of Logic Programming*, 8(4):527–543, 2008.

[11] Juan Carlos Nieves, Mauricio Osorio, and Claudia Zepeda. Expressing extension-based semantics based on stratified minimal models. In Hiroakira Ono, Makoto Kanazawa, and Ruy J. G. B. de Queiroz, editors, *Proceedings of the 16th International Workshop on Logic, Language, Information and Computation* (*WoLLIC 2009*)*, Tokyo, Japan, June 21-24*, volume 5514 of *Lecture Notes in Computer Science*, pages 305–319. Springer, 2009.

[12] Mauricio Osorio, Claudia Zepeda, Juan Carlos Nieves, and Ulises Cortés. Inferring acceptable arguments with answer set programming. In *Proceedings of the 6th Mexican International Conference on Computer Science (ENC 2005)*, pages 198–205. IEEE Computer Society, 2005.

[13] Michael Petritsch. Analysis of grounders and solvers for asp encodings of argumentation frameworks. Bachelor Thesis, Vienna University of Technology, 2010.

[14] Francesca Toni and Marek Sergot. Argumentation and answer set programming. In Marcello Balduccini and Tran Son, editors, *Logic Programming, Knowledge Representation, and Nonmonotonic Reasoning*, volume 6565 of *Lecture Notes in Computer Science*, pages 164–180. Springer Berlin / Heidelberg, 2011.

[15] Toshiko Wakaki and Katsumi Nitta. Computing argumentation semantics in answer set programming. In Hiromitsu Hattori, Takahiro Kawamura, Tsuyoshi Idé, Makoto Yokoo, and Yohei Murakami, editors, *New Frontiers in Artificial Intelligence* (*JSAI 2008*) *Conference and Workshops, Asahikawa, Japan, June 11-13*, volume 5447 of *Lecture Notes in Computer Science*, pages 254–269. Springer, 2008.