ArgSemSAT: Solving Argumentation Problems Using SAT

Federico CERUTTI [a,1], Massimiliano GIACOMIN [b], and Mauro VALLATI [c]
[a] *Department of Computing Science, King's College, University of Aberdeen, UK*
[b] *Department of Information Engineering, University of Brescia, Italy*
[c] *School of Computing and Engineering, University of Huddersfield, UK*

Abstract. In this paper we describe the system ArgSemSAT which includes algorithms which we proved to overcome current state-of-the-art performances in enumerating preferred extensions.

Keywords. argumentation, argumentation semantics, extensions enumeration

1. Introduction

Dung's theory of abstract argumentation frameworks provides a fundamental reference in computational argumentation in virtue of its simplicity, generality, and ability to capture a variety of more specific approaches as special cases. An abstract argumentation framework (*AF*) consists of a set of arguments and an *attack* relation between them. The concept of *extension* plays a key role in this simple setting, where an *extension* is intuitively a set of arguments which can "survive the conflict together". Different notions of extensions and of the requirements they should satisfy correspond to alternative *argumentation semantics*.

The main computational problems in abstract argumentation are related to extensions and can be partitioned into two classes: *decision* problems and *construction* problems. In particular, *extension enumeration* requires to construct *all* extensions prescribed for a given *AF*: its solution provides complete information concerning the justification status of arguments and subsumes the solutions to the other problems. On the practical side, few results are available on the development of efficient algorithms for abstract argumentation and their empirical assessment. In [1, 2] SAT-based approaches have been proposed to solve the extension enumeration problem for preferred semantics.

2. Overview of ArgSemSAT

ArgSemSAT is a collection of algorithms[2] mainly developed for enumerating preferred extensions which proved [1, 2] to be more efficient, in terms of time performance, than current state-of-the-art approaches.

[1] Corresponding Author
[2] http://tiny.cc/argsemsat

Among other algorithms, `ArgSemSAT` implements both PrefSAT [1], and SCC-P [2]. PrefSAT first solves a SAT problem whose solutions correspond to the complete extensions of an AF. Then, a hill-climbing approach is used to find a maximal w.r.t. set inclusion complete extension, i.e. a preferred extension. Previous states of the search — extensions already found — are excluded from subsequent search steps.

SCC-P [2] implements the SCC-recursiveness schema [3] which exploits the partial order of strongly connected components (SCCs). First, the extensions of the frameworks restricted to the initial (i.e. not receiving any attack) SCCs are computed and combined together. Then each SCC which is attacked only by initial SCCs is considered: the extensions of such a SCC are locally computed and merged with those already obtained. Then the subsequent (w.r.t. the partial order) SCCs are considered until no remaining SCCs are left to process. The schema is recursive: for each SCC and for each extension selected in the previous SCCs (1) all arguments attacked by the extension are suppressed; (2) the procedure is recursively applied to the remaining part of the SCC. The base of the recursion is reached when there is only one SCC: in this case a solver similar to PrefSAT is called. Unsurprisingly, SCC-P proved [2] to be more efficient than PrefSAT on AFs with a significant number of SCCs. In the other cases, PrefSAT is much more efficient [1, 2] than the other state-of-the-art solvers.

`ArgSemSAT` is written in C++ and integrates two alternative SAT solvers: PrecoSAT [4] and Glucose [5], respectively the winner of the SAT Competition 2009 on the Application track; and the winner of the SAT Competition in 2011 and of the SAT Challenge 2012 on the Application track. `ArgSemSAT` provides a CLI, can read Aspartix-like input file, and returns the computed extensions.

3. Conclusions

`ArgSemSAT` is at its first stage of development, but the implementation showed [1, 2] to overcome, in terms of efficiency, the current state-of-the-art techniques for enumerating preferred extensions. Moreover, as evidence in [1, 2], there are several theoretical open questions that, once addressed, could lead to a further increase in the performance of `ArgSemSAT`.

References

[1] Federico Cerutti, Paul E. Dunne, Massimiliano Giacomin, and Mauro Vallati. Computing Preferred Extensions in Abstract Argumentation: A SAT-Based Approach. In *TAFA 2013*, pages 176–193. Springer, 2014.

[2] Federico Cerutti, Massimiliano Giacomin, Mauro Vallati, and Marina Zanella. A SCC Recursive Meta-Algorithm for Computing Preferred Labellings in Abstract Argumentation. In *Proceedings of KR 2014*, 2014.

[3] Pietro Baroni, Massimiliano Giacomin, and Giovanni Guida. SCC-recursiveness: a general schema for argumentation semantics. *Artificial Intelligence*, 168(1-2):165–210, 2005.

[4] Armin Biere. P{re,ic}oSAT@SC'09. In *SAT Competition 2009*, 2009.

[5] Gilles Audemard and Laurent Simon. GLUCOSE 2.1. `http://www.labri.fr/perso/lsimon/glucose/`, 2012.