A Tool for Reasoning in Assumption-based Argumentation using Tree-decompositions

Andrei Popescu
Johannes P. Wallner
Institute of Software Technology
Graz University of Technology
Graz, Austria

***Abstract*—ASTRA is system for reasoning in assumption-based argumentation (ABA) that exploits tree-decompositions. The solver takes an ABA framework as input, and uses dynamic-programming (DP) algorithms for reasoning. This is achieved through the D-FLAT framework, which allows for declaratively specifying DP algorithms in answer set programming (ASP). The DP algorithms operate on a tree decomposition of a given ABA framework.**

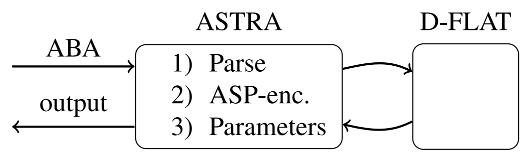

Fig. 1. System architecture of ASTRA

I. Introduction

In this system description we present ASTRA, a solver submission to the assumption-based argumentation (ABA) [1] track of the fifth International Competition on Computational Models of Argumentation (ICCMA), which follows successful previous editions [2]–[5]. The solver utilizes tree-decompositions and tree-width [6] to perform reasoning in ABA frameworks.

Informally, tree-width is a notion on graphs, which can be interpreted as measuring "closeness" of graphs to a tree. Tree-width can be formalized using the concept of tree-decompositions, which decompose a given graph into a tree-decomposition. Optimal tree-decompositions then give the tree-width of a given graph. In addition, tree-decompositions are utilized as a datastructure for dynamic programming algorithms to operate on. In this approach, for a given problem that can be represented on graphs, a tree-decomposition is constructed, following a (often bottom-up) computation on the tree-decomposition.

Our solver makes use of the recent D-FLAT framework [7]–[9] that enables specification of dynamic programming algorithms on tree-decompositions in answer set programming (ASP) [10], [11]. D-FLAT internally constructs a tree-decomposition of the given instance, using the *htd* library [12], and performs the dynamic programming algorithm as specified in ASP.

II. Background

We briefly recap ABA frameworks and credulous and skeptical reasoning in ABA. For a detailed overview, see, e.g., the chapter by [13]. As specified in the competition, we consider the logic programming fragment of ABA [1].

An ABA framework consists of a set of atoms $\mathcal{L}$, a set of assumptions $\mathcal{A} \subseteq \mathcal{L}$, a set of rules of the form $h \leftarrow b_1, \ldots, b_n$ with $b_i \in \mathcal{L}$ and $h \in \mathcal{L} \setminus \mathcal{A}$, and a contrary function assigning assumptions $a \in \mathcal{A}$ a contrary $\overline{a} = x \in \mathcal{L}$.

A subset of the assumptions $A \subseteq \mathcal{A}$ attacks an assumption $a \in \mathcal{A}$ if one can derive $\overline{a}$ using the rules in $\mathcal{R}$, starting from A. A subset of the assumptions is conflict-free if the set does not attack itself. A subset of assumptions A defends another set of assumptions B if whenever a set of assumptions C attacks B, we find that A attacks C.

We define a conflict-free set of assumptions to be admissible if the set defends itself. If, additionally, every assumption set defended is included, the set is called a complete assumption set. A conflict-free assumption set is stable if every assumption $\{a\} \subseteq \mathcal{A}$ is attacked by the set (i.e., every singleton assumption set outside is attacked).

An atom is credulously accepted under a semantics, such as admissibility, if the atom is derivable from at least one assumptions set of the semantics. Dually, an atom is skeptically accepted if the atom is derivable from every assumption set under the semantics.

III. System Architecture

We briefly describe the architecture of our solver, which also can be seen in Figure 1.

The solver accepts as input an ABA framework, a query, and a semantics, in the input specified by the ICCMA 2023 competition. Our solver contains ASP encodings describing how to solve the ABA reasoning task on tree-decompositions. The underlying graph structure of an ABA instance is given as follows: the vertices are composed of the atoms $\mathcal{L}$ and rules $\mathcal{R}$ and an atom is connected to a rule whenever the atom occurs in the rule (head or body). Two atoms are connected if one is the contrary of the other. The ASP encodings then specify what D-FLAT has to compute in each bag of the tree-decomposition. The query is added into the problem instance by our solver.

Subsequently, the dynamic programming algorithm is executed by D-FLAT, according to our ASP specification. Based on the output of D-FLAT, our solver computes the answer to the posed query.

IV. Supported Reasoning Tasks

The solver ASTRA supports the following ABA reasoning tasks:

- DC-CO, DC-ST: credulous acceptance under complete and stable semantics, and
- DC-CO, DC-ST: skeptical acceptance under complete and stable semantics, and
- SE-CO, SE-ST: return of one acceptable set under complete and stable semantics.

V. Conclusions

In this system description we presented our solver ASTRA, an entry to the fifth ICCMA competition for the ABA track. Our solver is based on dynamic programming on tree-decomposition of the input ABA framework, where we utilize the recent D-FLAT framework for declaratively specifying the dynamic programming algorithm in ASP.

Acknowledgements

This research was supported the Austrian Science Fund (FWF) P35632.

References

[1] A. Bondarenko, P. M. Dung, R. A. Kowalski, and F. Toni, "An abstract, argumentation-theoretic approach to default reasoning," *Artif. Intell.*, vol. 93, pp. 63–101, 1997.

[2] M. Thimm and S. Villata, "The first international competition on computational models of argumentation: Results and analysis," *Artif. Intell.*, vol. 252, pp. 267–294, 2017.

[3] S. A. Gaggl, T. Linsbichler, M. Maratea, and S. Woltran, "Summary report of the second international competition on computational models of argumentation," *AI Mag.*, vol. 39, no. 4, pp. 77–79, December 2018.

[4] S. Bistarelli, L. Kotthoff, F. Santini, and C. Taticchi, "Summary report for the third international competition on computational models of argumentation," *AI Mag.*, vol. 42, no. 3, pp. 70–73, 2021.

[5] J. Lagniez, E. Lonca, J. Mailly, and J. Rossit, "Introducing the fourth international competition on computational models of argumentation," in *Proc. SAFA*, ser. CEUR Workshop Proceedings, S. A. Gaggl, M. Thimm, and M. Vallati, Eds., vol. 2672. CEUR-WS.org, 2020, pp. 80–85.

[6] H. L. Bodlaender, "A tourist guide through treewidth," *Acta Cybern.*, vol. 11, no. 1-2, pp. 1–21, 1993.

[7] M. Abseher, B. Bliem, G. Charwat, F. Dusberger, M. Hecher, and S. Woltran, "The D-FLAT system for dynamic programming on tree decompositions," in *Proc. JELIA*, ser. Lecture Notes in Computer Science, E. Fermé and J. Leite, Eds., vol. 8761. Springer, 2014, pp. 558–572.

[8] B. Bliem, G. Charwat, M. Hecher, and S. Woltran, "D-FLAT^2: Subset minimization in dynamic programming on tree decompositions made easy," *Fundam. Informaticae*, vol. 147, no. 1, pp. 27–61, 2016.

[9] M. Abseher, B. Bliem, M. Hecher, M. Moldovan, and S. Woltran, "Dynamic programming on tree decompositions with D-FLAT," *Künstliche Intell.*, vol. 32, no. 2-3, pp. 191–192, 2018.

[10] M. Gelfond and V. Lifschitz, "The stable model semantics for logic programming," in *Proc. ICLP*, R. A. Kowalski and K. A. Bowen, Eds. MIT Press, 1988, pp. 1070–1080.

[11] I. Niemelä, "Logic programs with stable model semantics as a constraint programming paradigm," *Ann. Math. Artif. Intell.*, vol. 25, no. 3-4, pp. 241–273, 1999.

[12] M. Abseher, N. Musliu, and S. Woltran, "htd - A free, open-source framework for (customized) tree decompositions and beyond," in *Proc. CPAIOR*, ser. Lecture Notes in Computer Science, D. Salvagnin and M. Lombardi, Eds., vol. 10335. Springer, 2017, pp. 376–386.

[13] K. Čyras, X. Fan, C. Schulz, and F. Toni, "Assumption-based argumentation: Disputes, explanations, preferences," in *Handbook of Formal Argumentation*, P. Baroni, D. Gabbay, M. Giacomin, and L. van der Torre, Eds. College Publications, 2018, ch. 7, pp. 365–408.