Extension Enforcement under Grounded Semantics in Abstract Argumentation

Andreas Niskanen
University of Helsinki, Finland

Johannes P. Wallner
TU Wien, Austria

Matti Järvisalo
University of Helsinki, Finland

Abstract

The study of dynamics in abstract argumentation gives rise to optimization problems that are NP-hard also under the grounded semantics, in contrast to argument acceptance problems over argumentation frameworks (AF). Developing efficient systems for AF reasoning under grounded semantics has received less attention compared to other central AF semantics under which acceptance is NP-hard. In particular, grounded semantics is not currently supported by recent systems for extension enforcement, despite (or due to) its non-triviality. In this work, we propose and empirically evaluate three first approaches to enforcement under grounded semantics. While each of the approaches is based on employing constraint optimization solvers, we show empirically that there are significant differences in the scalability of the approaches.

1 Introduction

Argumentation is a vibrant area of modern KR and AI research. Abstract argumentation provides via argumentation frameworks (AFs) (Dung 1995) one of the central knowledge representation and reasoning formalisms for the study of computational aspects of argumentation. In contrast to the case of various other central semantics in abstract argumentation (Baroni, Caminada, and Giacomin 2011), the well-studied acceptance problems over AFs are polynomial-time decidable under the grounded semantics (Dung 1995), one of the central AF semantics. Thereby developing efficient systems for AF reasoning under the grounded semantics has received less attention in comparison to the other important AF semantics under which deciding acceptance is NP-hard (Dunne and Wooldridge 2009; Dunne and Bench-Capon 2002; Dimopoulos and Torres 1996). However, the recent line of research on dynamics in abstract argumentation (Baumann and Brewka 2010; Cayrol, de Saint-Cyr, and Lagasquie-Schiex 2010; Baumann 2012; Baumann and Brewka 2015; Coste-Marquis et al. 2015; Diller et al. 2015; de Saint-Cyr et al. 2016)—motivated by the fact that argumentation is intrinsically a dynamic process—gives rise to interesting optimization problems which have proven to be NP-hard also under the grounded semantics. Specifically, extension enforcement (Baumann and Brewka 2010) under the grounded semantics, where the task is to make a given set of arguments the grounded extension of a given AF by changing the attack relation of the AF, has been recently shown to be NP-hard (Wallner, Niskanen, and Järvisalo 2017) under the distance measure of minimizing the number of changes to the attack structure. In terms of application scenarios, enforcement as a formal problem situates e.g. within the field of persuasion, where a certain goal is to be achieved by bringing new information to a debate (de Saint-Cyr et al. 2016).

First algorithms with implemented systems for extension enforcement (Coste-Marquis et al. 2015; Niskanen, Wallner, and Järvisalo 2016), covering various other semantics have been proposed. However, no systems for extension enforcement under grounded semantics are currently available, despite—or due to—the fact that supporting this semantics has turned out to be computationally non-trivial. In this work, we bridge this gap by proposing and empirically evaluating several algorithmic approaches to enforcement under grounded semantics. The approaches are based on declarative techniques, each making use of distinct declarative aspects. In particular, we propose (i) to extend an earlier-proposed answer set programming (ASP) encoding (Egly, Gaggl, and Woltran 2010) for the grounded extension to the realm of extension enforcement; (ii) a new direct maximum satisfiability (MaxSAT) encoding for the problem; as well as (iii) a MaxSAT-based counterexample-guided abstraction refinement (CEGAR) (Clarke et al. 2003; Clarke, Gupta, and Strichman 2004) procedure that makes use of earlier theoretical observations (Boella, Kaci, and van der Torre 2009a; 2009b; Rienstra, Sakama, and van der Torre 2015) for achieving a non-trivial, tighter refinement step. We show empirically that there are significant differences in the scalability of the approaches when applied on extension enforcement instances arising from benchmarks from the latest ICCMA AF solver competition (Gaggl et al. 2016), with the two new MaxSAT-based approaches dominating the ASP encoding.

2 Argumentation Frameworks

We recall argumentation frameworks (Dung 1995) and their semantics (Baroni, Caminada, and Giacomin 2011).

Definition 1. An *argumentation framework (AF)* is a pair $F = (A, R)$ where A is a finite 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* (in F) by a set $S \subseteq A$ if, for each $b \in A$ s.t. $(b,a) \in R$, there is a $c \in S$ s.t. $(c,b) \in R$.

Semantics for AFs are defined through a function σ which assigns to each AF $F = (A,R)$ a set $\sigma(F) \subseteq 2^A$ of extensions. Here we consider the functions *adm*, *com*, and *grd*, which stand for admissible, complete, and grounded extensions, respectively.

Definition 2. Given an AF $F = (A,R)$, the *characteristic function* $\mathscr{F}_F : 2^A \to 2^A$ of F is $\mathscr{F}_F(S) = \{x \in A \mid x \text{ is defended by } S\}$. A set $S \subseteq A$ is *conflict-free* ($S \in cf(F)$), if there are no $a,b \in S$, such that $(a,b) \in R$. For a conflict-free set $S \in cf(F)$, it holds that $S \in adm(F)$ iff $S \subseteq \mathscr{F}_F(S)$; $S \in com(F)$ iff $S = \mathscr{F}_F(S)$; and $S \in grd(F)$ iff S is the $\subseteq$-least fixed-point of $\mathscr{F}_F$.

For any AF F we have $cf(F) \supseteq adm(F) \supseteq com(F) \supseteq grd(F)$. A σ-extension is an extension under a semantics $\sigma \in \{adm, com, grd\}$.

Example 1. Let $F = (A,R)$ be an AF with $A = \{a,b,c,d,e\}$ and $R = \{(a,b),(b,c),(c,d),(d,c),(d,e),(e,e)\}$. We have $adm(F) = \{\emptyset, \{a\}, \{a,c\}, \{a,d\}, \{d\}\}$, $com(F) = \{\{a\},\{a,c\},\{a,d\}\}$, and $grd(F) = \{\{a\}\}$.

3 Extension Enforcement

We focus on argument-fixed extension enforcement (Coste-Marquis et al. 2015; Wallner, Niskanen, and Järvisalo 2017) under grounded semantics. In *strict* extension enforcement, given an AF $F = (A,R)$, a subset of its arguments $T \subseteq A$, and an AF semantics σ, the goal is to modify R such that T becomes a σ-extension. In *non-strict* extension enforcement, it suffices that T is a subset of a σ-extension. We denote by

$$enf(s,F,T,\sigma) = \{R' \mid F' = (A,R'),\ T \in \sigma(F')\},$$

the set of attack structures that strictly (s) enforce T under σ for an AF F, and for non-strict (ns) enforcement by

$$enf(ns,F,T,\sigma) = \{R' \mid F' = (A,R'),\ \exists T' \in \sigma(F') : T' \supseteq T\}.$$

The Hamming distance between two attack structures is $|R\Delta R'| = |R \setminus R'| + |R' \setminus R|$, i.e., the number of changes between R and R'. We consider extension enforcement as the optimization problem of minimizing the Hamming distance between the original and modified attack structures.

Grounded Extension Enforcement ($M \in \{s, ns\}$)
Input: AF $F = (A,R)$, $T \subseteq A$
Task: Find an AF $F^* = (A,R^*)$ with

$$R^* \in \operatorname*{arg\,min}_{R' \in enf(M,F,T,grd)} |R\Delta R'|.$$

The corresponding decision problem, asking whether there is a solution attack structure R' with $|R\Delta R'| \leq k$ for a given integer k, is NP-complete (Wallner, Niskanen, and Järvisalo 2017). Note that $T = \emptyset$ is trivial for both the strict and non-strict case. Further, for any $R' \in enf(M,F,T,grd)$, if $T \neq \emptyset$, there is an unattacked argument in R' (the grounded extension is empty iff all arguments are attacked).

Listing 1: Module π_{enf}

```
mAtt(X,Y) ← not n_mAtt(X,Y), arg(X), arg(Y).
n_mAtt(X,Y) ← not mAtt(X,Y), arg(X), arg(Y).
← enf(X), not in(X).
← in(X), not enf(X), strict.
 n_root(X) ← arg(X), arg(Y), mAtt(Y,X).
 root(X) ← arg(X), not n_root(X).
 root_exists ← enf(X), root(X), strict.
 root_exists ← arg(X), root(X), not strict.
← not root_exists.
⇝ mAtt(X,Y), not att(X,Y). [1,X,Y]
⇝ not mAtt(X,Y), att(X,Y). [1,X,Y]
```

4 Grounded Enforcement via ASP

We give an ASP encoding for grounded enforcement by extending an encoding for the simpler problem of computing the grounded extension (Egly, Gaggl, and Woltran 2010). Here we only briefly recall the ASP language features we make use of; for more details, see (Gebser et al. 2012). We use programs with weak constraints of the form "$\rightsquigarrow b_1,\dots,b_n.[w,t_1,\dots,t_m]$", with each b_i an atom, each t_j a term, and w an integer; intuitively, if an answer set violates this constraint, then each tuple $(t_1,\dots,t_m)$ contributes cost w. An optimal answer set is one with minimum cost.

To encode enforcement under grounded semantics, we conjoin Egly, Gaggl, and Woltran's (2010) encoding for the grounded extension, π_{ground}, with the encoding π_{enf} given in Listing 1. The encoding computes the grounded extension for an attack structure in predicate **in** which depends now on **mAtt**, the modified attack structure. Encoding π_{enf} guesses a modified attack structure and constrains the answer sets so that the enforced arguments (**enf**) are a subset of **in** (and under **strict** enforcement to coincide with **in**). The next four rules compute **root**s (unattacked arguments) of the modified attack structure, and check whether there is such a root. While these rules are redundant (every grounded extension has an unattacked argument), we observed that solvers benefit from them. Finally, the weak constraints ensure that unit weight is given to each changed attack.

5 Grounded Enforcement via MaxSAT

As an alternative to the ASP approach, we propose a direct MaxSAT encoding for grounded extension enforcement. Recall that a propositional clause is a disjunction of literals, i.e., positive and negative Boolean variables. Satisfaction of a clause by a truth assignment τ is defined in the standard way. An instance $\varphi = (\varphi_h, \varphi_s)$ of the *(partial) MaxSAT* problem consists of a set φ_h of *hard* clauses and a set φ_s of *soft* clauses. An assignment τ that satisfies every clause in φ_h is a *solution* to φ. The *cost* of a solution τ to φ is $\mathrm{C}(\varphi,\tau) = \sum_{c \in \varphi_s}(1 - \tau(c))$, i.e., the number of soft clauses not satisfied by τ. A solution τ is *optimal* if $\mathrm{C}(\varphi,\tau) \leq \mathrm{C}(\varphi,\tau')$ holds for any solution τ' to φ. The MaxSAT problem asks to find an optimal solution to a given φ.

Let $F = (A,R)$ be the AF of an enforcement instance. For each $a,b \in A$, define a Boolean variable $r_{a,b}$ with the interpretation $\tau(r_{a,b}) = 1$ iff the attack (a,b) is included in the attack structure R' of the solution AF $F' = (A,R')$. We minimize the Hamming distance between the original and solution attack structures via the soft clauses $\varphi_s(F) = \bigwedge_{a,b\in A} r'_{a,b}$, where $r'_{a,b}$ is $r_{a,b}$ if $(a,b) \in R$, and else $\neg r_{a,b}$.

The computation of the grounded extension of F' is encoded via hard clauses. The idea is to map each argument a in the grounded extension of F' to a level number $Level(a)$ corresponding to the number of applications of $\mathscr{F}_{F'}$ to $\emptyset$ after which the argument is included, that is, $Level(a) = \min\{n \in \mathbb{Z}_+ \mid a \in \mathscr{F}^n_{F'}(\emptyset)\}$. To encode this, we define Boolean variables l^a_n for each $a \in A$ and $n = 1,\dots,\lceil |A|/2 \rceil$ with the interpretation $\tau(l^a_n) = 1$ iff $Level(a) \leq n$, and impose the following constraints.

$$\forall a \in A\colon\ \varphi^a_1 = \left(l^a_1 \leftrightarrow \bigwedge_{b\in A} \neg r_{b,a}\right)$$

$$\forall a \in A, n \geq 2\colon\ \varphi^a_n = \left(l^a_n \leftrightarrow \bigwedge_{b\in A}\left(r_{b,a} \rightarrow \bigvee_{c\in A}(r_{c,b} \wedge l^c_{n-1})\right)\right)$$

$$\forall a \in A\colon\ \varphi^a_{prop} = \bigwedge_{n=2}^{\lceil |A|/2 \rceil} \left(l^a_{n-1} \rightarrow l^a_n\right)$$

$$\varphi_{first} = \bigvee_{a\in A} l^a_1$$

Here φ^a_1 encodes that for argument a, $Level(a) = 1$ if and only if a is unattacked. Likewise, φ^a_n for $n \geq 2$ encodes that $Level(a) \leq n$ if and only if a is defended by the arguments on levels $1,\dots,n-1$. The formulas φ^a_{prop} and φ_{first} are redundant, but help guiding the MaxSAT solver in practice. In analogy with **root_exists** of the ASP encoding, φ_{first} encodes that every grounded extension contains an unattacked argument.

Let $T \subseteq A$ be a set to be enforced under grounded. For non-strict enforcement, the unit clauses $\bigwedge_{a\in T} l^a_{\lceil |A|/2 \rceil}$ make T a subset of the grounded extension. For strict enforcement, we need in addition the clauses $\bigwedge_{a\notin T} \neg l^a_{\lceil |A|/2 \rceil}$ to make sure that no other arguments are part of the grounded extension. Furthermore, for $a,b \in T$, the unit clause $(\neg r_{a,b})$ encodes the fact that $r_{a,b}$ must be false. For strict enforcement l^a_n is false for all n and for all $a \notin T$; thereby we can replace the bound $\lceil |A|/2 \rceil$ by $|T|$.

6 Grounded Enforcement via CEGAR

Instead of a direct MaxSAT encoding, another way to solve extension enforcement under grounded can be derived based on the fact that the grounded extension is one of the complete (and admissible) extensions. This gives rise to a counterexample-guided abstraction refinement (CEGAR) procedure for grounded extension enforcement. The approach iteratively optimally enforces T under complete semantics using a direct MaxSAT encoding, as an abstraction of grounded enforcement. If T is the grounded extension of the AF resulting from the enforcement, the AF is an optimal solution to grounded extension enforcement. If T is not the grounded extension of the obtained AF, a constraint ruling out the AF is added to the MaxSAT encoding as a refinement to the abstraction, and a MaxSAT solver is again invoked. For obtaining a tighter and non-trivial abstraction refinement and ruling out more non-solutions from further consideration after each MaxSAT solver call, we make use of theoretical observations from (Boella, Kaci, and van der Torre 2009a; 2009b; Rienstra, Sakama, and van der Torre 2015) on what type of changes to an attack structure can and can not influence the grounded extension of an AF.

In more detail, we propose the CEGAR algorithm presented as Algorithm 1, adapting (Wallner, Niskanen, and Järvisalo 2017). For strict enforcement, we choose complete as the semantics for the abstraction, and admissible for non-strict (since complete and admissible coincide for non-strict). In both cases we make use of the same soft clauses φ_s as in the previous section. Strict enforcement of T under complete semantics is encoded via the formula

$$\textsc{ext}(s,F,T,com) = \bigwedge_{a,b\in T} \neg r_{a,b} \wedge \bigwedge_{a\in T}\bigwedge_{b\in A\setminus T} (r_{b,a} \rightarrow \bigvee_{c\in T} r_{c,b}) \wedge \bigwedge_{a\in A\setminus T}\bigvee_{b\in A} (r_{b,a} \wedge \bigwedge_{c\in T} \neg r_{c,b}).$$

For non-strict enforcement under admissible, we also define for each $a \in A$ a Boolean variable x_a with the interpretation $\tau(x_a) = 1$ iff argument a is included in the extension of the solution AF. The hard clauses

$$\textsc{ext}(ns,F,T,adm) = \bigwedge_{a\in T} x_a \wedge \bigwedge_{a,b\in A} \left(r_{a,b} \rightarrow (\neg x_a \vee \neg x_b)\right) \wedge \bigwedge_{a,b\in A} \left((x_a \wedge r_{b,a}) \rightarrow \bigvee_{c\in A} (x_c \wedge r_{c,b})\right)$$

are used to solve non-strict enforcement under admissible.

In addition, we augment the abstraction by including clauses $\textsc{root}(s,F,T) = \bigwedge_{a\in T} \varphi^a_1 \wedge \bigvee_{a\in T} l^a_1$ (analogous to encoding roots in the ASP and MaxSAT encodings) for strict enforcement, stating that T should contain at least one unattacked argument, which together with $\textsc{ext}(s,F,T,com)$ implies that all unattacked arguments will be contained in T. For non-strict, we add clauses $\textsc{root}(ns,F,T) = \bigwedge_{a\in A} \varphi^a_1 \wedge \bigvee_{a\in A} l^a_1 \wedge \bigwedge_{a\in A\setminus T} (l^a_1 \rightarrow x_a)$ expressing that all unattacked arguments should be contained in an admissible superset.

The abstraction is solved in the main loop using MaxSAT, obtaining the attack relation $\textsc{ConstructAF}(\tau) = \{(a,b) \in A \times A \mid \tau(r_{a,b}) = 1\}$. We then check if T is (for strict) or is

Algorithm 1 CEGAR-based extension enforcement under grounded semantics with $M \in \{s, ns\}$.

```
1: if $M = s$ then $\sigma \leftarrow com$ else $\sigma \leftarrow adm$
2: $\varphi_h \leftarrow \textsc{ext}(M,F,T,\sigma) \wedge \textsc{root}(M,F,T)$
3: while true do
4:     $(c,\tau) \leftarrow \textsc{MaxSAT}(\varphi_h, \varphi_s(F))$
5:     $F' \leftarrow (A, \textsc{ConstructAF}(\tau))$
6:     if $M = s$ and $T = grd(F')$ then return $F'$
7:     if $M = ns$ and $T \subseteq grd(F')$ then return $F'$
8:     $\varphi_h \leftarrow \varphi_h \wedge \textsc{refine}(F')$
```

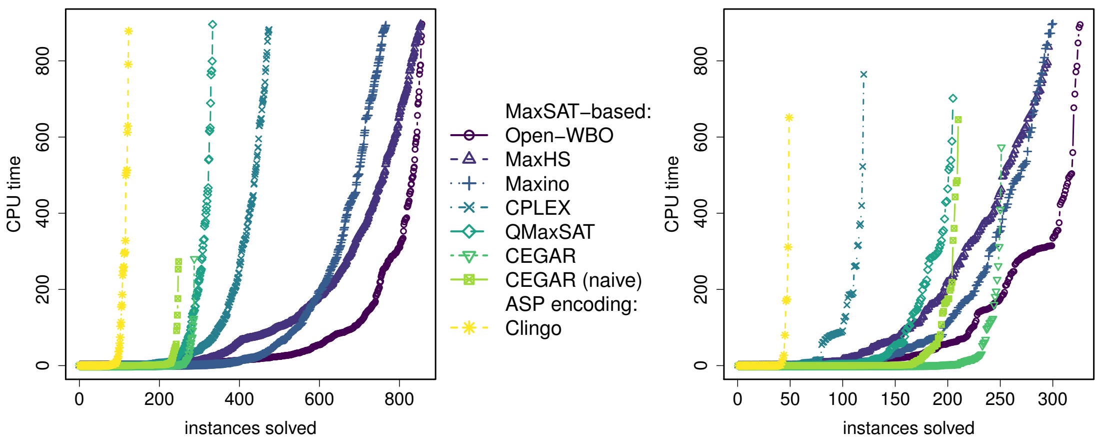

Figure 1: Comparison of approaches for strict (left) and non-strict (right) enforcement under grounded.

included in (for non-strict) the grounded extension of the AF $F' = (A, R')$. If the check fails, we proceed by refining the abstraction. At this point we make use of theoretical observations from (Boella, Kaci, and van der Torre 2009a; 2009b; Rienstra, Sakama, and van der Torre 2015). Specifically, in the labelling-based setting (Baroni, Caminada, and Giacomin 2011), under grounded semantics an argument $a \in in_{grd}(F)$ if $a \in grd(F)$, $a \in out_{grd}(F)$ if there is a $b \in grd(F)$ such that $(b,a) \in R$, and $a \in undec_{grd}(F)$ otherwise. Let

$$\begin{aligned}\textsc{AttAdd}(F) = \{(a,b) \notin R \mid (a,b) \in\ & (in_{grd}(F) \times in_{grd}(F)) \cup \\ & (in_{grd}(F) \times undec_{grd}(F)) \cup \\ & (out_{grd}(F) \times in_{grd}(F)) \cup \\ & (undec_{grd}(F) \times in_{grd}(F))\}\end{aligned}$$

be the attacks that may alter the grounded extension of (A,R) when added to R; e.g., if both a and b are part of the grounded extension, adding attack (a,b) makes this set conflicting. Further, let $\textsc{AttRem}(F) = \{(a,b) \in R \mid (a,b) \in (in_{grd}(F) \times out_{grd}(F)) \cup (undec_{grd}(F) \times undec_{grd}(F))\}$ be the attacks that can change the grounded extension of (A,R) when removed from R. This yields a non-trivial abstraction refinement $\textsc{refine}(F) = \bigvee_{(a,b)\in \textsc{AttAdd}(F)} r_{a,b} \vee \bigvee_{(a,b)\in \textsc{AttRem}(F)} \neg r_{a,b}$, stating that in the subsequent iterations, we should add (respectively remove) an attack not present (respectively present) in the current solution in such a way that we obtain a different grounded extension.

7 Experiments

We evaluate the three approaches to grounded extension enforcement using state-of-the-art solvers: Clingo v5.2.2 (Gebser et al. 2016) for ASP, and, for the direct MaxSAT encoding, MaxHS (Davies and Bacchus 2013), Maxino (Alviano, Dodaro, and Ricca 2015), Open-WBO (Martins, Manquinho, and Lynce 2014), and QMaxSAT (Koshimura et al. 2012), using the MaxSAT Evaluation 2017 versions, as well as the integer programming (IP) solver CPLEX v12.7 using a standard translation from MaxSAT to IP (Ansótegui and Gabàs 2013; Saikko, Berg, and Järvisalo 2016). We used Open-WBO as the underlying MaxSAT solver in the CEGAR approach.

We generated benchmarks from instances from the 2017 ICCMA competition (Gaggl et al. 2016) (sets A, B, and C) as follows. For each AF with at most 300 arguments, we computed the grounded extension of the AF. Let $p_{\text{flip}} = 0.05$. Then, for each argument a, if a is in the grounded extension, we include a in T with probability $1 - p_{\text{flip}}$, and if not, we include it with probability p_{flip}. If the resulting set is empty, we rerun the procedure. Intuitively, this creates a non-empty 'distorted' version of the original grounded extension. We repeated this procedure 5 times for each AF, resulting in a total of 1340 enforcement instances.

A summary of the results is presented in Figure 1, showing the number of solved instances (x-axis) under a per-instance time limit (y-axis) up to 900 s. Overall, we observe that non-strict enforcement is expectedly harder to solve due to the less constrained search space. The MaxSAT and CEGAR approaches clearly outperform the ASP approach, potentially partially due to the size of the ASP programs after grounding. All MaxSAT solvers perform relatively well. Among all the approaches Open-WBO—the best-performing solver on unweighted instances in the 2017 MaxSAT Evaluation—on the direct MaxSAT encoding is the dominant approach, also when compared to the commercial CPLEX solver. CEGAR is competitive with MaxSAT on non-strict, on which we also see a clear benefit from the non-trivial abstraction refinement compared to a naive refinement in which only the exact AF structure found is ruled out from further consideration. While CEGAR solved less instances than the best MaxSAT solver, CEGAR tends to be fastest on instances which it solves. Further, CEGAR was able to solve some non-strict instances with 250 arguments, while the MaxSAT and ASP approaches solved only instances with less than 150 and 50 arguments, respectively.

8 Conclusions

We proposed the first declarative approaches to the NP-hard extension enforcement problem under grounded semantics. Each of the approaches also allows for integrating structural constraints e.g. on which particular attacks and non-attacks may be subjected to change. The alternative MaxSAT and

CEGAR approaches show noticeably better scaling than the ASP encoding based on an earlier ASP encoding for computing the grounded extension. The approaches may be of interest for potential further optimization problems related to argumentation dynamics which turn out to be NP-hard under the grounded semantics.

Acknowledgements

Work supported in part by Academy of Finland (grants 276412 and 312662), DoCS Doctoral Programme in Computer Science and Research Funds of the University of Helsinki, and Austrian Science Fund (FWF): P30168-N31.

References

Alviano, M.; Dodaro, C.; and Ricca, F. 2015. A MaxSAT algorithm using cardinality constraints of bounded size. In *Proc. IJCAI*, 2677–2683. AAAI Press.

Ansótegui, C., and Gabàs, J. 2013. Solving (weighted) partial MaxSAT with ILP. In *Proc. CPAIOR*, volume 7874 of *LNCS*, 403–409. Springer.

Baroni, P.; Caminada, M.; and Giacomin, M. 2011. An introduction to argumentation semantics. *Knowl. Eng. Rev.* 26(4):365–410.

Baumann, R., and Brewka, G. 2010. Expanding argumentation frameworks: Enforcing and monotonicity results. In *Proc. COMMA*, volume 216 of *FAIA*, 75–86. IOS Press.

Baumann, R., and Brewka, G. 2015. AGM meets abstract argumentation: Expansion and revision for Dung frameworks. In *Proc. IJCAI*, 2734–2740. AAAI Press.

Baumann, R. 2012. What does it take to enforce an argument? Minimal change in abstract argumentation. In *Proc. ECAI*, volume 242 of *FAIA*, 127–132. IOS Press.

Boella, G.; Kaci, S.; and van der Torre, L. W. N. 2009a. Dynamics in argumentation with single extensions: Abstraction principles and the grounded extension. In *Proc. ECSQARU*, volume 5590 of *LNCS*, 107–118. Springer.

Boella, G.; Kaci, S.; and van der Torre, L. W. N. 2009b. Dynamics in argumentation with single extensions: Attack refinement and the grounded extension (extended version). In *ArgMAS Revised Selected and Invited Papers*, volume 6057 of *LNCS*, 150–159. Springer.

Cayrol, C.; de Saint-Cyr, F. D.; and Lagasquie-Schiex, M. 2010. Change in abstract argumentation frameworks: Adding an argument. *J. Artif. Intell. Res.* 38:49–84.

Clarke, E. M.; Grumberg, O.; Jha, S.; Lu, Y.; and Veith, H. 2003. Counterexample-guided abstraction refinement for symbolic model checking. *J. ACM* 50(5):752–794.

Clarke, E. M.; Gupta, A.; and Strichman, O. 2004. SAT-based counterexample-guided abstraction refinement. *IEEE T-CAD* 23(7):1113–1123.

Coste-Marquis, S.; Konieczny, S.; Mailly, J.; and Marquis, P. 2015. Extension enforcement in abstract argumentation as an optimization problem. In *Proc. IJCAI*, 2876–2882. AAAI Press.

Davies, J., and Bacchus, F. 2013. Exploiting the power of MIP solvers in MaxSAT. In *Proc. SAT*, volume 7962 of *LNCS*, 166–181. Springer.

de Saint-Cyr, F. D.; Bisquert, P.; Cayrol, C.; and Lagasquie-Schiex, M. 2016. Argumentation update in YALLA (Yet Another Logic Language for Argumentation). *Int. J. Approx. Reason.* 75:57–92.

Diller, M.; Haret, A.; Linsbichler, T.; Rümmele, S.; and Woltran, S. 2015. An extension-based approach to belief revision in abstract argumentation. In *Proc. IJCAI*, 2926–2932. AAAI Press.

Dimopoulos, Y., and Torres, A. 1996. Graph theoretical structures in logic programs and default theories. *Theoret. Comput. Sci.* 170(1-2):209–244.

Dung, P. M. 1995. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artif. Intell.* 77(2):321–358.

Dunne, P. E., and Bench-Capon, T. J. M. 2002. Coherence in finite argument systems. *Artif. Intell.* 141(1/2):187–203.

Dunne, P. E., and Wooldridge, M. 2009. Complexity of abstract argumentation. In *Argumentation in Artificial Intelligence*. Springer. 85–104.

Egly, U.; Gaggl, S.; and Woltran, S. 2010. Answer-set programming encodings for argumentation frameworks. *Argument & Computation* 1(2):147–177.

Gaggl, S. A.; Linsbichler, T.; Maratea, M.; and Woltran, S. 2016. Introducing the second international competition on computational models of argumentation. In *Proc. SAFA*, 4–9. CEUR-WS.org.

Gebser, M.; Kaminski, R.; Kaufmann, B.; and Schaub, T. 2012. *Answer Set Solving in Practice*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers.

Gebser, M.; Kaminski, R.; Kaufmann, B.; Ostrowski, M.; Schaub, T.; and Wanko, P. 2016. Theory solving made easy with Clingo 5. In *ICLP Tech. Commun.*, 2:1–2:15.

Koshimura, M.; Zhang, T.; Fujita, H.; and Hasegawa, R. 2012. QMaxSAT: A partial Max-SAT solver. *J. SAT* 8(1/2):95–100.

Martins, R.; Manquinho, V. M.; and Lynce, I. 2014. Open-WBO: A modular MaxSAT solver,. In *Proc. SAT*, volume 8561 of *LNCS*, 438–445. Springer.

Niskanen, A.; Wallner, J. P.; and Järvisalo, M. 2016. Pakota: A system for enforcement in abstract argumentation. In *Proc. JELIA*, volume 10021 of *LNCS*, 385–400. Springer.

Rienstra, T.; Sakama, C.; and van der Torre, L. W. N. 2015. Persistence and monotony properties of argumentation semantics. In *TAFA Revised Selected Papers*, volume 9524 of *LNCS*, 211–225. Springer.

Saikko, P.; Berg, J.; and Järvisalo, M. 2016. LMHS: A SAT-IP hybrid MaxSAT solver. In *Proc. SAT*, volume 9710 of *LNCS*, 539–546. Springer.

Wallner, J. P.; Niskanen, A.; and Järvisalo, M. 2017. Complexity results and algorithms for extension enforcement in abstract argumentation. *J. Artif. Intell. Res.* 60:1–40.