μ–TOKSIA AT ICCMA'21

Andreas Niskanen
HIIT, Department of Computer Science
University of Helsinki
Finland
andreas.niskanen@helsinki.fi

Matti Järvisalo
HIIT, Department of Computer Science
University of Helsinki
Finland

ABSTRACT

We shortly describe the μ-TOKSIA system, a SAT-based abstract argumentation reasoner participating in the 4th International Competition on Computational Models of Argumentation (ICCMA 2021).

https://bitbucket.org/andreasniskanen/mu-toksia

1 System Architecture

μ-TOKSIA [1] is a SAT-based system for reasoning in argumentation frameworks [2] implemented in C++ and released under the open-source MIT license. Features for dynamic abstract argumentation [3] are dropped from the ICCMA'21 version,[1] since the corresponding track was cancelled. There are four main components in the implementation:

- `AF.h`: Argumentation framework as a class, which maps arguments (strings) to indices (integers) and back, stores the attackers of each argument, and maps argument indices to corresponding Boolean variables.
- `SATSolver.h`: A generic interface to a SAT solver, which declares functions for adding a clause to the solver, for solving the current instance (under given assumptions), and for performing unit propagation.
- `Encodings.h`: Implements the SAT encodings of argumentation semantics [4], including functions for conflict-free, admissible, complete, and stable semantics, and a function for defining the range. Each function takes as input an AF and a SAT solver and instantiates the SAT solver with the corresponding semantics.
- `CredAcceptance.h`, `SkeptAcceptance.h`, `SingleExtension.h`, `CountExtensions.h`: Implementations of the SAT-based algorithms for the corresponding reasoning tasks.

The ICCMA'21 version of μ-TOKSIA includes CryptoMiniSat (version 5.8.0) [5] as the underlying SAT solver. In contrast to [1], we use CryptoMiniSat in its default configuration, without explicitly setting the default polarity, since we observed in our initial experiments that this can degrade performance of the new version of CryptoMiniSat. For ICCMA'21, we submit three versions of μ-TOKSIA: single-thread, 4-thread, and 8-thread. In multi-threaded versions CryptoMiniSat is run in parallel.[2] However, no parallelization is performed within μ-TOKSIA for the time being.

All algorithms employ incremental SAT solving via the assumptions interface, that is, for iterative calls the state of the SAT solver maintained. This means that only a single SAT solver is instantiated during a single run of the program (with the exception of counting extensions under semi-stable and stage semantics, where one SAT solver is used for iteratively maximizing the range, and another for enumerating complete extensions or confict-free sets with a fixed range).

As μ-TOKSIA was originally designed for dynamic tasks, the ICCMA'21 version turns out to be quite minimalistic: the header files contain 157 lines of code, and the source files only 1504 lines of code.

[1] Somewhat ironically, given the name of the solver which stems from "changes" in Finnish language (*muutoksia*).
[2] https://www.msoos.org/2018/07/cryptominisat-and-parallel-sat-solving/

2 Supported Tasks and Semantics

All tasks of the static abstract argumentation track are supported:

- credulous acceptance (**DC**-σ),
- skeptical acceptance (**DS**-σ),
- finding one extension (**SE**-σ),
- extension counting (**CE**-σ),

under the semantics $\sigma \in \{\mathbf{CO}, \mathbf{PR}, \mathbf{ST}, \mathbf{SST}, \mathbf{STG}, \mathbf{ID}\}$ (complete, preferred, stable, semi-stable, stage, and ideal, respectively).

3 Algorithms

For a detailed description of the SAT encodings and algorithms, we refer the reader to [1]. Briefly put, for complete and stable semantics, credulous and skeptical acceptance are solved by a single SAT solver call, as is finding a single extension (with the exception of skeptical acceptance and finding one extension under complete, for which polynomial-time unit propagation yields the grounded extension—this feature is however dropped from the parallel version). On the other hand, for preferred, semi-stable, and stage semantics, a CEGAR procedure for credulous and skeptical acceptance is implemented as described in [6] (with the exception of credulous acceptance under preferred, which is solved via complete). A single extension can be found by iteratively maximizing a complete extension (for preferred and semi-stable) or a conflict-free set (for stage semantics). Maximization is range-based for semi-stable and stage semantics. Finally, we compute the ideal extension by computing the union of complete extensions, removing arguments that are attacked by this union, and maximizing a complete extension within this set. This procedure is also adapted to acceptance under ideal semantics.

The newly introduced task of extension counting is implemented simply via extension enumeration by incrementing a counter instead of printing an extension. Enumeration is implemented by iteratively blocking solutions via adding clauses. In contrast to [1], we block the exact extension found only for complete semantics, instead blocking subsets of the extension found also for stable semantics (as stable extensions clearly cannot be in a subset relation), similarly as under preferred semantics. In addition, for semi-stable and stage semantics, when enumerating extensions corresponding to a fixed subset-maximal range found via an iterative SAT-based maximization procedure, we also block subsets (as conflict-free sets with the same range clearly cannot be in a subset relation).

References

[1] Andreas Niskanen and Matti Järvisalo. μ-toksia: An efficient abstract argumentation reasoner. In *Proc. KR*, pages 800–804, 2020.

[2] 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.

[3] Andreas Niskanen and Matti Järvisalo. Algorithms for dynamic argumentation frameworks: An incremental SAT-based approach. In *Proc. ECAI*, volume 325 of *FAIA*, pages 849–856. IOS Press, 2020.

[4] Philippe Besnard and Sylvie Doutre. Checking the acceptability of a set of arguments. In *Proc. NMR*, pages 59–64, 2004.

[5] Mate Soos, Karsten Nohl, and Claude Castelluccia. Extending SAT solvers to cryptographic problems. In *Proc. SAT*, volume 5584 of *LNCS*, pages 244–257. Springer, 2009.

[6] Wolfgang Dvorák, Matti Järvisalo, Johannes Peter Wallner, and Stefan Woltran. Complexity-sensitive decision procedures for abstract argumentation. *Artif. Intell.*, 206:53–78, 2014.