Complexity Results and Algorithms for Extension Enforcement in Abstract Argumentation*

Johannes P. Wallner and Andreas Niskanen and Matti Järvisalo
Helsinki Institute for Information Technology HIIT, Department of Computer Science, University of Helsinki, Finland

Abstract
Understanding the dynamics of argumentation frameworks (AFs) important in the study of argumentation in AI. In this work, we focus on the so-called extension enforcement problem in abstract argumentation. We provide a nearly complete computational complexity map of fixed-argument extension enforcement under various major AF semantics, with results ranging from polynomial-time algorithms to completeness for the second-level of the polynomial hierarchy. Complementing the complexity results, we propose algorithms for NP-hard extension enforcement based on constrained optimization. Going beyond NP, we propose novel counterexample-guided abstraction refinement procedures for the second-level complete problems and present empirical results on a prototype system constituting the first approach to extension enforcement in its generality.

Introduction
Argumentation is a core topic in Artificial Intelligence (AI) (Bench-Capon and Dunne 2007), with applications in e.g. decision support (Amgoud and Prade 2009), legal reasoning (Bench-Capon, Prakken, and Sartor 2009), and multi-agent systems (McBurney, Parsons, and Rahwan 2012). Argumentation frameworks (AFs) (Dung 1995) provide the fundamental formal model for many approaches to argumentation in AI. Syntactically, AFs are directed graphs, where arguments are abstract entities represented by vertices. Conflicts among arguments are formalized as attacks, and represented with directed edges between arguments. Semantics of AFs—several of which have been proposed—specify criteria for arguments’ acceptance resulting in sets of jointly acceptable arguments called extensions.

Argumentation is inherently a dynamic process. Recently, several works have focused on fundamental aspects of argumentation dynamics (Baumann 2012a; Baumann and Brewka 2015; Bisquert et al. 2013; Coste-Marquis et al. 2014a; 2014b; Delobelle, Konieczny, and Vesic 2015; Diller et al. 2015). In this work, we focus on extension enforcement (Baumann 2012b; Bisquert et al. 2013; Coste-Marquis et al. 2015), a specific form of AF dynamics with connections to belief revision, concerned with finding changes to a given AF in order to support a desired point of view, represented as a set of arguments, under pre-specified semantics.

While the complexity landscape of non-dynamic problems on AFs, including the credulous and skeptical reasoning tasks for a given fixed AF, is already well-established (Dunne and Wooldridge 2009), the complexity of extension enforcement under different semantics and problem variants has not been thoroughly studied until now. Furthermore, while several efficient systems for the NP-hard variants of non-dynamic problems are available (Cerutti et al. 2014; Cerutti, Giacomin, and Vallati 2014; Dvořák et al. 2014; Egly, Gaggl, and Woltran 2010; Nofal, Atkinson, and Dunne 2014), to our best knowledge the single existing system for extension enforcement was only recently proposed (Coste-Marquis et al. 2015), and currently supports extension enforcement only w.r.t. specific AF semantics (the stable semantics). This paper aims at bridging these gaps.

Our main contributions are the following.

- We provide a nearly complete computational complexity map of fixed-argument extension enforcement, where the task is to enforce a given extension by modifying the attack relation of a given AF. Our results cover nine standard AF semantics and both the so-called strict and non-strict variants of extension enforcement. For examples, we provide polynomial-time algorithms for strict enforcement under the admissible and stable semantics (the latter of which was in fact proposed to be solved using the NP-machinery of integer programming (IP) by Coste-Marquis et al. (2015)); show that most non-strict enforcement problems are NP-complete, along with strict enforcement under the complete and grounded semantics; and establish second-level completeness for strict enforcement under preferred and semi-stable semantics as well as for non-strict semi-stable and stage semantics.

- We propose algorithms for the NP-hard variants of the enforcement problems based on applying constraint optimization solvers. We detail maximum satisifiability (MaxSAT) encodings for the NP-complete problem variants, and, perhaps most interestingly, propose novel counterexample-guided abstraction refinement (CEGAR) (Clarke et al. 2003; Clarke, Gupta, and Strich-
man (2004) procedures for the second-level $\Sigma^P_R$-complete variants using optimization solvers as functional NP oracles. We provide an overview of an empirical evaluation of a prototype system implementation that supports the considered extension enforcement variants.

While our main focus is fixed-argument extension enforcement, we also shortly discuss the normal, strong, and weak variants (Baumann 2012b) of enforcement.

### Preliminaries

We recall concepts related to argumentation frameworks (Dung 1995), their semantics (Baroni, Caminada, and Giacomin 2011), and enforcement operators (Baumann 2012b; Coste-Marquis et al. 2015).

**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 $b \in A$ such that $(b, a) \in R$, there exists a $c \in S$ such that $(c, b) \in R$.

**Example 1.** Let $F = (A, R)$ be an AF with $A = \{a, b, c, d\}$ and $R = \{(b, a), (b, c), (c, a), (c, d), (d, b)\}$. The corresponding graph representation is shown in Figure 1.

Semantics for argumentation frameworks are defined through a function $\sigma$ which assigns to each AF $F = (A, R)$ a set $\sigma(F) \subseteq 2^A$ of extensions. We consider for $\sigma$ the functions naive, stb, adm, com, grd, prf, sem, and stg which stand for naive, stable, admissible, complete, grounded, preferred, semi-stable, and stage extensions, respectively. These semantics are defined as follows.

**Definition 2.** Given an AF $F = (A, R)$, the characteristic function $F_F : 2^A \rightarrow 2^A$ of $F$ is defined as $F_F(S) = \{x \in A \mid x \text{ is defended by } S\}$. Moreover, for a set $S \subseteq A$, we define the range of $S$ as $S^+_R = S \cup \{y \mid (y, x) \in R, y \in S\}$.

**Definition 3.** Let $F = (A, R)$ be an AF. A set $S \subseteq A$ is conflict-free (in $F$), if there are no $a, b \in S$, such that $(a, b) \in R$. We denote the collection of conflict-free sets of $F$ by $\text{cf}(F)$. For a conflict-free set $S \in \text{cf}(F)$, it holds that

- $S \in \text{naive}(F)$ if there is no $T \in \text{cf}(F)$ with $S \subseteq T$;
- $S \in \text{stb}(F)$ if $S^+_R = A$;
- $S \in \text{adm}(F)$ if $S \subseteq F_F(S)$;
- $S \in \text{com}(F)$ if $S = F_F(S)$;
- $S \in \text{grd}(F)$ if $S$ is the least fixed-point of $F_F$;
- $S \in \text{prf}(F)$ if $S \in \text{adm}(F)$ and there is no $T \in \text{adm}(F)$ with $S \subset T$;
- $S \in \text{sem}(F)$ if $S \in \text{adm}(F)$ and there is no $T \in \text{adm}(F)$ with $S^+_R \subset T^+_R$;

![Figure 1: Example argumentation framework](image)

**Extension enforcement** $(x \in \{s, ns\})$

**Input:** AF $F = (A, R)$, $T \subseteq A$, and semantics $\sigma$.

**Task:** Find an AF $F^* = (A, R^*)$ with $R^* \in \text{arg min}_{R' \in \text{enf}_s(F, T)} |R \Delta R'|$.

**Example 2.** In AF $F$ from Example 1 we have $\text{com}(F) = \{\emptyset\}$. A way to strictly enforce $\{a\}$ as a complete extension is to remove attacks $(b, a)$ and $(c, a)$ (Figure 2a). Adding $(d, c)$ makes $\{a, d\}$ a complete extension, and thus $\{a\}$ becomes non-strictly enforced (Figure 2b).

In the decision problems for extension enforcement, we are given an AF $F = (A, R)$, a set $T \subseteq A$, and an integer $k \geq 0$, and are asked to decide if there is an $F' = (A, R')$ with $|R \Delta R'| \leq k$ that enforces $T$ non-strictly (resp. strictly). For the complexity results, in addition to the standard complexity classes P, NP, and coNP, recall that the class $\Sigma^P_R$ consists of problems which can be decided by a non-deterministic polynomial-time algorithm with access to an NP oracle.
Complexity Analysis

An overview of the complexity results of this paper is given in Table 1. We begin our analysis by considering non-strict enforcement. A basic observation is that, to enforce a set $T$ under semantics $\sigma$, all attacks “inside” $T$ need to be removed, since all considered semantics are based on conflict-free sets. For non-strict enforcement under conflict-free and naive semantics, this modification turns out to be optimal.

**Proposition 1.** Non-strict enforcement for conflict-free and naive semantics is in $P$.

**Proof.** (sketch) Let $F = (A, R)$ be an AF and $T \subseteq A$ the set to be enforced. Define $F^* = (A, R^*)$ with $R^* = R \setminus (T \times T)$. Now $T \in \text{cf}(F^*)$ and thus there is a $T' \in \text{naive}(F^*)$ with $T' \subseteq T$. For any $R' \subseteq A \times A$ with $|R \Delta R'| < |R \Delta R^*|$ it holds that $T$ and all supersets of $T$ are not conflict-free in $F' = (A, R')$. Thus $F^*$ is an optimal solution.

For the remaining semantics, non-strict enforcement is presumably harder. This follows from the fact that it is computationally hard to check whether there is a superset of $T$ that is a $\sigma$-extension of the input AF $F$.

**Proposition 2.** Non-strict enforcement

- for admissible, complete, preferred, and stable semantics is $\Sigma_2^P$-complete; and
- for semi-stable and stage semantics is $\Sigma_2^P$-complete.

**Proof.** Hardness in all cases follows from a reduction from the credulous acceptance problem for the same semantics $\sigma$, where we have to decide whether an $a$ is contained in one $\sigma$-extension of a given AF $F$. We reduce this problem to non-strict enforcement by defining $T = \{a\}$. Then $T$ can be non-strictly enforced under $\sigma$ with 0 changes iff $a$ is credulously accepted. Complexity of credulous reasoning is analyzed in (Caminada, Carnielli, and Dunne 2012; Coste-Marquis, Devred, and Marquis 2005; Dimopoulos and Torres 1996; Dung 1995; Dvořák and Woltran 2010). Membership for all problems follows from a guess and check (verifying if a given set is admissible or stable can be checked in $P$; for semi-stable and stage this problem is in coNP).

Coste-Marquis et al. (2015) established that the union of non-strict and strict enforcement under stable semantics is $NP$-hard. As a more fine-grained analysis, by Proposition 2 non-strict enforcement is in itself $NP$-complete; furthermore, in the following we will show that strict enforcement under stable semantics is in fact in $P$.

From the previous propositions it might appear that the main source of intractability does not originate from the modifications of the attack structure, but from (credulous) acceptance problems associated with the semantics under consideration. However, even for the computationally simple grounded semantics, non-strict enforcement turns out to be $NP$-complete. This suggests that for admissibility-based semantics the non-determinism introduced by changes in the attack structure is enough for $NP$-hardness.

**Theorem 3.** Non-strict enforcement for grounded semantics is $NP$-complete.

We move on to strict enforcement. Here we establish polynomial-time results for stable and admissible semantics.

**Proposition 4.** Strict enforcement for conflict-free, naive, admissible, and stable semantics is in $P$.

**Proof.** (sketch) Let $F = (A, R)$ be an AF, $T \subseteq A$, and $\sigma \in \{\text{cf, naive, adm, stb}\}$. For each $\sigma$, we define a polynomial-time computable $F^*_\sigma = (A, R^*_\sigma)$ that is an optimal solution to the strict enforcement problem under $\sigma$. We assume that $T \neq \emptyset$; otherwise the problem is trivial. Let $t_0 \in T$ be an arbitrary but fixed argument. For all considered semantics we have to remove conflicts inside $T$.

- $\sigma = \text{cf}$: let $R^*_\text{cf} = R \setminus (T \times T)$;
- $\sigma = \text{naive}$: add a self-attack to arguments $a \in A \setminus T$ where $T \cup \{a\}$ would be conflict-free otherwise, i.e. $R^*_\text{naive} = (R \setminus (T \times T)) \cup \{(a, a) \mid a \in A \setminus T, \exists (b, a) \in R \text{ with } b \in T \cup \{a\}\}$;
- $\sigma = \text{stb}$: add attacks $(t_0, a)$ with $a \in A \setminus T$ where $a$ is not attacked by $T$, i.e. $R^*_\text{stb} = (R \setminus (T \times T)) \cup \{(t_0, a) \mid a \in A \setminus T, \exists (t, a) \in R \text{ with } t \in T\}$;
- $\sigma = \text{adm}$: for each attack from $a \in A \setminus T$ to $t \in T$ that is not counterattacked by $T$ add $(t_0, a)$, i.e. $R^*_\text{adm} = (R \setminus (T \times T)) \cup \{(t_0, a) \mid a \in A \setminus T, \exists (a, t) \in R \text{ s.t. } t \in T \setminus \{t\}' \in T\}$.

In contrast to admissible semantics, strict enforcement for complete and grounded semantics is $NP$-complete. Intuitively, admissibility together with the fact that we must not defend arguments outside any desired set can be used for reducing satisfiability of Boolean formulas to strict enforcement under complete or grounded semantics.

**Theorem 5.** Strict enforcement for complete and grounded semantics is $NP$-complete.

For preferred and semi-stable semantics, we see a jump in complexity: the corresponding problems are in fact $\Sigma_2^P$-complete. Intuitively, in addition to the source of intractability that strict enforcement under complete semantics brings, one has to take into account that modifications to the attack structure might give rise to supersets of $T$ that are admissible. Hardness can be proven by a reduction from satisfiability of quantified Boolean formulas.

**Theorem 6.** Strict enforcement for preferred and semi-stable semantics is $\Sigma_2^P$-complete.

<table>
<thead>
<tr>
<th>$\sigma$</th>
<th>strict</th>
<th>non-strict</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conflict-free</td>
<td>in $P$</td>
<td>in $P$</td>
</tr>
<tr>
<td>Naive</td>
<td>in $P$</td>
<td>in $P$</td>
</tr>
<tr>
<td>Admissible</td>
<td>in $P$</td>
<td>NP-c</td>
</tr>
<tr>
<td>Stable</td>
<td>in $P$</td>
<td>NP-c</td>
</tr>
<tr>
<td>Complete</td>
<td>NP-c</td>
<td>NP-c</td>
</tr>
<tr>
<td>Grounded</td>
<td>NP-c</td>
<td>NP-c</td>
</tr>
<tr>
<td>Preferred</td>
<td>$\Sigma_2^P$</td>
<td>NP-c</td>
</tr>
<tr>
<td>Semi-stable</td>
<td>$\Sigma_2^P$</td>
<td>$\Sigma_2^P$</td>
</tr>
<tr>
<td>Stage</td>
<td>coNP hard</td>
<td>in $\Sigma_2^P$</td>
</tr>
</tbody>
</table>

Table 1: Complexity results for extension enforcement
Figure 3: Strict enforcement under preferred semantics

As an illustrative example of when complete and preferred semantics differ w.r.t. strict enforcement, see Figure 3. Here strictly enforcing \( \{ t \} \) under complete semantics requires no modifications. In contrast, under preferred semantics each other argument requires one distinct modification.

Finally, for stage semantics we give straightforward bounds. Hardness follows from coNP-hardness of verifying if a set is a stage extension (Dvořáková and Woltran 2010).

**Corollary 7.** Strict enforcement for stage semantics is in \( \Sigma_2^P \) and coNP hard.

We conjecture that strict enforcement for stage semantics is indeed \( \Sigma_2^P \)-complete; this is the only missing piece in the complexity map (recall Table 1) established in this paper.

**Extension Enforcement via MaxSAT**

In this section we present declarative encodings that can be used for solving extension enforcement optimally under the considered semantics. We employ maximum satisfiability (MaxSAT) as a well-suited declarative language. For non-strict enforcement under stable semantics, our encoding is essentially the same as the integer programming formulation presented by Coste-Marquis et al. (2015). Here we present MaxSAT encodings for the various semantics, as well as develop counterexample-guided abstraction refinement algorithms for solving the \( \Sigma_2^P \)-complete problem variables by applying the NP-encodings.

We recall the MaxSAT problem. For a variable \( x \), there are two literals, \( x \) and \( \neg x \). A clause is a disjunction (\( \lor \)) of literals. A truth assignment is a function from variables to \{0, 1\}. A clause \( c \) is satisfied by a truth assignment \( \tau \) if \( \tau(c) = 1 \) or \( \tau(x) = 0 \) for a literal \( x \) in \( c \), otherwise \( \tau \) does not satisfy \( c \) (\( \tau(c) = 0 \)).

An instance \( \varphi = (\varphi_h, \varphi_s) \) of the Partial MaxSAT problem consists of a set \( \varphi_h \) of hard clauses and a set \( \varphi_s \) of soft clauses. Any truth assignment \( \tau \) that satisfies every clause in \( \varphi_h \) is a solution to \( \varphi \). The cost of a solution \( \tau \) to \( \varphi \) is \( \text{cost}(\varphi, \tau) = \sum_{c \in \varphi_h} (1 - \tau(c)) \), i.e., the number of soft clauses not satisfied by \( \tau \). A solution \( \tau \) is optimal for \( \varphi \) if \( \text{cost}(\varphi, \tau) \leq \text{cost}(\varphi, \tau') \) holds for any solution \( \tau' \) to \( \varphi \). Given \( \varphi \), the Partial MaxSAT problem asks to find an optimal solution to \( \varphi \). From here on, we refer to partial MaxSAT simply as MaxSAT.

We now present MaxSAT encodings for NP extension enforcement problems. Let \( F = (A, R) \) be an AF and \( T \subseteq A \) the set to be enforced under semantics \( \sigma \). We use variables \( x_a \) and \( r_{a,b} \) for \( a, b \in A \) with the interpretation "\( x_a = 1 \) iff \( a \) is in an extension", and "\( r_{a,b} = 1 \) iff the attack \( (a, b) \) occurs in the modified AF".

For all semantics, the soft clauses are given by

\[
\varphi_{\text{soft}}(F) = \bigwedge_{a,b \in A} r'_{a,b}, \quad \text{where}
\] \[
r'_{a,b} \leftrightarrow \begin{cases} r_{a,b} & \text{if } (a, b) \in R \\ \neg r_{a,b} & \text{if } (a, b) \notin R, \end{cases}
\]

i.e., a violated soft clause, contributing unit cost to the cost of a solution, implies that the corresponding attack has been modified (removed or added).

We now define clauses enforcing that the given set \( T \) must be part of a (\( \sigma \))-extension for the modified AF encoded via the attack variables \( r_{a,b} \). For non-strict enforcement (short-hand \( ns \)), we define \( \varphi_{ns}(F, T) = \bigwedge_{a \in T} x_a \) which encodes that the given set must be part a \( \sigma \)-extension. For strict enforcement (short-hand \( s \)), there is no need to encode arguments’ statuses as variables (their values are fixed), i.e., variables \( x_a \) are not required. For encoding the semantics, we adapt Boolean formulas from (Besnard and Doutre 2004), originally for non-dynamic problems, to extension enforcement. We start with non-strict enforcement. For conflict-free sets, if an attack between two arguments is present, then only one of them can be in a conflict-free set.

\[
\varphi_{ns}^c F, T, \varphi = \varphi_{ns}(F, T) \land \bigwedge_{a,b \in A} (r_{a,b} \rightarrow (\neg x_a \lor \neg x_b))
\]

For admissible semantics (recall that non-strict enforcement for admissible, complete, and preferred semantics coincides), if \( a \) is in an admissible set and there is an attack on \( a \), then a defender together with a defending attack must be assigned to 1.

\[
\varphi_{ad} F, T, \varphi = \varphi_{ns}(F, T) \land \bigwedge_{a,b \in A} (x_a \land \neg r_{b,a}) \rightarrow (x_b \lor \neg r_{c,b})
\]

Stable semantics can be encoded in the following way. If an argument is not in the stable extension, an attacker in the set together with an attack in the new AF has to be found.

\[
\varphi_{st} F, T, \varphi = \varphi_{ns}(F, T) \land \bigwedge_{a \in A} (\neg x_a \rightarrow \bigvee_{b \in A} (x_b \land r_{b,a}))
\]

We move on to strict enforcement for conflict-free sets.

\[
\varphi_s^c F, T, \varphi = \bigwedge_{a,b \in T} \neg r_{a,b},
\]

i.e., strict enforcement simply consists of removing all attacks inside \( T \). For admissible semantics, we need a defending counter-attack for each attack on set \( T \).

\[
\varphi_{ad} F, T, \varphi = \varphi_{ns}(F, T) \land \bigwedge_{a \in T} (r_{b,a} \rightarrow \bigvee_{c \in T} (r_{c,b})
\]

In the encoding for strict enforcement under complete semantics, we need to ensure that for each argument outside \( T \) there is an attack on it that is not defended against by \( T \).

\[
\varphi_{com} F, T, \varphi = \varphi_{ad}(F, T) \land \bigwedge_{a \in A \setminus T} (r_{b,a} \land \bigwedge_{c \in T} (r_{c,b})
\]

In summary, for semantics \( \sigma \), an optimal solution to the MaxSAT problem \( \varphi = (\varphi_s^c F, T, \varphi_{\text{soft}}) \) corresponds to an optimum solution to the strict enforcement problem (if \( x = s \)) or the non-strict enforcement problem (if \( x = ns \)).
Extension Enforcement Beyond NP

The second-level complexity of strict enforcement under preferred and semi-stable semantics, as well as non-strict enforcement under preferred, semi-stable, and stage semantics, hinders direct use of NP optimization solvers for these problems. However, the NP semantics, such as complete, overapproximate the preferred and semi-stable semantics, and conflict-free sets overapproximate stage semantics. This implies that the NP encodings can be used as base abstractions within a counterexample-guided abstraction refinement (CEGAR) approach to solving the second-level extension enforcement problems. As a general outline, in CEGAR an (over)abstraction of the set of solutions of interest is iteratively refined until an actual solution to the original problem instance is encountered. At each iteration, the abstraction is solved, typically using an NP oracle (such as a SAT solver). A thus obtained candidate solution is checked with another oracle call. If the oracle reports that the candidate is not an actual solution, a counterexample is obtained, and the abstraction is refined further based on the counterexample. This is repeated until no counterexamples are found, at which point the candidate solution is an actual solution.

Let \( F = (A, R) \) be an AF and \( T \subseteq A \) the set to be enforced under semantics \( \sigma \in \{ \text{prf}, \text{sem}, \text{stg} \} \). Let \( x \in \{ ns, s \} \) be the type of enforcement. Our procedures are presented in a unifying way as Algorithm 1. First, we select the “base” semantics \( \chi \) for enforcement that acts as our first abstraction: conflict-free sets for stage and admissible, complete semantics otherwise. In the loop an optimal solution for non-strict or strict enforcement under semantics \( \chi \) is computed by e.g. a MaxSAT or an IP solver, and represented by the truth assignment \( \tau \). We extract the AF \( F' = (A, R') \), with \( R' = \{(a, b) \mid a, b \in A, \tau(r_{a,b}) = 1 \} \) from \( \tau \). Then we check whether \( F' \) is also a solution to enforcement under semantics \( \sigma \). For strict enforcement, we have to check whether \( T \in \sigma(F') \) holds. For non-strict enforcement, we check whether \( T' \in \sigma(F') \) with \( T' = \{ a \in A \mid \tau(x_a) = 1 \} \). We encode the base semantics as in (Besnard and Doutre 2004). Conflict-free sets are encoded by \( \psi^c(F') = \land_{a \in A} (x_a \lor \neg x_a) \), admissible semantics by \( \psi^ad(F') = \land_{(a,b) \in R'} (x_a \lor \neg x_b) \), and complete semantics by \( \psi^{com}(F') = \land_{(b,a) \in R'} (x_a \lor \neg x_b) \).

Algorithm 1 Enforcement for \( \sigma \in \{ \text{prf}, \text{sem}, \text{stg} \} \) with \( x = s \) or \( \sigma \in \{ \text{sem}, \text{stg} \} \) with \( x = ns \).

```
1: if \( \sigma \in \{ \text{prf}, \text{sem} \} \) then \( \chi \leftarrow \text{com else } \chi \leftarrow \text{cf} \)
2: \( \psi \leftarrow \varphi^\chi(F, T) \)
3: if \( x = ns \) then \( \psi \leftarrow \psi \land \delta(A) \)
4: while true do
5: (c, \( \tau \)) \leftarrow \text{MAXSAT}(\psi, \varphi^\text{soft})
6: result \leftarrow \text{SAT}(\Gamma^\sigma(\tau))
7: if result = unsatisfiable then
8: return (c, \( \tau \))
9: else
10: \( \psi \leftarrow \psi \land \text{REFINE}(\tau, x) \)
```

In words, we search for a superset of \( T \) that is a \( \chi \)-extension. If this formula is satisfiable, then there is a counterexample witnessing that \( T \) is not a \( \sigma \)-extension in \( F' \). For the refinement step, we define the shorthand \( \gamma(\tau) = \land_{(a,b) \in R'} r_{a,b} \land \land_{(a,b) \subseteq A} \neg r_{a,b} \). We refine the current abstraction \( \psi \) by \( \text{REFINE}(\tau, s) = \neg \gamma(\tau) \) in the strict case which rules out \( R' \), and in the non-strict case we additionally rule out all \( \chi \)-extensions which is a subset of the range of \( T' \) by \( \text{REFINE}(\tau, ns) = \neg \gamma(\tau) \lor \lor_{a \in A} \lor_{a \in R'} x_a^+ a \). In this case, we define range variables dependent on the attack variables via the shorthand \( \delta(A) = \land_{a \in A} (x_a^+ \lor (x_a \lor \lor_{a \in R}(r_{a,b} \land x_b))) \) (Line 3).

Algorithm 1 solves strict enforcement for \( \{ \text{prf}, \text{sem}, \text{stg} \} \) and non-strict enforcement for \( \{ \text{sem}, \text{stg} \} \) optimally, as at each iteration the current abstraction is solved optimally.

**Experiments**

We present empirical results on a prototype system implementation (available at http://cs.helsinki.fi/group/coreo/pakota/) for extension enforcement, which supports the considered NP-hard strict and non-strict extension enforcement variants, and allows for using both MaxSAT solvers as well as IP solvers via standard translation of MaxSAT into IP (Ansoyegui and Gabás 2013).

We generated enforcement instances as follows. Let \( \lvert A \rvert = 25, 50, \ldots \) denote the number of arguments in the AF to be generated, and \( \lvert T \rvert \) the size of subset \( T \subseteq A \) of arguments to be enforced. For a fixed edge probability \( p \), we sampled directed graphs by independently picking an edge to the AF with probability \( p \) (but disallowing self-attacks). For each \( \lvert A \rvert \) and \( p \in \{ 0.05, 0.1, 0.2, 0.3 \} \), we sampled five directed graphs. For each AF, we picked randomly at random five sets of arguments \( T \subseteq A \) to be enforced for each \( \lvert T \rvert / \lvert A \rvert \in \{ 0.05, 0.1, 0.2, 0.3 \} \). For each number of arguments \( \lvert A \rvert \), this gave 400 enforcement problem instances.

We used the OpenWBO (Martins, Manquinho, and Lynce 2014) MaxSAT solver—among the solvers in the 2015 MaxSAT Evaluation on Partial MaxSAT—and the CPLEX IP solver. The experiments were run on 2.83-GHz Intel Xeon E5440 quad-core machines with 32-GB memory and Debian GNU/Linux 8 using a per-instance timeout of 900 seconds.

We present results for a choice of four major AF semantics: the NP-complete enforcement problems of strict complete and non-strict admissible and stable, as well as the \( \Sigma^P_2 \)-complete strict preferred. As the only system available for
comparison, we consider the recently proposed IP-based approach to non-strict stable by Coste-Marquis et al. (2015) using CPLEX. The performance of CPLEX on our encoding and the approach of Coste-Marquis et al. for non-strict stable (Fig. 4 left) essentially coincide also empirically, corroborating the fact that the encodings are essentially the same. Interestingly, the relative performance of OpenWBO and CPLEX varies noticeably depending on the combination of (non)strictness and the semantics; CPLEX dominates on non-strict stable, while OpenWBO is better on strict complete; compare Fig. 4 middle and right. While OpenWBO tends to produce more timeouts, the median runtimes (Fig. 4 left) of OpenWBO are noticeably lower than those of CPLEX. For the challenging $\Sigma_2^P$-complete problem for strict preferred, our prototype implementation of the proposed CEGAR approach, using OpenWBO and complete as the base abstraction, already performs well, solving instances with 200 arguments and beyond (Fig. 5).

**Other Extension Enforcement Variants**

Finally, we shortly discuss other variants of extension enforcement, namely, under normal, strong, and weak expansions. Baumann and Brewka (2010) consider enforcement under so-called expansions of an AF $F = (A, R)$, which result in $F' = (A \cup A', R \cup R')$ with new arguments $A'$ and new attacks $R'$ s.t. $A \cap A' = R \cap R' = \emptyset$ and one of $A'$ or $R'$ is non-empty. An expansion is normal if for each $(a, b) \in R'$ we have $a \in A'$ or $b \in A'$; strong if $a \notin A'$; and weak if $b \notin A$. The tasks for enforcement under these variants are the same as for strict and non-strict, with the additional requirement that the enforcing AF is to be a normal, strong, or weak expansion. Considering these variants, the proof of Proposition 2 implies the following for a fixed $A'$.

**Corollary 8.** Non-strict enforcement under normal, strong, or weak expansions

- for admissible, complete, preferred, and stable semantics is NP-complete; and

- for semi-stable and stage semantics is $\Sigma_2^P$-complete.

As pointed out by Coste-Marquis et al. (2015), enforcement under expansions can be encoded via additional hard constraints for a fixed set $A'$ of additional arguments; the same holds for the MaxSAT encodings. Furthermore, our CEGAR algorithm (Algorithm 1) with adapted hard constraints can also be applied for semi-stable and stage semantics under normal, strong, or weak expansions. Also, our approach allows for further enforcement variants, e.g., any combination of (i) imposing other constraints on the way the attack structure may be changed, e.g., utilizing hard unit clauses to state that certain attacks must not be removed, (ii) attaching weights to attacks and searching for weight-minimum changes, or (iii) removal of arguments (by removing the argument and all attacks the argument is involved in). However, we note that it is not clear under which condition the cost of optimal solutions is preserved when restricting these problems by considering a fixed set $A'$ (e.g. a singleton set). In fact, the following example shows that the costs of optimal solutions—and the sets of optimal solutions—do

![Figure 5: CEGAR on strict preferred](image)

![Figure 6: Enforcement under weak expansions](image)
not in general coincide. Consider strictly enforcing \( \{t_1, t_2\} \) under semi-stable semantics. Weak expansion in the AF in Figure 6 requires two new arguments (for each argument \( t_1 \) and \( t_2 \) we need a new argument to extend their range); restricting \( \mathcal{A}' \) to singleton sets would not yield any solutions.

**Conclusions**

We presented both new complexity results and novel algorithms, based on a declarative optimization approach, for several variants of fixed-argument extension enforcement. As the main contributions, on the theoretical side we presented a nearly complete computational complexity map of the considered problem variants. Complementing the theoretical analysis, we proposed algorithms for the variants, ranging from polytime results to procedures going beyond NP for the second-level complete problem variants. Further improving the efficiency of the approach both via understanding what makes enforcement instances hard and via employing optimization solvers incrementally with the CEGAR approach, as well as extensions to other types of argumentation dynamics, are important aspects of future work.

**References**


