prefMaxSAT: Exploiting MaxSAT for Enumerating Preferred Extensions

Mauro Vallati[1], Federico Cerutti[2], Wolfgang Faber[1], and Massimiliano Giacomin[3]

[1] School of Computing and Engineering,
University of Huddersfield, UK
{m.vallati,w.faber}@hud.ac.uk

[2] School of Natural and Computing Science,
Kings College, University of Aberdeen, UK
f.cerutti@abdn.ac.uk

[3] Department of Information engineering,
University of Brescia, Italy
massimiliano.giacomin@unibs.it

Abstract. In this paper we introduce **prefMaxSAT**, a solver that exploits an efficient encoding of preferred extensions search for abstract argumentation, using the MaxSAT approach.

1 Introduction

The main computational problems in abstract argumentation include *decision* and *construction* problems, and turn out to be computationally intractable for most of argumentation semantics [6]. In this paper we focus on the *extension enumeration* problem for the preferred semantics, i.e. constructing *all* preferred extensions for a given AF: its solution provides complete information about the justification status of arguments and subsumes the solutions to the other problems.

In this paper, we propose an efficient encoding of preferred extensions search using unweighted MaxSAT. The maximum satisfiability problem is the problem of identifying the maximum number of clauses, of a given boolean formula, that can be made true together by an assignment of the involved variables. In unweighted MaxSAT, two class of clauses are considered: hard and soft. The former must be always satisfied, while the number of soft clauses satisfied is the object of the maximisation. The interested reader is referred to [7] for a detailed introduction. MaxSAT can be considered as a generalisation of the SAT problem, which looks for a variable assignment that satisfies all the clauses at the same time.

2 Background

An argumentation framework [5] consists of a set of arguments[4] and a binary attack relation between them.

Definition 1. *An* argumentation framework *(AF) is a pair* $\Gamma = \langle \mathcal{A}, \mathcal{R} \rangle$ *where* $\mathcal{A}$ *is a set of arguments and* $\mathcal{R} \subseteq \mathcal{A} \times \mathcal{A}$*. We say that* $\boldsymbol{b}$ attacks $\boldsymbol{a}$ *iff* $\langle \boldsymbol{b}, \boldsymbol{a} \rangle \in \mathcal{R}$*, also denoted as* $\boldsymbol{b} \rightarrow \boldsymbol{a}$*. The set of attackers of an argument* $\boldsymbol{a}$ *will be denoted as* $\boldsymbol{a}^- \triangleq \{\boldsymbol{b} : \boldsymbol{b} \rightarrow \boldsymbol{a}\}$*, the set of arguments attacked by* $\boldsymbol{a}$ *will be denoted as* $\boldsymbol{a}^+ \triangleq \{\boldsymbol{b} : \boldsymbol{a} \rightarrow \boldsymbol{b}\}$*.*

Each *AF* has an associated directed graph where the vertices are the arguments, and the edges are the attacks.

The basic properties of conflict–freeness, acceptability, and admissibility of a set of arguments are fundamental for the definition of argumentation semantics.

Definition 2. *Given an AF* $\Gamma = \langle \mathcal{A}, \mathcal{R} \rangle$*:*

- *a set* $S \subseteq \mathcal{A}$ *is a* conflict–free *set of* Γ *if* $\nexists\ \boldsymbol{a}, \boldsymbol{b} \in S$ *s.t.* $\boldsymbol{a} \rightarrow \boldsymbol{b}$*;*
- *an argument* $\boldsymbol{a} \in \mathcal{A}$ *is* acceptable *with respect to a set* $S \subseteq \mathcal{A}$ *of* Γ *if* $\forall \boldsymbol{b} \in \mathcal{A}$ *s.t.* $\boldsymbol{b} \rightarrow \boldsymbol{a}$*,* $\exists\ \boldsymbol{c} \in S$ *s.t.* $\boldsymbol{c} \rightarrow \boldsymbol{b}$*;*
- *a set* $S \subseteq \mathcal{A}$ *is an* admissible set *of* Γ *if* S *is a conflict–free set of* Γ *and every element of* S *is acceptable with respect to* S *of* Γ*.*

An argumentation semantics σ prescribes for any *AF* Γ a set of *extensions*, denoted as $\mathcal{E}_\sigma(\Gamma)$, namely a set of sets of arguments satisfying the conditions dictated by σ. Here we recall the definitions of preferred (denoted as $\mathcal{PR}$) semantics.

Definition 3. *Given an AF* $\Gamma = \langle \mathcal{A}, \mathcal{R} \rangle$*, a set* $S \subseteq \mathcal{A}$ *is a* preferred extension *of* Γ*, i.e.* $S \in \mathcal{E}_{\mathcal{PR}}(\Gamma)$*, iff* S *is a maximal (w.r.t.* $\subseteq$*) admissible set of* Γ*.*

As discussed in [3, 4] the search for admissible sets can be encoded using propositional logic formulae.

Definition 4. *Given an AF* $\Gamma = \langle \mathcal{A}, \mathcal{R} \rangle$*,* $\mathcal{L}$ *a propositional language, and* $v : \mathcal{A} \mapsto \mathcal{L}$*:*

$$adm_\Gamma = \bigwedge_{\boldsymbol{a} \in \mathcal{A}} \left(\left(v(\boldsymbol{a}) \supset \bigwedge_{\boldsymbol{b} \rightarrow \boldsymbol{a}} \neg v(\boldsymbol{b}) \right) \wedge \left(v(\boldsymbol{a}) \supset \bigwedge_{\boldsymbol{b} \rightarrow \boldsymbol{a}} \bigvee_{\boldsymbol{c} \rightarrow \boldsymbol{b}} v(\boldsymbol{c}) \right) \right)$$

The models of adm_Γ *corresponds to the admissible sets of* Γ*.*

[4] In this paper we consider only *finite* sets of arguments: see [2] for a discussion on infinite sets of arguments.

3 The MaxSAT Encoding

The approach we propose, called **prefMaxSAT**, is based on MaxSAT to identify the maximal admissible extensions, namely preferred extensions. Each step of the search process requires the solution of a MaxSAT problem. Precisely, the algorithm is based on the idea of encoding the constraints corresponding to admissible labellings of an AF as a MaxSAT problem, and then iteratively producing and solving modified versions of the initial problem.

Given an AF $\Gamma = \langle \mathcal{A}, \mathcal{R} \rangle$ we are interested in identifying a boolean formula, composed by hard and soft clauses, such that each assignment satisfying all the hard clauses of the formula corresponds to an admissible labelling, and each assignment satisfying the hard clauses and maximising the number of soft clauses satisfied corresponds to a preferred labelling.

Definition 5. *Given an AF $\Gamma = \langle \mathcal{A}, \mathcal{R} \rangle$, $\mathcal{L}$ a propositional language and $v : \mathcal{A} \mapsto \mathcal{L}$, the unweighted MaxSAT encoding for preferred semantics of Γ, Π_Γ, is given by the conjunction of the hard clauses listed below:*

$$\bigwedge_{\{\boldsymbol{a} \in \mathcal{A} | \boldsymbol{a}^- = \emptyset\}} v(\boldsymbol{a}) \tag{1}$$

$$\bigwedge_{\{\boldsymbol{a} \in \mathcal{A} | \boldsymbol{a}^- \neq \emptyset\}} \left(\bigwedge_{\boldsymbol{b} \to \boldsymbol{a}} (\neg v(\boldsymbol{a}) \vee \neg v(\boldsymbol{b})) \right) \tag{2}$$

$$\bigwedge_{\{\boldsymbol{a} \in \mathcal{A} | \boldsymbol{a})^- \neq \emptyset\}} \left(\neg v(\boldsymbol{a}) \vee \left(\bigvee_{\{\boldsymbol{c} \in \mathcal{A} | \exists \boldsymbol{b},\ \boldsymbol{c} \to \boldsymbol{b} \ \wedge\ \boldsymbol{b} \to \boldsymbol{a}\}} v(\boldsymbol{b}) \right) \right) \tag{3}$$

and by the conjunction of the following soft clauses:

$$\bigwedge_{\boldsymbol{a} \in \mathcal{A}} v(\boldsymbol{a}) \tag{4}$$

The hard clauses of Definition 5 can be related to the definition of admissible sets (Def. 2): clauses (2) enforce the conflict–freeness; clauses (3) ensure that each argument in the admissible set is defended.

Finally, soft clauses (4) are used for maximising the number of arguments included in the admissible set, thus identifying preferred extensions. To this end, let us consider a function α that, given a variable assignment T, returns $S \subseteq \mathcal{A}$ such that $\mathbf{a} \in S$ iff $v(\mathbf{a}) \equiv \top$ in T.

4 Implementation Details

To enumerate all the preferred extension, **prefMaxSAT** exploits a MaxSAT solver able to prove unsatisfiability too: it accepts as input a CNF formula,

composed by soft and hard clauses, and returns a variable assignment maximally satisfying the formula if it exists. If no variable assignment satisfies the hard constraints, ε should be returned.

Initially, the original AF is encoded in a CNF, as depicted in Definition 5. The CNF is then provided to the MaxSAT solver. If a variable assignment that maximally satisfies the formula is returned: (i) the corresponding labelling is saved in the list of found preferred extensions; (ii) a hard clause for eliminating the solution is added to the CNF; (iii) a hard clause forcing to include different arguments is added to the CNF; (iv) the process is repeated. If the MaxSAT solver returned ε, **prefMaxSAT** ends and provides the set of found preferred extensions.

The algorithm has been implemented in C++, and exploits the open-wbo MaxSAT framework [8], using glucose3.0 [1]. It should be noted that **prefMaxSAT** can be used with any MaxSAT system supporting the DIMACS format.

Acknowledgement

The authors would like to acknowledge the use of the University of Huddersfield Queensgate Grid in carrying out this work.

References

1. Audemard, G., Simon, L.: Lazy clause exchange policy for parallel sat solvers. In: Theory and Applications of Satisfiability Testing–SAT 2014, pp. 197–205 (2014)
2. Baroni, P., Cerutti, F., Dunne, P.E., Giacomin, M.: Automata for Infinite Argumentation Structures. Artif. Intell. 203(0), 104–150 (May 2013)
3. Besnard, P., Doutre, S.: Checking the acceptability of a set of arguments. In: Proc. of the 10th Int. Workshop on Non-Monotonic Reasoning (NMR 2004). pp. 59–64 (2004)
4. Charwat, G., Dvoák, W., Gaggl, S.A., Wallner, J.P., Woltran, S.: Methods for solving reasoning problems in abstract argumentation A survey. Artificial Intelligence 220, 28–63 (Mar 2015)
5. Dung, P.M.: On the Acceptability of Arguments and Its Fundamental Role in Nonmonotonic Reasoning, Logic Programming, and n-Person Games. Artificial Intelligence 77(2), 321–357 (1995)
6. Dunne, P.E., Wooldridge, M.: Complexity of abstract argumentation. In: Rahwan, I., Simari, G. (eds.) Argumentation in AI, chap. 5, pp. 85–104. Springer-Verlag (2009)
7. Li, C.M., Manya, F.: Maxsat, hard and soft constraints. Handbook of satisfiability 185, 613–631 (2009)
8. Martins, R., Manquinho, V., Lynce, I.: Open-wbo: a modular maxsat solver. In: Theory and Applications of Satisfiability Testing–SAT 2014, pp. 438–445 (2014)