Instantiations and Computational Aspects of Non-Flat Assumption-based Argumentation

Tuomo Lehtonen[1] , **Anna Rapberger**[2] , **Francesca Toni**[2] ,
Markus Ulbricht[3] and **Johannes P. Wallner**[4]
[1]University of Helsinki, Department of Computer Science
[2]Imperial College London, Department of Computing
[3]Leipzig University, ScaDS.AI Dresden/Leipzig
[4]Graz University of Technology, Institute of Software Technology

Abstract

Most existing computational tools for *assumption-based argumentation (ABA)* focus on so-called *flat* frameworks, disregarding the more general case. In this paper, we study an instantiation-based approach for reasoning in possibly *non-flat* ABA. We make use of a semantics-preserving translation between ABA and bipolar argumentation frameworks (BAFs). By utilizing compilability theory, we establish that the constructed BAFs will in general be of exponential size. To keep the number of arguments and computational cost low, we present three ways of identifying redundant arguments. Moreover, we identify fragments of ABA which admit a poly-sized instantiation. We propose two algorithmic approaches for reasoning in non-flat ABA; the first utilizes the BAF instantiation while the second works directly without constructing arguments. An empirical evaluation shows that the former outperforms the latter on many instances, reflecting the lower complexity of BAF reasoning. This result is in contrast to flat ABA, where direct approaches dominate instantiation-based solvers.

1 Introduction

Formal argumentation constitutes a prominent branch of AI that studies and develops computational approaches to reason argumentatively [Baroni *et al.*, 2018]. The heterogeneity of this field is reflected in various formalizations and application domains, such as legal reasoning, medical sciences, and e-democracy [Atkinson *et al.*, 2017]. Computational approaches to solve key reasoning tasks are critical to the deployment of formal argumentation.

Argumentation formalisms are often classified as either abstract or structured . Abstract argumentation [Dung, 1995] is concerned with acceptability of arguments based exclusively on the relations between them. Structured argumentation formalisms [Besnard *et al.*, 2014] capture an entire argumentative workflow [Caminada and Amgoud, 2007]: starting from knowledge bases, a process of argument generation or instantiation is prescribed, upon which *semantics* can be deployed to find acceptable arguments or conclusions thereof.

Computational approaches to structured argumentation have gained increased attention in the research community. The biannual International Competition on Computational Models of Argumentation (ICCMA) [Thimm and Villata, 2017; Gaggl *et al.*, 2020; Lagniez *et al.*, 2020; Bistarelli *et al.*, 2021; Järvisalo *et al.*, 2023] has recently included a dedicated track for assumption-based argumentation (ABA) [Bondarenko *et al.*, 1997]—one of the prominent approaches to structured argumentation . Several algorithmic approaches to ABA or other well-known structured argumentation formalisms like ASPIC$^+$ [Modgil and Prakken, 2013] have been proposed recently [Craven and Toni, 2016; Lehtonen *et al.*, 2017; Bao *et al.*, 2017; Karamlou *et al.*, 2019; Lehtonen *et al.*, 2020; Lehtonen *et al.*, 2021a; Lehtonen *et al.*, 2021b; Diller *et al.*, 2021; Lehtonen *et al.*, 2022b; Thimm, 2017; Lehtonen *et al.*, 2022a; Lehtonen *et al.*, 2023].

Within this surge of algorithmic efforts, many solutions focus on restricted fragments of structured argumentation. For instance, most algorithms for ABA (e.g. [Craven and Toni, 2016; Lehtonen *et al.*, 2021a; Diller *et al.*, 2021]) focus on the *flat ABA* fragment, which imposes a strong restriction on the knowledge base by excluding derivations of 'assumptions'. The general ABA language, not restricted to the flat fragment and referred to in this paper as *non-flat ABA*, is able to capture more expressive settings such as auto-epistemic reasoning [Bondarenko *et al.*, 1997] and multi-agent settings where merging information from different sources can result in a non-flat knowledge base [Ulbricht *et al.*, 2024]. More broadly, non-flat ABA can capture situations where dependencies between assumptions need to be taken into account, and is therefore strictly more expressive than flat ABA. However, the few algorithms for non-flat ABA heavily restrict the kind of ABA frameworks (ABAFs) to which they apply (in [Karamlou *et al.*, 2019] to *bipolar* ABA frameworks [Cyras *et al.*, 2017]). A possible reason for the focus on flat ABA is the lower computational complexity. As shown by Dimopoulos *et al.* (2002) and Čyras *et al.* (2021a), all major reasoning tasks exhibit a one level jump in the polynomial hierarchy when going from flat to non-flat ABA, which, again, reflects the increased expressiveness of the general (not-flat) case.

In this paper we fill this gap and address computational challenges of non-flat ABA. In particular, we investigate algorithmic approaches to reasoning in non-flat ABA, establish

new theoretical insights and evaluate them in practice.

Algorithms for flat ABAFs can be divided into (i) instantiating a semantics-preserving argumentation framework (AF) [Dung, 1995] and then performing the reasoning task in the AF; and (ii) direct approaches operating on the given ABAF without instantiation. There are benefits to instantiation-based approaches, such as explainability in the form of more abstract, cognitively tractable explanations [Čyras *et al.*, 2021b]. For performance, direct approaches, specifically ones using modern declarative methods, such as answer set programming (ASP) [Gelfond and Lifschitz, 1988; Niemelä, 1999], typically have an edge over instantiation-based solvers in structured argumentation (e.g. [Lehtonen *et al.*, 2021a; Lehtonen *et al.*, 2023]). However, in the case of non-flat ABA this might plausibly not be the case. A recent instantiation [Ulbricht *et al.*, 2024] translates a given non-flat ABAF into a bipolar AF (BAF) [Amgoud *et al.*, 2008] such that reasoning in the constructed BAF is computationally milder compared to the initial ABAF . This is in contrast to the classical translation of flat ABAFs into AFs, where the complexity of reasoning is as high for the AFs.

In this work, we study the advantage this lower complexity can give for reasoning in non-flat ABAFs via instantiation to BAFs. While ideally the resulting BAF would be of polynomial size, similarly to a recently proposed translation from flat ABA to AF [Lehtonen *et al.*, 2023], we show that this is impossible in non-flat ABA. Instead, we investigate redundancies that can be eliminated to optimize the instantiation.

In more detail, our main contributions are:

- We present a result based on compilability theory, suggesting that an instantiation in BAFs cannot avoid an exponential number of arguments. Section 3.1
- Motivated by the lower complexity in instantiated BAFs compared to non-flat ABAFs, we show how to efficiently instantiate BAFs. We present three redundancy notions for argument generation. Section 3.2
- Towards a greater reach for applications, we also identify fragments of non-flat ABA with milder complexity: atomic and additive non-flat ABAFs. Section 3.3
- We propose two algorithmic approaches for reasoning in non-flat ABAFs. The first one efficiently instantiates a BAF via ASP using our redundancy notions, followed by SAT-based reasoning on the BAF. Section 4

  The second one performs reasoning directly on the given non-flat ABAF using iterative ASP calls, similarly to state-of-the-art approaches for other beyond-NP structured argumentation problems. Section 5
- We show empirically that both algorithms are competitive, with relative performance depending on the benchmark set. This contrasts with the dominance of non-instantiation approaches for other structured argumentation formalisms. Section 6

We focus on complete, grounded, stable and a complete-based version of preferred semantics. In a technical appendix [Lehtonen *et al.*, 2024], we expand on proof details, encodings, and how to approach admissible-based semantics, which was shown to be more involved [Ulbricht *et al.*, 2024].

2 Background

We recall preliminaries for assumption-based argumentation frameworks (ABAFs) [Bondarenko *et al.*, 1997; Čyras *et al.*, 2018], bipolar argumentation frameworks (BAFs) [Amgoud *et al.*, 2008], instantiation of BAFs to capture ABAFs [Ulbricht *et al.*, 2024] and basics of computational complexity .

Assumption-based Argumentation. We assume a deductive system $(\mathcal{L}, \mathcal{R})$, with $\mathcal{L}$ restricted to a set of atoms and $\mathcal{R}$ a set of rules over $\mathcal{L}$. A rule $r \in \mathcal{R}$ has the form $a_0 \leftarrow a_1, \ldots, a_n$ with $a_i \in \mathcal{L}$. We denote the head of r by $head(r) = a_0$ and the (possibly empty) body of r with $body(r) = \{a_1, \ldots, a_n\}$.

Definition 2.1. An ABAF is a tuple $(\mathcal{L}, \mathcal{R}, \mathcal{A}, {}^-)$, where $(\mathcal{L}, \mathcal{R})$ is a *deductive system*, $\mathcal{A} \subseteq \mathcal{L}$ is a non-empty set of *assumptions*, and ${}^- : \mathcal{A} \to \mathcal{L}$ is a (total) *contrary* function.

Here, we focus on finite ABAFs, i.e., $\mathcal{L}$ and $\mathcal{R}$ are finite.

An atom $p \in \mathcal{L}$ is derivable from assumptions $S \subseteq \mathcal{A}$ and rules $R \subseteq \mathcal{R}$, denoted by $S \vdash_R p$, if there is a finite rooted labeled tree G such that the root is labeled with p, the set of labels for the leaves of G is S or $S \cup \{\top\}$, and for every inner node v of G there is a rule $r \in R$ such that v is labelled with $head(r)$, the number of children of v is $|body(r)|$ and every child of v is labelled with a distinct $a \in body(r)$ or $\top$ if $body(r) = \emptyset$. We call such a G a *tree-based argument*. A *sub-argument* of G is a finite rooted labelled sub-tree G' such that the leafs of G' are a subset of the leafs of G. We will denote with A_D the set of all tree-based arguments of an ABAF D. Following Toni (2014), we will often compactly denote a tree-based argument with root p and leafs S as $S \vdash p$.

For $S \subseteq \mathcal{A}$, we let $\overline{S} = \{\overline{a} \mid a \in S\}$. By $Th_D(S) = \{p \in \mathcal{L} \mid \exists S' \subseteq S : S' \vdash_R p\}$ we denote the set of all conclusions derivable from S. Note that $S \subseteq Th_D(S)$ as each $a \in \mathcal{A}$ is derivable via $\{a\} \vdash_\emptyset a$. For $S \vdash p \in A_D$, we let $asms(S \vdash p) = S$; for $E \subseteq A_D$ we let $asms(E) = \bigcup_{x \in E} asms(x)$.

Definition 2.2. Let $D = (\mathcal{L}, \mathcal{R}, \mathcal{A}, {}^-)$ be an ABAF, $S, T \subseteq \mathcal{A}$, and $a \in \mathcal{A}$. The *closure* $cl(S)$ of S is $cl(S) = Th_D(S) \cap \mathcal{A}$. S is *closed* iff $S = cl(S)$; S *attacks* T iff $\overline{b} \in Th_D(S)$ for some $b \in T$; S *defends* a iff for each closed $V \subseteq \mathcal{A}$ s.t. V attacks a, S attacks V; S *defends itself* iff S defends each $b \in S$.

An ABAF where each set of assumptions is closed is called *flat*. We refer to an ABAF not restricted to be flat as *non-flat*.

For $a \in \mathcal{A}$, we also say S attacks a if S attacks the singleton $\{a\}$, and we write $cl(a)$ instead of $cl(\{a\})$.

A set $S \subseteq \mathcal{A}$ is *conflict-free* in D, denoted $E \in cf(D)$, if E is not self-attacking; S is *admissible*, denoted $S \in adm(D)$, if S is closed, conflict-free and it defends itself. We focus on grounded, complete, $\subseteq$-maximal complete, and stable ABA semantics (abbr. grd, com, prf', stb), as follows.[1]

Definition 2.3. Let $D = (\mathcal{L}, \mathcal{R}, \mathcal{A}, {}^-)$ be an ABAF and $S \subseteq \mathcal{A}$ be a set of assumptions s.t. $S \in adm(S)$. We say

[1] In the paper we show results for grd, com, stb and prf' defined as $\subseteq$-maximal complete sets (a variant to the usual prf semantics given as $\subseteq$-maximal admissible sets). We give results for adm and the standard prf in [Lehtonen *et al.*, 2024]. Although the intersection of all complete sets has been originally termed *well-founded semantics* [Bondarenko *et al.*, 1997], we stick to the usual convention in the argumentation literature and call it "grounded".

- $S \in com(D)$ iff it contains every $T \subseteq \mathcal{A}$ it defends,
- $S \in prf'(D)$ iff S is $\subseteq$-maximal in $com(D)$,
- $S \in grd(D)$ iff $S = \bigcap_{T \in com(D)} T$, and
- $S \in stb(D)$ iff S attacks each $x \in \mathcal{A} \setminus S$.

In this paper we stipulate that the empty intersection is interpreted as $\emptyset$, thus if $com(D) = \emptyset$, then $grd(D) = \emptyset$.

Example 2.4. We let D be an ABAF with assumptions $\mathcal{A} = \{a, b, c\}$, rules $\mathcal{R} = \{(p \leftarrow a), (q \leftarrow b), (c \leftarrow p, q)\}$, and $\overline{b} = p$. The set $\{a, c\}$ is ($\subseteq$-maximal) complete and stable in D since $\{a, c\}$ is unattacked and $\{a\}$ attacks b. Note that $\{a, b\}$ is not closed since $cl(\{a, b\}) = \{a, b, c\}$.

Bipolar Argumentation. Bipolar argumentation features adversarial and supportive relations between arguments.

Definition 2.5. A *bipolar argumentation framework (BAF)* $\mathcal{F}$ is a tuple $\mathcal{F} = (A, \mathrm{Att}, \mathrm{Sup})$ where A represents a set of arguments, $\mathrm{Att} \subseteq A \times A$ models *attack*, and $\mathrm{Sup} \subseteq A \times A$ models *support* between them.

Thus, in the tradition of Dung's abstract *argumentation frameworks (AFs)* (1995), arguments in BAFs are considered abstract, but BAFs extend AFs by integrating support. We call $F = (A, \mathrm{Att})$ the underlying AF of $\mathcal{F} = (A, \mathrm{Att}, \mathrm{Sup})$.

For two arguments $x, y \in A$, if $(x, y) \in \mathrm{Att}$ ($(x, y) \in \mathrm{Sup}$) we say that x *attacks* (*supports*) y as well as x *attacks* (*supports*) (the set) $E \subseteq A$ given that $y \in E$.

We utilize the BAF semantics introduced by Ulbricht *et al.* (2024). For a set $E \subseteq A$, we let $cl(E) = E \cup \{a \in A \mid \exists e \in E : (e, a) \in \mathrm{Sup}\}$. The set E is *closed* if $E = cl(E)$; E is *conflict-free* in $\mathcal{F}$, denoted $E \in cf(\mathcal{F})$, if for no $x, y \in E$, $(x, y) \in \mathrm{Att}$; E *defends* $a \in A$ if E attacks each closed set $S \subseteq A$ which attacks a. The *characteristic function* of $\mathcal{F}$ is $\Gamma(E) = \{a \in A \mid E \text{ defends } a\}$. We say E is *admissible*, denoted $E \in adm(\mathcal{F})$, if E is closed, conflict-free, and $E \subseteq \Gamma(E)$.

Definition 2.6. Let $\mathcal{F}$ be a BAF. For a set $E \in adm(\mathcal{F})$,

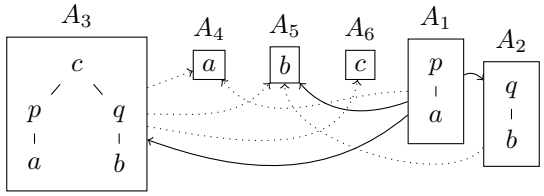
- $E \in com(\mathcal{F})$ iff $E = \Gamma(E)$;
- $E \in grd(\mathcal{F})$ iff $E = \bigcap_{S \in com(\mathcal{F})} S$;
- $E \in prf'(\mathcal{F})$ iff $E \subseteq$-maximal in $com(\mathcal{F})$;
- $E \in stb(\mathcal{F})$ iff E attacks all $a \in A \setminus E$.

ABA and Bipolar Argumentation. Each ABAF can be captured as a BAF as follows [Ulbricht *et al.*, 2024].

Definition 2.7. For an ABAF $D = (\mathcal{L}, \mathcal{R}, \mathcal{A}, {}^-)$, the *instantiated BAF* $\mathcal{F}_D = (A, \mathrm{Att}, \mathrm{Sup})$ is given by $A = A_D$ and

$$\mathrm{Att} = \{(S \vdash p, T \vdash q) \in A \times A \mid p \in \overline{T}\},$$
$$\mathrm{Sup} = \{(S \vdash p, \{a\} \vdash a) \in A \times A \mid a \in cl(S)\}.$$

Example 2.8. The ABAF from Example 2.4 yields the following BAF, with attacks as solid and supports as dashed lines, and arguments depicted as trees (with root at the top).



This BAF construction captures semantics for ABAFs, as recently shown [Ulbricht *et al.*, 2024].

Theorem 2.9. *Let D be an ABAF, $\mathcal{F}_D = (A, \mathrm{Att}, \mathrm{Sup})$ the associated BAF, and $\sigma \in \{com, prf', grd, stb\}$. Then*

- *if $E \in \sigma(\mathcal{F}_D)$, then $asms(E) \in \sigma(D)$, and*
- *if $S \in \sigma(D)$, then $\{x \in A \mid asms(x) \subseteq S\} \in \sigma(\mathcal{F}_D)$.*

Computational Complexity. We assume the reader to be familiar with the polynomial hierarchy. We focus on the credulous reasoning task. For a BAF $\mathcal{F}$ and a semantics σ, an argument $a \in A$ is *credulously accepted* if $a \in E$ for some $E \in \sigma(\mathcal{F})$; for an ABAF D and a semantics σ, a conclusion $p \in \mathcal{L}$ is *credulously accepted* if $p \in Th_D(E)$ for some $E \in \sigma(D)$. The induced decision problems are denoted by $Cred_\sigma^{BAF}$ and $Cred_\sigma^{ABA}$, respectively. We note that $Cred_{com}^{\mathcal{C}} = Cred_{prf'}^{\mathcal{C}}$ for $\mathcal{C} \in \{BAF, ABA\}$.

Deciding credulous reasoning for non-flat ABA is Σ_2^{P}-complete for complete, DP_2-complete for grounded, and NP-complete for stable semantics [Dimopoulos *et al.*, 2002; Čyras *et al.*, 2021a]. As recently shown [Ulbricht *et al.*, 2024], the corresponding decision problems for BAFs exhibit lower complexity for main semantics: DP-completeness for grounded and NP-completeness for complete and stable.

3 BAF Generation in Theory

For instantiation-based algorithms the number of arguments is critical for the run time performance. Direct instantiation methods which compute all tree-based arguments may yield unfeasibly large argumentation graphs. This has already been pointed out for flat ABA, which motivated the study of means to reduce their size [Lehtonen *et al.*, 2023]. In this section, we analyze redundancies for the non-flat case as well. However, our first observation is that the computation of exponentially many arguments can in general not be avoided for non-flat ABA. A result of this kind is, to the best of our knowledge, novel for structured argumentation.

3.1 A Lower Bound For Non-Flat Instantiations

We give a formal line of reasoning that it is impossible to instantiate a given non-flat ABAF in a "reasonable" way and thereby obtain a polynomial-sized graph (i.e., AF or BAF).

In more detail, we show a complexity result based on compilability theory [Cadoli *et al.*, 2002] stating that, unless the polynomial hierarchy collapses, one cannot transform a given non-flat ABAF D into some structure χ with the following properties: i) χ is of polynomial size w.r.t. D; and ii) in χ one can decide in polynomial time whether a given set E of assumptions is admissible or complete in D. Since verifying admissible sets in non-flat ABA is NP-complete, it is clearly impossible to construct such χ in polynomial time. However, we do not need this restriction. That is, even given exponential time, such χ cannot be constructed. Conceptually, this result excludes usual argument-centric instantiations of D into AFs or BAFs, because here we expect checking whether a set of arguments is admissible to be tractable.

That is, under complexity theoretic assumptions, our result states that it is impossible to apply instantiations of D into a

polynomial-sized AF or BAF which have the usual correspondences between sets of arguments and sets of assumptions.

Theorem 3.1. *Unless the polynomial hierarchy collapses one cannot transform a non-flat ABAF into a polynomial-sized AF or BAF from which one can in polynomial-time decide whether a given set of assumptions is admissible or complete.*

3.2 Towards Feasible BAF Instantiations

The previous subsection shows that instantiation of non-flat ABAFs will require exponentially many arguments in general. We strive to construct as few as possible nonetheless. We identify three redundancy notions to reduce the number of arguments. We consider an arbitrary but fixed semantics $\sigma \in \{com, grd, stb, prf'\}$ throughout this subsection.

Derivation Redundancy. We call the first notion derivation redundant arguments as it spots "inefficient" derivations.

Definition 3.2. For an ABAF D and its set of arguments A_D, we call an argument $(S \vdash p) \in A_D$ *derivation redundant* iff there is an argument $(S' \vdash p) \in A_D$ with $S' \subsetneq S$.

Example 3.3. Recall Example 2.4. Suppose we consider an ABAF D' by setting $\mathcal{R}' = \mathcal{R} \cup \{c \leftarrow p\}$. Then we would obtain a new argument $\{a\} \vdash c$. Then the existing argument A_3 representing $\{a, b\} \vdash c$ becomes derivation redundant.

We observe that derivation redundant arguments can be removed without altering the sets of accepted assumptions.

Proposition 3.4. *Let D be an ABAF and $\mathcal{F}_D$ the corresponding BAF. Let $x \in A_D$ be derivation redundant and let $\mathcal{G}$ be the BAF after removing the argument x from $\mathcal{F}_D$. Then*

$$\{asms(E) \mid E \in \sigma(\mathcal{F}_D)\} = \{asms(E) \mid E \in \sigma(\mathcal{G})\}.$$

Expendable Arguments. We derive another redundancy notion based on the conclusion of arguments: if $x = (S \vdash p)$ is an argument where p is neither an assumption nor a contrary, then x merely represents an intermediate derivation step. Arguments of this kind do not need to be instantiated if one is interested in assumption extensions only.

Definition 3.5. For an ABAF D and its set of arguments A_D we call an argument $(S \vdash p) \in A_D$ *expendable* iff $p \notin \mathcal{A} \cup \overline{\mathcal{A}}$.

Example 3.6. In our Example 2.4, for instance the argument A_2 representing $\{b\} \vdash q$ is expendable since q is neither a contrary nor an assumption.

Since arguments of this kind have no out-going attacks, they do not contribute to the semantics of the instantiated BAF. However, keep in mind that we still construct relevant super-arguments of expendable ones.

Proposition 3.7. *Let D be an ABAF and $\mathcal{F}_D$ the corresponding BAF. Let $x \in A_D$ be an expendable argument and let $\mathcal{G}$ be the BAF after removing the argument x from $\mathcal{F}_D$. Then*

$$\{asms(E) \mid E \in \sigma(\mathcal{F}_D)\} = \{asms(E) \mid E \in \sigma(\mathcal{G})\}.$$

Assumption Redundancy. This final redundancy notion is specific to non-flat ABAFs. It states that arguments that make use of assumptions in intermediate steps can be neglected. This requires rules with assumptions in their head and is thus not possible for flat ABA.

Definition 3.8. For an ABAF $D = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^-)$ and its set of arguments A_D, an argument $x = (S \vdash p) \in A_D$ is *assumption redundant* iff it contains a sub-argument x' s.t.

- x' is a proper sub-argument of x, i.e., $x' \neq x$,
- x is of the form $S' \vdash a$ where $S' \subseteq S$ and $a \in \mathcal{A}$.

Example 3.9. Suppose we augment Example 2.4 with the additional rule "$a \leftarrow b$". This would lead to a novel argument A_7 for p by first applying the rule "$a \leftarrow b$" and then "$p \leftarrow a$". However, since a is an assumption itself, it is more efficient to infer p from a directly, which is represented by the argument A_1. Thus A_7 would be assumption redundant.

As for the other redundancy notions, arguments of this kind can be removed without altering the semantics.

Proposition 3.10. *Let D be an ABAF and $\mathcal{F}_D$ the corresponding BAF. Let $x \in A_D$ be an assumption redundant argument and let $\mathcal{G}$ be the BAF after removing x from $\mathcal{F}_D$. Then*

$$\{asms(E) \mid E \in \sigma(\mathcal{F}_D)\} = \{asms(E) \mid E \in \sigma(\mathcal{G})\}.$$

Summary. With our redundancy notions, the instantiated BAF can be reduced as follows. From A_D we construct the set A_D^* of *non-redundant* arguments by i) first removing all derivation redundant arguments from A_D; ii) then removing all expendable arguments from the result of i); and finally iii) removing all assumption redundant arguments from the result of ii). Now we define the redundancy-free core of D.

Definition 3.11. Let $D = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^-)$ be an ABAF. The BAF $\mathcal{G} = (A, \mathrm{Att}, \mathrm{Sup})$ is the *non-redundant core* of D where $A = \{(S, p) \mid S \vdash p \text{ is an argument in } A_D^*\}$, Att is the set of all attacks between arguments in A, and Sup is the set of all supports between arguments in A.

Due to Propositions 3.4, 3.7 and 3.10 this representation is semantically equivalent.

Corollary 3.12. *Let D be an ABAF and $\mathcal{F}_D$ its corresponding BAF. If $\mathcal{G}$ is the redundancy-free core of D, then we have that $\{asms(E) \mid E \in \sigma(\mathcal{F}_D)\} = \{asms(E) \mid E \in \sigma(\mathcal{G})\}$.*

By our previous results, the non-redundant core preserves the semantics of the given ABAF, since only redundant arguments are omitted and the representation streamlined. By applying this representation, we still have exponentially many arguments in general, but finiteness is guaranteed.

Proposition 3.13. *The redundancy-free core $\mathcal{G}$ of an ABAF D has at most $|2^{\mathcal{A}}| \cdot |\mathcal{L}|$ arguments.*

3.3 Fragments

As we just saw, non-flat ABA instantiations will in general have exponentially many arguments. In this subsection, we investigate fragments inducing fewer arguments. Since reasoning in BAFs is milder than in non-flat ABAFs, we expect such fragments to admit lower complexity. Indeed, if the core computation is polynomial, then the complexity drops.

Proposition 3.14. *Let $\mathcal{C}$ be a class of ABAFs s.t. the non-redundant core of D can be computed in polynomial time. Then, the computational complexity of reasoning problems in D is not harder than in the instantiated BAF, i.e., $Cred_\sigma^{ABA}$ is in* NP *for $\sigma \in \{adm, com, prf', stb\}$ and in* DP *for $\sigma = grd$.*

Indeed, in our empirical evaluation in Section 6 we will see that these fragments are milder in practice as well.

Atomic ABAFs The first fragment we consider is *atomic*, which has been studied for flat ABA as well [Rapberger and Ulbricht, 2023; Lehtonen *et al.*, 2023]. For an ABAF to be atomic, each rule body element has to be an assumption.

Definition 3.15. Let $D = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^-)$ be an ABAF. A rule $r \in \mathcal{R}$ is called *atomic* if $body(r) \subseteq \mathcal{A}$. The ABAF D is called *atomic* if each rule $r \in \mathcal{R}$ is atomic.

For flat ABA, this means that D has $|\mathcal{R}| + |\mathcal{A}|$ arguments (each rule induces exactly one tree-based argument) [Lehtonen *et al.*, 2023]. In contrast, the same is not immediate for non-flat ABA as the derivation of assumptions is allowed. Nevertheless, due to our notion of assumption redundancy from Definition 3.8, we can show that the number of non-redundant arguments is indeed linear in D.

Proposition 3.16. *The non-redundant core of atomic ABAFs D consists of at most $|\mathcal{R}| + |\mathcal{A}|$ many arguments.*

Example 3.17. Consider an ABAF D with $\mathcal{A} = \{a, b, c\}$, rules $\mathcal{R} = \{(p \leftarrow a), (q \leftarrow b), (a \leftarrow c)\}$, and an arbitrary contrary function. D is atomic since each rule body consists of assumptions. While it is possible to construct more that $|\mathcal{R}|$ + $|\mathcal{A}|$ arguments (by applying "$a \leftarrow c$" and then "$p \leftarrow a$"), these additional arguments are assumption redundant.

As a consequence, Proposition 3.14 applies to the class of atomic ABAFs, inducing lower complexity of reasoning.

Additive Closure In an atomic ABAF, each non-redundant argument has a derivation depth of 1. We can also bound the body size of rules, obtaining so-called *additive* ABAFs.

Definition 3.18. We call an ABAF $D = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^-)$ *additive* if for each rule $r \in \mathcal{R}$ it holds that $|body(r)| \leq 1$.

The reason for calling ABAFs of this kind additive is that they induce an additive Th_D mapping, that is, for each set S of assumptions, $Th_D(S) = \bigcup_{s \in S} Th_D(s)$.

Example 3.19. Consider an ABAF D with $\mathcal{A} = \{a, b, c\}$, $\mathcal{L} = \mathcal{A} \cup \{p, q, r, s\}$, rules $\mathcal{R} = \{(p \leftarrow a), (q \leftarrow b), (r \leftarrow p), (s \leftarrow q)\}$, and an arbitrary contrary function. This ABAF is additive since each rule body has size one. Consequently, each constructible argument is based on one assumption only (in contrast to e.g. A_3 in Example 2.8 relying on a and b).

Note that Bipolar ABA [Cyras *et al.*, 2017] is a special kind of non-flat ABA restricted to being both atomic and additive.

As suggested by the previous example, each argument in an additive ABAF has the form $\{\} \vdash p$ or $\{a\} \vdash p$ for some assumption a. This induces the following bound.

Proposition 3.20. *The non-redundant core of an additive ABAF D consists of at most $(|\mathcal{A}| + 1) \cdot |\mathcal{L}|$ many arguments.*

Also in this case, Proposition 3.14 is applicable, including the computational benefits of lower complexity.

4 Algorithms for Non-flat ABA via BAFs

We introduce an approach for solving reasoning problems in non-flat ABA by instantiating a BAF and solving the corresponding reasoning problem in the BAF. We focus on credulous reasoning for complete (and so preferred′) and stable semantics. We employ efficient declarative methods, namely answer set programming (ASP) [Gelfond and Lifschitz, 1988; Niemelä, 1999] for generating arguments, and Boolean satisfiability (SAT) [Biere *et al.*, 2021] for BAF reasoning.

Listing 1: Program Π_{gen_arg}

```
1 {in(X) : assumption(X)}.
2 derivable(X) ← assumption(X), in(X).
3 derivable(X) ← head(R,X), usable_rule(R).
4 usable_rule(R) ← head(R,_), derivable(X) : body(R,X).
```

4.1 BAF Generation

We introduce a novel approach for generating non-redundant arguments (see Section 3.2) from an ABAF, using the state-of-the-art ASP solver CLINGO [Gebser *et al.*, 2016].

Firstly, we present an ABAF $D = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^-)$ with $\mathcal{R} = \{r_1, \ldots, r_n\}$ in ASP as follows.

$$\begin{aligned}\texttt{ABA}(D) = &\{\mathbf{assumption}(a). \mid a \in \mathcal{A}\} \cup \\ &\{\mathbf{head}(i, b). \mid r_i \in \mathcal{R}, b = head(r_i)\} \cup \\ &\{\mathbf{body}(i, b). \mid r_i \in \mathcal{R}, b \in body(r_i)\} \cup \\ &\{\mathbf{contrary}(a, b). \mid b = \overline{a}, a \in \mathcal{A}\}.\end{aligned}$$

Then, we introduce the ASP program Π_{gen_arg} (see Listing 1), which by itself enumerates each assumption set and determines what can be derived from it. The following procedure limits redundancy in the set of arguments.[2] Firstly, by Proposition 3.4, we eliminate derivation redundant arguments. We use CLINGO heuristics [Gebser *et al.*, 2013] for this, adding the heuristic that each assumption is by default not in. Secondly, by Proposition 3.7, one retains semantic equivalence even if expendable arguments are ignored. As we are interested in credulous acceptance, we also construct arguments for atoms whose acceptance we want to query. Optionally, instead of a query atom, a set of atoms can be specified, so that the acceptance of any of these atoms can be decided from a single BAF. We enumerate answers to $\texttt{ABA}(D) \cup \Pi_{arg\_gen}$ together with a constraint that **derivable**(a) holds, for each atom of interest a, resulting in answer sets corresponding to non-derivation redundant, non-expendable arguments for a. The attack and support relations over the arguments can be straightforwardly determined based on the assumptions and conclusion of each argument.

4.2 SAT Encodings for BAFs

We move on to proposing SAT encodings that capture the BAF semantics of interest , following Definition 2.6. Our encodings are based on the standard SAT encodings for AFs [Besnard and Doutre, 2004] with modifications to capture the additional aspects of BAFs. Given a BAF $\mathcal{F} = (A, \mathrm{Att}, \mathrm{Sup})$ and the closure $cl(a)$ of each argument $a \in A$ (which can be precomputed in polynomial time), we capture conflict-free, closed, self-defending, and complete sets of arguments, respectively, as follows. We use variables x_a for $a \in A$ to denote that a is in the extension.

Conflict-freeness is encoded as for AFs: $cf(\mathcal{F}) = \bigwedge_{(a,b) \in Att}(\neg x_a \vee \neg x_b)$. For closedness, if an argument a

[2] We chose to include assumption redundant arguments still, leaving their elimination to future work.

is in an extension, then each argument b that a supports must be in the extension, too: $closed(\mathcal{F}) = \bigwedge_{(a,b)\in Sup}(\neg x_a \vee x_b)$.

Note that a set of arguments E defends an argument b if and only if E attacks the closure of every argument a that attacks b [Ulbricht *et al.*, 2024, Lemma 3.4]. Thus an extension E defends itself iff it holds that, if $b \in E$, then for each a that attacks b, some argument d that attacks an argument c in the closure of a must be in the extension.

$$self\_defense(\mathcal{F}) = \bigwedge_{(a,b)\in Att} \left(x_b \rightarrow \bigvee_{\substack{(d,c)\in Att,\\ c\in cl(a)}} x_d \right)$$

For a complete extension E, any argument b must be in E if it holds that an argument in the closure of each argument a that attacks b is attacked by an argument in the extension.

$$defended(\mathcal{F}) = \bigwedge_{b\in A} \left(\left(\bigwedge_{(a,b)\in Att} \bigvee_{\substack{(d,c)\in Att,\\ c\in cl(a)}} x_d \right) \rightarrow x_b \right)$$

Taken together, we encode complete semantics by $com(\mathcal{F}) = cf(F) \wedge closed(\mathcal{F}) \wedge self\_defense(\mathcal{F}) \wedge defended(\mathcal{F})$.

We can encode stability similarly to the standard encoding for AFs, with only the addition that an extension is closed: $stb(F) = cf(\mathcal{F}) \wedge closed(\mathcal{F}) \wedge \bigwedge_{a\in A} \left(x_a \vee \bigvee_{(b,a)\in Att} x_b \right)$.

To decide credulous acceptance under $\sigma \in \{com, stb\}$, we add the clause $cred(\mathcal{F}, \alpha) = \bigvee_{a\in A_\alpha} x_a$ where $A_\alpha \subseteq \mathcal{A}$ is the set of arguments concluding α.

Proposition 4.1. *Given an ABAF $D = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^-)$, $\sigma \in \{com, stb\}$ and $\alpha \in \mathcal{L}$, let $\mathcal{F}$ be the BAF constructed from D via the procedure of Section 4.1. Then $\sigma(\mathcal{F}) \wedge cred(\mathcal{F}, \alpha)$ is satisfiable iff α is credulously accepted under σ.*

5 ASP Algorithms for Non-flat ABA

In this section we introduce an ASP-based approach for credulous acceptance in non-flat ABA without constructing arguments. We propose a counterexample-guided abstraction refinement (CEGAR) [Clarke *et al.*, 2004; Clarke *et al.*, 2003] algorithm for complete semantics, inspired by state-of-the-art algorithms for other problems in structured argumentation, including flat ABA, that are hard for the second level of the polynomial hierarchy [Lehtonen *et al.*, 2021b; Lehtonen *et al.*, 2022b; Lehtonen *et al.*, 2022a]. For stable semantics, we propose an ASP encoding , reflecting the fact that credulous acceptance is NP-complete under stable semantics.

We use ASP to generate candidates based on an abstraction of the original problem, and another ASP solver to verify whether a counterexample to the candidate being a solution exists. We use the ASP encoding of ABAFs from Section 4.1, and the notation $solve(\Pi)$ to refer to solving the ASP program Π and assume that $solve(\Pi)$ either returns an answer set or reports that the program is unsatisfiable. Details of the ASP programs are available in [Lehtonen *et al.*, 2024].

We introduce Algorithm 1 for credulous acceptance under complete semantics. As subprocedures, we introduce

Algorithm 1: Credulous acceptance, complete semantics

Require: ABA framework $F = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^-)$, $s \in \mathcal{L}$
Ensure: return YES if s is credulously accepted under complete semantics in F, NO otherwise
1: **while** $C := solve(\Pi_{abs})$ **do**
2: $\quad flag := T$
3: $\quad$ **if** $solve(\Pi_{not\_adm}(C))$ unsatisfiable **then**
4: $\quad\quad$ **for** $a \in \mathcal{A}$ s.t. $a \notin C$ and a not attacked by C **do**
5: $\quad\quad\quad$ **if** $solve(\Pi_{defends}(C, a))$ **then** $flag := F$; break
6: $\quad\quad$ **if** $flag = T$ **then return** YES
7: $\quad$ Add constraint excluding C to Π_{abs}
8: **return** NO

three ASP programs: the abstraction Π_{abs}, a program that checks for a counterexample to admissibility Π_{not_adm}, and $\Pi_{defends}(C, a)$ for checking counterexamples to completeness, i.e., if particular assumptions are defended. The abstraction Π_{abs} admits as answers closed and conflict-free assumption sets from which the query is derivable, along with a stronger condition for complete semantics. Namely, if an assumption is not in the candidate and not attacked by the set of undefeated assumptions, the candidate cannot be complete (since this assumption cannot be attacked by any closed assumption set in particular, and is thus defended by the candidate). The program Π_{not_adm} is satisfiable iff a closed set of assumptions that is not attacked by the candidate attacks the candidate, implying that the candidate is not admissible. Finally, given a candidate $C \subseteq \mathcal{A}$ and an assumption $a \in \mathcal{A}$, the program $\Pi_{defends}(C, a)$ is satisfiable iff there is a set of assumptions that is not attacked by C, is closed, and attacks a. In such a case the candidate does not defend a.

Algorithm 1 iteratively generates candidates (Line 1) and checks admissibility (Line 3). In case a candidate is admissible, it is further checked whether there is an assumption that is not in the candidate or attacked by the candidate (Line 4), such that this assumption is defended by the candidate (Line 5). If not, the candidate is complete and thus the queried atom is credulously accepted (Line 6). Otherwise (if C is either not admissible or not complete) the abstraction is refined by excluding the candidate from further consideration (Line 7). Finally, if no complete assumption set is found, the queried atom is not credulously accepted (Line 8).

Proposition 5.1. *Given an ABAF $D = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^-)$ and a query $\alpha \in \mathcal{L}$, Algorithm 1 outputs YES if and only if α is credulously accepted under complete semantics in D.*

Stable semantics can be encoded by slightly adapting the encoding for flat ABAFs [Lehtonen *et al.*, 2021a]. Namely, we add a constraint for the assumption set being closed. For finding the credulous acceptance of $s \in \mathcal{L}$, we add a constraint requiring that s is derivable from the assumption set.

6 Empirical Evaluation

We present an evaluation of the algorithms proposed in Sections 4 and 5, named ABABAF[3] and ASPFORABA[4], respec-

[3] Available at https://bitbucket.org/lehtonen/ababaf.

[4] Available at https://bitbucket.org/coreo-group/aspforaba.

tively. We implemented both approaches in Python, using CLINGO (version 5.5.1) [Gebser *et al.*, 2016] for ASPFORABA and for generating the arguments in ABABAF. We used PYSAT (version 0.1.7) [Ignatiev *et al.*, 2018] with GLUCOSE (version 1.0.3) [Audemard and Simon, 2009; Eén and Sörensson, 2003] as the SAT solver in ABABAF. We used 2.50 GHz Intel Xeon Gold 6248 machines under a per-instance time limit of 600 seconds and memory limit of 32 GB.

Lacking a standard benchmark library for non-flat ABA, we generated two benchmark sets adapted from flat ABA benchmarks [Järvisalo *et al.*, 2023]. Set 1 has the following parameters: number of atoms in $\{80, 120, 160, 200\}$, ratio of atoms that are assumptions in $\{0.2, 0.4\}$, ratio of assumptions occurring as rule heads in $\{0.2, 0.5\}$, and both number of rules deriving any given atom and rule size (number of atoms in the body of rule) selected at random from the interval $[1, n]$ for $n \in \{1, 2, 5\}$. We call the maximum rules per atom mr and maximum rule size ms; instances with $ms = 1$ are additive. For benchmark set 2, we limited mr and ms to $\{2, 5\}$, and generated instances a certain distance from atomic. For this, a *slack* parameter specifies how many atoms in each rule body can be non-assumptions. Here slack is $0, 1$ or 2, the first resulting in atomic ABAFs. We generated 5 instances for each combination of parameters for both benchmark sets.

Table 1 summarizes results for credulous acceptance under complete semantics. On benchmark set 1 (Table 1, left), generally ABABAF performs better than ASPFORABA when the parameters take lower values. ABABAF outperforms ASPFORABA when $ms = 1$, corresponding to additive ABA frameworks, but also for $ms = 2$ when mr is 1 or 2. ASPFORABA, on the other hand, performs best with $mr = 5$. The results for benchmark set 2 (for credulous acceptance under complete semantics) can be seen in Table 1 (right). On atomic instances ($slack = 0$) ABABAF outperforms ASPFORABA. As the slack increases, the performance of ABABAF decreases, while ASPFORABA somewhat improves. Evidently, ABABAF is able to take advantage of the lower complexity of additive and atomic instances. The results suggest that the approaches are complementary and their relative performance varies by parameters.

We show the run times of both algorithms against the number of arguments a given instance gives rise to in Figure 1 (for instances solved byABABAF; timeouts of ASPFORABA shown as 600 seconds). For ABABAF, there is a clear correlation between run time and size of the constructed BAF, as can be expected given that the arguments construction is time-consuming and the BAF is the input for a SAT call. In contrast, BAF size does not predict run time for ASPFORABA, since ASPFORABA does not instantiate arguments.

We also evaluated our algorithms under stable semantics. As expected given the lower complexity compared to complete semantics, ASPFORABA performs better, solving all instances. On the other hand, ABABAF constructs the BAF for stable semantics too, and accordingly the performance of ABABAF is significantly worse than ASPFORABA, with multiple timeouts. This highlights that a crucial component in the performance of ABABAF on complete semantics is the lower complexity of deciding acceptance in BAF compared to the corresponding ABAF.

		#solved (mean run time (s))			
ms	mr	ABABAF		ASPFORABA	
1	1	80	(0.2)	56	(9.3)
	2	80	(0.4)	66	(8.4)
	5	80	(6.2)	79	(0.1)
2	1	70	(0.6)	62	(6.8)
	2	54	(17.1)	40	(24.2)
	5	50	(13.2)	66	(14.4)
5	1	80	(2.3)	80	(0.1)
	2	54	(4.0)	72	(5.3)
	5	11	(24.9)	43	(37.1)

		#solved (mean run time (s))			
$slack$	ms	ABABAF		ASPFORABA	
0	2	147	(3.8)	104	(41.5)
	5	109	(7.3)	77	(51.3)
1	2	112	(15.4)	112	(18.1)
	5	51	(20.0)	89	(31.3)
2	2	122	(21.0)	118	(8.2)
	5	59	(6.6)	102	(31.2)

Table 1: Number of solved instances and mean run time over solved instances under complete semantics in benchmark sets 1 (left) and 2 (right). There are 80 and 160 instances per row in sets 1 and 2.

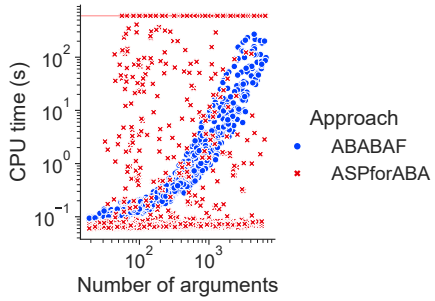


Figure 1: The effect of the number of arguments an ABAF gives rise to on the run time of our approaches w.r.t. complete semantics.

7 Conclusion

Efficient algorithmic solutions for assumption-based argumentation are widely studied; however, despite the wide variety of different solvers for assumption-based reasoning, most existing approaches only focus on the flat ABA fragment. In this work we investigated theoretical foundations and algorithms for non-flat ABA reasoning, which has higher complexity than in flat ABA.

Our redundancy notions gave rise to two fragments of ABA, i.e., atomic and additive ABA, in which reasoning exhibits the same complexity as for flat ABA. We proposed two algorithmic approaches, one instantiating a BAF and one without argument construction. The former faces an exponential number of arguments in general, but we proposed and applied redundancy notions, and employed state-of-the-art solving techniques. For the latter, we adapted state-of-the-art algorithms for structured argumentation. We showed empirically that, in contrast to previous instantiation-based approaches for structured argumentation, our novel instantiation-based approach is able to outperform the direct approach, in particular for problems on the second level of the polynomial hierarchy.

For future work, we want to extend our implementation to further common semantics and reasoning tasks such as skeptical acceptance ; this is an additional challenge, since e.g. skeptical reasoning under preferred semantics lies on the third level of the polynomial hierarchy. Another promising line of future work is to develop methods for computing explanations for ABA in spirit of the work by Dung *et al.* (2006), utilizing our instantiation-based implementation.

Acknowledgements

This research was funded by the Austrian Science Fund (FWF) P35632 and University of Helsinki Doctoral Programme in Computer Science DoCS; by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No. 101020934, ADIX) and by J.P. Morgan and by the Royal Academy of Engineering under the Research Chairs and Senior Research Fellowships scheme; by the Federal Ministry of Education and Research of Germany and by Sächsische Staatsministerium für Wissenschaft, Kultur und Tourismus in the programme Center of Excellence for AI-research "Center for Scalable Data Analytics and Artificial Intelligence Dresden/Leipzig", project identification number: ScaDS.AI. The authors wish to thank the Finnish Computing Competence Infrastructure (FCCI) for supporting this project with computational and data storage resources.

References

[Amgoud *et al.*, 2008] Leila Amgoud, Claudette Cayrol, Marie-Christine Lagasquie-Schiex, and P. Livet. On bipolarity in argumentation frameworks. *Int. J. Intell. Syst.*, 23(10):1062–1093, 2008.

[Atkinson *et al.*, 2017] Katie Atkinson, Pietro Baroni, Massimiliano Giacomin, Anthony Hunter, Henry Prakken, Chris Reed, Guillermo Ricardo Simari, Matthias Thimm, and Serena Villata. Towards artificial argumentation. *AI Mag.*, 38(3):25–36, 2017.

[Audemard and Simon, 2009] Gilles Audemard and Laurent Simon. Predicting learnt clauses quality in modern SAT solvers. In *Proc. IJCAI*, pages 399–404, 2009.

[Bao *et al.*, 2017] Ziyi Bao, Kristijonas Čyras, and Francesca Toni. ABAplus: Attack reversal in abstract and structured argumentation with preferences. In *Proc. PRIMA*, volume 10621 of *Lecture Notes in Computer Science*, pages 420–437. Springer, 2017.

[Baroni *et al.*, 2018] Pietro Baroni, Dov Gabbay, Massimiliano Giacomin, and Leendert van der Torre, editors. *Handbook of Formal Argumentation*. College Publications, 2018.

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

[Besnard *et al.*, 2014] Philippe Besnard, Alejandro Javier García, Anthony Hunter, Sanjay Modgil, Henry Prakken, Guillermo Ricardo Simari, and Francesca Toni. Introduction to structured argumentation. *Argument Comput.*, 5(1):1–4, 2014.

[Biere *et al.*, 2021] Armin Biere, Marijn Heule, Hans van Maaren, and Toby Walsh, editors. *Handbook of Satisfiability - Second Edition*, volume 336 of *Frontiers in Artificial Intelligence and Applications*. IOS Press, 2021.

[Bistarelli *et al.*, 2021] Stefano Bistarelli, Lars Kotthoff, Francesco Santini, and Carlo Taticchi. Summary report for the third international competition on computational models of argumentation. *AI Mag.*, 42(3):70–73, 2021.

[Bondarenko *et al.*, 1997] Andrei Bondarenko, Phan Minh Dung, Robert A. Kowalski, and Francesca Toni. An abstract, argumentation-theoretic approach to default reasoning. *Artif. Intell.*, 93:63–101, 1997.

[Cadoli *et al.*, 2002] Marco Cadoli, Francesco M. Donini, Paolo Liberatore, and Marco Schaerf. Preprocessing of intractable problems. *Inf. Comput.*, 176(2):89–120, 2002.

[Caminada and Amgoud, 2007] Martin Caminada and Leila Amgoud. On the evaluation of argumentation formalisms. *Artif. Intell.*, 171(5-6):286–310, 2007.

[Clarke *et al.*, 2003] Edmund M. Clarke, Orna Grumberg, Somesh Jha, Yuan Lu, and Helmut Veith. Counterexample-guided abstraction refinement for symbolic model checking. *J.ACM*, 50(5):752–794, 2003.

[Clarke *et al.*, 2004] Edmund M. Clarke, Anubhav Gupta, and Ofer Strichman. SAT-based counterexample-guided abstraction refinement. *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.*, 23(7):1113–1123, 2004.

[Craven and Toni, 2016] Robert Craven and Francesca Toni. Argument graphs and assumption-based argumentation. *Artif. Intell.*, 233:1–59, 2016.

[Cyras *et al.*, 2017] Kristijonas Cyras, Claudia Schulz, and Francesca Toni. Capturing bipolar argumentation in non-flat assumption-based argumentation. In *Proc. PRIMA*, volume 10621 of *Lecture Notes in Computer Science*, pages 386–402. Springer, 2017.

[Čyras *et al.*, 2018] Kristijonas Čyras, Xiuyi Fan, Claudia Schulz, and Francesca Toni. Assumption-based argumentation: Disputes, explanations, preferences. In *Handbook of Formal Argumentation*, chapter 7, pages 365–408. College Publications, 2018.

[Čyras *et al.*, 2021a] Kristijonas Čyras, Quentin Heinrich, and Francesca Toni. Computational complexity of flat and generic assumption-based argumentation, with and without probabilities. *Artif. Intell.*, 293:103449, 2021.

[Čyras *et al.*, 2021b] Kristijonas Čyras, Antonio Rago, Emanuele Albini, Pietro Baroni, and Francesca Toni. Argumentative XAI: a survey. In *Proc. IJCAI*, pages 4392–4399. ijcai.org, 2021.

[Diller *et al.*, 2021] Martin Diller, Sarah Alice Gaggl, and Piotr Gorczyca. Flexible dispute derivations with forward and backward arguments for assumption-based argumentation. In *Proc. CLAR*, volume 13040 of *Lecture Notes in Computer Science*, pages 147–168. Springer, 2021.

[Dimopoulos *et al.*, 2002] Yannis Dimopoulos, Bernhard Nebel, and Francesca Toni. On the computational complexity of assumption-based argumentation for default reasoning. *Artif. Intell.*, 141(1/2):57–78, 2002.

[Dung *et al.*, 2006] Phan Minh Dung, Robert A Kowalski, and Francesca Toni. Dialectic proof procedures for assumption-based, admissible argumentation. *Artif. Intell.*, 170(2):114–159, 2006.

[Dung, 1995] Phan Minh Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reason-

ing, logic programming and n-person games. *Artif. Intell.*, 77(2):321–358, 1995.

[Eén and Sörensson, 2003] Niklas Eén and Niklas Sörensson. An extensible sat-solver. In Enrico Giunchiglia and Armando Tacchella, editors, *Proc. SAT*, volume 2919 of *Lecture Notes in Computer Science*, pages 502–518. Springer, 2003.

[Gaggl *et al.*, 2020] Sarah A. Gaggl, Thomas Linsbichler, Marco Maratea, and Stefan Woltran. Design and results of the second international competition on computational models of argumentation. *Artif. Intell.*, 279, 2020.

[Gebser *et al.*, 2013] Martin Gebser, Benjamin Kaufmann, Javier Romero, Ramón Otero, Torsten Schaub, and Philipp Wanko. Domain-specific heuristics in answer set programming. In *Proc. AAAI*, pages 350–356. AAAI Press, 2013.

[Gebser *et al.*, 2016] Martin Gebser, Roland Kaminski, Benjamin Kaufmann, Max Ostrowski, Torsten Schaub, and Philipp Wanko. Theory solving made easy with Clingo 5. In *Technical Communications of ICLP*, OASICS, pages 2:1–2:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2016.

[Gelfond and Lifschitz, 1988] Michael Gelfond and Vladimir Lifschitz. The stable model semantics for logic programming. In *Proc. ICLP/SLP*, pages 1070–1080. MIT Press, 1988.

[Ignatiev *et al.*, 2018] Alexey Ignatiev, Antonio Morgado, and Joao Marques-Silva. PySAT: A Python toolkit for prototyping with SAT oracles. In *Proc. SAT*, volume 10929 of *Lecture Notes in Computer Science*, pages 428–437. Springer, 2018.

[Järvisalo *et al.*, 2023] Matti Järvisalo, Tuomo Lehtonen, and Andreas Niskanen. Design of ICCMA 2023, 5th international competition on computational models of argumentation: A preliminary report (invited paper). In *Proc. First International Workshop on Argumentation and Applications*, volume 3472 of *CEUR Workshop Proceedings*, pages 4–10. CEUR-WS.org, 2023.

[Karamlou *et al.*, 2019] Amin Karamlou, Kristijonas Čyras, and Francesca Toni. Complexity results and algorithms for bipolar argumentation. In *Proc. AAMAS*, pages 1713–1721. IFAAMAS, 2019.

[Lagniez *et al.*, 2020] Jean-Marie Lagniez, Emmanuel Lonca, Jean-Guy Mailly, and Julien Rossit. Introducing the fourth international competition on computational models of argumentation. In *Proc. SAFA*, volume 2672 of *CEUR Workshop Proceedings*, pages 80–85. CEUR-WS.org, 2020.

[Lehtonen *et al.*, 2017] Tuomo Lehtonen, Johannes P. Wallner, and Matti Järvisalo. From structured to abstract argumentation: Assumption-based acceptance via AF reasoning. In *Proc. ECSQARU*, volume 10369 of *Lecture Notes in Computer Science*, pages 57–68. Springer, 2017.

[Lehtonen *et al.*, 2020] Tuomo Lehtonen, Johannes P. Wallner, and Matti Järvisalo. An answer set programming approach to argumentative reasoning in the ASPIC+ framework. In *Proc. KR*, pages 636–646. IJCAI, 2020.

[Lehtonen *et al.*, 2021a] Tuomo Lehtonen, Johannes P. Wallner, and Matti Järvisalo. Declarative algorithms and complexity results for assumption-based argumentation. *J. Artif. Intell. Res.*, 71:265–318, 2021.

[Lehtonen *et al.*, 2021b] Tuomo Lehtonen, Johannes P. Wallner, and Matti Järvisalo. Harnessing incremental answer set solving for reasoning in assumption-based argumentation. *Theory Pract. Log. Program.*, 21(6):717–734, 2021.

[Lehtonen *et al.*, 2022a] Tuomo Lehtonen, Johannes P. Wallner, and Matti Järvisalo. Algorithms for reasoning in a default logic instantiation of assumption-based argumentation. In *Proc. COMMA*, volume 353 of *Frontiers in Artificial Intelligence and Applications*, pages 236–247. IOS Press, 2022.

[Lehtonen *et al.*, 2022b] Tuomo Lehtonen, Johannes P. Wallner, and Matti Järvisalo. Computing stable conclusions under the weakest-link principle in the ASPIC+ argumentation formalism. In *Proc. KR*, pages 215–225. IJCAI, 2022.

[Lehtonen *et al.*, 2023] Tuomo Lehtonen, Anna Rapberger, Markus Ulbricht, and Johannes P. Wallner. Argumentation frameworks induced by assumption-based argumentation: Relating size and complexity. In *Proc. KR*, pages 440–450. IJCAI, 2023.

[Lehtonen *et al.*, 2024] Tuomo Lehtonen, Anna Rapberger, Francesca Toni, Markus Ulbricht, and Johannes P. Wallner. Instantiations and computational aspects of non-flat assumption-based argumentation. *CoRR*, abs/2404.11431, 2024.

[Modgil and Prakken, 2013] Sanjay Modgil and Henry Prakken. A general account of argumentation with preferences. *Artif. Intell.*, 195:361–397, 2013.

[Niemelä, 1999] Ilkka Niemelä. Logic programs with stable model semantics as a constraint programming paradigm. *Ann. Math. Artif. Intell.*, 25(3-4):241–273, 1999.

[Rapberger and Ulbricht, 2023] Anna Rapberger and Markus Ulbricht. On dynamics in structured argumentation formalisms. *J. Artif. Intell. Res.*, 77:563–643, 2023.

[Thimm and Villata, 2017] Matthias Thimm and Serena Villata. The first international competition on computational models of argumentation: Results and analysis. *Artif. Intell.*, 252:267–294, 2017.

[Thimm, 2017] Matthias Thimm. The tweety library collection for logical aspects of artificial intelligence and knowledge representation. *Künstliche Intell.*, 31(1):93–97, 2017.

[Toni, 2014] Francesca Toni. A tutorial on assumption-based argumentation. *Argument Comput.*, 5(1):89–117, 2014.

[Ulbricht *et al.*, 2024] Markus Ulbricht, Nico Potyka, Anna Rapberger, and Francesca Toni. Non-flat ABA is an instance of bipolar argumentation. In *Proc. AAAI*, pages 10723–10731. AAAI Press, 2024.