Complexity Results and Algorithms for Bipolar Argumentation

Amin Karamlou
Imperial College London
London, United Kingdom
mak514@imperial.ac.uk

Kristijonas Čyras
Imperial College London
London, United Kingdom
k.cyras@imperial.ac.uk

Francesca Toni
Imperial College London
London, United Kingdom
f.toni@imperial.ac.uk

ABSTRACT

Bipolar Argumentation Frameworks (BAFs) admit several interpretations of the support relation and diverging definitions of semantics. Recently, several classes of BAFs have been captured as instances of bipolar Assumption-Based Argumentation, a class of Assumption-Based Argumentation (ABA). In this paper, we establish the complexity of bipolar ABA, and consequently of several classes of BAFs. In addition to the standard five complexity problems, we analyse the rarely-addressed extension enumeration problem too. We also advance backtracking-driven algorithms for enumerating extensions of bipolar ABA frameworks, and consequently of BAFs under several interpretations. We prove soundness and completeness of our algorithms, describe their implementation and provide a scalability evaluation. We thus contribute to the study of the as yet uninvestigated complexity problems of (variously interpreted) BAFs as well as of bipolar ABA, and provide the lacking implementations thereof.

KEYWORDS

Complexity; Structured Argumentation; Bipolar Argumentation

1 INTRODUCTION

Human-understandable agent interaction is an important topic in multi-agent systems. Argumentation has been widely-used to model agent interaction, e.g. [1, 5, 30, 32], especially in the form of debates, e.g. [18, 25, 33, 35]. Bipolar argumentation (see e.g. [6, 9, 12]) in particular has been shown to be applicable in capturing, formalising and executing debates, e.g. [3, 26, 34]. Thus, issues pertaining to practical deployment of bipolar argumentation are of great importance.

Bipolar Argumentation Frameworks (BAFs) (see e.g. [6, 9, 20]) constitute one prominent class of formalisms for bipolar argumentation. In particular, they admit various interpretations of support and diverging definitions of semantics, which arguably impinge their practical deployment. Recently, bipolar Assumption-Based Argumentation (*bipolar ABA*) [12] has been shown to subsume BAFs under various interpretations of support (to which we henceforth refer to as *various BAFs*), thus allowing for the consolidation of theoretical foundations of bipolar argumentation. However, the complexity of bipolar ABA and various BAFs is largely unknown (except for some complexity problems in one form of BAFs, namely deductive BAFs [6], as given in [19]). What is more, implementations of bipolar argumentation are generally lacking too (except for deductive BAFs, as given in [17]). This is despite the fact that computational problems in bipolar argumentation have tremendous potential for practical use, for instance, in knowing the effectiveness of answering questions such as "Does there exist a winner of the debate?", or in yielding all the 'winning' arguments.

In this paper, we address the above issues and provide complexity results as well as implementations for bipolar ABA, and therefore indirectly for various BAFs. Specifically, we analyse the (non-empty) existence, verification, (credulous and sceptical) acceptance and enumeration complexity problems in bipolar ABA under the semantics capturing various BAFs. We establish that bipolar ABA is equally as complex as abstract argumentation (AA) [14]. We then give algorithms for extension enumeration in bipolar ABA, which effectively capture solutions to other complexity problems too. We describe an implementation of bipolar ABA as well as various BAFs and complement it with a scalability evaluation, showing that our system is fit for practical deployment.

The paper is organised as follows. Section 2 provides background on argumentation and complexity theory, as well as existing complexity results for AA. We give the complexity results for bipolar ABA in Section 3. In Section 4 we advance new algorithms for implementing bipolar argumentation. We describe the software system implementing these algorithms in Section 5, alongside evaluating the system's scalability practically. We review related work in Section 6 and discuss conclusions and future work in Section 7.

2 BACKGROUND

We here give background on argumentation and complexity.

2.1 Argumentation

We start with background on argumentation.

2.1.1 Assumption-Based Argumentation (ABA). Background on ABA and its restriction Bipolar ABA follows [4, 11, 12, 38].

An *ABA framework* is a tuple $(\mathcal{L}, \mathcal{R}, \mathcal{A}, ^-)$, where:

- $(\mathcal{L}, \mathcal{R})$ is a deductive system with $\mathcal{L}$ a language (i.e. a set of sentences) and $\mathcal{R}$ a set of rules of the form $\varphi_0 \leftarrow \varphi_1, \ldots, \varphi_m$ with $m \geqslant 0$ and $\varphi_i \in \mathcal{L}$ for $i \in \{0, \ldots, m\}$; φ_0 is the *head* and $\varphi_1, \ldots, \varphi_m$ the *body*; if $m = 0$, then $\varphi_0 \leftarrow \varphi_1, \ldots, \varphi_m$ has an empty body, and is written as $\varphi_0 \leftarrow \top$, where $\top \notin \mathcal{L}$;
- $\mathcal{A} \subseteq \mathcal{L}$ is a non-empty set of *assumptions*;
- $^- : \mathcal{A} \to \mathcal{L}$ is a total map: for $\alpha \in \mathcal{A}$, the $\mathcal{L}$-sentence $\overline{\alpha}$ is referred to as the *contrary* of α.

For the remainder of this section, we assume as given a fixed but otherwise arbitrary ABA framework $\mathcal{F} = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^-)$.

- A *deduction for* $\varphi \in \mathcal{L}$ *supported by* $A \subseteq \mathcal{A}$ *and* $R \subseteq \mathcal{R}$, denoted $A \vdash^R \varphi$, is a finite tree with: the root labelled by φ; leaves labelled by $\top$ or assumptions, with A being the set of all such assumptions; the children of non-leaves ψ labelled by the elements of the body of some ψ-headed rule in $\mathcal{R}$, with R being the set of all such rules.
- $A \subseteq \mathcal{A}$ *attacks* $B \subseteq \mathcal{A}$, denoted $A \rightsquigarrow_{\mathsf{ABA}} B$, if there is a deduction $A' \vdash^R \overline{\beta}$ such that $\beta \in B$, $A' \subseteq A$ and $R \subseteq \mathcal{R}$. If it is not the case that A attacks B, we may write $A \not\rightsquigarrow_{\mathsf{ABA}} B$.

Let $A \subseteq \mathcal{A}$: (1) The *closure* of A is $Cl(A) = \{\alpha \in \mathcal{A} : \exists A' \vdash^R \alpha, A' \subseteq A, R \subseteq \mathcal{R}\}$. (2) A is *closed* iff $A = Cl(A)$. (3) $\mathcal{F}$ is *flat* iff every $A \subseteq \mathcal{A}$ is closed. (4) A is *conflict-free* iff $A \not\rightsquigarrow_{\mathsf{ABA}} A$. (5) A *defends* $\alpha \in \mathcal{A}$ iff for all closed $B \subseteq \mathcal{A}$ with $B \rightsquigarrow_{\mathsf{ABA}} \{\alpha\}$ it holds that $A \rightsquigarrow_{\mathsf{ABA}} B$. We also say A defends $B \subseteq \mathcal{A}$ if A defends every $\beta \in B$.

We use the following ABA semantics. A set $E \subseteq \mathcal{A}$, also called an *extension*, is: (1) *admissible* iff it is closed, conflict-free and defends itself. (2) *preferred* iff it is $\subseteq$-maximally admissible. (3) *stable* iff it is closed, conflict-free and $E \rightsquigarrow_{\mathsf{ABA}} \{\alpha\}\ \forall \alpha \in \mathcal{A} \setminus E$. (4) *set-stable* iff it is closed, conflict-free and $E \rightsquigarrow_{\mathsf{ABA}} Cl(\{\alpha\})\ \forall \alpha \in \mathcal{A} \setminus E$.

In $\mathcal{F}$, a stable extension is set-stable, a set-stable extension is preferred, and if $\mathcal{F}$ is flat, then a set-stable extension is also stable.

The restricted class Bipolar ABA is defined thus. An ABA framework $(\mathcal{L}, \mathcal{R}, \mathcal{A}, ^-)$ is *bipolar* iff every rule in $\mathcal{R}$ is of the form $\varphi \leftarrow \alpha$, where $\alpha \in \mathcal{A}$ and either $\varphi \in \mathcal{A}$ or $\varphi = \overline{\beta}$ for some $\beta \in \mathcal{A}$.

Bipolar (just as flat) ABA frameworks admit admissible and preferred but not, in general, stable or set-stable extensions.

2.1.2 Abstract Argumentation (AA). We give background on AA following [14]. An *AA framework* (AF) is a pair $(Args, \hookrightarrow)$ with a (finite) set $Args$ of arguments and a binary attack relation $\hookrightarrow$ on $Args$. Notions of conflict-freeness and defence, as well as semantics of admissible, preferred and stable extensions are defined verbatim as for ABA, but with (sets of) arguments replacing (sets of) assumptions and the closure condition dropped. (As in flat ABA, set-stable and stable semantics coincide.)

2.2 Elements of Complexity

We assume knowledge of fundamental time and space complexity classes, as well as the concepts of hardness and completeness [31]. Thus, we here recap the complexity problems studied in argumentation, as well as established results for AFs.

2.2.1 Enumeration. We first give (the less standard) enumeration problems and related complexity classes following [24]. An *enumeration problem* is a pair (L, Sol) such that $L \subseteq \Sigma^*$ (for an alphabet Σ containing at least two symbols) and $Sol : \Sigma^* \to 2^{\Sigma^*}$ is a function such that for all $x \in \Sigma^*$, we have that the set of *solutions* $Sol(x)$ is finite, and $Sol(x) = \emptyset$ iff $x \notin L$. An *enumeration algorithm* $\mathcal{A}$ for an enumeration problem $\mathcal{P} = (L, Sol)$ outputs, on input x, exactly the elements from $Sol(x)$ without duplicates. For enumeration algorithms, we use the RAM model of computation [24].

The complexity classes OutputP and nOP are defined thus. Let $\mathcal{P} = (L, Sol)$ be an enumeration problem. $\mathcal{P} \in$ OutputP if there exists an enumeration algorithm $\mathcal{A}$ for $\mathcal{P}$ and some $m \in \mathbb{N}$, such that on every input x, algorithm $\mathcal{A}$ terminates in time $\mathcal{O}((|x| + |Sol(x)|)^m)$. Problems *not* in OutputP constitute the class nOP.

The following decision problem – **MANYSOL**$(\mathcal{P})$: Given $x \in L$ and a positive integer m in unary notation, is $|Sol(x)| \geq m$? – is strongly related to the enumeration problem $\mathcal{P} = (L, Sol)$: If **MANYSOL**$(\mathcal{P}) \notin$ P, then $\mathcal{P} \notin$ OutputP. We will use **MANYSOL** in our analysis of the enumeration problem in bipolar ABA.

2.2.2 Problems of Interest. We now state the problems we are interested in. In the following, $\mathcal{F}$ stands for a bipolar ABA framework and $\sigma \in \{adm, prf, set\text{-}stb\}$ denotes a semantics, where *adm*, *prf* and *set-stb* abbreviate admissible, preferred and set-stable, respectively.

1. Existence ($\mathbf{EX}^{\mathcal{F}}_{\sigma}$): Does $\mathcal{F}$ admit a σ extension?
2. Non-Empty Existence ($\mathbf{NE}^{\mathcal{F}}_{\sigma}$): Does $\mathcal{F}$ admit a non-empty σ extension?
3. Verification ($\mathbf{VER}^{\mathcal{F}}_{\sigma}(A)$): Given $A \subseteq \mathcal{A}$, is A a σ extension of $\mathcal{F}$?
4. Credulous Acceptance ($\mathbf{CA}^{\mathcal{F}}_{\sigma}(a)$): Given $a \in \mathcal{L}$, is there a σ extension A of $\mathcal{F}$ such that $A' \vdash^R a$ for some $A' \subseteq A$ and $R \subseteq \mathcal{R}$?
5. sceptical Acceptance($\mathbf{SA}^{\mathcal{F}}_{\sigma}(a)$): Given $a \in \mathcal{L}$, is it the case that for every σ extension A of $\mathcal{F}$ it holds that $A' \vdash^R a$ for some $A' \subseteq A$ and $R \subseteq \mathcal{R}$?
6. Extension Enumeration($\mathbf{EE}^{\mathcal{F}}_{\sigma}$): Return all σ extensions of $\mathcal{F}$.

The above complexity problems admit natural counterparts in BAFs (as well as AFs). In fact, the only difference is in the credulous and sceptical acceptance problems, for which instead of asking for deductions as in bipolar ABA, one asks for containment in extensions in B(AFs), see e.g. [15, 16, 19]. As various BAFs are captured in bipolar ABA via a polynomial mapping [12], our complexity results for bipolar ABA in this paper will cover various BAFs too.

Existing complexity results for AFs are summarised in Table 1 (*stb* stands for stable); see [15, 16] for surveys of these results.

Table 1: Existing complexity results for AFs. (Here and henceforth, Y stands for 'Yes' and N stands for 'No'.

sem	Ex	NE	VER	CA	SA	EE
adm	Trivial (Y)	NP-c	P	NP-c	Trivial (N)	nOP
prf	Trivial (Y)	NP-c	coNP-c	NP-c	Π^{P}_2-c	nOP
stb	NP-c	NP-c	P	NP-c	coNP-c	nOP

3 COMPLEXITY RESULTS

In this section, we prove new complexity results for the complexity problems in bipolar ABA. Table 2 summarises our results.

Table 2: Complexity results for bipolar ABA frameworks.

sem	Ex	NE	VER	CA	SA	EE
adm	Trivial (Y)	NP-c	P	NP-c	Trivial (N)	nOP
prf	Trivial (Y)	NP-c	coNP-c	NP-c	Π^{P}_2-c	nOP
set-stb	NP-c	NP-c	P	NP-c	coNP-c	nOP

These results show that the problems for bipolar ABA frameworks belong to precisely the same complexity classes as their corresponding problems for AFs. As a consequence, the same results apply to the various BAFs investigated in [12].

We first present prerequisite results needed for all of the problems, then we study verification, before moving on to existence, acceptance, and enumeration problems. Note that because there exists a polynomial time mapping between AFs and bipolar ABA frameworks [12], all the computational problems for bipolar ABA are at least as hard as their AF counterparts.

Throughout, unless stated otherwise, we assume as given a fixed but otherwise arbitrary bipolar ABA framework $\mathcal{F} = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^-)$.

3.1 Prerequisite Results

The derivability problem for ABA frameworks is as follows.

Derivability($\mathbf{DER}^{\mathcal{F}}(A, \alpha)$): Given a set $A \subseteq \mathcal{A}$ and $\alpha \in \mathcal{L}$, does there exist a deduction of the form $A \vdash^{R} \alpha$?

Proposition 3.1. *$\mathbf{DER}^{\mathcal{F}}(A, \alpha)$ is* NL*-complete (thus in* P*).*

Proof.
Membership. The following algorithm operates in logspace and nondeterministically solves the $\mathbf{DER}^{\mathcal{F}}(A, \alpha)$ problem: (1) Create a variable β. (2) Set β equal to an arbitrary element of A. If $\beta = \alpha$, output 'yes'. Otherwise continue. (3) Initiate a counter $k = 0$. (4) Pick an arbitrary rule $R \in \mathcal{R}$ s.t. $\{\beta\}$ is the body of R. If no such rule exists, output 'no'. Otherwise continue. (5) If the head of R is equal to α, output 'yes'. Otherwise continue. (6) set β equal to the head of R. Increment k by 1. If $k \geq |\mathcal{A}|$, output 'no'. Otherwise return to step 4. Note that this algorithm operates in log space since the space usage of counter $k \leq \log(|\mathcal{R}|)$.
Hardness. We provide a (logspace) reduction from Reachability, the canonical NL-complete problem [31].
Reachability ($\mathbf{RCH}(G, s, t)$): Given a directed graph G and vertices s and t of G, is there a path from s to t in G?
The mapping below transforms a directed graph G into a language $\mathcal{L}$ and a set of bipolar ABA rules $\mathcal{R}$:
- $\mathcal{L} = \mathcal{A} = \{x : x \text{ is a node of } G\}$,
- $\mathcal{R} = \{y \leftarrow x : x \text{ and } y \text{ are nodes of } G \text{ and there exists an edge from } x \text{ to } y \text{ in } G\}$,
- $\overline{\alpha} = \alpha$ for $\alpha \in \mathcal{A}$.

This is a logspace transformation since we only need two counters to track the node and edge being considered at any point. Moreover, there is a path from s to t in G iff $s = t$ or there is a chain of rules $R \subseteq \mathcal{R}$ of the form $t \leftarrow \gamma_n \leftarrow \ldots \leftarrow \gamma_2 \leftarrow \gamma_1 \leftarrow s$. This is precisely the condition in which $\mathbf{DER}^{\mathcal{F}}(\{s\}, t)$ would output 'yes'. Thus $\mathbf{RCH}(G, s, t)$ is logspace reducible to $\mathbf{DER}^{\mathcal{F}}(A, \alpha)$. This means that $\mathbf{DER}^{\mathcal{F}}(A, \alpha)$ is NL-hard. □

We now analyse the fundamental properties of conflict-freeness and closure pertaining to all semantics considered in this paper.
Conflict-Freeness ($\mathbf{CF}^{\mathcal{F}}(A)$): Given $A \subseteq \mathcal{A}$, is A conflict-free in $\mathcal{F}$?
Closure ($\mathbf{CL}^{\mathcal{F}}(A)$): Given $A \subseteq \mathcal{A}$, is A closed in $\mathcal{F}$?

Proposition 3.2. *$\mathbf{CF}^{\mathcal{F}}(A)$ and $\mathbf{CL}^{\mathcal{F}}(A)$ are in* P.

Proof. We present P-time algorithms for both problems.
$\mathbf{CF}^{\mathcal{F}}(A)$: For each $\alpha \in A$, use an NL oracle for $\mathbf{DER}^{\mathcal{F}}(A, \overline{\alpha})$ to check if $A \rightsquigarrow_{\mathtt{ABA}} \{\alpha\}$. If it does, output 'no'. Else, output 'yes'.
$\mathbf{CL}^{\mathcal{F}}(A)$: For each $\alpha \notin A$, use an NL oracle for $\mathbf{DER}^{\mathcal{F}}(A, \alpha)$ to check if $A \vdash^{R} \alpha$ for some $R \subseteq \mathcal{R}$. If it does, output 'no'. Otherwise, output 'yes'. □

We will use the following result, which says that in a bipolar ABA framework, no sentence is deducible without any assumptions.

Lemma 3.3. *There is no deduction in $\mathcal{F}$ of the form $\emptyset \vdash^{R} \varphi$ for any $\varphi \in \mathcal{L}$ and $R \subseteq \mathcal{R}$.*

Proof. Bipolar ABA frameworks do not contain any facts (i.e. rules of the form $\alpha \leftarrow \top$). Hence, it is impossible to have $\top$ as the child of any node in a bipolar ABA deduction. As a result, a deduction of the form $\emptyset \vdash^{R} \overline{\varphi}$ does not exist. □

3.2 Verification

We now analyse the complexity of the verification problem under the admissible, preferred and set-stable semantics. In order to prove the results for admissible semantics, we first introduce the notion of minimal attacks in ABA.

Definition 3.4. $A \subseteq \mathcal{A}$ **minimally attacks** $B \subseteq \mathcal{A}$, denoted by $A \rightsquigarrow_{\mathtt{min\text{-}ABA}} B$, iff $A \rightsquigarrow_{\mathtt{ABA}} B$ and there is no $A' \subset A$ s.t. $A' \rightsquigarrow_{\mathtt{ABA}} B$.

Lemma 3.5. *All minimal attacks are of the form $\{\alpha\} \rightsquigarrow_{\mathtt{min\text{-}ABA}} B$ where $\alpha \in \mathcal{A}$ and $B \subseteq \mathcal{A}$.*

Proof. Assume there are $A \subseteq \mathcal{A}$ and $B \subseteq \mathcal{A}$ s.t. $A \rightsquigarrow_{\mathtt{min\text{-}ABA}} B$ and $|A| \neq 1$. Then we have two cases: (1) $|A| = \emptyset$: Lemma 3.3 implies that $A \not\rightsquigarrow_{\mathtt{ABA}} B$. This contradicts $A \rightsquigarrow_{\mathtt{min\text{-}ABA}} B$. (2) $|A| > 1$: In order for $A \rightsquigarrow_{\mathtt{ABA}} B$ there must exist $\alpha \in A$ and $\beta \in B$ where either $\alpha = \overline{\beta}$ or there exists a chain of rules $\overline{\beta} \leftarrow \gamma_n \leftarrow \ldots \leftarrow \gamma_2 \leftarrow \gamma_1 \leftarrow \alpha$. In both cases we have $\{\alpha\} \rightsquigarrow_{\mathtt{ABA}} B$. However, $\{\alpha\} \subset A$ so we have a contradiction to the definition of minimal attacks. In any event, $A = \{\alpha\}$ where $\alpha \in \mathcal{A}$ as required. □

Proposition 3.6. *$\mathbf{VER}^{\mathcal{F}}_{adm}(A)$ is in* P.

Proof. (1) Use a P oracle for $\mathbf{CF}^{\mathcal{F}}(A)$ to check if A is conflict-free. If it is not, output 'no'. Otherwise continue. (2) Use a P oracle for $\mathbf{CL}^{\mathcal{F}}(A)$ to check if A is closed. If it is not output 'no'. Otherwise continue. (3) For each assumption $\beta \notin A$, call an NL oracle for $\mathbf{DER}^{\mathcal{F}}(A, \overline{\alpha})$ $|A|$ times (once for each $\alpha \in A$) to check if $\{\beta\} \rightsquigarrow_{\mathtt{ABA}} A$. If it does and $A \not\rightsquigarrow_{\mathtt{ABA}} \{\beta\}$ output 'no'. Otherwise continue. (4) Output 'yes'.

Note that step 3 is sufficient to check that A defends itself. This follows from the fact that if $B \rightsquigarrow_{\mathtt{ABA}} A$ then $\{\beta\} \rightsquigarrow_{\mathtt{min\text{-}ABA}} A$ for some $\beta \in B$ (Lemma 3.5). From the definition of attacks, it follows that if $A \rightsquigarrow_{\mathtt{ABA}} \{\beta\}$ then $A \rightsquigarrow_{\mathtt{ABA}} B$ as well. Moreover, since step 1 of the algorithm checks that A is conflict-free, we know that $\beta \notin A$. So it suffices to prove that A defends itself against singleton sets of assumptions which are not contained within it. □

Proposition 3.7. *$\mathbf{VER}^{\mathcal{F}}_{prf}(A)$ is* coNP*-Complete.*

Proof. Membership comes from the following non-deterministic, P-time algorithm, adapted from [13] , which solves the $\mathbf{coVER}^{\mathcal{F}}_{\text{prf}}(A)$ problem: (1) Use a P oracle for $\mathbf{VER}^{\mathcal{F}}_{\text{adm}}(A)$ to check if A is admissible. If it is not, output 'yes'. Otherwise continue. (2) Guess an assumption set $A' \supset A$. (3) Use a P oracle for $\mathbf{VER}^{\mathcal{F}}_{\text{adm}}(A')$ to check if A' is admissible. If it is, output 'yes'. Otherwise output 'no'. □

Proposition 3.8. *$\mathbf{VER}^{\mathcal{F}}_{set\text{-}stb}(A)$ is in* P.

Proof. We present the following P-time algorithm: (1) Use a P oracle for $\mathbf{CF}^{\mathcal{F}}(A)$ to check if A is conflict-free. If it is not, output 'no'. Else continue. (2) Use a P oracle for $\mathbf{CL}^{\mathcal{F}}(A)$ to check if A is closed. If it is not, output 'no'. Else continue. (3) For each $\alpha \notin A$, calculate $Cl(\{\alpha\})$ by calling an NL oracle for $\mathbf{DER}^{\mathcal{F}}$ $|\mathcal{A}|$ times, and then check if $A \rightsquigarrow_{\mathtt{ABA}} Cl(\{\alpha\})$ using an additional $|Cl(\{\alpha\})|$ oracle calls. If it does not, output 'no'. Otherwise, output 'yes'. □

3.3 Existence and Acceptance

Before proving the remainder of our results we make the following observations.

Proposition 3.9. *$\boldsymbol{EX}^{\mathcal{F}}_{adm}$, $\boldsymbol{NE}^{\mathcal{F}}_{adm}$ and $\boldsymbol{CA}^{\mathcal{F}}_{adm}$ are respectively equivalent to $\boldsymbol{EX}^{\mathcal{F}}_{prf}$, $\boldsymbol{NE}^{\mathcal{F}}_{prf}$ and $\boldsymbol{CA}^{\mathcal{F}}_{prf}$.*

Proof. Follows from the fact that every preferred extension is admissible and every admissible extension is a subset of some preferred assumption set [13, Prop1]. □

Proposition 3.10. *$\boldsymbol{EX}^{\mathcal{F}}_{set\text{-}stb}$ is equivalent to $\boldsymbol{NE}^{\mathcal{F}}_{set\text{-}stb}$.*

Proof. Assume $\emptyset$ is set-stable in $\mathcal{F}$. As $\mathcal{A} \neq \emptyset$, there is $\alpha \in \mathcal{A}$ s.t. $\emptyset \rightsquigarrow_{\mathtt{ABA}} Cl(\{\alpha\})$. But this contradicts Lemma 3.3. Thus, $\emptyset$ is never set-stable in $\mathcal{F}$, and so existence of a set-stable extension is equivalent to the existence of a non-empty set-stable extension. □

3.3.1 Existence. We now consider (non-empty) existence.

Proposition 3.11. *$\boldsymbol{Ex}^{\mathcal{F}}_{adm}$ and $\boldsymbol{Ex}^{\mathcal{F}}_{prf}$ are constant, with answer 'yes'.*

Proof. $\emptyset$ is conflict-free, defends itself, and, by Lemma 3.3, is closed. Hence, $\emptyset$ is admissible, which establishes the claim for $\mathbf{Ex}^{\mathcal{F}}_{\text{adm}}$. The claim for $\mathbf{Ex}^{\mathcal{F}}_{\text{prf}}$ then follows from Proposition 3.9. □

Now we switch our attention to the non-emptiness problem.

Proposition 3.12. *$\boldsymbol{NE}^{\mathcal{F}}_{adm}$, $\boldsymbol{NE}^{\mathcal{F}}_{prf}$, $\boldsymbol{NE}^{\mathcal{F}}_{set\text{-}stb}$ and $\boldsymbol{Ex}^{\mathcal{F}}_{set\text{-}stb}$ are* NP*-Complete.*

Proof. The following non-deterministic, P-time algorithm proves membership for admissible and set-stable semantics. The results for $\mathbf{NE}^{\mathcal{F}}_{\text{prf}}$ and $\mathbf{Ex}^{\mathcal{F}}_{\text{set-stb}}$ follow from Propositions 3.9 and 3.10.

(1) Guess an assumption set $A \subseteq \mathcal{A}$. (2) Use a P oracle for $\mathbf{VER}^{\mathcal{F}}_{\text{adm}}(A)$ (or $\mathbf{VER}^{\mathcal{F}}_{\text{set-stb}}(A)$) to check if A is an admissible (or set-stable) extension. If not, output 'no'. Otherwise Output 'yes'. □

3.3.2 Credulous and Sceptical Acceptance. We now turn to acceptance problems.

Proposition 3.13. *$\boldsymbol{CA}^{\mathcal{F}}_{adm}(\alpha)$, $\boldsymbol{CA}^{\mathcal{F}}_{prf}(\alpha)$, and $\boldsymbol{CA}^{\mathcal{F}}_{set\text{-}stb}(\alpha)$ are* NP*-complete, $\boldsymbol{SA}^{\mathcal{F}}_{set\text{-}stb}(\alpha)$ is* coNP*-complete, and $\boldsymbol{SA}^{\mathcal{F}}_{prf}(\alpha)$ is Π^{P}_2-complete.*

Proof. Membership uses the following algorithm, adapted from [13], solving $\mathbf{CA}^{\mathcal{F}}_{\sigma}(\alpha)$ and $\mathbf{coSA}^{\mathcal{F}}_{\sigma}(\alpha)$, and our previous results for $\mathbf{VER}^{\mathcal{F}}_{\sigma}(A)$. The result for $\mathbf{CA}^{\mathcal{F}}_{\text{prf}}(\alpha)$ follows from Proposition 3.9.

(1) Guess $A \subseteq \mathcal{A}$. (2) Use a $\mathbf{VER}^{\mathcal{F}}_{\sigma}(A)$ oracle to check if A is a σ extension. If it is not, output 'no'. Otherwise continue. (3) Use an NL oracle for $\mathbf{DER}^{\mathcal{F}}$ to check that the formula under consideration is derivable (or not derivable for $\mathbf{coSA}^{\mathcal{F}}_{\sigma}(\alpha)$) from A and $\mathcal{R}$. If it is not, output 'no'. Otherwise, output 'yes'. □

Sceptical acceptance under admissible semantics is trivial.

Proposition 3.14. *$\boldsymbol{SA}^{\mathcal{F}}_{adm}(\alpha)$ is constant, with answer 'no'.*

Proof. $\emptyset$ is admissible (as in the proof of Proposition 3.11), so any sentence derivable from all admissible extensions is derivable from $\emptyset$. However, by Lemma 3.3, no such sentence exists. □

We are left to address the enumeration problem.

3.4 Extension Enumeration

We here establish the complexity of **EE** in bipolar ABA using the **MANYSOL** problem (see Section 2.2).

Proposition 3.15. *$\boldsymbol{EE}^{\mathcal{F}}_{adm}$, $\boldsymbol{EE}^{\mathcal{F}}_{prf}$ and $\boldsymbol{EE}^{\mathcal{F}}_{set\text{-}stb}$ are in* nOP*, assuming* P ≠ NP.

Proof. $\mathbf{MANYSOL}(\mathbf{EE}^{\text{AF}}_{\sigma})$ is NP-hard for $\sigma \in \{adm, prf, stb\}$ [24]. Because AFs can be mapped into flat bipolar ABA in P-time [12] and since stable and set-stable semantics coincide for flat ABA, $\mathbf{MANYSOL}(\mathbf{EE}^{\mathcal{F}}_{\sigma})$ is NP-hard for $\sigma \in \{adm, prf, set\text{-}stb\}$. □

This completes the complexity analysis of bipolar ABA as summarised in Table 2.

4 ALGORITHMS

We have shown that many of the standard problems for bipolar ABA are non-tractable. As such, practical algorithms for solving them must make use of advanced techniques and heuristics. We now propose such algorithms for the $\mathbf{EE}^{\mathcal{F}}_{\sigma}$ problem. Note that, effectively, $\mathbf{EE}^{\mathcal{F}}_{\sigma}$ answers the other standard problems too. Having all the σ extensions of $\mathcal{F}$ one can establish (non-empty) existence immediately, verification by checking membership in the set of enumerated extensions, and (credulous and sceptical) acceptance by using the efficient algorithm for derivation (cf. Proposition 3.1).

The algorithms in this section make use of a backtracking strategy. They recursively traverse a binary tree from left to right, where the root node is the empty set and the tree forks to a left (or right) node by including (or excluding) an assumption. If the current node represents a valid extension, it is added to the solution set. Backtracking occurs whenever the procedure is going down a path which will never lead to a correct solution, at this point, it moves back up the tree and takes a different path instead.

4.1 Enumeration of Preferred Extensions

We first give a basic algorithm for enumerating preferred extensions that conveys the main ideas.

In what follows a labelling is a total mapping $Lab : \mathcal{A} \to \{$IN, OUT, UNDEC, BLANK, MUST_OUT$\}$.

4.1.1 Basic Algorithm. We define some labellings which correspond to the different states of our algorithm while traversing the binary tree. These definitions and the algorithms following them are inspired by the corresponding work for AFs, particularly [28].

Definition 4.1. A labelling Lab of $\mathcal{F}$ is:

- the **initial labelling** of $\mathcal{F}$ iff $Lab = \{(\alpha, \text{BLANK}) : \alpha \in \mathcal{A} \setminus S\} \cup \{(\beta, \text{UNDEC}) : \beta \in S\}$ where $S \subseteq \mathcal{A}$ is the set of all $\gamma \in \mathcal{A}$ s.t. $\{\gamma\} \rightsquigarrow_{\mathtt{min\text{-}ABA}} Cl(\{\gamma\})$.
- a **terminal labelling** of $\mathcal{F}$ iff for each $\alpha \in \mathcal{A}$, $Lab(\alpha) \neq$ BLANK.
- a **hopeless labelling** of $\mathcal{F}$ iff there exists an $\alpha \in$ MUST_OUT s.t. for all $\beta \in \mathcal{A}$, if $\{\beta\} \rightsquigarrow_{\mathtt{min\text{-}ABA}} \{\alpha\}$ then $Lab(\beta) \in \{$OUT, UNDEC$\}$.
- an **admissible labelling** of $\mathcal{F}$ iff Lab is a terminal labelling of $\mathcal{F}$ and MUST_OUT = $\emptyset$.
- a **preferred labelling** of $\mathcal{F}$ iff Lab is an admissible labelling of $\mathcal{F}$ and $\{x : Lab(x) = \text{IN}\}$ is maximal (w.r.t. $\subseteq$) among all admissible labellings of $\mathcal{F}$.

In the above, the initial labelling corresponds to the root of the binary tree. Terminal labellings correspond to leaf nodes of the tree. If there are no MUST_OUT assumptions in a terminal labelling, then we have an admissible labelling. Preferred labellings are those which are maximally admissible. Finally, hopeless labellings are those which are guaranteed to not reach an admissible labelling.

Next, we define two procedures of our algorithm, which correspond to taking the left or right path down our binary tree.

Definition 4.2. Let Lab be a labelling of $\mathcal{F}$, and $\alpha \in \mathcal{A}$.

- The **left-transition** of Lab to the new labelling Lab' using α is defined by: (1) $Lab' \leftarrow Lab$. (2) For each $\beta \in Cl(\{\alpha\})$, $Lab'(\beta) \leftarrow IN$. (3) For each $\gamma \in \mathcal{A}$, if $\alpha \rightsquigarrow_{\texttt{min-ABA}} Cl(\{\gamma\})$, $Lab'(\gamma) \leftarrow$ OUT. (4) For each $\delta \in \mathcal{A}$, with $Lab(\delta) \neq$ OUT, if $\delta \rightsquigarrow_{\texttt{min-ABA}} Cl(\{\alpha\})$, $Lab'(\delta) \leftarrow$ MUST_OUT.
- The **right-transition** of Lab to the new labelling Lab' using α is defined by: (1) $Lab' \leftarrow Lab$. (2) For each β in $\mathcal{A}$, if $\alpha \in Cl(\{\beta\})$, $Lab'(\beta) \leftarrow UNDEC$.

A left transition starts by labelling all assumptions in the closure of some target assumption as IN. We then label any assumptions whose closure is minimally attacked by the target assumption as OUT. After that, we can add those assumptions which minimally attack the closure of the target assumptions to MUST_OUT. In a right-transition, we label all assumptions whose closure contains the target assumption as UNDEC.

Algorithm 1 enumerates all the preferred extensions of $\mathcal{F}$.

Proposition 4.3. *Algorithm 1 solves the* $\boldsymbol{EE}^{\mathcal{F}}_{prf}$ *problem.*

Proof outline. *Completeness.* Algorithm 1 builds every closed, conflict-free subset of $\mathcal{F}$. This is guaranteed by our definitions of initial labelling, left-transition and right-transition.

Soundness. We need to show that the generated sets are maximal and admissible. Maximality is ensured by line 4 together with the fact that maximal sets are constructed first (by performing left-transitions before right-transitions). For admissibility we need to show that the sets in E are closed, conflict-free and defend themselves. Closure is guaranteed because as soon as a new assumption is labelled IN, so is every element in its closure. Conflict-freeness is guaranteed since any assumption attacked by the set of IN assumptions is immediately labelled OUT. Defence is guaranteed by our usage of the MUST_OUT label and hopeless labellings. □

Figure 1 shows an example of Algorithm 1 calculating the preferred extensions of a bipolar ABA framework. The algorithm starts with the initial labelling and forks to the left and right by performing the appropriate transition procedure (represented by left and right arrows in the figure). Leaf nodes are identified as either admissible or preferred extensions (although only preferred ones are saved). Moreover, the figure shows how the notion of hopeless labellings reduces the search-space of the algorithm.

4.1.2 Improved Algorithm. We now discuss several improvements of the basic algorithm given above, similarly to [28].

Algorithm 1 can be improved by introducing influential assumptions. The idea is to select the most influential assumption for a left-transition to reach a terminal or hopeless labelling faster.

Algorithm 1: Enumerate_Preferred($\mathcal{F}$, Lab, E)

input: $\mathcal{F} = (\mathcal{L}, \mathcal{R}, \mathcal{A}, \overline{\ })$ is a bipolar ABA framework.
$Lab : \mathcal{A} \rightarrow$ {IN, OUT, UNDEC, BLANK, MUST_OUT}
$E \subseteq 2^{\mathcal{A}}$

output: $Lab : \mathcal{A} \rightarrow$ {IN, OUT, UNDEC, BLANK, MUST_OUT}
$E \subseteq 2^{\mathcal{A}}$

1 **if** *Lab is a hopeless labelling* **then return**;
2 **if** *Lab is a terminal labelling* **then**
3 **if** *Lab is an admissible labelling* **then**
4 **if** $\{x : Lab(x) = IN\}$ *is not a subset of any set in* E **then**
5 $E \leftarrow E \cup \{\{x : Lab(x) = \text{IN}\}\}$
6 **return**;
7 Select any assumption $\alpha \in \mathcal{A}$ with $Lab(\alpha) =$ BLANK;
8 Get a new labelling Lab' by applying the left-transition of Lab using x;
9 Call Enumerate_Preferred($\mathcal{F}$, Lab', E);
10 Get a new labelling Lab' using the right-transition of Lab with x; Call Enumerate_Preferred($\mathcal{F}$, Lab', E).

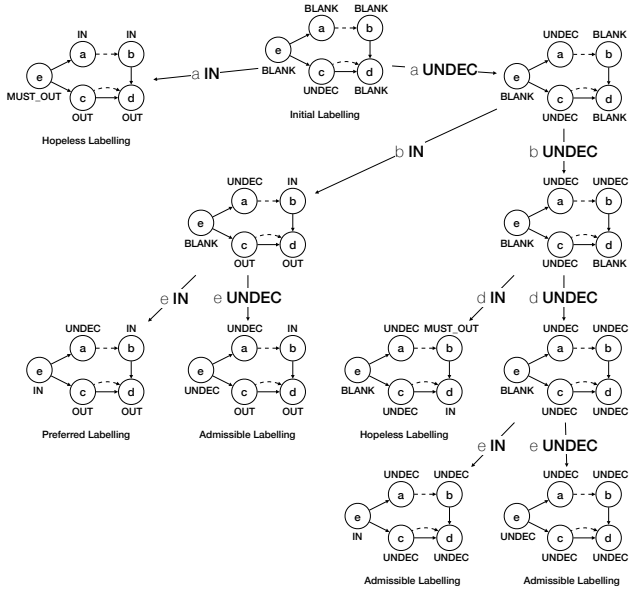

Figure 1: The behaviour of Algorithm 1 in calculating preferred extensions of a specific bipolar ABA framework. In the above, solid lines correspond to rules of the form $\overline{\alpha} \leftarrow \beta$ and dotted lines to rules of the form $\alpha \leftarrow \beta$, where $\alpha, \beta \in \mathcal{A}$.

Definition 4.4. Let Lab be a labelling of $\mathcal{F}$, and $\alpha \in (A)$ be such that $Lab(\alpha) =$ BLANK. Then α is **influential** iff for all $\beta \in \mathcal{A}$ with $Lab(\beta) =$ BLANK, $h(\alpha) \geq h(\beta)$ where $h(x)$ is defined as the number of rules in $\mathcal{R}$ which contain the assumption x in their head or body.

Another improvement comes from realising that assumptions which are minimally attacked only by assumptions labelled OUT or MUST_OUT have to be labelled IN if the labelling is to evolve

into a preferred one. This is because they must be defended by any admissible set reachable from the current labelling.

Definition 4.5. Let *Lab* be a labelling of $\mathcal{F}$. Then $\alpha \in \mathcal{A}$ is a **must_in assumption** iff $Lab(\alpha)$ = BLANK and for all $\beta \in \mathcal{A}$ where $\beta \rightsquigarrow_{\texttt{min-ABA}} \{\alpha\}$, $Lab(\alpha) \in$ {OUT, MUST_OUT}. The **labelling propagation** of *Lab* consists of the following actions: (1) If there is no must_in assumption, halt. (2) Pick a must_in assumption α. (3) Do $Lab(\alpha) \leftarrow$ IN. (4) For each $\beta \in Cl(\{\alpha\})$, do $Lab(\beta) \leftarrow IN$. (5) For each $\gamma \in \mathcal{A}$, if $\alpha \rightsquigarrow_{\texttt{min-ABA}} Cl(\{\gamma\})$, do $Lab(\gamma) \leftarrow$ OUT. (6) For each $\delta \in \mathcal{A}$ with $Lab(\delta) \neq$ OUT, if $\delta \rightsquigarrow_{\texttt{min-ABA}} Cl(\{\alpha\})$, do $Lab'(\delta) \leftarrow$ MUST_OUT. (7) Return to step 1.

We now propose Algorithm 2 which adds the following improvements to Algorithm 1. (1) The left transition is performed using the most influential assumption. (2) The recursive call after a right transition is replaced with a while loop structure. (3) Hopeless labellings are checked for every time a labelling changes. (4) Labelling propagation is added to the start of the algorithm.

Algorithm 2: Enumerate_Preferred($\mathcal{F}$, *Lab*, *E*)

input : $\mathcal{F} = (\mathcal{L}, \mathcal{R}, \mathcal{A}, \overline{\ })$ is a bipolar ABA framework.
$Lab : \mathcal{A} \rightarrow$ {IN, OUT, UNDEC, BLANK, MUST_OUT}
$E \subseteq 2^{\mathcal{A}}$
output : $Lab : \mathcal{A} \rightarrow$ {IN, OUT, UNDEC, BLANK, MUST_OUT}
$E \subseteq 2^{\mathcal{A}}$

```
1  Propagate Lab;
2  if Lab is a hopeless labelling then return;
3  while Lab is not a terminal labelling do
4      Select a new assumption α ∈ A s.t. α is influential;
5      Get a new labelling called Lab' by applying the
         left-transition of Lab using α;
6      if Lab' is not a hopeless labelling then
7          Call Enumerate_Preferred(F, Lab', E);
8      Update Lab by applying the right-transition of Lab
         using α;
9      if Lab is a hopeless labelling then return;
10 if Lab is an admissible labelling then
11     if {x : Lab(x) = IN} is not a subset of any set in E then
12         E ← E ∪ {{x : Lab(x) = IN}}.
```

Proposition 4.6. *Algorithm 2 solves the* $\mathbf{EE}^{\mathcal{F}}_{prf}$ *problem.*

Proof outline. We show that none of the changes introduced in Algorithm 2 compromise soundness or completeness. (1) Selecting the most influential assumption does not compromise left and right transitions, because by definition this assumption will be labelled BLANK. (2) Changing the right transition to be performed as a while loop doesn't change the order of operations. (3) Checking for hopeless labellings does not have any side effects, so doing it more often will not either. (4) Labelling propagation excludes only admissible labellings which are not preferred. □

4.2 Enumeration of Admissible and Set-Stable Extensions

We next give algorithms for enumerating admissible and set-stable extensions of $\mathcal{F}$.

4.2.1 Enumeration of Admissible Extensions. Algorithm 2 can be adapted to find admissible extensions. To achieve this, we first need to drop the maximality check. Moreover, the labelling propagation step needs to be removed. Indeed, if we do not remove it, then there is a risk that some admissible sets will be overlooked since these sets do not necessarily contain every assumption that they defend. This modification is achieved in Algorithm 3.

Algorithm 3: Enumerate_Admissible($\mathcal{F}$, *Lab*, *E*)

input : $\mathcal{F} = (\mathcal{L}, \mathcal{R}, \mathcal{A}, \overline{\ })$ is a bipolar ABA framework.
$Lab : \mathcal{A} \rightarrow$ {IN, OUT, UNDEC, BLANK, MUST_OUT}
$E \subseteq 2^{\mathcal{A}}$
output : $Lab : \mathcal{A} \rightarrow$ {IN, OUT, UNDEC, BLANK, MUST_OUT}
$E \subseteq 2^{\mathcal{A}}$

```
1  if Lab is a hopeless labelling then return;
2  while Lab is not a terminal labelling do
3      Select a new assumption α ∈ A s.t. α is influential;
4      Get a new Lab called Lab' by applying the
         left-transition of Lab using α;
5      if Lab' is not a hopeless labelling then
6          Call Enumerate_Admissible(F, Lab', E);
7      Update Lab by applying the right-transition of Lab
         using α;
8      if Lab is a hopeless labelling then return;
9  if Lab is an admissible labelling then
10     E ← E ∪ {{x : Lab(x) = IN}}
```

Therefore, we have the following result:

Proposition 4.7. *Algorithm 3 solves the* $\mathbf{EE}^{\mathcal{F}}_{adm}$ *problem.*

4.2.2 Enumeration of Set-Stable Extensions. Algorithm 2 can also be adapted to find set-stable extensions. By definition any set-stable extension attacks the closure of all assumption sets it does not contain. Thus, the UNDEC label is no longer useful since any assumption which would have been labelled UNDEC in the case of preferred semantics should now be labelled MUST_OUT. We change some of our definitions accordingly.

Definition 4.8. Let *Lab* be a labelling of $\mathcal{F}$. Then:

- *Lab* is the **initial set-stable labelling** of $\mathcal{F}$ iff $Lab = \{(\alpha, \text{BLANK}) : \alpha \in \mathcal{A} \setminus S\} \cup \{(\beta, \text{MUST\_OUT}) : \beta \in S\}$ where $S \subseteq \mathcal{A}$ is the set of all $\gamma \in \mathcal{A}$ s.t. $\{\gamma\} \rightsquigarrow_{\texttt{min-ABA}} Cl(\{\gamma\})$.
- Let α be an assumption in $\mathcal{A}$. Then the **set-stable right-transition** of *Lab* to the new labelling *Lab'* using α is defined by actions: (1) $Lab' \leftarrow Lab$. (2) For each $\delta \in Cl(\{\alpha\})$ with $Lab(\delta) \neq$ OUT, $Lab'(\delta) \leftarrow$ MUST_OUT.
- *Lab* is a **set-stable labelling** of $\mathcal{F}$ iff *Lab* is a terminal labelling of $\mathcal{F}$ and MUST_OUT = $\emptyset$.

The modifications are achieved in Algorithm 4. Thus, as with enumeration of admissible extensions, we have the following result:

Proposition 4.9. *Algorithm 4 solves the* $\boldsymbol{EE}^{\mathcal{F}}_{\textit{set-stb}}$ *problem.*

Algorithm 4: Enumerate_Set-stable($\mathcal{F}$, *Lab*, *E*)

input : $\mathcal{F} = (\mathcal{L}, \mathcal{R}, \mathcal{A}, \overline{\ })$ is a bipolar ABA framework.
Lab : $\mathcal{A} \rightarrow$ {IN, OUT, BLANK, MUST_OUT}
$E \subseteq 2^{\mathcal{A}}$

output : *Lab* : $\mathcal{A} \rightarrow$ {IN, OUT, BLANK, MUST_OUT}
$E \subseteq 2^{\mathcal{A}}$

```
Propagate Lab;
if Lab is a hopeless labelling then return;
while Lab is not a terminal labelling do
    Select a new assumption α ∈ A s.t. α is influential;
    Get a new labelling called Lab' by applying the
      left-transition of Lab using α;
    if Lab' is not a hopeless labelling then
        Call Enumerate_Set-stable(F, Lab', E);
    Update Lab by applying the right-transition-set-stable
      of Lab using α;
    if Lab is a hopeless labelling then return;
if Lab is a set-stable labelling then
    E ← E ∪ {{x : Lab(x) = IN}}
```

We note that while conceptually similar algorithms exist for enumerating extensions of AFs (see e.g. [8, 27, 28]), adapting them to be used for bipolar ABA frameworks is not a trivial task. Specifically, the algorithms described in this section for admissible and preferred semantics are more complex than existing ones for AFs because extensions in bipolar ABA need to be closed, and the notion of defence in ABA is more involved than the corresponding notion of acceptance in abstract argumentation. Moreover, the ideas relating to set-stable labellings (Definition 4.8) are new, as is Algorithm 4.

5 IMPLEMENTATION AND EVALUATION

We now discuss our implementation of the algorithms mentioned in the previous section. In addition to directly enumerating extensions of bipolar ABA frameworks, our system is also capable of calculating extensions of other argumentation frameworks (particularly AFs and BAFs (as defined in [6, 20, 29]) by utilising extension-preserving mappings from these formalisms into bipolar ABA as discussed in [12]. This makes our tool more versatile than existing systems [8], almost none of which calculate extensions of BAFs.

5.1 Implementation

We now describe the control flow of our system as depicted graphically in Figure 2.

(1) **Input argumentation framework.** The user inputs an argumentation framework to the system and specifies which semantics they would like the system to calculate extensions under. The argumentation framework can be an AF, one of the various BAFs, or a bipolar ABA framework.
(2) **Parse argumentation framework.** The system parses, and generates an internal representation of, the input framework.
(3) **Perform standard mapping.** If the input framework is not a bipolar ABA framework, the system transforms it into a bipolar ABA framework using the mappings defined in [12].
(4) **Perform labelling algorithms.** The bipolar ABA framework is inputted to the labelling algorithms defined in section 4.
(5) **Output Extensions.** Our system terminates after outputting the extensions calculated by the labelling algorithms.

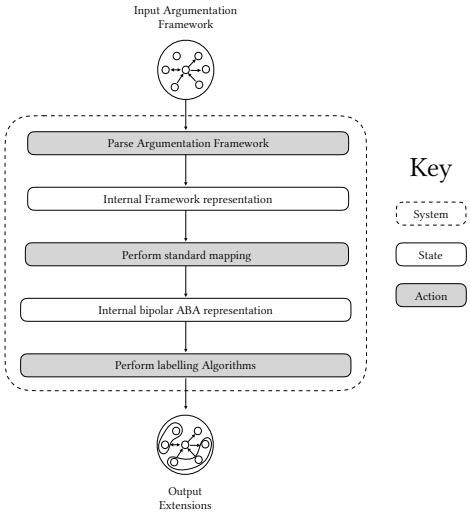

Figure 2: Control flow diagram of the system for computing extensions of bipolar argumentation frameworks.

5.1.1 Evaluation. In order to test the scalability of our system, we generated 405 bipolar ABA frameworks of increasing size. To do this we adapted an existing benchmark generator from [10], originally used to create flat ABA frameworks, and ensured that bipolar ABA frameworks are generated instead.

We input a tuple of parameters $(N_s, N_a, N_{rh}, N_{rph})$ to the generator in order to create our frameworks. The parameters are defined as follows: (1) N_s is the total number of sentences in the framework, i.e., $|\mathcal{L}|$. (2) N_a is the number of assumptions, i.e., $|\mathcal{A}|$, given as a percentage of the number of sentences. (3) N_{rh} is the number of distinct sentences to be used as rule heads, given as an integer. (4) N_{rph} is the number of rules per distinct rule head, given as an interval [min, max], where min and max are integers.

The specific parameters used were $(N_s, 37\%, N_s/2, [2, N_s/8])$ with the value of N_s starting at 16, and increasing by 8 between subsequent frameworks. The largest framework consisted of 3248 sentences, 1202 assumptions and 174,365 rules.

We measured the elapsed time between inputting a framework and outputting its extensions, under the admissible, preferred and set-stable semantics, for all generated frameworks. The elapsed times were very similar for all three semantics. Figure 3 shows the time taken to calculate extensions for each framework, averaged over the three semantics. These experiments were run on a home machine, with 16GB of memory and a 2.9GHZ, 2 core CPU.

The results show that even for the largest frameworks considered, our algorithms calculate extensions in under 25 seconds. We do see the performance begin to deteriorate as the size of the frameworks increase. This is expected since the backtracking method

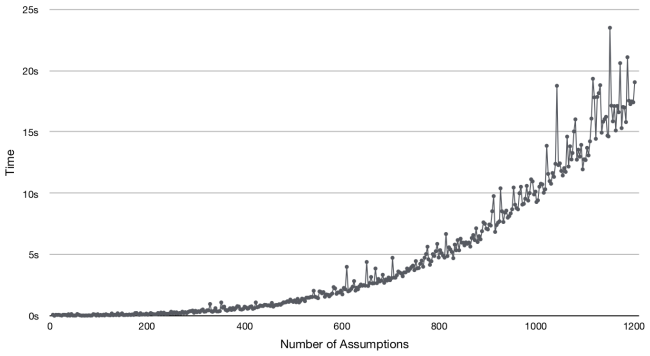

Figure 3: Time taken to calculate extensions of generated bipolar ABA frameworks (averaged over semantics).

we rely on operates in $\mathcal{O}(2^n)$ in the worst case. Overall, these results demonstrate the feasibility of our algorithms as a means of generating extensions of large argumentation frameworks.

All in all, we have presented a scalable system for computing and enumerating all extensions of bipolar ABA frameworks under the semantics considered in this paper. Consequently, the system computes and enumerates extensions of various formulations of BAFs. In addition, it allows to answer questions to all the standard complexity problems considered in this paper.

6 RELATED WORK

In [19], the authors studied the complexity of BAFs under deductive support as in [6]. Specifically, Fazzinga et al. analysed the verification problem **VER** and established that it is in P under admissible, stable, complete and grounded semantics, and in coNP under preferred semantics. We instead studied the complexity of bipolar ABA, and thus indirectly of BAFs not only under deductive support, but also under other interpretations of support and with diverging semantics, as captured in bipolar ABA [12]. In addition, we analysed all the complexity problems standard in argumentation, namely **EX**, **NE**, **VER**, **CA**, **SA** and **EE**. To our knowledge, these problems have not been investigated for BAFs, except for the work of Fazzinga et al.. We restricted our study to the admissible, preferred and set-stable semantics of bipolar ABA used to capture various BAFs, but we will extend our analysis to other semantics in the future.

Complexity of ABA was investigated in [13]. Dimopoulos et al. studied general non-flat ABA with respect to the complexity of the derivation problem in the underlying deductive system of an ABA framework, as well as various instances of ABA, including the (flat) logic programming instance, called LP-ABA. Specifically, they established the generic upper bounds for **VER**, as well as both upper bounds and instance-specific lower bounds for **CA** and **SA** under admissible, preferred and stable semantics. We note that **DER** in LP-ABA belongs to P [13], and AFs can be mapped in P-time into LP-ABA [37]. Thus, the results proven in this paper apply to LP-ABA as well. In particular, results provided in Section 3 complement the original work of Dimopoulos et al. on LP-ABA by giving new lower bounds for **EX**, **NE** and **VER** problems.

The Tweety libraries [36] provide implementations of various argumentation formalisms including AFs and ABA. Tweety can enumerate extensions of ABA frameworks under five semantics, including admissible, preferred and stable, but not set-stable semantics. Tweety essentially takes a brute force approach. For example, to compute the preferred extensions, it first generates all possible sets of assumptions and checks which ones are admissible. It then iterates through all these and checks which are maximal. This is very slow, as witnessed e.g. in a framework with ten assumptions, where Tweety takes more than 5 minutes to calculate extensions. In contrast, we showed our algorithms to be efficient in situations with hundreds of assumptions (on the same hardware).

In [17], Egly et al. provide an implementation of deductive BAFs [6], but not of other approaches to BAFs. Their system reduces the problems to instances of answer set programming whereas ours works by directly calculating extensions. There are also other implementations of structured argumentation formalisms (see [7] for a recent survey), and those relevant to ABA (e.g. [21–23]) are reviewed in [2]. Except for the Tweety libraries discussed above, to the best of our knowledge no other implementations of non-flat ABA in general, or bipolar ABA in particular, exist.

7 CONCLUSIONS AND FUTURE WORK

In this paper, we established the computational complexity of six problems, namely (non-empty) existence, verification, (credulous and sceptical) acceptance and enumeration, for bipolar Assumption-Based Argumentation (ABA) under the admissible, preferred and set-stable semantics. Our results carry over to various Bipolar Argumentation Frameworks (BAFs) that are instances of bipolar ABA. We also provided novel algorithms for extension enumeration, consequently addressing the remaining problems, for bipolar ABA. Using these algorithms, we gave an implementation of bipolar ABA and various BAFs, and showed that it scales well. We have therefore provided solid theoretical foundations and realised an implementation underlying the practical deployment of bipolar argumentation.

In the future, we plan on extending our analysis to generalisations of bipolar ABA. We will explore whether empowering these frameworks with new capabilities, such as support for factual rules or rules with multiple elements in their body, will lead to an increase in complexity. Moreover, we plan to extend our labelling algorithms to work for all ABA frameworks. Such algorithms will find use in an even wider range of practical scenarios than those described in this paper, due to the higher expressive power of generic ABA.

Acknowledgements. The authors were supported by the EPSRC project **EP/P029558/1** ROAD2H: *Resource Optimisation, Argumentation, Decision Support and Knowledge Transfer to Create Value via Learning Health Systems.*

Data access statement: All data created during this research is available at github.com/AminKaram/BipolarABASolver. For more information please contact Amin Karamlou at mak514@ic.ac.uk.

REFERENCES

[1] Leila Amgoud and Mathieu Serrurier. 2008. Agents that Argue and Explain Classifications. *Autonomous Agents and Multi-Agent Systems* 16, 2 (2008), 187–209. https://doi.org/10.1007/s10458-007-9025-6

[2] Ziyi Bao, Kristijonas Čyras, and Francesca Toni. 2017. ABAplus: Attack Reversal in Abstract and Structured Argumentation with Preferences. In *PRIMA 2017: Principles and Practice of Multi-Agent Systems - 20th International Conference (Lecture Notes in Computer Science)*, Bo An, Ana L. C. Bazzan, João Leite, Serena

Villata, and Leendert van der Torre (Eds.). Springer, Nice, 420–437. https://doi.org/10.1007/978-3-319-69131-2_25

[3] Pietro Baroni, Serena Borsato, Antonio Rago, and Francesca Toni. 2018. The "Games of Argumentation" Web Platform. In *Computational Models of Argument - Proceedings of COMMA 2018, Warsaw, Poland, 12-14 September 2018*. 447–448. https://doi.org/10.3233/978-1-61499-906-5-447

[4] Andrei Bondarenko, Phan Minh Dung, Robert Kowalski, and Francesca Toni. 1997. An Abstract, Argumentation-Theoretic Approach to Default Reasoning. *Artificial Intelligence* 93, 97 (1997), 63–101. https://doi.org/10.1016/S0004-3702(97)00015-5

[5] Álvaro Carrera and Carlos Iglesias. 2015. A Systematic Review of Argumentation Techniques for Multi-Agent Systems Research. *Artificial Intelligence Review* 44, 4 (2015), 509–535. https://doi.org/10.1007/s10462-015-9435-9

[6] Claudette Cayrol and Marie-Christine Lagasquie-Schiex. 2013. Bipolarity in argumentation graphs: Towards a better understanding. *Int. J. Approx. Reasoning* 54, 7 (2013), 876–899. https://doi.org/10.1016/j.ijar.2013.03.001

[7] Federico Cerutti, Sarah Alice Gaggl, Matthias Thimm, and Johannes Peter Wallner. 2017. Foundations of Implementations for Formal Argumentation. *FLAP* 4, 8 (2017). http://www.collegepublications.co.uk/downloads/ifcolog00017.pdf

[8] Günther Charwat, Wolfgang Dvorák, Sarah Alice Gaggl, Johannes Peter Wallner, and Stefan Woltran. 2015. Methods for solving reasoning problems in abstract argumentation - A survey. *Artif. Intell.* 220 (2015), 28–63. https://doi.org/10.1016/j.artint.2014.11.008

[9] Andrea Cohen, Sebastian Gottifredi, Alejandro Javier García, and Guillermo Ricardo Simari. 2014. A survey of different approaches to support in argumentation systems. *Knowledge Eng. Review* 29, 5 (2014), 513–550. https://doi.org/10.1017/S0269888913000325

[10] Robert Craven and Francesca Toni. 2016. Argument graphs and assumption-based argumentation. *Artif. Intell.* 233 (2016), 1–59. https://doi.org/10.1016/j.artint.2015.12.004

[11] Kristijonas Čyras, Xiuyi Fan, Claudia Schulz, and Francesca Toni. 2018. Assumption-Based Argumentation: Disputes, Explanations, Preferences. In *Handbook Of Formal Argumentation*, Pietro Baroni, Dov M Gabbay, Massimiliano Giacomin, and Leendert van der Torre (Eds.). Vol. 1. College Publications.

[12] Kristijonas Čyras, Claudia Schulz, and Francesca Toni. 2017. Capturing Bipolar Argumentation in Non-flat Assumption-Based Argumentation. In *PRIMA 2017: Principles and Practice of Multi-Agent Systems - 20th International Conference (Lecture Notes in Computer Science)*, Bo An, Ana L. C. Bazzan, João Leite, Serena Villata, and Leendert van der Torre (Eds.). Springer, Nice, 386–402. https://doi.org/110.1007/978-3-319-69131-2_23

[13] Yannis Dimopoulos, Bernhard Nebel, and Francesca Toni. 2002. On the computational complexity of assumption-based argumentation for default reasoning. *Artif. Intell.* 141, 1/2 (2002), 57–78. https://doi.org/10.1016/S0004-3702(02)00245-X

[14] Phan Minh Dung. 1995. On the Acceptability of Arguments and its Fundamental Role in Nonmonotonic Reasoning, Logic Programming and n-Person Games. *Artif. Intell.* 77, 2 (1995), 321–358. https://doi.org/10.1016/0004-3702(94)00041-X

[15] Paul E. Dunne and Michael Wooldridge. 2009. Complexity of Abstract Argumentation. In *Argumentation in Artificial Intelligence*. 85–104. https://doi.org/10.1007/978-0-387-98197-0_5

[16] Wolfgang Dvorák and Paul E. Dunne. 2017. Computational Problems in Formal Argumentation and their Complexity. *FLAP* 4, 8 (2017). http://www.collegepublications.co.uk/downloads/ifcolog00017.pdf

[17] Uwe Egly, Sarah Alice Gaggl, and Stefan Woltran. 2010. Answer-set programming encodings for argumentation frameworks. *Argument & Computation* 1, 2 (2010), 147–177. https://doi.org/10.1080/19462166.2010.486479

[18] Xiuyi Fan and Francesca Toni. 2012. Agent Strategies for ABA-based Information-Seeking and Inquiry Dialogues. In *20th European Conference on Artificial Intelligence (Frontiers in Artificial Intelligence and Applications)*, Luc De Raedt, Christian Bessière, Didier Dubois, Patrick Doherty, Paolo Frasconi, Fredrik Heintz, and Peter J. F. Lucas (Eds.), Vol. 242. IOS Press, Montpelier, 324–329. https://doi.org/10.3233/978-1-61499-098-7-324

[19] Bettina Fazzinga, Sergio Flesca, and Filippo Furfaro. 2018. Probabilistic Bipolar Abstract Argumentation Frameworks: Complexity Results. In *27th International Joint Conference on Artificial Intelligence*. International Joint Conferences on Artificial Intelligence Organization, Stockholm, 1803–1809. https://doi.org/10.24963/ijcai.2018/249

[20] Dov M Gabbay. 2016. Logical Foundations for Bipolar and Tripolar Argumentation Networks: Preliminary Results. *Journal of Logic and Computation* 26, 1 (2016), 247–292. https://doi.org/10.1093/logcom/ext027

[21] Alejandro Javier García and Guillermo Ricardo Simari. 2014. Defeasible Logic Programming: DeLP-servers, Contextual Queries, and Explanations for Answers. *Argument & Computation* 5, 1 (2014), 63–88. https://doi.org/10.1080/19462166.2013.869767

[22] Thomas F Gordon and Douglas Walton. 2016. Formalizing Balancing Arguments. In *Computational Models of Argument (Frontiers in Artificial Intelligence and Applications)*, Pietro Baroni, Thomas F Gordon, Tatjana Scheffler, and Manfred Stede (Eds.), Vol. 287. IOS Press, Potsdam, 327–338. https://doi.org/10.3233/978-1-61499-686-6-327

[23] Antonis C Kakas and Pavlos Moraitis. 2003. Argumentation Based Decision Making for Autonomous Agents. In *2nd International Joint Conference on Autonomous Agents & Multiagent Systems*. ACM Press, Melbourne, 883–890. https://doi.org/10.1145/860575.860717

[24] Markus Kröll, Reinhard Pichler, and Stefan Woltran. 2017. On the Complexity of Enumerating the Extensions of Abstract Argumentation Frameworks. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*. 1145–1152. https://doi.org/10.24963/ijcai.2017/159

[25] Peter McBurney and Simon Parsons. 2009. Dialogue Games for Agent Argumentation. In *Argumentation in Artificial Intelligence*, Guillermo Ricardo Simari and Iyad Rahwan (Eds.). Springer, Chapter 13, 261–280. https://doi.org/10.1007/978-0-387-98197-0_13

[26] Stefano Menini, Elena Cabrio, Sara Tonelli, and Serena Villata. 2018. Never Retreat, Never Retract: Argumentation Analysis for Political Speeches. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*. 4889–4896. https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16393

[27] Samer Nofal, Katie Atkinson, and Paul E. Dunne. 2014. Algorithms for decision problems in argument systems under preferred semantics. *Artif. Intell.* 207 (2014), 23–51. https://doi.org/10.1016/j.artint.2013.11.001

[28] Samer Nofal, Katie Atkinson, and Paul E. Dunne. 2016. Looking-ahead in backtracking algorithms for abstract argumentation. *Int. J. Approx. Reasoning* 78 (2016), 265–282. https://doi.org/10.1016/j.ijar.2016.07.013

[29] Farid Nouioua and Vincent Risch. 2010. Bipolar Argumentation Frameworks with Specialized Supports. In *22nd IEEE International Conference on Tools with Artificial Intelligence*, Vol. 1. IEEE, Arras, 215–218. https://doi.org/10.1109/ICTAI.2010.37

[30] Santi Ontañón and Enric Plaza. 2007. An Argumentation-Based Framework for Deliberation in Multi-Agent Systems. In *Argumentation in Multi-Agent Systems: 4th International Workshop, ArgMAS*, Iyad Rahwan, Simon Parsons, and Chris Reed (Eds.). Springer, Honolulu, 178–196. https://doi.org/10.1007/978-3-540-78915-4-12

[31] Christos H. Papadimitriou. 1994. *Computational complexity*. Addison-Wesley.

[32] Simon Parsons, Carles Sierra, and Nick Jennings. 1998. Agents that reason and negotiate by arguing. *Journal of Logic and Computation* 8, 3 (1998), 261–292.

[33] Henry Prakken and Giovanni Sartor. 1998. Modelling Reasoning with Precedents in a Formal Dialogue Game. *Artificial Intelligence and Law* 6, 2-4 (1998), 231–287. https://doi.org/10.1023/A:1008278309945

[34] Antonio Rago and Francesca Toni. 2017. Quantitative Argumentation Debates with Votes for Opinion Polling. In *PRIMA 2017: Principles and Practice of Multi-Agent Systems - 20th International Conference, Nice, France, October 30 - November 3, 2017, Proceedings*. 369–385. https://doi.org/10.1007/978-3-319-69131-2_22

[35] Antonio Rago, Francesca Toni, Marco Aurisicchio, and Pietro Baroni. 2016. Discontinuity-Free Decision Support with Quantitative Argumentation Debates. In *Principles of Knowledge Representation and Reasoning: Proceedings of the Fifteenth International Conference, KR 2016, Cape Town, South Africa, April 25-29, 2016*. 63–73. http://www.aaai.org/ocs/index.php/KR/KR16/paper/view/12874

[36] Matthias Thimm. 2017. The Tweety Library Collection for Logical Aspects of Artificial Intelligence and Knowledge Representation. *KI* 31, 1 (2017), 93–97. https://doi.org/10.1007/s13218-016-0458-4

[37] Francesca Toni. 2012. Reasoning on the Web with Assumption-Based Argumentation. In *Reasoning Web. Semantic Technologies for Advanced Query Answering - 8th International Summer School 2012, Vienna, Austria, September 3-8, 2012. Proceedings*. 370–386. https://doi.org/10.1007/978-3-642-33158-9_10

[38] Francesca Toni. 2014. A Tutorial on Assumption-Based Argumentation. *Argument & Computation* 5, 1 (2014), 89–117. https://doi.org/10.1080/19462166.2013.869878