ASPARTIX-D: ASP Argumentation Reasoning Tool - Dresden

Sarah Alice Gaggl and Norbert Manthey

Institute of Artificial Intelligence, Technische Universität Dresden, Germany

Abstract. ASPARTIX-D is a system designed to evaluate abstract argumentation frameworks. It consists of collection of answer-set programming (ASP) encodings together with an optimized ASP (resp. SAT) solver configuration for each reasoning problem. The system meets the requirements of the first International Competition on Computational Models of Argumentation (ICCMA 2015).

Keywords: abstract argumentation, ASPARTIX-D, answer-set programming, SAT solving, system

1 Motivation

ASPARTIX-D is the version of ASPARTIX [3,2] which has been further developed in Dresden. In particular, necessary modifications have been performed for the participation in the first International Competition on Computational Models of Argumentation (ICCMA 2015)[1]. ASPARTIX-D consists of a collection of answer-set programming (ASP) encodings together with dedicated solvers to compute certain abstract argumentation reasoning tasks. In the following we introduce the necessary background of abstract argumentation frameworks and give an overview of the performed evaluation and final configuration of the system.

2 Semantics and Reasoning Tasks

Abstract argumentation frameworks (AFs) are defined according to [1].

Definition 1. *An* argumentation framework (AF) *is a pair $F = (A, R)$ where A is a set of arguments and $R \subseteq A \times A$ is the attack relation. The pair $(a, b) \in R$ means that a attacks b. An argument $a \in A$ is* defended *by a set $S \subseteq A$ if, for each $b \in A$ such that $(b, a) \in R$, there exists a $c \in S$ such that $(c, b) \in R$.*

Semantics for argumentation frameworks are given via a function σ which assigns to each AF $F = (A, R)$ a set $\sigma(F) \subseteq 2^A$ of extensions. We shall consider here for σ the functions ST, CO, PR, and GR which stand for stable, complete, preferred, and grounded semantics respectively.

[1] `http://argumentationcompetition.org/`.

Definition 2. *Let $F = (A, R)$ be an AF. A set $S \subseteq A$ is* conflict-free (in F)*, if there are no $a, b \in S$, such that $(a, b) \in R$. $cf(F)$ denotes the collection of conflict-free sets of F. For a conflict-free set $S \in cf(F)$, it holds that*

- *$S \in ST(F)$, if each $a \in A \setminus S$ is attacked by S;*
- *$S \in CO(F)$, if each $a \in A$ defended by S is contained in S;*
- *$S \in GR(F)$, if $S \in CO(F)$ and there is no $T \in CO(F)$ with $T \subset S$;*
- *$S \in PR(F)$, if $S \in CO(F)$ and there is no $T \in CO(F)$ with $T \supset S$.*

Typical reasoning tasks for any AF $F = (A, R)$ and a semantics σ are the following.

- *DC-σ credulous reasoning*, decide whether $a \in A$ is contained in any $S \in \sigma(F)$;
- *DS-σ skeptical reasoning*, decide whether $a \in A$ is contained in each $S \in \sigma(F)$;
- *EE-σ enumerate* all extensions $S \in \sigma(F)$;
- *SE-σ* return *some* extension $S \in \sigma(F)$.

2.1 Answer-Set Programming Encodings

ASPARTIX-D is a collection of ASP encodings as described in [3] and optimization encodings which we call in the following *metasp* encodings as given in [2]. These metasp encodings make use of the metasp optimization front-end for the ASP-package *gringo* & *clasp* (see [5] for more details).

The input AF should be specified in the *ASPARTIX syntax*, i.e. for each arguments $a \in A$ one specifies a fact $\mathrm{arg}(a).$ and for each attack $(a, b) \in R$ the fact $\mathrm{att}(a, b).$ should be generated. A typical call of ASPARTIX-D for the reasoning task DC-ST looks as follows.

```
./aspartix.sh -p DC-ST -f <file> -fo apx -a <argument>
```

In general ASP encodings are designed to return all (resp. n) solutions to a given problem. For credulous and skeptical reasoning we are only interested in a *YES* or *NO* decision. As the ASP encodings use predicates $\mathrm{in}/1$ and $\mathrm{out}/1$ to guess the extensions we can perform the following simple modifications. In case of credulous reasoning we just add the argument $a \in A$ in question as the fact $\mathrm{in}(a)$ to the program and check if there is one answer set. If this is the case then, the ASP-solver found one witness extension containing the argument a. Otherwise, if the program is unsatisfiable, we know there is no extension which contains a. For skeptical reasoning we perform a similar modification, where we add the fact $\mathrm{out}(a)$ to the program. If the program is satisfiable, i.e. an answer set is found, we know that the argument a can not be in each extension of the semantics σ. However, if the program is unsatisfiable, we obtain that a is skeptically inferred.

3 Evaluation

The main goal of the evaluation was to find the most suitable encodings & solver configuration. As the potassco ASP solvers[2] showed to perform very well for our purpose

[2] `http://potassco.sourceforge.net`

we decided to test several options from the ASP solver *Clingo 4.4* for the original encodings. Furthermore, we considered the *gringo3.0.5* & *clasp3.1.1* grounder & solver combination for the metasp encodings and the *lp2sat* & *riss* SAT Solver [4,7,6], for $DC\text{-}\{ST, CO, GR\}$ and $DS\text{-}\{ST, CO, GR\}$.

As benchmarks, we considered a collection of frameworks which have been used by different research groups for testing before consisting of structured and randomly generated AFs, resulting in 5829 frameworks. In particular we used parts of the instances Federico Cerutti provided to us which have been generated towards an increasing number of SCCs [8]. Further benchmarks were used to test the system *dynpartix* and we included the instances provided by the ICCMA 2015 organizers.

The computation has been performed on an Intel Xeon E5-2670 running at 2.6 GHz. From the 16 available cores we used only every fourth core to allow a better utilization of the CPU's cache. We applied a 15 minutes timeout and a maximum of 6.5 GB of main memory.

The tests showed in case of $EE\text{-}PR$ and $SE\text{-}PR$ the metasp encodings outperform all other options. Also surprisingly the metasp encodings for grounded semantics gave the best results, even though they are not adequate from a complexity point of view (see [2]). This might be due to the fact that the original encodings for grounded semantics use a certain loop construction which have a bad influence on the performance. For the decision tasks EE and SE of stable and complete semantics the *clingo* option `--project` returned the best results. Moreover, the *lp2sat* & *riss* combination gave better results for the reasoning tasks $DC\text{-}CO$ and $DS\text{-}CO$ on real problem instances.

The results of the evaluation led to the following final configuration.

Task	**Used configuration**
GR:	metasp encodings for all reasoning tasks
DC-ST	original
DC-CO	*lp2sat* & *riss*
DC-PR	`-configuration=auto`
DS-ST	original
DS-CO	*lp2sat* & *riss*
DS-PR	original
EE-ST	`--project`
EE-CO	`--project`
EE-PR	metasp
SE-ST	`--project`
SE-CO	`--project`
SE-PR	metasp

The system as well as the benchmarks used for the evaluation are available at `https://ddll.inf.tu-dresden.de/web/Sarah_Alice_Gaggl/ASPARTIX-D`.

4 Conclusion

The evaluation of different ASP-solver configurations and encoding types clearly showed that with optimized encodings one can obtain better results than only with specific solver options. However, the *clingo* option `--project` showed good results for several reasoning tasks. Furthermore, on real problem instances the SAT solver performed better for credulous and skeptical reasoning of complete semantics.

For future work we plan to expand the evaluation also for other argumentation semantics, as for nearly all of them ASP encodings exist (see `http://www.dbai.tuwien.ac.at/research/project/argumentation/systempage/#download`).

Acknowledgments: We thank the Center for Information Services and High Performance Computing (ZIH) at TU Dresden for generous capacity of parallel computation resources. Parts of this project were funded by the Deutsche Forschungsgemeinschaft (DFG) under the number HO 1294/11-1.

References

1. 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.
2. Wolfgang Dvořák, Sarah A. Gaggl, Johannes P. Wallner, and Stefan Woltran. Making use of advances in answer-set programming for abstract argumentation systems. In Hans Tompits, Salvador Abreu, Johannes Oetsch, Jörg Pührer, Dietmar Seipel, Masanobu Umeda, and Armin Wolf, editors, *Proceedings of the 19th International Conference on Applications of Declarative Programming and Knowledge Management (INAP 2011), Revised Selected Papers*, volume 7773 of *LNAI*, pages 114–133. Springer, 2013.
3. Uwe Egly, Sarah A. Gaggl, and Stefan Woltran. Answer-set programming encodings for argumentation frameworks. *Argument & Computation*, 1(2):147–177, 2010.
4. Martin Gebser, Tomi Janhunen, and Jussi Rintanen. Answer set programming as SAT modulo acyclicity. In Torsten Schaub, Gerhard Friedrich, and Barry O'Sullivan, editors, *Proceedings of the 21st European Conference on Artificial Intelligence (ECAI 2014)*, volume 263 of *FAIA*, pages 351–356. IOS Press, 2014.
5. Martin Gebser, Roland Kaminski, and Torsten Schaub. Complex optimization in answer set programming. *Theory and Practice of Logic Programming*, 11(4-5):821–839, 2011.
6. Norbert Manthey. Coprocessor 2.0 – A flexible CNF simplifier. In Alessandro Cimatti and Roberto Sebastiani, editors, *Proceedings of the 15th International Conference on Theory and Applications of Satisfiability Testing (SAT 2012)*, volume 7317 of *LNCS*, pages 436–441. Springer Berlin Heidelberg, 2012.
7. Norbert Manthey. Riss 4.27. In Anton Belov, Daniel Diepold, Marijn J.H. Heule, and Matti Järvisalo, editors, *Proceedings of SAT Competition 2014*, volume B-2014-2 of *Department of Computer Science Series of Publications B*, pages 65–67. University of Helsinki, Finland, 2014.
8. Mauro Vallati, Federico Cerutti, and Massimiliano Giacomin. Argumentation frameworks features: an initial study. In Torsten Schaub, Gerhard Friedrich, and Barry O'Sullivan, editors, *Proceedings of the 21st European Conference on Artificial Intelligence (ECAI 2014)*, volume 263 of *FAIA*, pages 1117–1118. IOS Press, 2014.