Journal of Artificial Intelligence Research 71 (2021) 265-318 Submitted 10/2020; published 06/2021

Declarative Algorithms and Complexity Results for Assumption-Based Argumentation

Tuomo Lehtonen TUOMO.LEHTONEN@HELSINKI.FI
Helsinki Institute for Information Technology HIIT,
Department of Computer Science, University of Helsinki, Finland

Johannes P. Wallner WALLNER@IST.TUGRAZ.AT
Institute of Software Technology, Graz University of Technology, Austria

Matti Järvisalo MATTI.JARVISALO@HELSINKI.FI
Helsinki Institute for Information Technology HIIT,
Department of Computer Science, University of Helsinki, Finland

Abstract

The study of computational models for argumentation is a vibrant area of artificial intelligence and, in particular, knowledge representation and reasoning research. Arguments most often have an intrinsic structure made explicit through derivations from more basic structures. Computational models for structured argumentation enable making the internal structure of arguments explicit. Assumption-based argumentation (ABA) is a central structured formalism for argumentation in AI. In this article, we make both algorithmic and complexity-theoretic advances in the study of ABA. In terms of algorithms, we propose a new approach to reasoning in a commonly studied fragment of ABA (namely the logic programming fragment) with and without preferences. While previous approaches to reasoning over ABA frameworks apply either specialized algorithms or translate ABA reasoning to reasoning over abstract argumentation frameworks, we develop a direct declarative approach to ABA reasoning by encoding ABA reasoning tasks in answer set programming. We show via an extensive empirical evaluation that our approach significantly improves on the empirical performance of current ABA reasoning systems. In terms of computational complexity, while the complexity of reasoning over ABA frameworks is well-understood, the complexity of reasoning in the ABA$^+$ formalism integrating preferences into ABA is currently not fully established. Towards bridging this gap, our results suggest that the integration of preferential information into ABA via so-called reverse attacks results in increased problem complexity for several central argumentation semantics.

1. Introduction

The study of computational models of argumentation is a vibrant area of artificial intelligence research, in particular in the field of knowledge representation and reasoning (Baroni et al., 2018). The aim is to draw conclusions from internally inconsistent or incomplete knowledge bases using formalisms in which reasoning can be done algorithmically. In contrast to classical logic, for example, formal argumentation represents defeasible reasoning, where new information may revoke previously reached conclusions. Indeed, as knowledge representation formalisms, computational models of argumentation capture various related paradigms, including forms of non-monotonic reasoning and logic programming (LP) (Dung, 1995). In terms of applications, formal argumentation has been shown to be relevant in various different settings (Atkinson et al., 2017), including legal (Prakken &

Sartor, 2015), e-government (Atkinson et al., 2006), and medical applications (Craven et al., 2012; Čyras et al., 2020), as well as multi-agent systems (Fan et al., 2014).

Two main types of computational models of argumentation are the so-called *structured* and *abstract* formalisms. In abstract argumentation the structure of individual arguments is completely abstract. Reasoning over abstract argumentation frameworks (AFs) (Dung, 1995) is restricted to the level of pair-wise knowledge of attacks between conflicting atomic arguments.

In contrast to AFs, in structured argumentation formalisms the structure of arguments is explicit. Arguments are constructed from atomic elements of a given formalism, usually a form of premises and a deductive system used for deriving facts (Besnard et al., 2014). Attacks between arguments also depend on the structure of the arguments and are constructed from other elements of the formalism, such as the notion of what contradicts a specific premise. In this way, structured argumentation formalisms provide approaches to representing arguments inferred from a knowledge base. The explicitness of the structure of arguments and attacks makes the complexity of reasoning in structured argumentation formalisms in many cases higher than in abstract argumentation (Dvořák & Dunne, 2018). While noticeable attention has recently been paid to computational approaches for AFs (as recently surveyed by Charwat et al., 2015; Cerutti et al., 2018), advancing understanding of the complexity of and algorithms for reasoning over structured argumentation frameworks has received less attention. Indeed, theoretical and algorithmic advances for computational models for structured argumentation can be perceived as more challenging, as the internal structure of arguments being explicit results in more complex formalisms compared to Dung's abstract frameworks.

Various different structured argumentation formalisms have been proposed, with different characteristics and properties, and thereby also being suited to some extent for different application scenarios. Prominent structured formalisms include assumption-based argumentation (ABA) (Bondarenko et al., 1997; Čyras et al., 2018; Dung et al., 2009; Toni, 2014), ASPIC$^+$ (Modgil & Prakken, 2013, 2018; Prakken, 2010), defeasible logic programming (DeLP) (García & Simari, 2004, 2014, 2018), deductive argumentation (Besnard & Hunter, 2008, 2018), and Carneades (Gordon et al., 2007). Different formalisms offer ways of integrating different types of preferential information, either directly as a built-in feature (as in the case of e.g. ASPIC$^+$) or as different types of extensions of the base formalism (as in the case of ABA$^+$ (Čyras, 2017; Bao et al., 2017; Čyras & Toni, 2016c, 2016a, 2016b) and p_ABA (Wakaki, 2017a, 2017b), two proposed approaches to accommodating preferential information within ABA, with different foreseeable application scenarios). This is motivated by the fact that preferences can be considered an important part of argumentative reasoning as they make it possible to include additional qualifying information about the situation that is being modelled, such as the plausibility of a fact or the wishes of an agent. An example of the latter is a study where a variant of ABA$^+$ was used to determine a set of recommendations to follow from clinical guidelines (Čyras et al., 2020). In this setting, preferences were used to represent the wishes of patients about their treatment: a patient might wish, for example, to avoid intense exercise. Then a treatment without intense exercise is chosen over ones involving it, if possible.

Our focus in this article is on ABA and its extension ABA$^+$; we delay further discussion on other formalisms until Section 6 focusing on related work. As one of the prominent structured argumentation formalisms, real-world scenarios of applications of ABA include medical decision making (Craven et al., 2012; Čyras et al., 2020), decision making in a multi-agent context (Fan et al., 2014), and game theory (Fan & Toni, 2016).

An ABA framework consists of assumptions, rules, sentences and contraries. Deductions of sentences are made from assumptions via the rules, and assumptions can be attacked by deducing

the contrary of the assumptions, contraries being sentences. In particular, we consider a commonly-studied logic programming fragment of ABA (Bondarenko et al., 1997). Attacks are defined between sets of assumptions: if a set of assumptions derives the contrary of some assumption, the set of assumptions attacks this assumption and all assumption sets containing the attacked assumption. Whereas in abstract argumentation sets of *arguments* are jointly accepted according to semantics, in ABA the semantics sanction acceptable sets of *assumptions*[1]. ABA$^+$ is a natural extension of ABA enabling modelling preferential information over assumptions (Čyras, 2017; Bao et al., 2017; Čyras & Toni, 2016c, 2016a, 2016b). Similarly to abstract argumentation, ABA has various semantics. An acceptable set of assumptions is such that the assumptions together do not derive a contrary of any of the assumptions in the set, and in addition fulfill some further criteria enforced by the chosen semantics.

As the main contributions of this article, we make both algorithmic and complexity-theoretic advances in the study of ABA and ABA$^+$ focusing on the commonly-studied logic programming fragment (Bondarenko et al., 1997).

On the algorithmic side, we propose a new computational approach to reasoning in the considered LP fragment of ABA with and without preferences. Previous approaches to reasoning in ABA are based on either specialized algorithms or translating ABA reasoning to reasoning over AFs. The dispute derivation approach (Gaertner & Toni, 2007a, 2007b, 2008; Dung et al., 2007; Craven et al., 2012; Toni, 2013; Craven et al., 2013) can answer credulous queries under admissible, grounded and ideal semantics. Dispute derivations for admissible and grounded semantics are implemented in the ABAGRAPH system (Craven & Toni, 2016). The translation-based approach (Dung et al., 2007; Caminada et al., 2013) implemented in the ABA2AF system (Lehtonen et al., 2017) transforms ABA frameworks to AFs, allowing for the use of AF solvers to query credulous and skeptical acceptance under admissible, preferred and stable semantics. Finally, the ABAPLUS system (Bao et al., 2017) for ABA$^+$ uses a similar translation, but takes preferences into account in the translation, and supports enumeration of acceptable assumption sets under complete, preferred, stable, grounded and ideal semantics. In contrast, we propose a "direct" declarative approach to ABA reasoning based on encoding ABA reasoning tasks directly in answer set programming (ASP) (Gelfond & Lifschitz, 1988; Niemelä, 1999; Brewka et al., 2011), without first translating ABA reasoning to reasoning over AFs. Motivated by the success of ASP encodings in reasoning over AFs in practice (Toni & Sergot, 2011; Egly & Woltran, 2006; Egly et al., 2008; Nieves et al., 2008; Wakaki & Nitta, 2008; Egly et al., 2010; Gaggl et al., 2015), we present novel ASP encodings for ABA and for ABA$^+$. Our focus is in particular on combinations of semantics and reasoning modes which give rise to NP-hard decision problems. Harnessing the power and capabilities of ASP solvers, we are able to cover various semantics (for ABA, admissible, complete, preferred, stable, grounded and ideal; for ABA$^+$, stable and grounded) and reasoning modes (credulous and skeptical reasoning as well as finding and enumerating acceptable assumption sets) in a relatively uniform manner. We also provide results from an extensive empirical evaluation of the approach, comparing its runtime performance with the earlier proposed algorithmic approaches to reasoning in ABA and ABA$^+$ for which implementations are available. Our approach significantly improves on the empirical performance of currently available reasoning systems, in particular approaches based on dispute derivations or translations to AFs. Our implementation is available at `https://bitbucket.org/coreo-group/aspforaba`.

1. A formulation that is equivalent for central ABA semantics uses acceptable sets of arguments instead of assumptions, where an argument is a derivation for a sentence (Toni, 2014).

While the complexity of reasoning over ABA frameworks is well-understood (Dimopoulos et al., 2002; Dunne, 2009; Dvořák & Dunne, 2018), the complexity of reasoning in the ABA$^+$ formalism under different semantics is at present in various cases open. Towards bridging this gap, we also give new complexity results for reasoning in the considered fragment of ABA$^+$, suggesting that the integration of preferential information into ABA results in increased problem complexity for several central argumentation semantics. In particular, we establish that the problem of verifying whether a given set of assumptions is admissible in ABA$^+$ is coNP-complete (and coNP-hard under complete and grounded semantics). Complexity of the verification problem is oftentimes a key indicator and stepping stone to complexity of further reasoning tasks in argumentation. Indeed, we establish that credulous acceptance of a sentence under admissibility is Σ_2^{P}-complete in ABA$^+$. Furthermore, for ABA$^+$ frameworks satisfying the property expressed by the so-called fundamental lemma (which states that an admissible assumption set stays admissible when an assumption that the set defends is added to it), the grounded assumption set can be computed in polynomial time when having access to an NP-oracle. Our results clearly set apart the complexity of core reasoning tasks in ABA and ABA$^+$: in the LP fragment of ABA, verification of admissibility is P-complete, credulous reasoning under admissible semantics is NP-complete, and all problems regarding grounded semantics are in P. As a particular case where no complexity jump is exhibited, we show that reasoning via stable semantics in ABA$^+$ has the same complexity as in ABA within the considered fragment.

The rest of this article is organized as follows. We start with an overview of the key definitions, semantics, and reasoning problems in assumption-based argumentation with and without preferences (Section 2) to the extent necessary for the rest of the article. We then (in Section 3) establish new complexity results for ABA$^+$ and point out properties essential for developing our ASP-based approach to reasoning in ABA and ABA$^+$. The ASP approach is detailed in Section 4. In Section 5 we present results from an empirical evaluation of the ASP-based approach, focusing on its runtime performance compared to the previously proposed and implemented approaches to reasoning in ABA and ABA$^+$ to the extent applicable. Finally, Section 6 gives an overview of related work and Section 7 concludes. Full proofs not included in the main text can be found in Appendix A.

Some of the results presented in this article have been preliminarily published in the proceedings of the 33rd AAAI Conference on Artificial Intelligence (AAAI-19) (Lehtonen et al., 2019). The current article noticeably extends the AAAI-19 article in several ways: we provide further complexity results (these include the first upper bound on the complexity of finding the grounded assumption set in ABA$^+$ and Σ_2^{P}-completeness for credulous reasoning under admissible semantics in ABA$^+$) and previously unpublished full formal proofs of all complexity results, and extend the ASP-based approach to further semantics (ideal for ABA and grounded for ABA$^+$) as well as present a new encoding of the grounded semantics (an erroneous encoding was proposed in the preliminary version of this work). Furthermore, the empirical evaluation has been extended to cover the supported semantics and reasoning tasks and further comparisons with previously proposed systems.

2. Assumption-Based Argumentation

We recall assumption-based argumentation (ABA) (Bondarenko et al., 1997; Toni, 2014; Čyras et al., 2018) and its generalization ABA$^+$ (Čyras & Toni, 2016b, 2016c; Bao et al., 2017) which equips ABA with preferences over assumptions.

We assume a deductive system $(\mathscr{L},\mathscr{R})$, where $\mathscr{L}$ is a formal language, i.e., a set of sentences, and $\mathscr{R}$ a set of inference rules over $\mathscr{L}$. A rule $r \in \mathscr{R}$ has the form $a_0 \leftarrow a_1,\ldots,a_n$ with $a_i \in \mathscr{L}$.

We denote the head of rule r by $head(r) = \{a_0\}$ and the (possibly empty) body of r with $body(r) = \{a_1, \ldots, a_n\}$. An ABA framework is composed of a deductive system and further information about which sentences can be provisionally assumed and which sentences are contrary to assumptions (inducing a conflict).

Definition 1. *An ABA framework is a tuple $F = (\mathscr{L}, \mathscr{R}, \mathscr{A}, ^-)$, where $(\mathscr{L}, \mathscr{R})$ is a deductive system, $\mathscr{A} \subseteq \mathscr{L}$ a non-empty set of assumptions, and $^-$ a function mapping assumptions $\mathscr{A}$ to sentences $\mathscr{L}$.*

In this work, we focus on a commonly used fragment of ABA, specified as follows. Notably, all implementations of ABA reasoning proposed thus far (see Section 5.1) assume this fragment of ABA.

Assumption 1. *In this paper we focus on the following commonly studied fragment of ABA. For a given ABA framework $F = (\mathscr{L}, \mathscr{R}, \mathscr{A}, ^-)$, we presume that*

- *$\mathscr{L}$ is a set of atoms,*
- *sets $\mathscr{L}$, $\mathscr{R}$, and $\mathscr{A}$ are finite,*
- *for each rule $r \in \mathscr{R}$:*
  - *$body(r)$ is finite,*
  - *the rule r is stated explicitly (given as input), and*
  - *the head of r is not an assumption, i.e., $head(r) \cap \mathscr{A} = \emptyset$;*

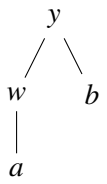
In words, the set $\mathscr{L}$ consists of atomic entities (no complex or compound structures), all components are finite, and no rule has an assumption as its head. ABA frameworks satisfying the last condition are referred to as flat ABA frameworks, which allow for simpler definitions and satisfaction of further properties than general frameworks. In general, different fragments of ABA have been proposed and studied, especially ones allowing for different deductive systems. Further, towards our computational results, we remark that we assume that rules are given explicitly. In particular, derivability (defined formally in the following) is assumed to be decidable in polynomial time (which is immediate by having the rules explicitly given as input).[2] Mainly for illustrative purposes, we sometimes do not explicitly mention all contraries, i.e., we mention an assumption without its contrary. In such cases it can be assumed that the contrary is not part of any rule head, which, in turn, implies non-derivability. From now on, unless stated otherwise, we assume that all ABA frameworks satisfy the properties stated in Assumption 1.

The fragment defined via Assumption 1 subsumes the so-called logic programming (LP) fragment of ABA (Bondarenko et al., 1997), in case the underlying ABA framework is finite. Formally, the LP fragment of ABA is defined as follows. Let HB be a set of atoms.[3] Define $HB_{not} = \{not\, a \mid a \in HB\}$. A ground normal logic programming rule r is of the form $a \leftarrow b_1, \ldots, b_n$ with $a \in HB$ and $b_i \in HB \cup HB_{not}$. A ground normal logic program is a set of such rules. Let π be a ground normal logic program over HB. An ABA framework $F = (\mathscr{L}, \mathscr{R}, \mathscr{A}, ^-)$ corresponds to π if

2. Note that, in general, the deductive system can be such that derivability is not decidable in polynomial time.
3. Bondarenko et al. (1997) use HB to denote the Herbrand Base, i.e., the set of all ground atoms formulated in a Herbrand universe. Yet, for our purposes HB may stand for a set of atoms, without specific reference to a Herbrand universe. Furthermore, we will later, in Section 4, make use of answer set programs, and recall their syntax and semantics.

$$\begin{aligned}
\text{sentences } \mathscr{L} &= \{a,b,c,d,w,x,y,z\} \\
\text{assumptions } \mathscr{A} &= \{a,b,c,d\} \\
\text{contraries} \quad & \overline{b} = x,\ \overline{c} = y,\ \overline{d} = z \\
\text{rules } \mathscr{R} &= \{(w \leftarrow a), (y \leftarrow b,w), (x \leftarrow c), (z \leftarrow a,b)\}
\end{aligned}$$

(a)



(b)

$(w \leftarrow a)$
$(y \leftarrow b,w)$

(c)

Figure 1: Example: (a) ABA framework, (b) tree-derivation for y, (c) and forward-derivation for y.

- $\mathscr{L} = HB \cup HB_{not}$,
- $\mathscr{R} = \pi$,
- $\mathscr{A} = HB_{not}$, and
- $^-$ is defined, for a $not\, a \in HB_{not}$, by $\overline{not\, a} = a$.

An ABA framework is within the LP fragment if there is a corresponding ground normal logic program.[4] The fragment defined via Assumption 1 is a minor generalization of the LP fragment of ABA for finite frameworks: (i) we allow $\mathscr{L}$ to contain more sentences, not only assumptions and a contrary of the assumption (e.g., we allow for sentences that are not a contrary of an assumption), and (ii) we allow for different names of sentences. Nevertheless, the fragment we consider exhibits the same complexity as the LP fragment. Also note that both the fragment we consider and the LP fragment are instances of flat ABA frameworks (i.e., assumptions cannot be derived).

Example 1. *An example ABA framework is illustrated in Figure 1(a). There are eight sentences in $\mathscr{L}$, four of which are assumptions ($\mathscr{A}$), and there are four rules in $\mathscr{R}$. In this framework, the assumption a has no derivable contrary, and thus the contrary of a is left unmentioned.*

A central concept in ABA and ABA$^+$ is how a sentence can be derived from a given set of assumptions and a set of rules. In ABA, several notions of derivability are studied (Dung et al., 2006, 2010), with tree-derivability ($\models$) being the commonly considered one. ABA (without preferences) can be equivalently defined via forward-derivability (Dung et al., 2006, 2010), denoted by the symbol $\vdash$. We employ the forward-derivability definition in our ASP encodings for reasoning in ABA under various semantics (see Section 4). In contrast, tree-derivability is central for defining ABA$^+$, and the equivalence with forward-derivability does not carry over in general to ABA$^+$. However,

4. Bondarenko et al. (1997) give two equivalent formulations of the LP fragment; we recalled the one without usage of an additional theory. Similarly as Dimopoulos et al. (2002), we assume a ground normal logic program.

as we will show, tree-derivability and forward-derivability remain equivalent also for ABA$^+$ under specific semantics, which allows for ASP encodings similar to those we develop for ABA.

Tree-derivability refers to a derivation via a (proof) tree that "starts" from the assumptions (leaves in the tree) and each internal node corresponds to a rule. The trees we consider here are finite rooted labeled trees. As usual, we call non-leaf nodes internal nodes (including the root), and the children of a node are those adjacent nodes on a path to a non-root leaf node (i.e., children are "downwards" towards the non-root leaves). A sentence $s \in \mathscr{L}$ is tree-derivable from a set of assumptions $X \subseteq \mathscr{A}$ and rules $R \subseteq \mathscr{R}$, denoted by $X \models_R s$, if there is a finite rooted labeled tree T with

1. the root is labeled with s,
2. the set of labels for the leaves of T is equal to X or $X \cup \{\top\}$, and
3. there is a surjective function f mapping internal nodes of T to rules R such that $f(v) = r$ implies
   - that the set of labels of children nodes of v is equal to $body(r)$, or $\{\top\}$ if $body(r) = \emptyset$, and
   - that the label of v is equal to $head(r)$.

In other words, a sentence s is tree-derivable via assumptions X and rules R if one can construct a tree with leaves labeled by $X \cup \{\top\}$ which start off the derivations. Each internal node has a corresponding rule in R, and the label of that internal node and the label of the children of that internal node correspond to head and body of a single rule, respectively. Further, each rule in R is present in at least one internal node (since f is assumed to be surjective, each rule in R is mapped onto by f). Finally, the root node is the derived sentence s. The symbol "$\top$" (which is not contained in $\mathscr{L}$) represents an empty rule body, signifying that the head of this rule is tree-derivable from the empty set. As a special case for assumptions, a derivation tree consisting of a single node is permitted in case that node is an assumption $a \in \mathscr{A}$ (i.e., assumptions are derivable from themselves).

While we focus on semantics of ABA defined on assumption sets (with the formal definition being introduced shortly), we mention here that derivation trees can be seen as argument structures. Under this view, there is an argument for a sentence $s \in \mathscr{L}$ if and only if there is a tree-derivation for s. In general, tree-derivations (derivation trees) may be arbitrarily large, e.g., by a rule $s \leftarrow s$ with $s \in \mathscr{L}$, one can chain this rule arbitrarily many times.

Example 2. *For the deductive system in Figure 1(a), the tree-derivation $\{a,b\} \models_R y$ for sentence y is shown in Figure 1(b). Here $R = \{(w \leftarrow a), (y \leftarrow b, w)\}$, i.e., the tree-derivation uses two rules.*

Unless not clear from the context, we will usually write $\models_R$ without explicitly defining R and assume that $R \subseteq \mathscr{R}$ from the given deductive system.

A sentence $a \in \mathscr{L}$ is forward-derivable from a set $X \subseteq \mathscr{A}$ via rules $\mathscr{R}$, denoted by $X \vdash_{\mathscr{R}} a$, if $a \in X$ or there is a sequence of rules $(r_1, \ldots, r_n)$ such that $head(r_n) = a$ and for each rule r_i we have $r_i \in \mathscr{R}$ and each sentence in the body of r_i is derived from rules earlier in the sequence or in X, i.e., $body(r_i) \subseteq X \cup \bigcup_{j<i} head(r_j)$.

Example 3. *For the deductive system in Figure 1(a), a forward-derivation for y from $X = \{a,b\}$ is shown in Figure 1(c).*

There is a natural correspondence between the two notions of derivability.

Proposition 1 (Dung et al., 2006, 2010)**.** *It holds that*

- *if $X \models_R s$ then $X \vdash_{\mathscr{R}} s$, and*
- *if $X \vdash_{\mathscr{R}} s$ there is an $X' \subseteq X$ and $R \subseteq \mathscr{R}$ such that $X' \models_R s$.*

That is, $\models$ is "stricter" (requires a tree derivation where all assumptions and rules are required for derivation), while $\vdash$ is simpler (no witness tree is required and redundant assumptions and rules are allowed). For intuition on why the correspondence holds, if there is a tree-derivation $X \models_R s$ for some $X \subseteq \mathscr{L}$ and $R \subseteq \mathscr{R}$, one can directly use the leaves and rules for showing $X \vdash_{\mathscr{R}} s$. On the other hand, if $X \vdash_{\mathscr{R}} s$ holds, then there exists a subset $R \subseteq \mathscr{R}$ and $X' \subseteq X$ which are "required" to forward-derive s. These form a tree-derivation. Note that forward-derivations may have redundant elements.[5]

Example 4. *Consider the deductive system in Figure 1. For this system it holds that $\{a,b,c\} \vdash_{\mathscr{R}} y$. However, $\{a,b,c\} \not\models_R y$ for any $R \subseteq \mathscr{R}$. To see this, note that one cannot form a tree-derivation with leaves a, b, and c, and root y (there are no rules that connect these sentences, i.e., the connectedness of a potential tree is violated). In this example, c is redundant in deriving y.*

If the type of derivation is not relevant, we generally simply refer to derivations. The deductive closure for an assumption set X w.r.t. rules $\mathscr{R}$ is given by $Th_{\mathscr{R}}(X) = \{a \in \mathscr{L} \mid X \vdash_{\mathscr{R}} a\}$. This can be equivalently defined with $\models$.

We move on to the definitions of attacks between assumption sets and the various semantics of ABA.

Definition 2. *Let $F = (\mathscr{L}, \mathscr{R}, \mathscr{A}, {}^-)$ be an ABA framework, and $A, B \subseteq \mathscr{A}$ be two sets of assumptions. Assumption set A attacks assumption set B in F if $A' \models_R \overline{b}$ for some $A' \subseteq A$, $R \subseteq \mathscr{R}$, and $b \in B$.*

That is, set A attacks B if one can derive the contrary of an assumption in B via the given deductive system. In this definition, we have used tree-derivations. There is an equivalent definition via forward-derivations: A attacks B if $A \vdash_{\mathscr{R}} \overline{b}$ for some $b \in B$. The two notions coincide (Dung et al., 2006, 2010). For attacks in ABA, we will primarily utilize forward-derivations.

The semantics of ABA are based on the concepts of conflict-freeness and defense, defined next.

Definition 3. *Let $F = (\mathscr{L}, \mathscr{R}, \mathscr{A}, {}^-)$ be an ABA framework. An assumption set $A \subseteq \mathscr{A}$ is conflict-free in F iff A does not attack itself. Set A defends assumption set $B \subseteq \mathscr{A}$ in F iff for all $C \subseteq \mathscr{A}$ that attack B it holds that A attacks C.*

We are now ready to recall the various semantics considered in this work.

5. We remark that here redundancy of forward-derivations refers to the fact that $X \vdash_{\mathscr{R}} s$ implies $X' \vdash_{\mathscr{R}} s$ for any $X' \supseteq X$. That is, forward-derivability is preserved under supersets (or is monotone in that sense), unlike tree-derivations where this fact does not hold in general. However, this does not imply that tree-derivations are free of redundancies. For instance, it might be the case that $X \models_R s$ and $X' \models_{R'} s$ for $X \subset X'$, e.g., when R and R' present alternative ways of deriving s, with the former making use of fewer assumptions. A way of addressing the redundancies of tree-derivations is replacing trees with graphs (Craven & Toni, 2016).

Table 1: Admissible assumption sets of Example 5 compared to other semantics σ and semantics $<$-σ with $a < d$.

assumption set	$Th_{\mathscr{R}}(\cdot)$	ABA semantics σ	ABA$^+$ semantics $<$-σ
$\emptyset$	$\emptyset$	*adm*	*adm*
$\{a\}$	$\{a,w\}$	*adm*, *com*, *grd*, *ideal*	*adm*
$\{c\}$	$\{c,x\}$	*adm*	*adm*
$\{c,d\}$	$\{c,d,x\}$	*adm*	*adm*
$\{a,c\}$	$\{a,c,w,x\}$	*adm*	*adm*
$\{a,b\}$	$\{a,b,w,y,z\}$	*adm*, *com*, *prf*, *stb*	-
$\{a,c,d\}$	$\{a,c,d,w,x\}$	*adm*, *com*, *prf*, *stb*	*adm*, *com*, *grd*, *prf*, *stb*, *ideal*

Definition 4. *Let $F = (\mathscr{L}, \mathscr{R}, \mathscr{A}, ^-)$ be an ABA framework. Further, let $A \subseteq \mathscr{A}$ be a conflict-free set of assumptions in F. In F, set A is*

- admissible *iff A defends itself;*
- complete *iff A is admissible and contains every assumption set defended by A;*
- grounded *iff A is the intersection of all complete assumption sets;*[6]
- preferred *iff A is admissible and there is no admissible set of assumptions B with $A \subset B$;*
- stable *iff each $\{x\} \subseteq \mathscr{A} \setminus A$ is attacked by A; and*
- ideal *iff A is the $\subseteq$-maximal admissible assumption set within the intersection of all preferred assumption sets.*

We use the term σ-assumption set for an assumption set under a semantics $\sigma \in \{$*adm*, *com*, *grd*, *stb*, *prf*, *ideal*$\}$, i.e., admissible, complete, grounded, stable, preferred, and ideal semantics, respectively.

Example 5. *For the example ABA framework in Figure 1(a), Table 1 lists all admissible assumption sets (first column), the deductive closure (second column), and all semantics whose criteria the corresponding assumption set satisfies (third column). Consider the assumption set $\{a,b\}$: the deductive closure also contains w, y, and z. The last two sentences are contrary to assumptions c and d respectively. This means that $\{a,b\}$ attacks assumption sets $\{c\}$ and $\{d\}$, and every assumption set containing either c or d. The set itself is attacked only by assumption sets containing the assumption c, since $Th_{\mathscr{R}}(\{c\}) = \{x\}$ and $\overline{b} = x$. Since every assumption set containing c is attacked by $\{a,b\}$, it holds that $\{a,b\}$ is admissible. This set is complete, since any assumption set $A \subseteq \mathscr{A}$ not contained in $\{a,b\}$ is not defended by $\{a,b\}$. To see that $\{a,b\}$ is also preferred, consider any proper superset A of $\{a,b\}$: then either $c \in A$ or $d \in A$, which violates conflict-freeness. Thus, no proper superset is admissible. Finally, $\{a,b\}$ is stable since it attacks all other assumptions.*

Further, the framework has three complete assumption sets: $\{a\}$, $\{a,b\}$, and $\{a,c,d\}$. The assumption set $\{a\}$ is the intersection of the complete assumption sets, and is moreover admissible.

6. In general (i.e., not necessarily flat) ABA frameworks grounded semantics is referred to as well-founded semantics. We consider only flat frameworks.

Thus $\{a\}$ is the grounded assumption set. The other two complete assumption sets are also preferred and stable assumption sets. As $\{a\}$ is the intersection of the preferred assumptions sets and is also admissible, it is the ideal assumption set.

We recall an alternative characterization of grounded semantics that we use for our ASP implementation in Section 4. For an ABA framework F, the grounded semantics can be equivalently defined via the function $def_F(A) = \{a \in \mathscr{A} \mid A \text{ defends } \{a\}\}$.

Proposition 2 (Bondarenko et al., 1997, Theorem 6.2)**.** *Let $F = (\mathscr{L}, \mathscr{R}, \mathscr{A}, ^-)$ be an ABA framework. It holds that the grounded assumption set E of F is equal to the least fixpoint of def_F.*

Two basic reasoning tasks on ABA are verifying whether a given set of assumptions is a σ-assumption set and enumerating all σ-assumption sets. In addition, often a relevant question is to find out whether a given sentence is acceptable under a semantics. To answer this question, two prominent reasoning modes are credulous and skeptical acceptance of sentences in an ABA framework.

Definition 5. *Let $F = (\mathscr{L}, \mathscr{R}, \mathscr{A}, ^-)$ be an ABA framework and σ be a semantics.*

- *In the verification problem the task is to check whether a given $S \subseteq \mathscr{A}$ is a σ-assumption set of F.*
- *In the enumeration problem the task is to report all σ-assumption sets of F.*

A given sentence $s \in \mathscr{L}$ is

- *credulously accepted in F under semantics σ iff there is a σ-assumption set A such that $s \in Th_{\mathscr{R}}(A)$; and*
- *skeptically accepted in F under semantics σ iff $s \in Th_{\mathscr{R}}(A)$ for all σ-assumption sets A.*

In flat ABA there is a unique grounded and ideal assumption set, implying coincidence of credulous and skeptical reasoning for both of these semantics. Grounded reasoning also coincides with skeptical reasoning under complete semantics. Since each preferred assumption set is complete, each complete assumption set is admissible, and each admissible assumption set is a subset of some preferred assumption set, it follows that credulous reasoning under admissible, complete, and preferred semantics coincide (Bondarenko et al., 1997; Čyras et al., 2018).

Example 6. *Continuing Example 5 (see also Table 1), every sentence is credulously accepted under admissible semantics (and therefore also under complete and preferred semantics). No sentence is skeptically accepted under admissible semantics (only sentences derivable from the empty set are skeptically accepted under admissible semantics). The sentences a and w are skeptically accepted under complete, preferred, and stable semantics.*

We move on to the ABA$^+$ formalism. ABA$^+$ extends ABA by including preferences over assumptions.

Definition 6. *An ABA$^+$ framework is a tuple $F = (\mathscr{L}, \mathscr{R}, \mathscr{A}, ^-, \leq)$, where $(\mathscr{L}, \mathscr{R}, \mathscr{A}, ^-)$ is an ABA framework and $\leq$ a preorder on $\mathscr{A}$.*

A preorder is a reflexive and transitive binary relation. The strict counterpart $<$ of $\leq$ is defined as usual by $a < b$ iff $a \leq b$ and $b \not\leq a$, for $a, b \in \mathscr{A}$. We focus on flat ABA$^+$ frameworks, where flatness is defined in the same manner as for ABA frameworks.

The definition of attacks from ABA frameworks is generalized as follows to $<$-attacks in ABA$^+$ frameworks, with "$<$" highlighting the fact that preferences, induced by $<$, are used for attacks.

Definition 7. *Let $(\mathscr{L}, \mathscr{R}, \mathscr{A}, {}^-, \leq)$ be an ABA$^+$ framework, and $A, B \subseteq \mathscr{A}$ be two sets of assumptions. Assumption set A $<$-attacks B in F iff*

- *$A' \models_R \overline{b}$, for some $A' \subseteq A$, $b \in B$, and $\nexists a' \in A'$ with $a' < b$, or*
- *$B' \models_R \overline{a}$ for some $a \in A$ and $B' \subseteq B$ such that $\exists b' \in B'$ with $b' < a$.*

In words, set A attacks B iff (i) from A, via subset A', one can tree-derive a contrary of an assumption $b \in B$ and no member in A' is strictly less preferred than b, or (ii) from B, via subset B' one can tree-derive a contrary of an assumption $a \in A$ and some member of B' is strictly less preferred than a. Attacks of type (i) are *normal* $<$-attacks and those of type (ii) *reverse* $<$-attacks, with the intuition that the (non-preference based) conflict in (i) succeeds and in case of (ii) is reversed by the preference relation.

Comparing ABA$^+$ $<$-attacks to ABA attacks, in an ABA$^+$ framework $(\mathscr{L}, \mathscr{R}, \mathscr{A}, {}^-, \leq)$ with $\leq = \emptyset$ (i.e., no preferences are imposed), the normal $<$-attacks coincide with attacks and there are no reverse $<$-attacks. That is, without preferences one can use the standard ABA definitions for attacks, and they coincide with normal $<$-attacks.

The notions of conflict-freeness, defense, and the semantics are straightforwardly generalized from ABA to ABA$^+$ by replacing attacks with $<$-attacks. For conflict-freeness, there is no need for a new definition, since if A attacks B in an ABA$^+$ framework, then either A $<$-attacks B or B $<$-attacks A. This means that each non-preference-based attack is either present as-is as a $<$-attack, or reversed, but never "lost". Thus in ABA$^+$ an assumption set is conflict-free using the same definition as for ABA.

Set $A \subseteq \mathscr{A}$ $<$-defends assumption set $B \subseteq \mathscr{A}$ iff for all $C \subseteq \mathscr{A}$ that $<$-attack B it holds that A $<$-attacks C. The definitions for the semantics for ABA$^+$ are generalized from ABA as follows.

Definition 8. *Let $F = (\mathscr{L}, \mathscr{R}, \mathscr{A}, {}^-, \leq)$ be an ABA$^+$ framework. Further, let $A \subseteq \mathscr{A}$ be a conflict-free set of assumptions in F. In F, set A is*

- $<$-admissible *iff A $<$-defends itself;*
- $<$-complete *iff A is admissible and contains every assumption set $<$-defended by A;*
- $<$-grounded *iff A is the intersection of all $<$-complete assumption sets;*
- $<$-preferred *iff A is $<$-admissible and there is no $<$-admissible set of assumptions B with $A \subset B$;*
- $<$-stable *iff each $\{x\} \subseteq \mathscr{A} \setminus A$ is $<$-attacked by A; and*
- $<$-ideal *iff A is a $\subseteq$-maximal admissible assumption set within the intersection of all $<$-preferred assumption sets.*

When referring to assumption sets for ABA$^+$, we additionally use the preference order $<$, e.g., we say $<$-admissible assumption set. The reasoning tasks for ABA$^+$ are the same as for ABA, after replacing σ with $<$-σ.

Example 7. *Recall the ABA framework from Example 5, shown in Table 1. If we extend the example framework by the preference $a < d$, several semantics change, as shown in the right most column in Table 1. Notably, $\{a,b\}$ is not $<$-admissible (and also not $<$-complete, $<$-preferred, or $<$-stable). The reason for this is that from $\{a,b\}$ one can tree-derive $z = \overline{d}$, which means $\{a,b\}$ is attacking (when not taking preferences into account) assumption set $\{d\}$. However, since d is strictly more preferred than a, it holds that this attack is reversed when using $<$-attacks. That is, $\{d\}$ reversely $<$-attacks $\{a,b\}$. In fact, $\{d\}$ is not $<$-attacked at all, implying that no set $<$-defends $\{a,b\}$. Under the preference, both $\{a\}$ and $\{d\}$ are $<$-unattacked. Further, $\{d\}$ $<$-defends $\{c\}$: $\{c\}$ is $<$-attacked only by $\{a,b\}$ (normally), which is countered by $\{d\}$ (reversely), and, thus, also by $\{a,d\}$. This means that $\{a,c,d\}$ is $<$-admissible. In fact, this set is also $<$-complete, $<$-grounded, $<$-preferred, and $<$-stable.*

The computational complexity of reasoning in ABA is well-established (Dimopoulos et al., 2002; Dunne, 2009), and is outlined in Table 2 for the considered fragment of ABA. We assume that the reader has knowledge about basic complexity classes, and the concepts of decision problems, reductions, hardness, and completeness. For details we refer the reader to the book on complexity theory by Papadimitriou (1994). We briefly recall complexity classes beyond NP to the extent necessary for our discussion. A decision problem is in Σ^{P}_2 if the problem can be decided via a non-deterministic polynomial time algorithm that can access an NP-oracle. The class Π^{P}_2 is the complementary class of Σ^{P}_2, i.e., contains all problems where there is a corresponding problem in Σ^{P}_2 where all "yes" instances are "no" instances and vice versa for "no" instances. The classes Δ^{P}_2 and Θ^{P}_2 contain problems that can be decided in polynomial time by an algorithm that can access an NP-oracle, with the latter class having the additional restriction that at most a logarithmic number of such calls may be used.

Complexity of credulous and skeptical reasoning in the LP fragment of ABA under admissible, preferred, and stable semantics was established by Dimopoulos et al. (2002), and under ideal semantics by Dunne (2009). Regarding hardness under ideal reasoning, we remark that Θ^{P}_2-hardness was shown under randomized reductions (Valiant & Vazirani, 1986). Complexity under complete and grounded reasoning was not made explicit; we provide a formal proof in the next section. Complexity of ABA with preferences has not, to our knowledge, been studied in-depth. Wakaki (2017b) showed complexity results for p_ABA, which presents an alternative way of handling preferences in ABA. However, the results are not directly transferable to ABA$^+$. We establish complexity results for the LP fragment of ABA$^+$ in the next section.

3. Properties and Complexity Results

We discuss properties essential for developing our ASP encodings, as well as establish complexity results for ABA$^+$ (results for ABA$^+$ summarized in Table 3). Recall that we focus on frameworks satisfying Assumption 1, which in particular subsumes the LP fragment of ABA and implies flatness for both ABA and ABA$^+$. Concretely, for ABA$^+$, we show complexity of credulous reasoning under admissible semantics (Σ^{P}_2-complete) and stable semantics (NP-complete) and skeptical reasoning under stable semantics (coNP-complete). We further show complexity of verifying whether a set of

Table 2: Complexity of deciding acceptance of a sentence in the LP fragment of ABA. In addition to the previously established results, a proof of the complexity of credulous reasoning under complete semantics and reasoning under grounded semantics is established as Corollary 6.

	Reasoning mode	
semantics	**credulous**	**skeptical**
admissible	NP-complete	in P
complete	NP-complete	in P
preferred	NP-complete	Π_2^P-complete
stable	NP-complete	coNP-complete
grounded	in P	in P
ideal	in Θ_2^P	in Θ_2^P

Table 3: Complexity results for deciding acceptance of a sentence and verifying whether a set of assumptions is a σ-assumption set in the LP fragment of ABA$^+$. All the results are established in this article. The complexity of credulous and skeptical reasoning under $<$-grounded semantics hold only for frameworks satisfying the FL-property; see Property 1.

		Reasoning mode	
semantics	**verifying σ-assumption set**	**credulous**	**skeptical**
$<$-stable	in P	NP-complete	coNP-complete
$<$-grounded	coNP-hard	Δ_2^P (FL)	Δ_2^P (FL)
$<$-admissible	coNP-complete	Σ_2^P-complete	?
$<$-complete	coNP-hard	?	?

assumption is a $<$-σ assumption set, with tight bounds for admissible semantics (coNP-complete) and hardness for both complete and grounded semantics (coNP-hard). For finding the grounded semantics in frameworks satisfying the property expressed by the fundamental lemma, we give a complexity upper bound (polynomial time when given access to an NP-oracle).

3.1 Properties and Complexity of ABA

The ABA framework is well understood and many properties and complexity results have been derived (Čyras et al., 2018; Dimopoulos et al., 2002). Towards our ASP encodings, we slightly re-state (in an equivalent way) some of the ABA semantics. Furthermore, we make observations on the complexity landscape of ABA reasoning that have not been explicated before.

We begin with a lemma for ABA which explicates that defending a set of assumptions is the same as defending each assumption of that set individually. Although the lemma follows fairly directly from the definitions, the analogous observation does in fact not hold for ABA$^+$.

Lemma 3. *Let $F = (\mathscr{L}, \mathscr{R}, \mathscr{A}, ^-)$ be an ABA framework, and $A, B \subseteq \mathscr{A}$. Set A defends set B iff set A defends each $\{b\} \subseteq B$.*

Proof. Assume that A defends B. Let $b \in B$ be arbitrary, and let C be any assumption set attacking $\{b\}$. By $\subseteq$-monotonicity of attacks, it holds that set C attacks B. Therefore A attacks C, and thus A defends $\{b\}$. Since b is arbitrary, A defends each $\{b\} \subseteq B$.

Assume that A defends each $\{b\} \subseteq B$. Let a $C \subseteq \mathscr{A}$ attack B. Then, by definition of attacks in ABA, there is a $b' \in B$ such that $C \vdash_{\mathscr{R}} \overline{b'}$. This implies that C attacks $\{b'\} \subseteq B$, and thus A attacks C. Therefore A defends B. □

We now highlight that defense of assumption sets and in turn ABA semantics can be equivalently defined via the set of attacked assumptions. Let $(\mathscr{L}, \mathscr{R}, \mathscr{A}, {}^-)$ be an ABA framework and $E \subseteq \mathscr{A}$ be a set of assumptions. Define $att(E) = \{a \in \mathscr{A} \mid E \text{ attacks } \{a\}\}$. Recall that in ABA an assumption set E attacks $\{a\}$ iff $E \vdash_{\mathscr{R}} \overline{a}$. For Proposition 4, the main ingredient in checking if a set is admissible or complete is the set $att(E)$ (which can be computed in polynomial time since $\vdash_{\mathscr{R}}$ is decidable in polynomial time).

Proposition 4. *Let $(\mathscr{L}, \mathscr{R}, \mathscr{A}, {}^-)$ be an ABA framework and $E \subseteq \mathscr{A}$ be a conflict-free set of assumptions. It holds that*

- *set E defends an assumption set $\{a\} \subseteq \mathscr{A}$ iff $\mathscr{A} \setminus att(E)$ does not attack $\{a\}$;*
- *set E is admissible iff $\mathscr{A} \setminus att(E)$ does not attack E; and*
- *set E is complete iff E is admissible and for each $b \in \mathscr{A} \setminus E$ it holds that $\{b\}$ is attacked by $\mathscr{A} \setminus att(E)$.*

Proof. Set E defends an assumption $\{a\} \subseteq \mathscr{A}$ iff for all $C \subseteq \mathscr{A}$ that attack $\{a\}$ it holds that E attacks C. Set E attacks any assumption set X with $X \cap att(E) \neq \emptyset$. Thus, E defends $\{a\}$ iff $\mathscr{A} \setminus att(E)$ does not attack $\{a\}$. The remaining statements follow directly from the previous observation, the corresponding definitions, and by Lemma 3. □

We make the complexity of reasoning in ABA under grounded and complete semantics explicit, which has been lacking in the literature.[7] Recall that the grounded assumption set of an ABA framework F is equal to the least fixpoint of def_F (see Proposition 2).

Corollary 5. *For a given ABA framework, one can in polynomial time*

- *compute the grounded assumption set, and*
- *verify whether a given set of assumptions is complete.*

Proof. It is sufficient to show that $def(A)$ can be computed in polynomial time for an ABA framework $(\mathscr{L}, \mathscr{R}, \mathscr{A}, {}^-)$ and $A \subseteq \mathscr{A}$. Since $\vdash_{\mathscr{R}}$ can be computed in polynomial time, one can compute $att(X)$ in polynomial time for a given $X \subseteq \mathscr{A}$. The assumptions defended by A are those that are not attacked by the set of assumptions that are not attacked by A. □

The complexity of acceptance under grounded and complete semantics follows from the preceding corollary together with the facts that the grounded assumption set is a $\subseteq$-minimal complete assumption set and the grounded assumption set is unique (Bondarenko et al., 1997, Theorem 6.2), and that credulous reasoning under admissible and complete semantics coincides (Bondarenko et al., 1997; Čyras et al., 2018).

Corollary 6. *It holds that*

7. Between the writing and publication of this article, these results were independently shown by Čyras et al. (2021).

- *credulous reasoning in ABA under complete semantics is NP-complete, and*
- *skeptical reasoning in ABA under complete semantics and credulous as well as skeptical reasoning under grounded semantics is decidable in polynomial time.*

We make the computation of the grounded assumption set via iterated defense explicit, since we will use the same approach in our encodings later. For an ABA framework $F = (\mathscr{L}, \mathscr{R}, \mathscr{A}, ^-)$ we define triples $\mathtt{S}_i = (I, D, U)$ for $0 \leq i \leq |\mathscr{A}|$ and $I, D, U \subseteq \mathscr{A}$.[8] For a triple $\mathtt{S}_i = (I, D, U)$ we use shorthands $I_i = I$, $D_i = D$, and $U_i = U$. The interpretation is that I_i is defended by I_{i-1}, D_i is attacked by I_i, and U_i is unattacked by I_i. We define the sequence $\mathtt{S}(F) = (\mathtt{S}_0, \ldots, \mathtt{S}_{|\mathscr{A}|})$ with

- $\mathtt{S}_0 = (\emptyset, att(\emptyset), \mathscr{A} \setminus att(\emptyset))$ and
- $\mathtt{S}_i = (I_i, D_i, U_i)$ with $I_i = \mathscr{A} \setminus att(U_{i-1})$, $D_i = att(I_i)$, and $U_i = \mathscr{A} \setminus D_i$.

We show that $I_{|\mathscr{A}|}$ is the grounded assumption set of F. By $def_F^i(\emptyset)$ we denote i applications of def on $\emptyset$, and for $i = 0$ we define $def_F^0(\emptyset) = \emptyset$.

Lemma 7. *Let $F = (\mathscr{L}, \mathscr{R}, \mathscr{A}, ^-)$ be an ABA framework and sequence $\mathtt{S}(F) = (\mathtt{S}_0, \ldots, \mathtt{S}_{|\mathscr{A}|})$ as above. It holds that $def_F^i(\emptyset) = I_i$, for $0 \leq i \leq |\mathscr{A}|$.*

Proof. We prove the claim by induction. For $i = 0$ the claim holds by definition. Let $i \geq 0$ and assume that $def_F^i(\emptyset) = I_i$. We show that $def_F^{i+1}(\emptyset) = I_{i+1}$ holds. It holds that

$$
\begin{aligned}
& a \in def_F^{i+1}(\emptyset) \\
\text{iff } & \{a\} \text{ is defended by } def_F^i(\emptyset) \\
\text{iff } & \mathscr{A} \setminus att(def_F^i(\emptyset)) \text{ does not attack } \{a\} && \text{(Proposition 4)} \\
\text{iff } & \mathscr{A} \setminus att(I_i) \text{ does not attack } \{a\} && \text{(Induction hypothesis)} \\
\text{iff } & \mathscr{A} \setminus D_i \text{ does not attack } \{a\} \\
\text{iff } & U_i \text{ does not attack } \{a\} \\
\text{iff } & a \in I_{i+1} && \square
\end{aligned}
$$

To see that $I_{|\mathscr{A}|}$ is the grounded assumption set, notice that $I_{|\mathscr{A}|}$ is a fixpoint of def: $def_F^i(\emptyset)$ is monotonically increasing (w.r.t. subsets) and at each iteration either a fixpoint is reached or at least one further assumption is added to the set. Thus a fixpoint is reached after at most $|\mathscr{A}|$ applications of def.

3.2 Derivability in ABA$^+$

Tree-derivability is used for defining $<$-attacks in ABA$^+$. In particular, tree-derivations ($\models$) and forward-derivations ($\vdash$) are not equivalent in terms of $<$-attacks. This can be seen by considering situations in which A normally $<$-attacks B: adding redundant assumptions to A (not used for deriving a sentence) may weaken the set and make it open for reverse $<$-attacks that are not present under $\models$. That is, reversing attacks can differ for the two derivability notions. The following is a concrete example for the fact that tree-derivability and forward-derivability give rise to different sets of $<$-attacks.

8. The components in the triple stand for in (I), defeated (D) and undefeated (U), respectively.

Example 8. *Consider the ABA$^+$ framework $(\mathscr{L},\mathscr{R},\mathscr{A},^-,\leq)$ with $\mathscr{A}=\{a,b,c\}$ and $\mathscr{R}=\{\overline{b}\leftarrow a\}$, and let $c<b$ be a preference in $\leq$. It holds that $A=\{a\}$ $<$-attacks $B=\{b\}$ normally, since $A\models_R\overline{b}$ with $R=\{\overline{b}\leftarrow a\}$. B does not $<$-attack A, since no contrary of a is derivable from B, and $a\not<b$. Let $A'=\{a,c\}$. In this case, $A'\vdash_{\mathscr{R}}\overline{b}$, but $A'\not\models_R\overline{b}$ for any $R\subseteq\mathscr{R}$, since there is no derivation tree with root $\overline{b}$ and both a and c as assumptions (leaves). In other words, c is redundant in deriving $\overline{b}$. Note that there is an assumption in A' that is less preferred than b, namely c. It holds that B does not reversely $<$-attack A'. To see this, consider any subset X of A'. Only when $X=A$ it holds that $X\models_R\overline{b}$. But there is no assumption in A less preferred than b. If one would replace tree-derivability with forward-derivability in the definition of $<$-attacks, then B would reversely $<$-attack A' since the redundant assumption c is weaker than b.*

However, in the special case of a conflict-free set reversely $<$-attacking a singleton set, forward-derivability can be used for normal and reverse $<$-attacks.

Lemma 8. *Let $(\mathscr{L},\mathscr{R},\mathscr{A},^-,\leq)$ be an ABA$^+$ framework, $A,B\subseteq\mathscr{A}$ sets of assumptions, and $b\in\mathscr{A}$. Then the following claims hold.*

(i) Set A normally $<$-attacks B iff $A'\vdash_{\mathscr{R}}\overline{b}$ for some $A'\subseteq A$, $b\in B$ and $\nexists a'\in A'$ with $a'<b$.

(ii) If A is conflict-free, then A reversely $<$-attacks $\{b\}$ iff $\{b\}\vdash_{\mathscr{R}}\overline{a}$ for some $a\in A$ and $b<a$.

Proof. (i) First recall from Proposition 1 that if $X\models_R s$, then $X\vdash_{\mathscr{R}}s$, and if $X\vdash_{\mathscr{R}}s$, then there is an $X'\subseteq X$ and $R\subseteq\mathscr{R}$ such that $X'\models_R s$.

Assume that assumption set A normally $<$-attacks B. Then $A'\models_R\overline{b}$, for some $A'\subseteq A$, $b\in B$, and $\nexists a'\in A'$ with $a'<b$. This implies $A'\vdash_{\mathscr{R}}\overline{b}$. For the other direction, assume that $A'\vdash_{\mathscr{R}}\overline{b}$ for some $A'\subseteq A$, $b\in B$, and $\nexists a'\in A'$ with $a'<b$. There is an $A''\subseteq A'$ such that $A''\models_R\overline{b}$ for some $R\subseteq\mathscr{R}$. Since $\nexists a'\in A'$ with $a'<b$, we have $\nexists a''\in A''$ with $a''<b$, and the first claim follows.

(ii) If A reversely $<$-attacks $\{b\}$, then by definition $\{b\}\models_R\overline{a}$ for some $a\in A$ such that $b<a$. Then $\{b\}\vdash_{\mathscr{R}}\overline{a}$. Assume, on the other hand, that $b<a$ and $\{b\}\vdash_{\mathscr{R}}\overline{a}$ for some $a\in A$. As A is conflict-free, it must then hold that $\{b\}\models_R\overline{a}$. This implies that A $<$-reversely attacks $\{b\}$. □

A corollary of Lemma 8 is that one can check whether a set of assumptions is $<$-stable via relying only on forward-derivability. To make this explicit, we slightly restate $<$-stable semantics in the following proposition, distinguishing between normal and reverse $<$-attacks.

Proposition 9. *Let $D=(\mathscr{L},\mathscr{R},\mathscr{A},^-,\leq)$ be an ABA$^+$ framework. A conflict-free set $E\subseteq\mathscr{A}$ is $<$-stable iff for all $\{b\}\subseteq\mathscr{A}$ that are not normally $<$-attacked by E, either $b\in E$ or $\{b\}$ is reversely $<$-attacked by E.*

Proof. Let $E\subseteq\mathscr{A}$ be a conflict-free assumption set. Assume that E is $<$-stable. Then E $<$-attacks any $\{b\}\subseteq\mathscr{A}\setminus E$. Let $X\subseteq\mathscr{A}$ be the set of assumptions not normally $<$-attacked by E. Suppose that there is an $x\in X$ such that neither $x\in E$ nor $\{x\}$ is reversely $<$-attacked by E. Since $\{x\}$ is not normally $<$-attacked by E, it holds that $\{x\}$ is not $<$-attacked by E. Since $x\notin E$, E is not $<$-stable, which is a contradiction.

Assume that for all $\{b\}$ that are not normally $<$-attacked by E, either $b\in E$ or $\{b\}$ is reversely $<$-attacked by E. Suppose E is not $<$-stable. Since E is conflict-free, it holds that there is an $x\in\mathscr{A}\setminus E$ such that E does not $<$-attack $\{x\}$. Thus E does not normally $<$-attack $\{x\}$. By assumption, either $x\in E$ or $\{x\}$ is reversely $<$-attacked by E. This is a contradiction to E not being $<$-stable. □

Proposition 9 will prove to be useful for the ASP encodings presented later on in this article.

3.3 On the Complexity of Reasoning in ABA$^+$

We move on to investigating the complexity of reasoning in ABA$^+$. Central to this is understanding the role of $<$-attacks. In both ABA and ABA$^+$ attacks satisfy $\subseteq$-monotonicity of the following form.

Lemma 10 (Čyras, 2017, Lemma 3.3)**.** *Let $(\mathscr{L},\mathscr{R},\mathscr{A},^-,\leq)$ be an ABA$^+$ framework, and A and B two sets of assumptions. Then for any $A \subseteq A'$ and $B \subseteq B'$ we have that if A $<$-attacks B, then A' $<$-attacks B'.*

That is, if A ($<$-)attacks $\{b\}$, then A ($<$-)attacks any B with $b \in B$.

Independently of monotonicity, attacks in ABA$^+$ may in cases not attack singleton sets of assumptions. While normal $<$-attacks in ABA$^+$ are similar to attacks in ABA, reverse $<$-attacks *originate* from singleton sets of assumptions and attack a set of assumptions. Put differently, for ABA it holds that if a set B is attacked by A, then there is at least one singleton set $\{b\} \subseteq B$ such that $\{b\}$ is attacked by A. However, this does not hold for reverse $<$-attacks in ABA$^+$.

Example 9. *Consider again the ABA$^+$ framework from Example 7 (this is the ABA$^+$ framework shown in Figure 1 together with preference $a < d$). Important for our purpose in this example is that from $\{a,b\}$ one can tree-derive $z = \overline{d}$ (via rule $z \leftarrow a,b$). Here $\{d\}$ reversely $<$-attacks $\{a,b\}$. However, $\{d\}$ $<$-attacks neither $\{a\}$ nor $\{b\}$, since there is no derivation possible from d or b (i.e., $Th_{\mathscr{R}}(\{d\}) = \{d\}$ and $Th_{\mathscr{R}}(\{b\}) = \{b\}$) and from a one can derive only w (via rule $w \leftarrow a$) which is not the contrary of d. Thus, $\{d\}$ does not $<$-attack any singleton subset of $\{a,b\}$.*

The difference between attacks in ABA and ABA$^+$ is significant from a computational perspective. In particular, for analyzing the complexity of reasoning in ABA$^+$ in terms of different types of attacks, we identify four types of (counter) $<$-attacks. Specifically, for two sets of assumptions A and B, we will distinguish between

(i) A normally $<$-attacking B,

(ii) A reversely $<$-attacking B,

(iii) A normally $<$-attacking each subset $B' \subseteq B$ that $<$-attacks A, and

(iv) A reversely $<$-attacking each subset $B' \subseteq B$ that $<$-attacks A.

The first three types of $<$-attacks can be checked in polynomial time.

Proposition 11. *Let $(\mathscr{L},\mathscr{R},\mathscr{A},^-,\leq)$ be an ABA$^+$ framework, and A and B two sets of assumptions. One can decide in polynomial time whether set A*

1. *normally $<$-attacks B,*
2. *reversely $<$-attacks B, or*
3. *normally $<$-attacks each subset $B' \subseteq B$ that $<$-attacks A.*

Proof. For Item 1, consider each $b \in B$ separately and compute $A_b = \{a \in A \mid a \not< b\}$. We have A_b normally $<$-attacks $\{b\}$ iff $A_b \vdash_{\mathscr{R}} \overline{b}$, the latter of which can be checked in polynomial time. If A_b

normally $<$-attacks $\{b\}$, then, by monotonicity, A normally $<$-attacks B. Likewise, if A normally $<$-attacks B, then there is a $b \in B$ such that A_b normally $<$-attacks $\{b\}$.

For Item 2, compute $Th_{\mathscr{R}}(B)$. For each $a \in A$, if $\overline{a} \in Th_{\mathscr{R}}(B)$, check whether it holds that $B' \models_R \overline{a}$ such that $B' \subseteq B$ and there is a $b' \in B'$ with $b' < a$. This check can be reduced to reachability in a directed graph due to Lemma 20 (shown in Appendix A), and can be done in polynomial time (the graph of this lemma can be constructed in polynomial time, and the reachability problem can be decided in polynomial time, also for each $b' < a$).

For Item 3, compute $X = \{b \in B \mid A \text{ normally } <\text{-attacks } \{b\}\}$, which can be done in polynomial time due to Item 1. By monotonicity A attacks each B' of B for which it holds that $B' \cap X \neq \emptyset$, so it suffices to check whether $B \setminus X$ $<$-attacks A (via Items 1 and 2). If so, then A does not counter all $<$-attacks via normal $<$-attacks. If $B \setminus X$ does not $<$-attack A, then by monotonicity it holds that no subset $B' \subseteq B \setminus X$ $<$-attacks A. □

We emphasize that Item 2 of Proposition 11 is polynomial-time decidable. This is because one does not need to find subset-minimal $B' \subseteq B$ that are reversely $<$-attacked by A. Item 3 is also polynomial-time decidable since normal $<$-attacks target only singleton sets of assumptions.

Before delving into attacks of type (iv), we note that the complexity of verifying that a set of assumptions is $<$-stable is obtained via Proposition 11. Note that the property of being conflict-free does not depend on the preference relation (Čyras, 2017, Theorem 3.5) and can be checked in polynomial time.

Theorem 12. *Verifying that a set of assumptions is $<$-stable in an ABA$^+$ framework is in P. Checking credulous (skeptical) acceptance of a sentence under $<$-stable semantics in ABA$^+$ is NP-complete (coNP-complete).*

Proof. For membership, given an assumption set A, one can check in polynomial time whether A is conflict-free. Further, by Proposition 11, one can check in polynomial time whether A $<$-attacks each $\{b\} \subseteq \mathscr{A} \setminus A$. Credulous (skeptical) acceptance can be checked by a non-deterministic guess of an assumption set, and verifying stability and whether the queried sentence can (cannot) be derived. Hardness carries over from ABA. □

In particular, credulous and skeptical reasoning under $<$-stable semantics in ABA$^+$ is of the same complexity as reasoning under stable semantics in ABA.

However, for the other semantics considered in this work, complexity results for reasoning in ABA do not directly carry over to ABA$^+$. In fact, we show next that it is coNP-hard to decide whether a set of assumptions counterattacks by reverse $<$-attacks all sets of assumptions that attack it, i.e., attacks of type (iv) from before Proposition 11.

Several of the subsequent complexity proofs will be based on the following reduction from the problem of deciding whether a given Boolean formula in conjunctive normal with a maximum of three literals per clause (3-CNF) is unsatisfiable, which is a classical coNP-complete problem. For the following, for a clause $c_j = l_1 \vee l_2 \vee l_3$ let $neg(c_j) = \neg l_1, \neg l_2, \neg l_3$ with $\neg l_i = x$ if $l_i = \neg x$ and $\neg l_i = \neg x$ if $l_i = x$. In our reduction, we translate a Boolean formula ϕ in 3-CNF to an ABA$^+$ framework. In particular, we translate literals l to sentences in the ABA$^+$ framework. We emphasize that a literal of the form $l = \neg x$ (negative literal) is interpreted as a (normal) sentence in the constructed ABA$^+$ framework. That is, "$\neg$" is not to be interpreted as a unary (logical) operator in any sense, within the constructed ABA$^+$ instance. For instance, a rule $x \leftarrow \neg y$ is to be interpreted that one can derive from the sentence "$\neg y$" the sentence "x" (and this rule does not apply for a contrary of y, if present).

Reduction 1. *Given a Boolean formula in 3-CNF $\phi = c_1 \wedge \cdots \wedge c_m$ over vocabulary $X = \{x_1, \ldots, x_n\}$ and clauses $C = \{c_1, \ldots, c_m\}$, the ABA$^+$ framework $\texttt{red}(\phi) = (\mathscr{L}, \mathscr{R}, \mathscr{A}, ^-, \leq)$ has $\mathscr{A} = \{a, b\} \cup X \cup \neg X$ for $\neg X = \{\neg x \mid x \in X\}$ and $\mathscr{R}$ consists of the following rules.*

- $d_i \leftarrow x_i$ *and* $d_i \leftarrow \neg x_i$ *for* $1 \leq i \leq n$.
- $\overline{a} \leftarrow d_1, \ldots, d_n$.
- $\overline{x_i} \leftarrow x_i$ *and* $\overline{\neg x_i} \leftarrow \neg x_i$ *for* $1 \leq i \leq n$.
- $\overline{b} \leftarrow x_i, \neg x_i, \overline{a}$ *for* $1 \leq i \leq n$.
- $\overline{b} \leftarrow neg(c_j), \overline{a}$ *for* $1 \leq j \leq m$.

Finally, let $\mathscr{L} = \mathscr{A} \cup \{\overline{z} \mid z \in \mathscr{A}\} \cup \{d_1, ..., d_n\}$ and let $\leq$ consist of $b > x_i$ and $b > \neg x_i$ for $1 \leq i \leq n$.

Reduction 1 is illustrated in Figure 2 for an example formula. We note that the ABA$^+$ framework $\texttt{red}(\phi)$ can be constructed in polynomial time for a given formula. For each variable x_i we have five rules and two contraries, for each clause c_j one rule, and one additional rule. The size of $\mathscr{R}$ is $|X| \cdot 5 + |C| + 1$ and the size of $\mathscr{L}$ equals $|X| \cdot 5 + 4$. The size of each rule is bounded, as well.

We now prove properties of Reduction 1 that we require for subsequent complexity proofs.

Lemma 13. *Let ϕ be a Boolean formula in 3-CNF. For ABA$^+$ $F = \texttt{red}(\phi)$ (recall Reduction 1), the following statements are equivalent.*

1. *$\{a, b\}$ reversely $<$-attacks all assumption sets X that $<$-attack $\{a, b\}$.*
2. *$\{a, b\}$ is $<$-admissible in F.*
3. *$\{a, b\}$ is $<$-grounded in F.*
4. *ϕ is unsatisfiable.*

Proof. $1 \Leftrightarrow 2$: Assume that statement 1 holds. Note that $\{a, b\}$ is conflict-free. Since any $<$-attack from an assumption set to $\{a, b\}$ is countered by a reverse $<$-attack from $\{a, b\}$, we know that $\{a, b\}$ is $<$-admissible. For the other direction, assume that statement 2 holds. Now $\{a, b\}$ is conflict-free. Note that $\{a, b\}$ does not normally $<$-attack any assumption set. Thus $\{a, b\}$ is $<$-admissible iff $\{a, b\}$ $<$-attacks all assumption sets that $<$-attack it iff $\{a, b\}$ reversely $<$-attacks all assumption sets that $<$-attack it.

$2 \Rightarrow 3$: Assume that statement 2 holds. We argue that $\{a, b\}$ is the unique $<$-complete assumption set of F and thus it is the $<$-grounded assumption set. First note that $\{a, b\}$ does not derive further sentences by the given rules, that is, $Th_{\mathscr{R}}(\{a, b\}) = \{a, b\}$. It holds that $\{b\}$ is not $<$-attacked, since all rules deriving $\overline{b}$ contain an assumption that is strictly less preferred than b, and thus any such potential attack is reversed. This implies that any $<$-complete assumption set contains b. However, $\{b\}$ is not $<$-complete. This follows from the facts that, by assumption, $\{a, b\}$ defends itself, and if $\{a, b\}$ defends itself, then $\{b\}$ defends $\{a, b\}$. To see the latter, consider any set X that is $<$-attacked by $\{a, b\}$. First note that X is not normally $<$-attacked by $\{a, b\}$, since no further sentence is derivable from $\{a, b\}$. Thus X is reversely $<$-attacked by $\{a, b\}$. Since b is part of a preference and a is not, it holds that $\{b\}$ reversely $<$-attacks X if $\{a, b\}$ $<$-attacks X. Thus $\{b\}$ $<$-defends $\{a, b\}$ against any $<$-attack. It follows that any $<$-complete assumption set is a superset

$$
\begin{array}{rcl}
\texttt{red}(\phi) & = & (\mathscr{L}, \mathscr{R}, \mathscr{A}, {}^{-}, \leq) \\
\text{sentences } \mathscr{L} & = & \{a, b, \overline{a}, \overline{b}, x_1, \neg x_1, x_2, \neg x_2, x_3, \neg x_3, \\
 & & \overline{x_1}, \overline{\neg x_1}, \overline{x_2}, \overline{\neg x_2}, \overline{x_3}, \overline{\neg x_3}, d_1, d_2, d_3\} \\
\text{assumptions } \mathscr{A} & = & \{a, b, x_1, x_2, x_3, \neg x_1, \neg x_2, \neg x_3\} \\
\text{rules } \mathscr{R} & = & \{(d_1 \leftarrow x_1),\ (d_1 \leftarrow \neg x_1),\ (d_2 \leftarrow x_2), \\
 & & (d_2 \leftarrow \neg x_2),\ (d_3 \leftarrow x_3),\ (d_3 \leftarrow \neg x_3), \\
 & & (\overline{a} \leftarrow d_1, d_2, d_3),\ (\overline{x_1} \leftarrow x_1),\ (\overline{\neg x_1} \leftarrow \neg x_1), \\
 & & (\overline{x_2} \leftarrow x_2),\ (\overline{\neg x_2} \leftarrow \neg x_2),\ (\overline{x_3} \leftarrow x_3), \\
 & & (\overline{\neg x_3} \leftarrow \neg x_3),\ (\overline{b} \leftarrow x_1, \neg x_1, \overline{a}), \\
 & & (\overline{b} \leftarrow x_2, \neg x_2, \overline{a}),\ (\overline{b} \leftarrow x_3, \neg x_3, \overline{a}), \\
 & & (\overline{b} \leftarrow \neg x_1, \neg x_2, x_3, \overline{a}),\ (\overline{b} \leftarrow x_1, \neg x_2, \neg x_3, \overline{a})\} \\
\text{preferences} & & b > x_1, b > x_2, b > x_3, \\
 & & b > \neg x_1, b > \neg x_2, b > \neg x_3
\end{array}
$$

$<$-attacking $\{a,b\}$	$<$-attacked by $\{a,b\}$
$\{x_1, \neg x_1, \ldots\}$	yes
$\{x_2, \neg x_2, \ldots\}$	yes
$\{x_3, \neg x_3, \ldots\}$	yes
$\{x_1, x_2, x_3\}$	no
$\{x_1, x_2, \neg x_3\}$	no
$\{x_1, \neg x_2, x_3\}$	no
$\{x_1, \neg x_2, \neg x_3\}$	yes
$\{\neg x_1, x_2, x_3\}$	no
$\{\neg x_1, x_2, \neg x_3\}$	no
$\{\neg x_1, \neg x_2, x_3\}$	yes
$\{\neg x_1, \neg x_2, \neg x_3\}$	no

Figure 2: Example of Reduction 1 for formula $\phi = (x_1 \vee x_2 \vee \neg x_3) \wedge (\neg x_1 \vee x_2 \vee x_3)$

of $\{a,b\}$. Moreover, any proper superset of $\{a,b\}$ is conflicting, since from any x_i or $\neg x_i$ one can derive the corresponding contrary $\overline{x_i}$ or $\overline{\neg x_i}$. In conclusion, if $\{a,b\}$ is $<$-admissible, $\{a,b\}$ is the unique $<$-complete assumption set, implying that $\{a,b\}$ is $<$-grounded.

$3 \Rightarrow 4$: Assume that statement 3 holds. Any proper superset of $\{a,b\}$ is conflicting (and any set of assumptions incomparable to $\{a,b\}$ is conflicting, as well), and, since $\{a,b\}$ is $<$-grounded, by definition no proper subset of $\{a,b\}$ can be $<$-complete. This implies that $\{a,b\}$ is $<$-complete, and thus $<$-admissible. For contradiction, suppose that ϕ is satisfiable. We show that then $\{a,b\}$ is not $<$-admissible. Let τ be a truth assignment that satisfies ϕ. Construct $B = \{x_i \mid \tau(x_i) = 1\} \cup \{\neg x_i \mid \tau(x_i) = 0\}$. Since τ assigns each variable in X to either true or false, by construction of the reduction, it holds that $B \vdash_{\mathscr{R}} \overline{a}$. Since τ satisfies ϕ, τ satisfies each clause c_j. Then B does not contain all sentences of any $neg(c_j)$. Thus, $B \nvdash_{\mathscr{R}} \overline{b}$ and so $\{b\}$ does not reversely $<$-attack B. Recall also that $\{a,b\}$ does not normally $<$-attack anything. This implies that $\{a,b\}$ is not $<$-admissible: B $<$-attacks $\{a,b\}$ by normally $<$-attacking a, but B is not $<$-attacked by $\{a,b\}$. Thus ϕ can not be satisfiable and instead statement 4 holds.

$4 \Rightarrow 2$: Assume that statement 4 holds. We show that this implies statement 2, completing the proof. For contradiction, assume that $\{a,b\}$ is not $<$-admissible. As $\{a,b\}$ is conflict-free, there is a $B \subseteq \mathscr{A}$ such that B $<$-attacks $\{a,b\}$ and $\{a,b\}$ does not $<$-attack B. Since $\{a,b\}$ can not be reversely $<$-attacked, B normally $<$-attacks $\{a,b\}$. If $B \vdash_{\mathscr{R}} \overline{b}$, then $\{a,b\}$ reversely $<$-attacks B (since b is preferred to any assumption required to derive either $\overline{a}$ or $\overline{b}$). Thus by assumption $B \nvdash_{\mathscr{R}} \overline{b}$ and $B \vdash_{\mathscr{R}} \overline{a}$. It follows that B contains exactly one of x_i or $\neg x_i$ for each $1 \leq i \leq n$ (one of each is required to derive $\overline{a}$, but if both are present, $\overline{b}$ is derived). This defines a truth assignment on X: true if x_i is present in B, false otherwise. Since B does not derive $\overline{b}$, all bodies of rules $\overline{b} \leftarrow neg(c_j), \overline{a}$ for $1 \leq j \leq m$ are unsatisfied. Thus B satisfies each clause: B does not contain all elements from $neg(c_j)$ iff B satisfies at least one literal in clause c_j iff B satisfies c_j. Since this holds for all clauses, B satisfies ϕ. This implies that statement 2 holds. □

Lemma 13 allows for establishing the complexity deciding whether a set of assumptions $<$-defends itself against attacks.

Proposition 14. *Deciding whether a given set of assumptions A in an ABA$^+$ framework reversely counterattacks all sets that $<$-attack A is coNP-complete.*

Proof. For membership, consider the complementary problem, i.e., given an ABA$^+$ framework and a set of assumptions A, check whether there is a set B of assumptions that $<$-attacks A and A does not reversely $<$-attack B. This problem is in NP, since one can non-deterministically guess B and check in polynomial time whether B $<$-attacks A and whether A does not reversely $<$-attack B (by Proposition 11). Hardness follows from Lemma 13 via Reduction 1. □

Proposition 14 suggests that in order to verify that A reversely $<$-counterattacks all $<$-attacks from B, one needs to check for *each subset* B' of B whether B' is $<$-attacking and, furthermore, whether A $<$-attacks B'; i.e., knowledge of $<$-attacks on B or any $\{b\} \subseteq B$ is not sufficient. In particular, this differentiates the complexity of verifying $<$-admissibility from $<$-stability (recall verification under $<$-stable is in P by Proposition 12).

Theorem 15. *Verifying that a set of assumptions is $<$-admissible in ABA$^+$ is coNP-complete.*

Proof. Hardness follows directly from Lemma 13 and Reduction 1. For membership in coNP, consider an arbitrary ABA$^+$ framework and a set of assumptions A. Conflict-freeness of A is decidable in polynomial time. By Proposition 11 one can determine in polynomial time all assumptions $a \in \mathscr{A}$ such that A normally $<$-attacks $\{a\}$. Denote this set of assumptions by $X = \{a \in \mathscr{A} \mid A \text{ normally } <\text{-attacks } \{a\}\}$. Any $<$-attack from an assumption set intersecting with X is $<$-defended against by A. Only reverse $<$-attacks from A can $<$-defend against $<$-attacks from $\mathscr{A} \setminus X$, since there is no normal attack from A against $\mathscr{A} \setminus X$. Thus it suffices to check whether all $B \subseteq (\mathscr{A} \setminus X)$ that $<$-attack A are counterattacked by a reverse $<$-attack from A. This check is in coNP by Proposition 14. □

The coNP-hardness of verifying that a set is $<$-grounded or $<$-complete is coNP-hard from Lemma 13 by observing that in the proof of Lemma 13, if ϕ is unsatisfiable, there is a unique $<$-complete assumption set.

Corollary 16. *Verifying that a set of assumptions is $<$-complete or $<$-grounded in ABA$^+$ is coNP-hard.*

In non-preferential ABA frameworks F one can determine the unique grounded assumption set in polynomial time via iteratively applying the function $def_F(\emptyset)$ that collects singleton assumptions defended from the empty set of assumptions until a fixed point is reached (the formal base for this can be seen in Proposition 2). That is, $def_F^i(\emptyset)$ results in the grounded assumption set for some $i \geq 0$.

In general preferential ABA$^+$ frameworks F, the same procedure can yield the $<$-grounded assumption set, in case a natural property is satisfied for F. In the seminal paper by Phan Minh Dung, the fundamental lemma (Dung, 1995, Lemma 10) states that if a set S of arguments is admissible and defends an argument a in an AF, then $S \cup \{a\}$ is admissible in the same AF. The analogous property does not hold in ABA$^+$ in general, but in case an ABA$^+$ framework F does satisfy the property, then iteratively applying def_F starting with the empty set of assumptions results in the $<$-grounded assumption set. This result was already stated by Čyras and Toni (2016a, proof of Theorem 14 (iv)) and Čyras (2017, proof of Theorem 4.7 (iv)). We next formally define (recall) the property on ABA$^+$ frameworks, and give a direct proof for the sake of clarity (recall that we assume ABA$^+$ frameworks to be finite, which simplifies the proof).

Property 1. *An ABA$^+$ framework $F = (\mathscr{L}, \mathscr{R}, \mathscr{A}, ^-, \leq)$ satisfies the FL-property if $A \cup \{x\}$ is $<$-adm in F for any $<$-admissible $A \subseteq \mathscr{A}$ in F and $x \in def_F(A)$.*

If an ABA$^+$ framework satisfies the FL-property, $def_F^i(\emptyset)$ is the $<$-grounded assumption for a sufficiently high $i \geq 0$, as stated next formally.

Lemma 17. *Let $F = (\mathscr{L}, \mathscr{R}, \mathscr{A}, ^-, \leq)$ be an ABA$^+$ framework satisfying the FL-property. It holds that $def_F^i(\emptyset)$ is the $<$-grounded assumption set of F, for some integer $i \geq 0$.*

Proof. We first show that def_F is $\subseteq$-monotone. Let $A \subseteq B \subseteq \mathscr{A}$. If $a \in def_F(A)$, then A $<$-attacks any C which $<$-attack $\{a\}$. Since $<$-attacks are $\subseteq$-monotone, it holds that B $<$-attacks C as well, thereby defending $\{a\}$. Thus, def_F is $\subseteq$-monotone.

Second, we show that iterating $def_F(\emptyset)$ results in a fixed point $def_F^i(\emptyset) = def_F^{i+1}(\emptyset)$, for some $i \geq 0$. It straightforwardly holds that $\emptyset \subseteq def_F(\emptyset)$. By $\subseteq$-monotonicity we have $def_F(\emptyset) \subseteq def_F^2(\emptyset)$, and, more generally, $def_F^i(\emptyset) \subseteq def_F^{i+1}(\emptyset)$. Since $def_F^{i+1}(\emptyset) \subseteq \mathscr{A}$, and $\mathscr{A}$ is finite, we conclude that there is an integer $i \geq 0$ such that $def_F^i(\emptyset) = def_F^{i+1}(\emptyset)$.

Let $G = def_F^i(\emptyset)$ for $def_F^i(\emptyset) = def_F^{i+1}(\emptyset)$. Thirdly, we show that if C is $<$-complete in F, then $G \subseteq C$. We show this by induction: we show $def_F^i(\emptyset) \subseteq C$, for any $i \geq 0$. Base case: $\emptyset \subseteq C$. Inductive step: assume that $def_F^i(\emptyset) \subseteq C$ holds for i. Then since def_F is $\subseteq$-monotone, $def_F^{i+1}(\emptyset) \subseteq def_F(C) = C$.

Finally, if $X \subseteq \mathscr{A}$ is $<$-admissible, we show that then $def_F(X)$ is $<$-admissible. If $def_F(X) = X$ then the claim follows immediately. It holds that $X \cup \{x\}$ is $<$-admissible for an $x \in def_F(X) \setminus X$ by assumption that the FL-property holds. Since $def_F(X) \subseteq def_F(X \cup \{x\})$, it follows that $X \cup \{x, y\}$ is $<$-admissible for an $y \in def_F(X) \setminus (X \cup \{x\})$ (i.e., we can add y to $X \cup \{x\}$ and preserve $<$-admissibility). Since $\mathscr{A}$ is finite, we can infer that $def_F(X)$ is $<$-admissible. In turn, G, as defined above, is $<$-grounded in F. □

One can directly bound the number of applications i by $|\mathscr{A}|$ (in each iteration either at least one assumption is added, or we arrived at a fixed point). In practice the fixed point may be reached much earlier.

The preceding formal results yield a complexity-theoretic upper bound for $<$-grounded semantics: in particular, we show that the $<$-grounded assumption set can be computed via a deterministic polynomial-time algorithm with access to an NP-oracle (i.e., the functional problem for $<$-grounded semantics is in FP^{NP}) for ABA$^+$ frameworks satisfying the FL-property.

Proposition 18. *The $<$-grounded assumption set can be computed via a deterministic polynomial-time algorithm that can access an NP-oracle in an ABA$^+$ framework satisfying the FL-property.*

Proof. Let $F = (\mathscr{L}, \mathscr{R}, \mathscr{A}, ^-, \leq)$ be an ABA$^+$ framework satisfying the FL-property. By Lemma 17, it holds that the least fixpoint of def_F is equal to the $<$-grounded assumption set. Consider the (sub)problem of checking whether a given set of assumptions X $<$-defends a singleton assumption set $\{a\}$. We claim that this problem is in coNP. To see this, consider the complementary problem, i.e., checking whether X does not $<$-defend $\{a\}$. Non-deterministically construct an assumption set $Y \subseteq \mathscr{A}$. Check whether Y $<$-attacks $\{a\}$ (decidable in polynomial time due to Proposition 11). If this is the case, check whether X does not $<$-attack Y (again decidable in polynomial time). By definition, it holds that there is a Y that $<$-attacks $\{a\}$ and is not $<$-attacked by X iff X does not $<$-defend $\{a\}$. The preceding observation implies that $def_F(X)$ can be computed with polynomial many calls to an NP-oracle. It holds that at most $|\mathscr{A}|$ iterative applications of def_F, starting with $\emptyset$, yields the $<$-grounded assumption set. Overall, the $<$-grounded assumption set can be computed via at most $|\mathscr{A}|^2$ many calls to an NP-oracle: at most $|\mathscr{A}|$ many calls for a single application of def_F and after at most $|\mathscr{A}|$ many such applications a fixpoint is reached. □

A property of ABA$^+$ that guarantees that the FL-property is satisfied is the Axiom of Weak Contraposition (WCP) (Čyras, 2017; Čyras & Toni, 2016a, 2016c). However, assuming WCP for an ABA$^+$ framework does not appear to yield overall milder complexity. In fact the hardness results of Proposition 14 and Theorem 15 hold when assuming WCP. We formally present the corresponding results, and definition of WCP, in Appendix A in Definition 9, Proposition 21, and Theorem 22. All complexity results presented in this section are independent of WCP, except Proposition 18, which assumes the FL-property. WCP implies the FL-property, but there can be other restrictions that imply the FL-property.

Finally, we turn to credulous reasoning under $<$-admissible semantics and establish that this problem is Σ_2^{P}-complete. In particular, this result shows that reasoning under preferences in ABA$^+$ faces significantly higher complexity in the general case. An intuition for the upper bound can be seen by a non-deterministic construction of an assumption set, and verifying whether this set is $<$-admissible, with the latter problem being coNP-complete (see Theorem 15).

Theorem 19. *Checking whether a sentence is credulously accepted w.r.t. $<$-admissible semantics in ABA$^+$ is Σ_2^P-complete.*

Proof. This more involved proof can be found in Appendix A. □

4. An ASP-Based Approach to Reasoning in ABA and ABA$^+$

Complementing our theoretical results, we develop an efficient approach to reasoning in ABA and ABA$^+$ by harnessing answer set programming (ASP) (Gelfond & Lifschitz, 1988; Niemelä, 1999). Before detailing the ASP encodings and ASP-based algorithms which allow for deciding skeptical and credulous acceptance as well as assumption set enumeration using ASP solvers, we shortly overview answer set programs as applied in this work.

4.1 Answer Set Programming

We recall preliminaries on answer set programming (ASP); the reader is also referred to overview articles on ASP for further background (Brewka et al., 2011; Gebser et al., 2012).

We employ the standard NP-complete language of normal answer set programs as the basis of our ASP-based approach. A (normal) answer set program π consists of rules r of the form

$$h \leftarrow b_1, \ldots, b_k, \mathit{not}\ b_{k+1}, \ldots, \mathit{not}\ b_m.$$

where h (the head of the rule) and each b_i (constituting the body of the rule) are atoms. An atom b_i is of the form $p(t_1,\ldots,t_n)$ with p a predicate of arity n, and each t_j either a constant or a variable. An answer set program, a rule, and an atom, respectively, is ground if it is free of variables. We will use the convention that the first letter of a variable is uppercase and that of constants is lowercase.

A rule is positive if $k = m$ (i.e., the body contains no negated atoms), a fact if $m = 0$ (i.e., the body is empty). A fact is shortened to "h." by omitting $\leftarrow$. A constraint $\leftarrow b_1, \ldots, b_k, \mathit{not}\ b_{k+1}, \ldots, \mathit{not}\ b_m$ is a shorthand for $x \leftarrow b_1, \ldots, b_k, \mathit{not}\ b_{k+1}, \ldots, \mathit{not}\ b_m, \mathit{not}\ x$, where x is a fresh ground atom not occurring anywhere else in the program. We do not make use of functions within ASP programs.

For a non-ground program, let GP be the set of rules obtained by applying all possible substitutions from the variables to the set of constants appearing in the program. An interpretation I, i.e., a subset of all the ground atoms, satisfies a positive rule $r = h \leftarrow b_1,\ldots,b_k$ iff presence of all positive body elements $b_1,\ldots,b_k$ in I implies that the head atom is in I, i.e., $\{b_1,\ldots,b_k\} \subseteq I$ implies $h \in I$. For a program π consisting only of positive rules, let $Cl(\pi)$ be the uniquely determined interpretation I that satisfies all rules in π and no subset of I satisfies all rules in π. Interpretation I is an answer set of a ground program π if $I = Cl(\pi^I)$ where $\pi^I = \{(h \leftarrow b_1,\ldots,b_k) \mid (h \leftarrow b_1,\ldots,b_k,\mathit{not}\, b_{k+1},\ldots,\mathit{not}\, b_m) \in \pi, \{b_{k+1},\ldots,b_m\} \cap I = \emptyset\}$ is the reduct. An interpretation I is an answer set of a non-ground program π if I is an answer set of GP of π. A program is satisfiable if the program has at least one answer set and unsatisfiable otherwise.

We employ the state-of-the-art ASP system CLINGO (Gebser et al., 2011, 2016). In addition to determining the existence of answer sets and enumerating all answer sets, CLINGO supports cautious reasoning via its cautious mode, in which case it computes the intersection of all answer sets (Gebser et al., 2016). We make use of this feature in our algorithm for computing the ideal assumption set. We also use the ASP optimization framework ASPRIN (Brewka et al., 2015) extending CLINGO. In particular, as detailed later on, we will make use of the answer set programs with optimization statements that enforce that only answer sets that are $\subseteq$-maximal w.r.t. a specified predicate p of arity one are returned (i.e., I is an optimal answer set if there is no answer set J such that $\{p(t) \mid p(t) \in I\} \subset \{p(t) \mid p(t) \in J\}$). This extension is in particular useful for computing preferred assumption sets.

4.2 ASP Encodings for ABA

In this section we present algorithms for reasoning in ABA based on applying ASP solvers on encodings of ABA and ABA$^+$ semantics and reasoning tasks. In particular, we provide algorithms for deciding the credulous and skeptical acceptability of sentences as well as for enumerating σ-assumption sets under admissible, complete, preferred, stable, grounded, and ideal semantics.

Except for the case of ideal semantics in ABA and $<$-grounded semantics in ABA$^+$, our approach is based on encoding ABA and ABA$^+$ semantics in ASP in such a way that A is a σ-assumption set of a given ABA framework F if and only if there is an answer set M of $\pi_\sigma \cup \mathtt{ABA}(F)$

where $A = \{a \mid \mathbf{in}(a) \in M\}$, π_σ is the ASP encoding for semantics σ and $\mathtt{ABA}(F)$ is the ASP encoding for F. For ideal and $<$-grounded semantics, we present more complex algorithms which use an ASP solver as a subprocedure.

4.2.1 REPRESENTING FRAMEWORKS AND QUERIES IN ASP

As the basis of the ASP approach, we start with an encoding of a given ABA framework $F = (\mathscr{L}, \mathscr{R}, \mathscr{A}, ^-)$ with $\mathscr{R} = \{r_1, \ldots, r_n\}$, in particular the assumptions, rules and contraries of the framework. We represent the framework F in ASP as the facts

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

The predicate $\mathbf{assumption}(a)$ indicates that a is an assumption, whereas $\mathbf{contrary}(a,b)$ indicates that b is the contrary of a. The rules are expressed via separate predicates for heads and bodies of rules, with a unique rule index linking them. The predicate $\mathbf{head}(i,b)$ is interpreted as b being the head of the rule with index i. Similarly, $\mathbf{body}(i,b)$ indicates that b is contained in the body of the rule with index i.

Example 10. *Consider the ABA framework F with*

$$\begin{aligned}\text{sentences } \mathscr{L} &= \{a,b,x,y\}, \\ \text{assumptions } \mathscr{A} &= \{a,b\}, \\ \text{contraries} \quad \overline{a} &= y,\ \overline{b} = x, \\ \text{rules } \mathscr{R} &= \{(x \leftarrow a,y), (y \leftarrow b)\}.\end{aligned}$$

F is represented in ASP as the set of facts

$$\begin{aligned}\mathtt{ABA}(F) = \{\ &\mathbf{assumption}(a).\ \mathbf{assumption}(b). \\ &\mathbf{head}(1,x).\ \mathbf{body}(1,a).\ \mathbf{body}(1,y). \\ &\mathbf{head}(2,y).\ \mathbf{body}(2,b). \\ &\mathbf{contrary}(a,y).\ \mathbf{contrary}(b,x).\ \}\end{aligned}$$

For reasoning about acceptance, we will furthermore us the predicate **query** to represent the sentence whose acceptance status is queried, i.e., a queried sentence $s \in \mathscr{L}$ will be specified in ASP by adding the fact

$$\mathbf{query}(s).$$

to the ASP encoding of semantics at hand.

For compactness of presentation, in the following we identify the set of constants for which a predicate holds by the predicate; e.g. we say "**in** attacks..." to mean that "the set of assumptions for which the predicate **in** holds, attack...".

Listing 1: Module π_{common}

```
1  in(X) ← assumption(X), not out(X).
2  out(X) ← assumption(X), not in(X).
3  supported(X) ← assumption(X), in(X).
4  supported(X) ← head(R,X), triggered_by_in(R).
5  triggered_by_in(R) ← head(R,_), supported(X) : body(R,X).
6  defeated(X) ← supported(Y), contrary(X,Y).
7  ← in(X), defeated(X).
```

4.2.2 CONFLICT-FREE SETS AND DERIVATIONS

We begin with an ASP module (subprogram) π_{common} (see Listing 1); this module encodes conflict-free assumption sets and will be as such common to the encodings of several semantics. Lines 1–2 encode a non-deterministic guess of a subset of the given assumptions, with **in** and **out** denoting what is inside and outside the set, respectively. Lines 3–5 encode forward-derivations via the ASP predicate **supported** (recall that in ABA we can focus on forward-derivations only). Formally, for an assumption set A represented via **in**, if $A \vdash_{\mathscr{R}} x$, then **supported**(x) is included in an answer set. Line 3 encodes the base case, i.e., assumptions in the set. Line 4 encodes that whenever a rule is "triggered", i.e., all its body elements are derivable from A, then the **head** of that rule shall be derived. Line 5 encodes the triggering of rules via a conditional construct.[9] Thus the rule is triggered when each sentence in its body is supported. The sixth rule derives attacked (defeated) assumptions by the guessed assumption set. The last rule is a constraint to enforce conflict-freeness: an assumption can not be both **in** and attacked by **in**.

4.2.3 CONSTRAINTS FOR CREDULOUS AND SKEPTICAL ACCEPTANCE

Before detailing encodings for additional semantics, we describe constraints which, together with the base module π_{common} common to several semantics, capture credulous and skeptical reasoning. We employ these constraints for capturing reasoning under semantics for which we do not need optimization via ASPRIN, i.e., all but preferred semantics.

The rule

$$\leftarrow \mathit{not}\,\mathbf{supported}(X), \mathbf{query}(X).$$

with predicate **supported** defined in π_{common} and **query** specifying the sentence queried for credulous acceptance, rules out answers where X is not derived from an assumption set. Thus this constraint captures credulous reasoning in terms of the query; the resulting answer set program does not have any answer sets iff the queried sentence is not credulously accepted. To capture skeptical reasoning, the rule

$$\leftarrow \mathbf{supported}(X), \mathbf{query}(X).$$

enforces a counterexample check for skeptical acceptance. In particular, the resulting answer set program does not have any answer sets iff the queried sentence is skeptically accepted.

9. The conditional **supported**(X) : **body**(R,X) holds when the ASP atoms **supported**(X) are present for each **body**(R,X) of the given rule index R. In particular, conditional literals allow for a succinct presentation of a rule with a variable sized body. Gebser et al. (2015) explain the construct in more detail.

4.2.4 STABLE, ADMISSIBLE, COMPLETE AND PREFERRED SEMANTICS

We continue by building on π_{common} to obtain encodings for stable, admissible, complete and preferred semantics.

The program for stable semantics π_{stb} is π_{common} conjoined with the constraint

$$\leftarrow \mathbf{out}(X), \mathit{not}\, \mathbf{defeated}(X).$$

This constraint enforces that all assumptions are either **in** or attacked by **in**, corresponding to the definition of stable semantics.

For admissibility, recall that a conflict-free assumption set A is admissible iff the set of assumptions that are not attacked by A does not attack A (Proposition 4). This is implemented as an ASP program in Listing 2. The idea is to check whether a contrary of an **in** assumption is derivable from undefeated assumptions. Checking if the undefeated assumptions attack **in** assumptions (on Lines 1–4) is done in a similar manner as checking what the **in** set attacks (on Lines 3–6 of π_{common}). The ASP encoding for admissibility is $\pi_{adm} = \pi_{common} \cup \Delta_{adm}$.

The program for complete semantics π_{com} is given by conjoining π_{adm} with the rule

$$\leftarrow \mathbf{out}(X), \mathit{not}\, \mathbf{attacked\_by\_undefeated}(X).$$

This constraint enforces that there are no assumptions that are **out** and defended by **in**. If an assumption X is out and not attacked by assumptions that are undefeated, then that assumption is defended but not in the defending assumption set. The constraint rules out this situation.

To compute preferred assumption sets, we use preferential optimization statements supported by the ASPRIN system (Brewka et al., 2015), which uses CLINGO. Recall that an admissible (or complete) assumption set A is preferred iff there is no superset of A that is also admissible (complete). This property can be presented as the preferential optimization statements

$$\#\mathbf{preference}(p1, \text{superset})\{\mathbf{in}(X) : \mathbf{assumption}(X)\}.$$
$$\#\mathbf{optimize}(p1).$$

which enforce that only answer sets are returned that are subset-maximal w.r.t. the **in** predicate (mirroring subset maximality).[10] In particular, adding this statement to the ASP encoding of admissibility allows for computing preferred assumptions sets using ASPRIN. ASPRIN has built-in support for querying, which allows for directly capturing credulous and skeptical reasoning under preferred semantics.

10. Concretely, the first line specifies the optimization ("preference") of supersets w.r.t. sets represented by **in**, and the second line that this optimization shall be applied.

Listing 2: Module Δ_{adm}

```
1 derived_from_undefeated(X) ← assumption(X), not defeated(X).
2 derived_from_undefeated(X) ← head(R,X), triggered_by_undefeated(R).
3 triggered_by_undefeated(R) ← head(R,_), derived_from_undefeated(X) : body(R,X).
4 attacked_by_undefeated(X) ← contrary(X,Y), derived_from_undefeated(Y).
5 ← in(X), attacked_by_undefeated(X).
```

Listing 3: Module π_{grd}

```
1  n_assumptions(N) ← #count{X : assumption(X)} = N.
2  iteration(0..N−1) ← n_assumptions(N).
3  in(X,I) ← iteration(J), assumption(X), not attacked_by_undefeated(X,J), J+1=I.
4  supported(X,I) ← assumption(X), in(X,I).
5  supported(X,I) ← head(R,X), triggered_by_in(R,I).
6  triggered_by_in(R,I) ← iteration(I), head(R,_), supported(X,I) : body(R,X).
7  defeated(X,I) ← supported(Y,I), contrary(X,Y).
8  derived_from_undefeated(X,I) ← iteration(I), assumption(X), not defeated(X,I).
9  derived_from_undefeated(X,I) ← head(R,X), triggered_by_undefeated(R,I).
10 triggered_by_undefeated(R,I) ← iteration(I), head(R,_), derived_from_undefeated(X,I) : body(R,X).
11 attacked_by_undefeated(X,I) ← contrary(X,Y), derived_from_undefeated(Y,I).
```

4.2.5 GROUNDED SEMANTICS

Our encoding for grounded semantics is based on Lemma 7, which states that, given an ABA framework F, the least fixpoint of def_F and thus the grounded assumption set of F can be computed via the sequence $\mathtt{S}(F) = (\mathtt{S}_0, \ldots, \mathtt{S}_{|\mathscr{A}|})$, where (i) $\mathtt{S}_0 = (\emptyset, att(\emptyset), \mathscr{A})$ and (ii) $\mathtt{S}_i = (I_i, D_i, U_i)$ with $I_i = \mathscr{A} \setminus att(U_{i-1})$, $D_i = att(I_i)$, and $U_i = \mathscr{A} \setminus D_i$. In words, I_i is the set of assumptions defended by I_{i-1}, U_i is the set of assumptions not attacked by I_i, and D_i is the set of assumptions attacked by I_i. The grounded assumption set of F equals $I_{|\mathscr{A}|}$.

We implement the computation of this sequence explicitly in ASP in Listing 3. Lines 1–2 set the number of iterations to equal the number of assumptions in the given framework. In particular, **#count** is an ASP construct that gives the number of elements in a given set. Here on Line 1 it is used to assert that the number of assumptions is N and letting **n_assumptions** hold for this N. The predicate **in** corresponds to I_i and **defeated** corresponds to D_i. The set U_i is given by *not* **defeated**. On Line 3, **in** is determined based on the previous iteration: assumptions that are defended by **in** of the previous iteration are included. Note that since **in** is defined for all I such that I-1 is an **iteration**, the first I for which **in** is computed is 1, giving the set of assumptions defended by $\emptyset$, and the last such I is $|\mathscr{A}|$. Lines 4–6 determine the sentences derivable from **in** at the current iteration and Line 7 defines **defeated** based on this. Lines 8–10 determine what is derivable from *not* **defeated**. Finally, Line 11 determines the assumptions that are attacked by *not* **defeated**, which is used for determining **in** at the next iteration. The grounded assumption set is given by **in** after the last iteration. In other words, every assumption X for which $\mathbf{in}(X, |\mathscr{A}|)$ holds is included in the grounded assumption set.

Remark 1. *The encoding for grounded semantics given in the preliminary version of this article (Lehtonen et al., 2019) is erroneous; for more details, see Appendix B.*

4.3 Ideal Semantics

We adapt the algorithm for computing the ideal assumption set from Dunne (2009) in Algorithm 1. Recall that the ideal assumption set is the maximal admissible set that is a subset of each preferred assumption set. This algorithm first computes an overapproximation of the ideal assumption set, namely all assumptions credulously accepted under admissible semantics (Lines 1–2) apart from the assumptions attacked by this set, resulting in $\mathscr{A}_{\mathrm{PSA}}$ (Lines 3–4). The algorithm then refines

Listing 4: Module π_{prefs}

```
1 preferred(X,Z) ← preferred(X,Y), preferred(Y,Z).
2 strictly_less_preferred(X,Y) ← preferred(Y,X), not preferred(X,Y).
3 no_less_preferred(X,Y) ← assumption(X), assumption(Y), not strictly_less_preferred(X,Y).
```

the answer towards the ideal assumption set by iteratively removing assumptions not defended by the set. Concretely in the loop of Lines 6–11, Γ (which equals $\mathscr{A}_{\text{PSA}}$ at first) is refined into the ideal assumption set by repeatedly identifying (on Line 9) the assumptions in Γ that are attacked by assumptions not attacked by Γ (computed on Line 8) and removing them from Γ (on Line 10). This computation yields the ideal assumption set (Dunne, 2009, Theorem 8). Note that all parts of the algorithm except Line 1 can be computed in polynomial time. In terms of ASP, we obtain the set of assumptions that are not credulously accepted under admissible semantics using the cautious reasoning mode of CLINGO on the ASP encoding of admissible semantics. The cautious mode gives the intersection of all answer sets to the given program, so the assumptions that are **out** in the cautious solution constitute the set of assumptions that are not credulously accepted under admissible semantics, i.e., the set $\mathscr{A}_{out}$.

Remark 2. *The algorithm as presented by Dunne (2009, Algorithm 3) differs slightly from Algorithm 1 by corrections to Lines 8–9; Lines 8–9 as originally presented by Dunne (2009) result in the original algorithm being incorrect. For more details, see Appendix C.*

4.4 ASP Encodings for ABA$^+$

We now turn to ABA$^+$, and present an ASP encoding for $<$-stable semantics and an algorithm for $<$-grounded semantics making iterative calls to an ASP solver.

In the following, for a given ABA$^+$ framework $F = (\mathscr{L}, \mathscr{R}, \mathscr{A}, ^-, \leq)$, let $\texttt{ABA}^+(F) = \texttt{ABA}(F) \cup \{\textbf{preferred}(x,y). \mid y \leq x\}$ be the facts representing the ABA$^+$ framework. The module π_{prefs} detailed

Algorithm 1 Computing the ideal assumption set (Dunne, 2009, Algorithm 3)

Require: ABA framework $F = (\mathscr{L}, \mathscr{R}, \mathscr{A}, ^-)$
Ensure: return the *ideal*-assumption set of F

1: $\mathscr{A}_{\text{out}} := \{a \in \mathscr{A} \mid a \text{ is not credulously accepted under admissible semantics}\}$
2: $\mathscr{A}_{\text{in}} := \mathscr{A} \setminus \mathscr{A}_{\text{out}}$
3: $\mathscr{A}_{\text{CA}} := \{a \in \mathscr{A} \mid \mathscr{A}_{\text{in}} \vdash_{\mathscr{R}} \overline{a}\}$
4: $\mathscr{A}_{\text{PSA}} := \mathscr{A}_{\text{in}} \setminus \mathscr{A}_{\text{CA}}$
5: $\Gamma := \mathscr{A}_{\text{PSA}}$
6: **repeat**
7: $\quad \Gamma_{\text{in}} := \Gamma$
8: $\quad \Xi := \{a \in A \mid \Gamma \nvdash_{\mathscr{R}} \overline{a}\}$
9: $\quad \Delta := \{a \in \Gamma_{\text{in}} \mid \Xi \vdash_{\mathscr{R}} \overline{a}\}$
10: $\quad \Gamma := \Gamma \setminus \Delta$
11: **until** $\Gamma_{\text{in}} = \Gamma$
12: **return** Γ

Listing 5: Module Δ_{stb+}

```
1  preferredly_supported(X,Y) ← no_less_preferred(X,Y), assumption(X), in(X).
2  preferredly_supported(X,Y) ← head(R,X), preferredly_triggered_by_in(R,Y).
3  preferredly_triggered_by_in(R,Y) ← head(R,_), assumption(Y),
       preferredly_supported(X,Y) : body(R,X).
4  normally_defeated(Y) ← preferredly_supported(X,Y), contrary(Y,X).
5  derived_from_undefeated_assumption(Z,Z) ← assumption(Z), not normally_defeated(Z).
6  derived_from_undefeated_assumption(Y,Z) ← head(R,Y), triggered_by_undefeated_assumption(R,Z).
7  triggered_by_undefeated_assumption(R,Z) ← head(R,_), assumption(Z),
       derived_from_undefeated_assumption(Y,Z) : body(R,Y).
8  in_attacked_by_normally_undefeated_assumption(X,Z) ← in(X), contrary(X,Y),
       derived_from_undefeated_assumption(Y,Z).
9  reversely_defeated(Z) ← strictly_less_preferred(Z,X),
       in_attacked_by_normally_undefeated_assumption(X,Z).
10 ← out(Y), not normally_defeated(Y), not reversely_defeated(Y).
```

in Listing 4 encodes transitivity of the preference relation and introduces the predicates for the strict counterpart of **preferred** (**strictly_less_preferred**) and its negation (**no_less_preferred**).

4.4.1 $<$-STABLE SEMANTICS

Finding $<$-stable assumption sets can be encoded in a similar manner as finding stable assumption sets. Recall the restatement of $<$-stable semantics from Proposition 9: a conflict-free assumption set A is $<$-stable iff each assumption $b \in \mathscr{A}$ that is not normally $<$-attacked by A is either in A or reversely $<$-attacked by A. Furthermore, by Lemma 8 one can check if a set of assumptions is $<$-stable by only using forward-derivability. Thus we can encode $<$-stable semantics via forward-derivability: perform an ASP guess of a set and check conflict-freeness with π_{common}, compute each $\{b\}$ that is normally $<$-attacked, and from the assumptions not normally $<$-attacked check whether they are reversely $<$-attacked. Crucially, these checks can be done in polynomial time (Proposition 11). Similarly to the ABA encodings, the $<$-stable assumption set is represented by the **in** predicate.

In more detail, our ASP encoding for $<$-stable semantics is $\pi_{stb+} = \Delta_{stb+}$ (Listing 5) $\cup \pi_{common} \cup \pi_{prefs}$. In Δ_{stb+}, the first four lines derive the assumptions that are normally $<$-attacked by **in**. This is achieved by modifying the **supported** predicate to take preferences into account. Concretely, the **preferredly_supported** predicate has a parameter, Y, for the assumptions which **in** might attack. For each Y, only the assumptions of **in** that are not less preferred than Y are taken into account when considering whether **in** derives the contrary of Y. Thus this predicate derives the normal $<$-attacks from **in** on Line 4. Lines 5–9 check whether **in** reversely $<$-attacks the assumptions that **in** does not normally $<$-attack. This is done by checking if the not normally $<$-attacked assumptions by themselves attack some assumption in **in** that is more preferred than the attacking assumption. Concretely, Lines 5–7 compute what the normally $<$-undefeated assumptions derive by themselves. Line 8 computes which assumptions of **in** each of these assumptions attack (when not taking preferences into account). Based on this and the preference relation, Line 9 computes which of these assumptions **in** reversely $<$-attacks. Finally, Line 10 enforces the condition of $<$-stable semantics: an assumption can not be **out** and neither normally nor reversely $<$-attacked by **in**.

Listing 6: Module $\pi_{grd+\ subroutine}$

```
1  suspect(X) ← assumption(X), not other(X).
2  other(X) ← assumption(X), not suspect(X).
3  preferredly_supported_by_suspects(X) ← target(Y), no_less_preferred(X,Y),assumption(X),
       suspect(X).
4  preferredly_supported_by_suspects(X) ← head(R,X),preferredly_triggered_by_suspects(R).
5  preferredly_triggered_by_suspects(R) ← head(R,_),
       preferredly_supported_by_suspects(X) : body(R,X).
6  target_normally_attacked ← target(Y), preferredly_supported_by_suspects(X), contrary(Y,X).
7  derived_from_target(X) ← target(X).
8  derived_from_target(X) ← head(R,X), triggered_by_target(R).
9  triggered_by_target(R) ← head(R,_), derived_from_target(X) : body(R,X).
10 suspect_attacked_by_target(X) ← suspect(X), contrary(X,Y), derived_from_target(Y).
11 target_reversely_attacked ← target(Y), strictly_less_preferred(Y,X), suspect_attacked_by_target(X).
12 supported_by_def(X,Y) ← suspect(Y), no_less_preferred(X,Y), assumption(X), def(X).
13 supported_by_def(X,Y) ← suspect(Y), head(R,X), triggered_by_def(R,Y).
14 triggered_by_def(R,Y) ← head(R,_), assumption(Y), supported_by_def(X,Y) : body(R,X).
15 suspect_normally_defeated_by_def ← supported_by_def(X,Y), contrary(Y,X).
16 supported_by_suspects(X) ← assumption(X), suspect(X).
17 supported_by_suspects(X) ← head(R,X), triggered_by_suspects(R).
18 triggered_by_suspects(R) ← head(R,_), supported_by_suspects(X) : body(R,X).
19 reaches_via_suspect(X,Y) ← triggered_by_suspects(R), head(R,Y), body(R,X).
20 reaches_via_suspect(X,Y) ← reaches_via_suspect(X,Z), reaches_via_suspect(Z,Y).
21 reaches_via_suspect(X,X) ← suspect(X).
22 suspect_reversely_defeated_by_def ← def(Y), contrary(Y,X), suspect(Z), reaches_via_suspect(Z,X),
       strictly_less_preferred(Z,Y).
23 ← suspect_normally_defeated_by_def.
24 ← suspect_reversely_defeated_by_def.
25 ← not target_normally_attacked, not target_reversely_attacked.
```

Algorithm 2 Computing the $<$-grounded assumption set

Require: ABA framework $F = (\mathscr{L}, \mathscr{R}, \mathscr{A}, ^-)$ satisfying the FL-property
Ensure: return the $<$-*grd*-assumption set of F

1: $grounded := \emptyset$
2: $\pi := \texttt{ABA}^+(F) \cup \pi_{prefs} \cup \pi_{grd+\ subroutine}$
3: **while** $change = true$ **do**
4: $\quad change := false$
5: $\quad S := \mathscr{A} \setminus grounded$
6: $\quad$ **while** $S \neq \emptyset$ **do**
7: $\quad\quad$ pick any $a \in S$
8: $\quad\quad$ **if** $\pi \cup \{\textbf{def}(x) \mid x \in grounded\} \cup \{\textbf{target}(a)\}$ is unsatisfiable **then**
9: $\quad\quad\quad grounded := grounded \cup \{a\}$
10: $\quad\quad\quad change := true$
11: $\quad\quad S := S \setminus \{a\}$
12: $\quad$ **end while**
13: **end while**
14: **return** *grounded*

4.4.2 $<$-GROUNDED SEMANTICS

We present a more complex ASP-based algorithm for finding the $<$-grounded assumption set in an ABA$^+$ framework that satisfies the FL-property. Recall that in an ABA$^+$ framework F that satisfies the FL-property, the $<$-grounded assumption set is the least fixpoint of the function $def_F(A)$[11], see Lemma 17. We implement the algorithm detailed in the proof of Proposition 18 that makes use of iterative computations of $<$-defense of singleton assumption sets. Our implementation uses an ASP encoding as a subroutine to determine whether a target assumption is $<$-defended by a given set of assumptions. To compute $def_F(A)$, one can use this encoding to check for each $a \in \mathscr{A}$ if A defends it. The encoding for the subroutine is π_{prefs} conjoined with $\pi_{grd+\ subroutine}$ (Listing 6). We present the procedure for computing the $<$-grounded assumption set as Algorithm 2. The outer loop starting on Line 3 computes the fixpoint of $def_F(\emptyset)$ by repeating the loop of Lines 6 to 12, which computes $def_F(grounded)$, where *grounded* is the thus far $<$-defended set. On Line 8 is the ASP subroutine determining whether a single assumption is $<$-defended by *grounded* (note that the single assumption is $<$-defended if the program is unsatisfiable).

Let us examine $\pi_{grd+\ subroutine}$ more closely. The currently $<$-defended assumptions are presented by the predicate **def** and the assumption whose status of being $<$-defended is being determined is presented by the predicate **target**. The encoding first non-deterministically guesses a set of assumptions (called suspects). The idea is to determine if the suspects attack the target without **def** attacking the suspects. On Lines 3-6 the set of assumptions normally $<$-attacked by the suspects is determined. Lines 7-11 compute the set of assumptions reversely $<$-attacked by the suspects. For this it suffices to check whether any suspect that the target attacks by itself is more preferred than the target. Lines 12–15 determine the suspects that **def** $<$-attacks normally. To compute the suspects that **def** reversely $<$-attacks, the reachability procedure of Proposition 11 (proven via Lemma 20) is implemented on Lines 16–22. Namely the predicate **reaches_via_suspect**(X,Y) holds when the

11. $def_F(A)$ gives the assumptions that the assumption set A $<$-defends.

suspects derive Y and there is a path from X to Y in a directed graph constructed by having an edge from all bodies of a rule to the head of the rule. Thus according to Proposition 11, Line 22 determines if a suspect is reversely defeated by a member of **def** by checking if a contrary of a member of **def** is reachable from a suspect that is less preferred than the member of **def**. Finally Lines 23–24 enforce that the suspects must not be $<$-attacked either normally or reversely by the $<$-defended set, and Line 25 enforces that the suspect set must $<$-attack the target either normally or reversely. Thus the program is satisfiable if and only if there is either a normal or reverse $<$-attack from any potential subset of the assumptions (the suspects) to the target assumption so that the currently $<$-defended set does not $<$-attack the suspects either normally or reversely. In other words, the target is $<$-defended by the currently $<$-defended set if and only if the program is unsatisfiable.

5. Empirical Evaluation

We present an overview of results from an empirical evaluation on the efficiency of reasoning in ABA and ABA$^+$ via the ASP-based approach presented in Section 4. In particular, we compare the performance of the ASP-based approach to that of state-of-the-art ABA and ABA$^+$ reasoning system implementations, and also present a further scalability study of the ASP-based approach. Before details on the empirical setup and results of the evaluation, we briefly overview earlier proposed algorithms and implemented systems for reasoning in ABA and ABA$^+$.

5.1 Systems for Reasoning in ABA and ABA$^+$

We briefly overview earlier proposed approaches to reasoning in ABA and ABA$^+$. Supported reasoning problems for each of the approaches are summarized in Table 4 for ABA and in Table 5 for ABA$^+$. Cerutti et al. (2018) provide a more thorough survey of argumentation systems.

Approaches to reasoning in ABA and ABA$^+$ can be categorized as one of three types of approaches: translation-based, specialized and direct declarative approaches.

The most notable specialized systems implement algorithms for so-called dispute derivations, which are procedures for determining the credulous acceptability of a given sentence under certain semantics (Toni, 2013; Craven et al., 2013; Craven & Toni, 2016). The most recent dispute derivation system is ABAGRAPH (Craven & Toni, 2016), improving on the earlier systems GRAPHARG (Craven et al., 2013), PROXDD (Toni, 2013) and CASAPI (Gaertner & Toni, 2007a), all implemented in Prolog. ABAGRAPH can reason credulously under admissible and grounded semantics, and also enumerate solutions, and further supports credulous reasoning under complete and preferred semantics via admissible semantics, as well as skeptical reasoning under complete semantics via grounded semantics.

Another specialized approach is implemented in TweetyProject, a Java library of various approaches to logical aspects of artificial intelligence and knowledge representation (Thimm, 2014, 2017). Tweety implements σ-assumption set enumeration under admissible, complete, stable, preferred, ideal and grounded semantics. However, Tweety is not optimized in terms of runtime behavior.

The translation-based approaches translate the input ABA frameworks to abstract argumentation frameworks and allow for the use of AF reasoning system implementations, such as ones based on ASP solvers (Egly et al., 2008; Gaggl et al., 2015). A problem of translation-based systems is that the translation between ABA and AF is itself not trivial in terms of runtime. In fact, there is evidence that the total runtimes of translation-based approaches such as ABA2AF is dominated

Table 4: ABA reasoning problem variants supported by different reasoning system implementations. ✓: the system natively supports the problem variant. *: the system supports the problem variant via another semantics. **: the system supports the problem variant via σ-assumption set enumeration.

system	**task**	*adm*	*com*	*stb*	*prf*	*grd*	*ideal*
ABAGRAPH	credulous	✓	*	-	*	✓	-
	skeptical	-	*	-	-	✓	-
	σ-assumption set enumeration	-	-	-	-	-	-
ABA2AF	credulous	✓	*	✓	✓	-	-
	skeptical	✓	-	✓	✓	-	-
	σ-assumption set enumeration	-	-	-	-	-	-
ASP	credulous	✓	✓	✓	✓	✓	✓
	skeptical	✓	✓	✓	✓	✓	✓
	σ-assumption set enumeration	✓	✓	✓	✓	✓	✓
translation of	credulous	-	-	✓	-	-	-
Caminada and Schulz (2017)	skeptical	-	-	✓	-	-	-
	σ-assumption set enumeration	-	-	✓	-	-	-
ABAPLUS	credulous	**	**	**	**	**	**
	skeptical	**	**	**	**	**	**
	σ-assumption set enumeration	✓	✓	✓	✓	✓	✓

by the translation phase (Lehtonen et al., 2017). The resulting AF instance may also be much larger than the input ABA framework; even though the complexity of deciding acceptance in AFs and ABA under the logic programming fragment considered in this work coincides for several reasoning tasks—suggesting that there should be efficient (polynomial-time) mappings for translating reasoning tasks in the ABA fragment into AF reasoning tasks—the currently known translations may produce exponentially larger AFs. The system ABA2AF translates ABA frameworks to AFs and uses ASP encodings on the AF-level (Lehtonen et al., 2017), supporting credulous and skeptical reasoning under admissible, stable and preferred semantics, and also provides an approach to credulous reasoning under complete semantics via admissible semantics.

A direct declarative approach encodes an ABA reasoning problem directly in a declarative language (such as ASP) and uses a solver for that language to solve the reasoning problem. This article provides such an approach, covering admissible, complete, stable, preferred, grounded and ideal semantics, allowing for credulous and skeptical reasoning as well as σ-assumption set enumeration. Recently a different approach, mapping ABA reasoning under different semantics via a single logic programming translation to different logic programming semantics has been presented by Caminada and Schulz (2017). While there is a lack of efficient implementations of declarative solvers for the other logic programming semantics, the answer set semantics (referred to as 2-valued stable semantics by Caminada and Schulz) corresponds to ABA stable semantics under their translation. In other words, one obtains the stable assumption sets of a given ABA framework by representing the framework in the specified way and solving the answer sets of this representation with an ASP solver. Thus restricting to stable semantics of ABA, the encoding presented by Caminada and Schulz (2017) can be compared to the one introduced in this work using an ASP solver for both. Their encoding has not been implemented before to the best of our knowledge; we provide in the following an empirical comparison showing that the two encodings result in similar runtime performance.

Table 5: ABA$^+$ reasoning problem variants supported by different reasoning system implementations. ✓: the system natively supports the problem variant. **: the system supports the problem variant via σ-assumption set enumeration.

System	Task	$<$-*com*	$<$-*stb*	$<$-*prf*	$<$-*grd* under FL-property	$<$-*ideal*
ASP	credulous	-	✓	-	✓	-
	skeptical	-	✓	-	✓	-
	σ-assumption set enum.	-	✓	-	✓	-
ABAPLUS	credulous	**	**	**	** (under WCP)	**
	skeptical	**	**	**	** (under WCP)	**
	σ-assumption set enum.	✓	✓	✓	✓(under WCP)	✓

To the best of our knowledge, the only system currently supporting reasoning in ABA$^+$ is ABAPLUS (Bao et al., 2017). ABAPLUS is a translation-based system, translating from ABA$^+$ to AFs and using Aspartix ASP encodings (Egly et al., 2010) on the AFs. The system supports σ-assumption set enumeration under $<$-stable, $<$-grounded, $<$-complete, $<$-preferred and $<$-ideal semantics; credulous and skeptical queries can be answered by enumerating all σ-assumption sets. Given an empty preference relation, ABAPLUS can also be applied on ABA. However, ABAPLUS insists that the input framework satisfies the Axiom of Weak Contraposition (WCP). We give the formal definition of WCP in the Appendix A in Definition 9. If an input framework does not satisfy WCP, ABAPLUS will modify the framework to one that satisfies WCP and reasons over the modified framework instead of the original input instance. Furthermore, the system places additional restrictions on the input framework: assumptions cannot be contraries and the same sentence cannot be the contrary of more than one assumption.

As representatives of state of the art, in the following we will focus on comparing the efficiency of our ASP-based approach to those of ABA2AF, ABAGRAPH and ABAPLUS.

5.2 Empirical Setup

Before presenting results from the evaluation, we describe the empirical setup.

5.2.1 BENCHMARKS

For comparing the direct ASP-based approach to ABAGRAPH and ABA2AF, we employed the 680 ABA frameworks, containing up to 90 sentences, and the associated queries used earlier by Craven and Toni (2016) and Lehtonen et al. (2017) in experiments on ABAGRAPH and ABA2AF (`http://robertcraven.org/proarg/experiments.html`). For acceptance problems, ten sentences from each framework were chosen as queries, following Craven and Toni (2016) and Lehtonen et al. (2017). Lehtonen et al. (2017) pointed out that many of the instances in the benchmark set are trivial.[12] We filtered the trivial instances out for the acceptance problems, leaving 1728 instances for credulous reasoning under admissible and grounded and 4613 for skeptical reasoning under

12. Instances in which the queried sentence is derivable from the empty set or not derivable at all in the given framework are trivial for some problems. In particular, sentences derivable from the empty set are always credulously accepted under admissible and grounded and skeptically accepted under stable semantics. Moreover, sentences which are not derivable at all in the framework are not credulously accepted under admissible nor grounded semantics. Since both of these cases can be detected by a polynomial time scan of the rules, these instances are not interesting in terms of computational performance.

stable semantics. For enumeration under preferred semantics, all of the 680 base frameworks were used.

The 680 ABA frameworks do not generally conform to the additional restrictions ABAPLUS places on frameworks (e.g., an assumption can not be a contrary of another assumption). To enable a comparison between the ASP-based approach and ABAPLUS on problems variants not supported by ABAGRAPH and ABA2AF, we generated further frameworks using a slightly modified version of the parameter family 4 introduced by Craven and Toni (2016), where each parameter is dependent on the number of sentences.[13] In each of the frameworks, 37% of the sentences are assumptions, again following Craven and Toni (2016). The number of rules deriving each sentence and the size of the rule bodies, respectively, are randomly chosen from the intervals $[1, \min(n_s/7, 20)]$ and $[1, \min(n_s/8, 20)]$, where n_s is the number of sentences. To avoid trivial instances, we let the size of the bodies start from one, so that no sentences are trivially derivable, and let all sentences be derivable by at least some rule. We further bound the number of rules deriving each sentence and the size of the rule bodies by 20 to avoid unnaturally complicated rules w.r.t. the number of sentences. For ABA complete and ideal semantics, we generated 20 frameworks for each number of sentences 10, 14, 18, 22, 26, 30, for a total of 120 frameworks. Furthermore, we extended these frameworks to ABA$^+$ by generating for each framework a preference relation. In particular, the preference relations were generated by choosing a random permutation $(a_i)_{0<i\leq n}$ of the assumptions, and for each $j < i$, set a_i to be preferred to a_j with a probability of 15% or 40% (half of the instances having each density). Finally, recall that ABAPLUS enforces WCP (cf. Def. 9) if it is not satisfied by changing the input framework. Hence for a fair performance comparison between the ASP-based approaches and ABAPLUS, we used ABAPLUS to modify the ABA$^+$ frameworks so that each of them satisfies WCP, and used the resulting frameworks as benchmarks for this comparison.

5.2.2 SYSTEMS AND COMPUTING SETUP

We compare our ASP-based approach to ABA2AF, ABAGRAPH and ABAPLUS to the extent the systems support the difference problem variants under all the problems variants covered by our ASP-based approach: ABA admissible, stable, grounded, preferred, complete, ideal, and ABA$^+$ $<$-grounded and $<$-stable. The tasks used for each of these semantics is restricted by the tasks supported by the systems.

For our ASP-based approach as well as ABA2AF, we used CLINGO version 5.2.2 (Gebser et al., 2016) as the ASP solver. We used version 3 of ASPRIN for the ASP approach for preferred semantics. For ABAGRAPH, we used SICStus Prolog version 4.5. All experiments were run on 2.83-GHz Intel Xeon E5440 quad-core machines with 32-GB RAM using a 600-second time limit per instance.

The implementation of our ASP-based approach is available at `https://bitbucket.org/coreo-group/aspforaba`.

5.3 Comparison against State of the Art

An overview of the results, comparing the ASP-based approach to the state-of-the-art systems as applicable, is shown in Table 6. In terms of the problem variants considered, we replicated the comparison between ABA2AF and ABAGRAPH on the enumeration of all solutions with respect to a

13. Unlike for the other parameter families considered by Craven and Toni (2016), the dependence between parameters and the number of sentences allows the frameworks to scale reasonably instead of e.g. the absolute number of assumptions being fixed while the number of sentences is increased.

Table 6: Runtime comparison. Mean (**mean**), median (**med.**) and cumulative running times (**cum.**) over solved instances, **#timeouts** is the number of timeouts. Number of instances: 1728 (*adm*, *grd*), 4613 (*stb*), 680 (*prf*), 120 (*com*, *ideal*, and ABA$^+$ <-*stb* and <-*grd*).

Problem	**Approach**	**#timeouts**	**Running times** (s)		
			mean	**med.**	**cum.**
ABA *adm*,	ASP	**0**	**0.030**	**0.012**	**53**
enum. w/query	ABAGRAPH	401	15.170	1.064	20131
	ABA2AF	393	18.650	0.588	24897
ABA *adm*,	ASP	**0**	**0.018**	**0.012**	**31**
cred. accep.	ABAGRAPH	200	8.464	1.056	12932
	ABA2AF	364	13.990	0.572	19078
ABA *stb*,	ASP	**0**	**0.008**	**0.004**	**38**
skept. accep.	ABA2AF	648	10.942	1.040	43386
ABA *grd*	ASP	**0**	**0.127**	**0.056**	**220**
accep.	ABAGRAPH	210	9.979	0.984	15148
ABA *prf*	ASP	**0**	**0.333**	**0.328**	**226**
enum. wo/query	ABA2AF	255	6.082	0.464	2585
ABA *com*	ASP	**0**	**0.005**	**0.004**	**1**
enum wo/query	ABAPLUS	9	15.287	0.268	1697
ABA *ideal*	ASP	**0**	**0.025**	**0.024**	**3**
	ABAPLUS	18	22.490	0.322	2293
ABA$^+$ <-*stb*	ASP	**0**	**0.018**	**0.008**	**2**
enum. wo/query	ABAPLUS	9	15.583	0.268	1729
ABA$^+$ <-*grd*	ASP	**0**	**0.380**	**0.168**	**46**
	ABAPLUS	9	15.611	0.268	1732

query sentence for ABA under admissible semantics from Lehtonen et al. (2017), and additionally considered skeptical reasoning under stable semantics, acceptance under grounded semantics, and enumeration of all σ-assumption sets (without a query) under preferred, complete and ideal semantics.[14] The choice of these tasks is largely forced by restrictions of the systems (recall Section 5.1): for grounded, ABAGRAPH only supports acceptance, and for ideal and complete, ABAPLUS only supports σ-assumption set enumeration without a query. For ABA$^+$, we compare the direct ASP-based approach to ABAPLUS under $<$-grounded and $<$-stable semantics.

The ASP-based approach, without any timeouts and very small cumulative running times, clearly outperforms the other systems on each problem, including both the ABA and ABA$^+$ problems, all of the available semantics and the different reasoning tasks. In contrast, the other systems exhibit high numbers of timeouts and large running times overall. The smallest average performance gap between ASP and one of the other tested systems is in ABA preferred, where the median runtime of ASP is $\approx 3/4$ of that of ABA2AF on instances that both systems could solve. Even for this problem variant, the cumulative runtime of the ASP-based approach is less than 1/10 of that of ABA2AF. In addition, ABA2AF timed out on more than a third of the instances, while the ASP-based approach could solve all.

Based on the number of timeouts, ABAPLUS seems to do reasonably well in the comparisons under complete and ideal as well as $<$-stable and $<$-grounded semantics. However, one should keep in mind that there are less of these instances (120 vs at least 680) and the instances are smaller (up to 30 sentences in a framework vs up to 90 sentences) than for the other tasks. The mean and cumulative runtimes show that ABAPLUS has problems solving these smaller instances as well, in contrast to the ASP approach.

5.4 Scalability of the ASP Approach

Due to the very good empirical performance of the ASP-based approach, we studied the scalability of the approach further with larger benchmarks. For this, we generated larger frameworks similarly as described in Section 5.2.1. In particular, we generated 10 frameworks for each number of sentences in $\{50, 250, 500, 1000, 1500, 2000, 2500, 3000, 3500, 4000\}$. We tested credulous acceptance of 10 arbitrary query sentences per ABA framework under admissible, complete and stable semantics, giving a total of 100 instances per number of sentences. For stable semantics, we additionally compared the encoding introduced in this work, denoted by *stb* in the table, and the one introduced by Caminada and Schulz (2017) (recall Section 5.1), denoted by *stb*-alt. As the ideal assumption set is unique, we did not use queries in the experiments for ideal semantics. For preferred we tested extension enumeration. For ABA$^+$, we generated six frameworks per each number of sentences, three with a preference ordering of density 15% and three of density 40%, for a total of 60 ABA$^+$ frameworks per number of sentences.[15] We then credulously queried ten arbitrary sentences per framework.

The results are shown in Table 7 and Figure 3. Depending on the semantics and reasoning task, the ASP approach can routinely solve instances with up to 3000 sentences for ABA and up to 1000 for ABA$^+$. This shows that the ASP approach scales significantly better than the previous state-of-the-art systems, as the benchmarks used in the comparisons earlier have only up to 90 sentences per

14. As there is always a unique grounded assumption set, credulous and skeptical acceptance coincide for grounded semantics. Enumerating ideal assumption sets amounts to finding the unique ideal assumption set.

15. These frameworks do not necessarily satisfy WCP.

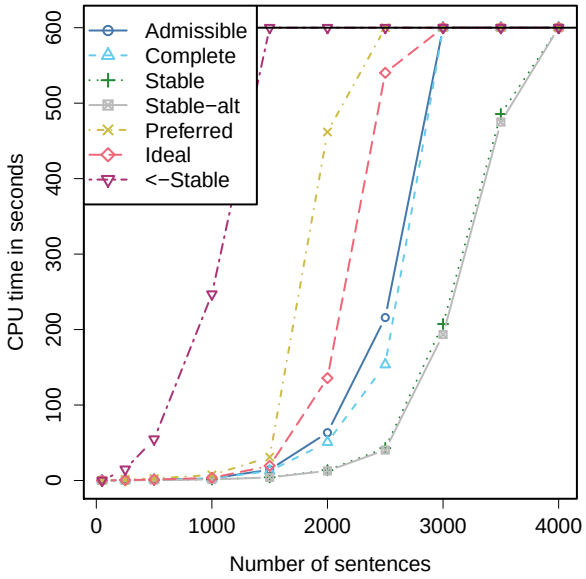


Figure 3: Median running times per number of sentences in the scalability experiments for each semantics. Timeouts are included as a running time of 600 seconds.

Table 7: Scalability of ASP on larger frameworks. Number of instances per $|\mathscr{L}|$: 100 (*adm*, *com*, *stb*), 10 (*prf*, *ideal*), 60 (*<-stb*).

	#timeouts (mean running time over solved instances (s))													
$\|\mathscr{L}\|$	**ABA** *adm*		**ABA** *com*		**ABA** *stb*		**ABA** *stb*-alt		**ABA** *prf*		**ABA** *ideal*		**ABA**$^+$ *<-stb*	
50	0	(0.01)	0	(0.01)	0	(0.01)	0	(0.01)	0	(0.3)	0	(0.03)	0	(0.1)
250	0	(0.4)	0	(0.4)	0	(0.3)	0	(0.3)	0	(1.3)	0	(0.5)	0	(12.6)
500	0	(0.8)	0	(0.9)	0	(0.6)	0	(0.6)	0	(2.8)	0	(1.2)	0	(53.1)
1000	0	(2.9)	0	(2.9)	0	(1.4)	0	(1.4)	0	(8.7)	0	(4.1)	0	(241.0)
1500	0	(13.8)	0	(12.3)	0	(4.3)	0	(4.1)	0	(32.4)	0	(18.2)	60	(0.0)
2000	0	(99.2)	0	(75.3)	0	(19.1)	0	(18.3)	5	(144.5)	0	(155.8)	60	(0.0)
2500	22	(126.1)	10	(201.1)	0	(74.2)	0	(72.7)	7	(268.8)	4	(292.5)	60	(0.0)
3000	70	(173.6)	58	(222.9)	18	(173.6)	18	(163.8)	10	(0.0)	9	(463.6)	60	(0.0)
3500	85	(211.3)	79	(253.6)	48	(231.8)	47	(228.1)	10	(0.0)	10	(0.0)	60	(0.0)
4000	89	(108.1)	87	(135.2)	81	(158.1)	81	(154.4)	10	(0.0)	10	(0.0)	60	(0.0)

framework. To further validate the performance gap between the ASP approach and the previous state-of-the-art, we ran ABAGRAPH under admissible semantics for these larger instances. It could only solve the 50-sentence instances and none of the larger ones within the time limit. We also observe that the two alternative encodings of stable semantics in ABA have very similar performance, with the average running times being within ten seconds of each others for each number of sentences. Interestingly, for ABA$^+$ <-stable, the density of the preference relation had little effect on the performance: both densities (15% and 40%) had the same behaviour in terms of timeouts, and their mean running time was within five seconds of each other's for each number of sentences.

6. Related Work

We give an overview on earlier work related to different aspects of this article.

6.1 Assumption-based Argumentation and other Structured Formalisms

Besnard et al. (2014) give an overview for structured argumentation formalisms. ABA, originally introduced by Bondarenko et al. (1997), has been extensively studied; see e.g. the survey articles by Dung et al. (2009), Čyras et al. (2018), Toni (2014). The complexity of reasoning in various forms of ABA has been studied by Dimopoulos et al. (2002), Dunne (2009), Karamlou et al. (2019), Čyras et al. (2021) and summarized by Dvořák and Dunne (2018). Results of Corollary 5 and Corollary 6 were independently shown by Čyras et al. (2021). Various application scenarios of ABA have been identified, including medical decision making (Craven et al., 2012; Čyras et al., 2020), decision making in a multi-agent context (Fan et al., 2014), and modelling game theoretical problems (Fan & Toni, 2016).

Beyond various forms of ABA, prominent structured formalisms include ASPIC$^+$ (Modgil & Prakken, 2013, 2018; Prakken, 2010), DeLP (García & Simari, 2004, 2014, 2018) and deductive argumentation (Besnard & Hunter, 2008). ASPIC$^+$ is a general formalism that supports both strict and defeasible rules and premises, and three kind of attacks (undercut, rebut and undermine), whereas attacks in ABA are in the form of undermining. In fact, ASPIC$^+$ captures the commonly-studied ABA fragment (without preferences) (Prakken, 2010; Modgil & Prakken, 2018) considered in this article. Indeed, extending the answer set programming approach to ABA developed in this article to different variants of ASPIC$^+$ is a promising avenue for further work; in fact, after the writing of this article, first steps to this direction have already been made (Lehtonen et al., 2020).

6.2 Integration of Preferences into Structured Formalisms

Beirlaen et al. (2018) give a recent survey of formal approaches to notions of strength of arguments, including approaches to preferences in structured argumentation formalisms. Structured argumentation formalisms generally incorporate preferences over some defeasible elements, but the way preferences are used varies between formalisms; in many cases, preferences can be handled in multiple ways within single formalisms as well. We will discuss here the differences and applicability of argumentation formalisms incorporating preferences.

Extending ABA with preferences, ABA$^+$ was introduced by Čyras and Toni (2016a, 2016c), Čyras (2017). The complexity of reasoning in ABA$^+$ was not investigated prior to our work. An application of ABA$^+$ in medical decision-making was proposed by Čyras et al. (2020), where a variant of ABA$^+$ was used to aid in selecting the proper clinical guideline recommendations to follow in a particular patient case. Preferences were used to factor the wishes of patients into the treatment plan: if a patient wishes, for example, to avoid intense exercise, a treatment involving it is avoided if possible.

ABA$^+$ has been shown to satisfy several rationality postulates for rule-based argumentation formalisms (Čyras, 2017; Čyras & Toni, 2016a). In addition, further desirable properties hold in a subset of ABA$^+$ frameworks, namely ones that satisfy the FL-property (see Property 1). This guarantees further properties, such as the existence of complete assumption sets and the existence of a unique grounded assumption set. The complexity results and ASP encoding for ABA$^+$ grounded semantics we introduce apply to frameworks having the FL-property. In particular, frameworks

that satisfy the Axiom of Weak Contraposition (WCP) also have the FL-property. In ASPIC$^+$, an analogous property to WCP also guarantees the satisfaction of the fundamental lemma (Modgil & Prakken, 2013). Our other results are independent of the FL-property, including, e.g., our approach to stable semantics in ABA$^+$.

In many argumentation formalisms, including ABA$^+$, preferences are used to modify the given framework (usually through the attack relation) so that it may be the case that in a framework with preferences, different sets of arguments are extensions than in the same framework without preferences (Čyras & Toni, 2016b; Modgil & Prakken, 2013; Besnard & Hunter, 2014; García & Simari, 2004). A contrasting approach is to use preferences to select among the extensions that the given framework has, which is a common approach in non-monotonic reasoning formalisms (Delgrande et al., 2004); in the realm of argumentation this approach was proposed in (Amgoud & Vesic, 2011). The intuition for modifying the framework instead of simply selecting among extensions is that preferences should play a role in information exchange in a dialectical setting instead of merely acting as constraints (Čyras, 2017). One approach to extending abstract argumentation with preferences, Preference-based Argumentation Frameworks (PAFs) (Amgoud & Vesic, 2014), incorporates attack reversals. There is a simple translation from PAFs to ABA$^+$ frameworks, while the reverse is not necessarily true (Čyras, 2017).

Care must be taken when applying argumentation formalisms to real world applications, including the interpretation of preferences. Wakaki (2017a) gives a simple example in legal reasoning when carelessly applying ABA$^+$ leads to counterintuitive results. Specifically, in the example an accused being guilty is set to be preferred to them being innocent, and then the reasoning outcome of ABA$^+$ violates the legal principle of "innocent until proven guilty". This illustrates how ABA$^+$ allows the preferences to result in new extensions compared to the framework without preferences instead of merely selecting among extensions. Different formalisms and different ways of preference-handling might be better suited to different applications.

Heyninck (2019) explores theoretical properties of two versions of ABA with preferences: one similar to ABA$^+$ and one without reverse attacks. Another approach to extending ABA with explicit preferences, p_ABA, uses preferences in a different way to ABA$^+$. In p_ABA, preferences are used to select the most preferred extensions (assumption sets) without modifying the attack relation beforehand. Wakaki (2017a) notes that p_ABA gives the intuitive result in the beforementioned legal reasoning example. Moreover, Wakaki (2017b) lists possible use cases for p_ABA via simple examples, including reasoning in extended decision frameworks and epistemic frameworks, as well as practical reasoning with competing goals. It is also possible to encode preferences implicitly using the existing tools in a formalism. For ABA, different approaches include those of Toni (2008), Fan and Toni (2014), Thang and Luong (2013).

Due to differences in how preferences are handled in ABA$^+$ and ASPIC$^+$ (notably unsuccessful attacks may be discarded in ASPIC$^+$, depending on the semantics, but reversed in ABA$^+$), a straightforward translation from ABA$^+$ to ASPIC$^+$ does not seem to be available (Čyras, 2017). ASPIC$^+$ allows for two distinct approaches to preferences: preferences over arguments and preferences over premises and defeasible rules. Furthermore, there is a variety of ways to lift the preferences to the argument level.

6.3 Algorithmic Approaches to Reasoning in Abstract and Structured Formalisms

Various algorithms for reasoning in different argumentation formalisms exist (Cerutti et al., 2018). Compared to the number of algorithms and system implementations recently developed for reasoning in abstract argumentation frameworks (Charwat et al., 2015), motivated also by the ICCMA competition series (Thimm & Villata, 2017; Gaggl et al., 2020), fewer system implementations are available for reasoning in structured argumentation formalisms. The algorithmic approaches implemented by different systems can be divided into specialized algorithmic approaches and reduction-based approaches which are based on declarative encodings of the reasoning problems at hand. The computational approach developed in this article is inspired by the success of ASP in argumentation reasoning, especially AF reasoning (Toni & Sergot, 2011; Egly & Woltran, 2006; Egly et al., 2008; Nieves et al., 2008; Wakaki & Nitta, 2008; Egly et al., 2010; Gaggl et al., 2015; Niskanen et al., 2019). Compared to ASP encodings of AF reasoning tasks, however, the more complex nature of ABA makes the declarative modelling more challenging compared to AFs, which is reflected in the more intricate encodings we present in this work.

The ABA systems ABAGRAPH (Craven & Toni, 2016), GRAPHARG (Craven et al., 2013), PROXDD (Toni, 2013), and CASAPI (Gaertner & Toni, 2007a) implement the specialized, so-called dispute derivation approach to reasoning in ABA. On the other hand, ABA2AF (Lehtonen et al., 2017) first translates ABA frameworks to AFs and uses ASP encodings on the AF-level. Notably, all of these systems are for the LP fragment of ABA also considered in this article. Bao et al. (2017) proposed a reasoning system for ABA$^+$, called ABAPLUS, which can enumerate extensions under multiple semantics by translating the ABA$^+$ framework to an AF and using an AF reasoner. ABAPLUS also assumes that the underlying ABA framework belongs to the LP fragment of ABA. A notable weakness of translating ABA to AFs is that the translation process can form an intrinsic bottleneck in terms of empirical performance (Lehtonen et al., 2017). In contrast, The ASP-based approach developed in this article directly encodes ABA reasoning tasks in ASP, thus circumventing issues with ABA-to-AF translations.

Karamlou et al. (2019) proposed labeling-based algorithms and an implementation for the bipolar fragment of ABA. While operating on a different fragment of ABA than considered in this work, an interesting avenue for future work would be to compare the approach of Karamlou et al. (2019) with our approach, in particular since the corresponding fragments exhibit the same computational complexity.

7. Conclusions

We provided algorithmic and complexity-theoretic contributions in the context of reasoning in assumption-based argumentation. From the algorithmic perspective, we proposed a new approach to reasoning in assumption-based argumentation with and without preferences via non-trivial ASP encodings of ABA reasoning tasks under several central argumentation semantics. The ASP-based approach allows for covering various semantics and reasoning modes in a relatively uniform manner. In practice, the approach significantly improves on the empirical performance of the current state-of-the-art approaches to ABA reasoning. This motivates further extensions of the approach to obtain algorithms for further reasoning problems in the context of ABA, in particular ones that are hard for the second-level of the polynomial hierarchy. Beyond the current article, it would also be interesting to investigate the potential of alternative encodings based on e.g. the labelling-based view (Sakama & Rienstra, 2017; Schulz & Toni, 2017). In addition to the various declarative al-

gorithms, we provided complexity results for central ABA$^+$ reasoning problems (credulous and skeptical reasoning) under several semantics, towards bridging the gap between the current knowledge on complexity of reasoning in ABA and ABA$^+$. Notably the computational complexity of credulous and skeptical acceptance is the same in ABA and ABA$^+$ under stable semantics. In contrast, credulous acceptance under admissible semantics in ABA$^+$ is on a higher complexity level compared to the corresponding problem in ABA. Our results on the complexity of the verification task for ABA$^+$ strongly suggest that the integration of preferential information into ABA via reverse attacks may increase the computational complexity of acceptance problems for other semantics as well. Beyond the LP fragments of ABA($^+$) covered in this article, it would be interesting to extend the ASP-based approach developed in this work to, e.g., general (non-flat) frameworks, which in general have higher computational complexity.

Acknowledgements

This work has been financially supported in part by Academy of Finland (grants 276412, 312662, 322869), University of Helsinki Doctoral Programme in Computer Science DoCS, and by the Austrian Science Fund (FWF): P30168-N31 and I2854.

Appendix A. Proofs

The following lemma is applied in the proof of Proposition 11 (Item 2), as well as in the ASP encoding for capturing $<$-grounded semantics (Listing 6).

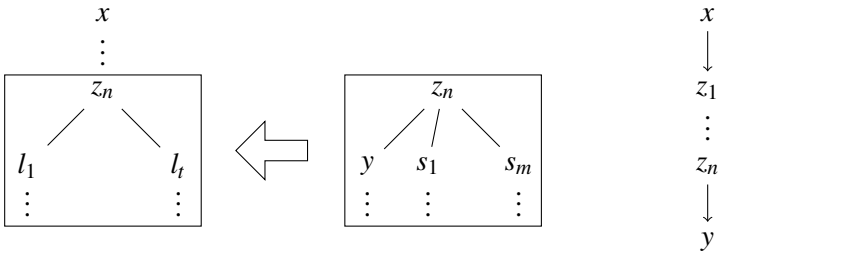
Lemma 20. *Let $F = (\mathscr{L}, \mathscr{R}, \mathscr{A}, ^-, \leq)$ be an ABA$^+$ framework, $x \in \mathscr{L}$, and A, A' be two sets of assumptions such that $A' \subseteq A$ and $A \vdash_{\mathscr{R}} x$. Further, let $G = (V, E)$ be a directed graph with*

- $V = Th_{\mathscr{R}}(A)$, *and*
- $E = \{(a,b) \mid \exists r \in \mathscr{R} : body(r) \subseteq Th_{\mathscr{R}}(A), a \in head(r), b \in body(r)\}$.

It holds that $X \models_R x$ with $X \subseteq A$ and $X \cap A' \neq \emptyset$ iff there is a directed path from x to a node in A' in G.

Proof. Assume that $X \models_R x$ with $X \cap A' \neq \emptyset$ for some $X \subseteq A$. Consider a derivation tree T witnessing $X \models_R x$. By presumption, at least one element $a' \in A'$ is present in the tree. This means that there is a path from x to a' in the tree. For each edge $e = (x, y)$ of this path, it holds that there is a rule $r \in R \subseteq \mathscr{R}$ such that $x \in head(r)$ and $y \in body(r)$. This implies that there is a directed path from x to a' in G (the same as in T).

For the other direction, assume that there is a directed path from x to a node in A' in G. We use an auxiliary definition of replacing subtrees in a rooted derivation tree T. Let T' be a rooted derivation tree with root s. We denote by $T[s/T']$ the uniform replacement of each node s and subtrees in T (i.e., this node labeled s and each edge and node connected from this node labeled s to a non-root leaf) by tree T'. In case of subtrees with a root s that also contain in the subtree another node labeled s, define the replacement on maximal subtrees that contain s as a root (we note that for our purposes the exact handling of subtrees containing several nodes labeled s is not crucial; replacing such subtrees in an arbitrary sequence, or replacing only one such subtree, results in the same overall result). Note that $T[s/T']$ is a derivation tree with the same root as T, and that the modified tree contains a node labeled s iff T contains a node labeled s.



Tree T deriving x via z_n Tree T' deriving z_n via y Part of graph G

Figure 4: Subtree replacement $T[z_n/T']$ for proof of Lemma 20

We show by induction on the length of a shortest directed path in G from x to elements in $s \in Th_{\mathscr{R}}(A)$ that there is a derivation tree deriving x using s. Let $N_i = \{y \in Th_{\mathscr{R}}(A) \mid$ there is a shortest path from x to y of length i in $G\}$.

Property P to show: For each $y \in N_i$ it holds that there is a derivation tree T of F with root x, all its leaves labeled by a subset of A or $\top$, and a node labeled by y. (It is not required that this node is a leaf.)

Induction base: Let $N_0 = \{x\}$. By presumption, $x \in Th_{\mathscr{R}}(A)$ and $A \vdash_{\mathscr{R}} x$. This implies that there is a $B \subseteq A$ such that $B \models_R x$. Thus, there is a derivation tree witnessing this that satisfies the condition of property P.

Induction step: Assume that the induction hypothesis holds, i.e., that P holds for N_i. We show that P holds for N_{i+1}. Let $y \in N_{i+1} \setminus N_i$. There is a shortest path from x to y: $p = (x, z_1, \ldots, z_n, y)$ of length $i+1$. Since $z_n \in N_i$ (there is a shortest path from x to z_n of length exactly i), it holds that there is a derivation tree T with root x, all its leaves labeled by some assumptions in A and z_n is a label of a node in the tree. By presumption, $A \vdash_{\mathscr{R}} y$. Thus, there is a $B \subseteq A$ with $B \models_R y$, and a corresponding witness derivation tree H'. It holds that all leaves of H' are in A (actually in B). Construct derivation tree T', as follows. Let the root of T' be z_n. Choose one rule $r \in \mathscr{R}$ such that $z_n \in head(r)$, $y \in body(r)$ and $body(r) \subseteq Th_{\mathscr{R}}(A)$. Such a rule exists because there is an edge from z_n to y by construction of G. Let $\{s_1, \ldots, s_m\}$ be the sentences in the body of r except for y. Construct derivation trees $H_1, \ldots, H_m$ with leaves in A (which must exist, since each is forward-derivable from A). Finish construction of T' by adding all nodes and edges from H' and $H_1, \ldots, H_m$ (viewing each node and edge as a fresh node and edge), and add edges from z_n to all roots of H' and $H_1, \ldots, H_m$. The resulting graph T' is a derivation tree with root z_n, and all leaves from assumptions in A. Replace each subtree with root z_n in T with T', i.e., consider $T'' = T[z_n/T']$. It holds that T'' has root x (the root of T is not changed, even if $z_n = x$), all leaves of T'' are in A (leaves might have changed, but both T and T' have all their leaves in A), and T'' has a node labeled y (since T' has a node labeled y). See Figure 4 for an illustration.

The previous proof by induction shows that if y is reachable from x in G, then there is a derivation tree with root x, leaves (assumptions) in A, and a node labeled by y. The claim of the lemma follows. □

Definition 9. *An ABA$^+$ framework $(\mathscr{L}, \mathscr{R}, \mathscr{A}, {}^-, \leq)$ satisfies the Axiom of Weak Contraposition (WCP) iff for each $A \subseteq \mathscr{A}$, $R \subseteq \mathscr{R}$, and $b \in \mathscr{A}$, if $A \models_R \overline{b}$ and $\exists a' \in A$ such that $a' < b$, then there is $a \leq$-minimal $a \in A$ such that $a < b$ and $A' \models_{R'} \overline{a}$ for some $A' \subseteq (A \setminus \{a\}) \cup \{b\}$ and $R' \subseteq \mathscr{R}$.*

Proposition 21. *Deciding whether a given set of assumptions A in an ABA$^+$ framework satisfying WCP reversely counterattacks all assumption sets that $<$-attack A is coNP-hard.*

Proof. We establish that coNP-hardness holds assuming WCP. In particular, adding the following rules to $\texttt{red}(\phi) = (\mathscr{L}, \mathscr{R}, \mathscr{A}, ^-, \leq)$ (recall Reduction 1) results in an ABA$^+$ framework that has WCP.

- $\overline{x_i} \leftarrow b, \neg x_i, \overline{a}$ and $\overline{\neg x_i} \leftarrow b, x_i, \overline{a}$ for each $1 \leq i \leq n$,
- $\overline{\neg l_1} \leftarrow b, \neg l_2, \neg l_3, \overline{a}$ and $\overline{\neg l_2} \leftarrow b, \neg l_1, \neg l_3, \overline{a}$, and $\overline{\neg l_3} \leftarrow b, \neg l_1, \neg l_2, \overline{a}$ for each clause $l_1 \vee l_2 \vee l_3$.

Denote the resulting ABA$^+$ framework by $F' = (\mathscr{L}, \mathscr{R}', \mathscr{A}, ^-, \leq)$. To see that F' has WCP, consider any $A \subseteq \mathscr{A}$ such that $A \models_R \overline{b}$ where $R \subseteq \mathscr{R}$ and $\exists a' \in A$ with $a' < b$. By construction, $a' \in X \cup \neg X$ and a' is minimal w.r.t. $<$. Due to the rules in $\mathscr{R}' \setminus \mathscr{R}$, it holds that $A' \models_{R'} \overline{a'}$, where $A' = (A \setminus \{a'\}) \cup \{b\}$ and $R' \subseteq \mathscr{R}'$. We show that a conflict-free set A defends another conflict-free set B in $\texttt{red}(\phi)$ iff A defends B in F', which implies that conflict-free sets and $<$-admissible sets coincide for $\texttt{red}(\phi)$ and F'. Then, in particular, all attacks to and from $\{a,b\}$ coincide for $\texttt{red}(\phi)$ and F', as $\{a,b\}$ is conflict-free. It follows that the reasoning in the proof of Lemma 13 can be applied for F'. To see this, consider two assumption sets A and B. Since the $<$-relation is the same in $\texttt{red}(\phi)$ and F' and the new rules of $\mathscr{R}'$ do not introduce new ways to derive $\overline{b}$ (which is the only assumption that is more preferred than any other assumption), A reversely $<$-attacks B in $\texttt{red}(\phi)$ iff it reversely $<$-attacks B in F'. Moreover, if A normally $<$-attacks B in $\texttt{red}(\phi)$, then A normally $<$-attacks B in F', since $Th_{\mathscr{R}}(X) \subseteq Th_{\mathscr{R}'}(X)$ for $X \subseteq \mathscr{A}$ and the $<$-relation is the same in $\texttt{red}(\phi)$ and F'. Assume A normally $<$-attacks B in F' but not in $\texttt{red}(\phi)$. This implies that there is an $A' \subseteq A$ such that $A' \models_R \overline{z}$ with $z \in X \cup \neg X$, by construction, and that $z \in B$. This implies that both A and B are conflicting assumption sets in both $\texttt{red}(\phi)$ and F', because any $z \in X \cup \neg X$ derives $\overline{z}$. Thus if A and B (or either of them) are conflict-free, then the $<$-attacks between A and B are the same in $\texttt{red}(\phi)$ and F'. □

Theorem 22. *Verifying that a set of assumptions is $<$-admissible in ABA$^+$ framework satisfying WCP is coNP-complete.*

Proof. This statement can be shown with an analogous reasoning as in the proof of Proposition 21. □

Proof of Theorem 19. For membership, non-deterministically construct an assumption set A for a given ABA$^+$ instance and query sentence q. Check whether $A \vdash_{\mathscr{R}} q$ (in polynomial time), and whether A is $<$-admissible (in coNP, due to Proposition 15).

For establishing hardness, let $\psi = \exists X \forall Y \phi$ be a closed quantified Boolean formula in prenex normal form where $\phi = c_1 \vee \cdots \vee c_s$ is an unquantified Boolean formula in disjunctive normal form (DNF) with conjunctions c_j. Formula ϕ is over variables $X = \{x_1, \ldots, x_n\}$ and $Y = \{y_1, \ldots, y_m\}$. Deciding whether ψ is satisfiable (true) is a Σ_2^{P}-complete problem. Construct (in polynomial time) the following ABA$^+$ instance. Similarly as in preceding proofs, if Z is a set, then $\neg Z = \{\neg z \mid z \in Z\}$ and $Z' = \{z' \mid z \in Z\}$. Let $\mathscr{A} = X \cup \neg X \cup X' \cup \neg X' \cup Y \cup Y' \cup \{p, r\}$ be the set of assumptions. Construct the following rules:

- $q \leftarrow d_{x_1}, \ldots, d_{x_n}, p, r$,
- for each $1 \leq i \leq n$:

  - $d_{x_i} \leftarrow x_i$ and $d_{x_i} \leftarrow \neg x_i$,
  - $\overline{\neg x_i} \leftarrow x_i$ and $\overline{x_i} \leftarrow \neg x_i$,
  - $\overline{\neg x'_i} \leftarrow x_i$ and $\overline{x'_i} \leftarrow \neg x_i$,
  - $d_{x'_i} \leftarrow x'_i$ and $d_{x'_i} \leftarrow \neg x'_i$, and
  - $\overline{r} \leftarrow x'_i, \neg x'_i$,

- for each $1 \leq i \leq m$:
  - $d_{y_i} \leftarrow y_i$ and $d_{y_i} \leftarrow \neg y_i$, and
  - $\overline{r} \leftarrow y_i, \neg y_i$,

- $\overline{r} \leftarrow asBody(c_j), \overline{p}$ for each $1 \leq j \leq s$, and

- $\overline{p} \leftarrow d_{x'_1}, \ldots, d_{x'_n}, d_{y_1}, \ldots, d_{y_m}$.

Here the function $asBody(c_j)$ is defined as the list of body elements $l_1, \ldots, l_t$ from the literals of the conjunction $c_j = l_1 \wedge \cdots \wedge l_t$. Complete the instance with $r > x'_i$, $r > \neg x'_i$, $r > y_i$, and $r > \neg y_i$ as the preference relation.

We claim that ψ is true (satisfiable) iff q is credulously accepted under $<$-admissible semantics in the ABA$^+$ framework. Assume that q is credulously accepted under $<$-admissible semantics. That is, there is an $<$-admissible assumption set A with $A \vdash_{\mathscr{R}} q$. By construction, A contains at least one of $\{x_i, \neg x_i\}$ for each $1 \leq i \leq n$ (necessary to derive sentences d_{x_i} that are required to derive q). By presumption, A is $<$-admissible and conflict-free. If there is an i such that $\{x_i, \neg x_i\} \subseteq A$ then A is not conflict-free: $A \vdash_{\mathscr{R}} \overline{x_i}$ and $A \vdash_{\mathscr{R}} \overline{\neg x_i}$. Therefore A contains for each i exactly one of x_i or $\neg x_i$. Further, $\{p, r\} \subseteq A$ by construction. Let τ be a partial truth assignment defined on X such that $\tau(x_i) = 1$ if $x_i \in A$ and $\tau(x_i) = 0$ if $\neg x_i \in A$. By the observations above, τ assigns a truth value to each variable in X. We claim that $\phi[\tau]$, the formula obtained by replacing each occurrence of a variable x_i in X by $\top$ if $\tau(x_i) = 1$ and $\bot$ if $\tau(x_i) = 0$, is tautological. This implies that ψ is satisfiable (since ϕ is satisfied no matter how the assignment on X given by τ is extended to the variables in Y). Suppose the contrary, i.e., that there is an extension of τ to τ' on all variables in ϕ (i.e. on variables $X \cup Y$) such that $\tau' \not\models \phi$. Let $B = \{x'_i \mid \tau'(x_i) = 1\} \cup \{\neg x'_i \mid \tau'(x_i) = 0\} \cup \{y_i \mid \tau'(y_i) = 1\} \cup \{\neg y_i \mid \tau'(y_i) = 0\}$. In words, B mirrors the assignment of τ' via variables x'_i, $\neg x'_i$, y_i, and $\neg y_i$. By construction, and since τ' assigns a value to all variables in $X \cup Y$ it holds that $\overline{p}$ is derivable from B (all auxiliary "decision" sentences d_z are derivable). Thus B $<$-attacks A. Since A is admissible, it follows that A $<$-attacks B. However, the contraries of y_i and $\neg y_i$ are not derivable and the contraries of x'_i and $\neg x'_i$ are derivable via x_i or $\neg x_i$, but since $x_i \in A$ iff $x'_i \in B$ and $\neg x_i \in A$ iff $\neg x'_i \in B$, it follows that from A no contrary of an assumption in B is derivable. This implies that A does not normally $<$-attack B, and must instead reversely $<$-attack B. By construction of the preference relation, this is only possible if $B \models_R \overline{r}$ (since r is the only assumption ranked lower than another one). Since B does not contain for any variable x_i or y_j both "assignments" (i.e., not both x'_i and $\neg x'_i$, and not both y_i and $\neg y_i$) it holds that, in order for B to derive $\overline{r}$, at least one c_j, for $1 \leq j \leq s$, is derivable from B. Let c_j be this sentence (conjunction). By construction of the corresponding rule that derives $\overline{r}$ it follows that $\tau' \models c_j$. Since ϕ is in DNF, $\tau' \models \phi$. This contradicts the presumption that ψ is not satisfied (true), since the same holds for any B (any extension to the variables in Y).

We now show the other direction. Assume that ψ is satisfiable (true). Then there is a partial assignment τ on variables X such that any extension τ' of τ to all variables in Y satisfies ϕ. We

claim that $A = \{x_i \mid \tau(x_i) = 1\} \cup \{\neg x_i \mid \tau(x_i) = 0\} \cup \{p, r\}$ is an $<$-admissible assumption set in the ABA$^+$ framework, which implies that q is credulously accepted under $<$-admissible semantics, since $A \vdash_{\mathscr{R}} q$. Suppose that A is not $<$-admissible. By the same line of reasoning as above, it holds that A is conflict-free. This implies that there is an assumption set B such that B $<$-attacks A, but A does not $<$-attack B. We show some properties of B. First, if A contains x_i ($\neg x_i$) then $\neg x_i \notin B$ ($x_i \notin B$), since then A would $<$-attack B. Similarly, B does not contain x'_i if $\neg x_i \in A$ and B does not contain $\neg x'_i$ if $x_i \in A$. Suppose that $B \vdash_{\mathscr{R}} \overline{r}$. Since $\overline{r}$ is only derivable if at least one of x'_i, $\neg x'_i$, y_i, or $\neg y_i$, for some i, is present in B, by construction of the preference relation it follows that A reversely $<$-attacks B if $B \vdash_{\mathscr{R}} \overline{r}$. Thus, $B \nvdash_{\mathscr{R}} \overline{r}$. It follows that $B \vdash_{\mathscr{R}} \overline{p}$, since from A one cannot derive any contrary of an assumption in B (B does not reversely $<$-attack A), and any normal $<$-attack from B onto A requires that $\overline{p}$ is derivable. These observations imply that B contains exactly one of x'_i or $\neg x'_i$ and exactly one of y_j or $\neg y_j$ for each $1 \leq i \leq n$ and $1 \leq j \leq m$ (if both would be present then $\overline{r}$ is derivable, if none are present then $\overline{p}$ is not derivable). This straightforwardly defines a truth assignment τ'' on $X \cup Y$: $\tau''(x_i) = 1$ iff $x'_i \in B$ and $\tau''(y_i) = 1$ iff $y_i \in B$. Observe that τ from above and τ'' are compatible on the X variables where τ is defined, since A does not normally $<$-attack B, and if $\tau(x_i) \neq \tau''(x_i)$, then A would normally $<$-attack B. We show that τ'' does not satisfy ϕ. Consider an arbitrary conjunction c_j in ϕ. Suppose τ'' satisfies c_j. Then, by similar reasoning as in the other direction, it holds that τ'' satisfies c_j iff B satisfies the body of a rule with $\overline{r}$ in the head (all body elements are present in B, since τ'' satisfies all literals in conjunction c_j). Thus, $B \vdash_{\mathscr{R}} \overline{r}$, which is a contradiction. Therefore, τ'' does not satisfy ϕ. This contradicts the presumption that ψ is satisfiable (true). Therefore, A is $<$-admissible and q is credulously accepted under $<$-admissible semantics. □

Appendix B. Counterexample to Grounded Encoding in Lehtonen et al. (2019)

The encoding for grounded semantics given in our preliminary work (Lehtonen et al., 2019) and replicated here in Listing 7 is erroneous. To see this, consider the ABA framework consisting of

$$\begin{aligned} \text{sentences } \mathscr{L} &= \{a, b, c, x, y, z\} \\ \text{assumptions } \mathscr{A} &= \{a, b, c\} \\ \text{contraries} \quad & \overline{b} = x,\ \overline{c} = z,\ \overline{a} = w \\ \text{rules } \mathscr{R} &= \{(x \leftarrow a), (y \leftarrow y), (y \leftarrow b), (z \leftarrow y).\} \end{aligned}$$

Listing 7 correctly assigns a to **in** as nothing attacks it. Assumption a attacks b. The only way to derive z, the contrary of c, is from b and thus c should also be added to **in**. However, Listing 7 does not achieve this. An assumption is assigned to **in** when its contrary is assigned to **out**, which happens when all rules deriving the contrary are **out**. However, the rule $(y \leftarrow y)$ is not determined to be **out** because y is not assigned to **out**. The rules of assigning sentences that are not derivable at all in the framework to **out** do not help, since y is derivable.

Appendix C. Counterexample to the Ideal Algorithm in Dunne (2009)

The algorithm for finding the ideal assumption set presented in (Dunne, 2009) is erroneous in terms of Lines 8–9. Instead of Lines 8–9 of Algorithm 1, as originally presented in (Dunne, 2009) the

Listing 7: Module π_{grd}

```
1  rule(R) ← head(R,_).
2  sentence(S) ← head(_,S).
3  sentence(S) ← contrary(_,S), not assumption(S).
4  derivable(X) ← assumption(X).
5  derivable(X) ← head(R,X), derivable_rule(R).
6  derivable_rule(R) ← head(R,_),derivable(X):body(R,X).
7  in(X) ← assumption(X), out(Y) : contrary(X,Y).
8  in(R) ← rule(R), in(X) : body(R,X).
9  in(S) ← in(R), head(R,S).
10 out(X) ← in(Y), contrary(X,Y).
11 out(R) ← out(X), body(R,X).
12 out(S) ← sentence(S), out(R) : head(R,S).
13 out(S) ← sentence(S), not derivable(S).
```

algorithm would have (apart from slight variation in notation) the lines

$$8.\ \Xi := \{a \in \mathscr{A}_{\text{out}} \mid \Gamma \nvdash_{\mathscr{R}} \overline{a}\}$$
$$9.\ \Delta := \{a \in \Gamma_{\text{in}} \mid \Xi \cup \mathscr{A}_{\text{CA}} \vdash_{\mathscr{R}} \overline{a}\}.$$

In words, Ξ would only contain assumptions in $\mathscr{A}_{\text{out}}$ instead of all assumptions, and Δ would contain assumptions attacked by $\Xi \cup \mathscr{A}_{\text{CA}}$ instead of ones attacked by Ξ. In effect, only attacks from $\mathscr{A}_{\text{out}} \cup \mathscr{A}_{\text{CA}}$ would be considered when deciding if Γ defends itself. In other words, attacks from a set containing assumptions from $\mathscr{A}_{\text{in}} \setminus \mathscr{A}_{\text{CA}} = \mathscr{A}_{\text{PSA}}$ would not be considered. This is not sufficient since there can be attacks from $\mathscr{A}_{\text{out}} \cup \mathscr{A}_{\text{PSA}}$ to Γ. By definition $\mathscr{A}_{\text{PSA}}$ can not by itself attack Γ, but nothing prevents assumptions from $\mathscr{A}_{\text{PSA}}$ forming a strict subset of a set of assumptions attacking an assumption in Γ. Thus the original algorithm can fail to identify assumptions in Γ that are not defended by Γ, namely, ones that are attacked by a set containing assumptions from $\mathscr{A}_{\text{out}}$ as well as $\mathscr{A}_{\text{PSA}}$.

For a concrete example where the original algorithm fails, consider the ABA framework with

$$\begin{aligned} \text{sentences } \mathscr{L} &= \{a,b,c,d,\overline{a},\overline{b},\overline{c},\overline{d}\} \\ \text{assumptions } \mathscr{A} &= \{a,b,c,d\} \\ \text{rules } \mathscr{R} &= \{(\overline{b} \leftarrow a),(\overline{a} \leftarrow b),(\overline{c} \leftarrow a),(\overline{c} \leftarrow b),(\overline{d} \leftarrow c,d)\} \end{aligned}$$

There are two preferred extensions, $\{a,d\}$ and $\{b,d\}$, since one can derive the contrary of all other assumptions except d from both a and b . Thus either of these assumptions joined with d attack all assumptions outside it and thus are subset-maximally admissible. The ideal assumption set is the maximal admissible set that is a subset of every preferred assumption set. In our example, the intersection of the preferred assumption sets is $\{d\}$. This set is not admissible, since $\{c,d\}$ attacks it without it defending itself. Thus the ideal assumption set is empty.

However, the algorithm as presented originally would return a non-empty assumption set. On lines 1–5 we obtain $\mathscr{A}_{\text{out}} = \{c\}$, $\mathscr{A}_{\text{in}} = \{a,b,d\}$, $\mathscr{A}_{\text{CA}} = \{a,b\}$, $\mathscr{A}_{\text{PSA}} = \{d\}$, and $\Gamma = \{d\}$. Then in the loop, $\Gamma_{\text{in}} = \{d\}$. On Line 8 the original algorithm checks which assumptions in $\mathscr{A}_{\text{out}}$ are not attacked by Γ. We get $\Xi = \{c\}$, since c is the only member of $\mathscr{A}_{\text{out}}$, and it also is not attacked

by Γ. On Line 9 any member of Γ that is attacked by $\Xi \cup \mathscr{A}_{\mathrm{CA}}$ is collected to Δ. In our case $\Xi \cup \mathscr{A}_{\mathrm{CA}} = \{a,b,c\}$, which does not attack d. Thus $\Delta = \emptyset$. Finally on Line 10, $\Gamma \setminus \emptyset = \Gamma$, so on Line 11 the algorithm terminates and on Line 12 incorrectly returns $\{d\}$ as the ideal assumption set.

References

Amgoud, L., & Vesic, S. (2011). Two roles of preferences in argumentation frameworks. In *Proc. ECSQARU*, Vol. 6717 of *Lecture Notes in Computer Science*, pp. 86–97. Springer.

Amgoud, L., & Vesic, S. (2014). Rich preference-based argumentation frameworks. *International Journal of Approximate Reasoning*, *55*(2), 585–606.

Atkinson, K., Baroni, P., Giacomin, M., Hunter, A., Prakken, H., Reed, C., Simari, G. R., Thimm, M., & Villata, S. (2017). Towards artificial argumentation. *AI Magazine*, *38*(3), 25–36.

Atkinson, K., Bench-Capon, T. J. M., & McBurney, P. (2006). PARMENIDES: facilitating deliberation in democracies. *Artificial Intelligence and Law*, *14*(4), 261–275.

Bao, Z., Čyras, K., & Toni, F. (2017). ABAplus: Attack reversal in abstract and structured argumentation with preferences. In An, B., Bazzan, A. L. C., Leite, J., Villata, S., & van der Torre, L. W. N. (Eds.), *Proc. PRIMA*, Vol. 10621 of *Lecture Notes in Computer Science*, pp. 420–437. Springer.

Baroni, P., Gabbay, D., Giacomin, M., & van der Torre, L. (Eds.). (2018). *Handbook of Formal Argumentation*. College Publications.

Beirlaen, M., Heyninck, J., Pardo, P., & Straßer, C. (2018). Argument strength in formal argumentation. *Journal of Applied Logics – IfCoLog Journal of Logics and their Applications*, *5*(3), 629–676.

Besnard, P., García, A. J., Hunter, A., Modgil, S., Prakken, H., Simari, G. R., & Toni, F. (2014). Introduction to structured argumentation. *Argument & Computation*, *5*(1), 1–4.

Besnard, P., & Hunter, A. (2008). *Elements of Argumentation*. MIT Press.

Besnard, P., & Hunter, A. (2014). Constructing argument graphs with deductive arguments: a tutorial. *Argument & Computation*, *5*(1), 5–30.

Besnard, P., & Hunter, A. (2018). A review of argumentation based on deductive arguments. In Baroni, P., Gabbay, D., Giacomin, M., & van der Torre, L. (Eds.), *Handbook of Formal Argumentation*, chap. 9, pp. 437–484. College Publications.

Bondarenko, A., Dung, P. M., Kowalski, R. A., & Toni, F. (1997). An abstract, argumentation-theoretic approach to default reasoning. *Artificial Intelligence*, *93*, 63–101.

Brewka, G., Delgrande, J. P., Romero, J., & Schaub, T. (2015). asprin: Customizing answer set preferences without a headache. In Bonet, B., & Koenig, S. (Eds.), *Proc. AAAI*, pp. 1467–1474. AAAI Press.

Brewka, G., Eiter, T., & Truszczynski, M. (2011). Answer set programming at a glance. *Communications of the ACM*, *54*(12), 92–103.

Caminada, M., Sá, S., Alcântara, J., & Dvořák, W. (2013). On the difference between assumption-based argumentation and abstract argumentation. In *Proc. BNAIC*, pp. 25–32. Delft University of Technology.

Caminada, M., & Schulz, C. (2017). On the equivalence between assumption-based argumentation and logic programming. *Journal of Artificial Intelligence Research*, *60*, 779–825.

Cerutti, F., Gaggl, S. A., Thimm, M., & Wallner, J. P. (2018). Foundations of implementations for formal argumentation. In Baroni, P., Gabbay, D., Giacomin, M., & van der Torre, L. (Eds.), *Handbook of Formal Argumentation*, chap. 15, pp. 688–767. College Publications.

Charwat, G., Dvořák, W., Gaggl, S. A., Wallner, J. P., & Woltran, S. (2015). Methods for solving reasoning problems in abstract argumentation - A survey. *Artificial Intelligence*, *220*, 28–63.

Craven, R., & Toni, F. (2016). Argument graphs and assumption-based argumentation. *Artificial Intelligence*, *233*, 1–59.

Craven, R., Toni, F., Cadar, C., Hadad, A., & Williams, M. (2012). Efficient argumentation for medical decision-making. In Brewka, G., Eiter, T., & McIlraith, S. A. (Eds.), *Proc. KR*, pp. 598–602. AAAI Press.

Craven, R., Toni, F., & Williams, M. (2013). Graph-based dispute derivations in assumption-based argumentation. In Black, E., Modgil, S., & Oren, N. (Eds.), *TAFA 2013 Revised Selected Papers*, Vol. 8306 of *Lecture Notes in Computer Science*, pp. 46–62. Springer.

Čyras, K. (2017). *ABA+: assumption-based argumentation with preferences*. Ph.D. thesis, Imperial College London, UK.

Čyras, K., Fan, X., Schulz, C., & Toni, F. (2018). Assumption-based argumentation: Disputes, explanations, preferences. In Baroni, P., Gabbay, D., Giacomin, M., & van der Torre, L. (Eds.), *Handbook of Formal Argumentation*, chap. 7, pp. 365–408. College Publications.

Čyras, K., Heinrich, Q., & Toni, F. (2021). Computational complexity of flat and generic assumption-based argumentation, with and without probabilities. *Artificial Intelligence*, *293*, 103449.

Čyras, K., Oliveira, T., Karamlou, A., & Toni, F. (2020). Assumption-based argumentation with preferences and goals for patient-centric reasoning with interacting clinical guidelines. *Argument & Computation*, *in press*.

Čyras, K., & Toni, F. (2016a). ABA+: Assumption-based argumentation with preferences. *CoRR*, *abs/1610.03024*.

Čyras, K., & Toni, F. (2016b). ABA+: Assumption-based argumentation with preferences. In Baral, C., Delgrande, J. P., & Wolter, F. (Eds.), *Proc. KR*, pp. 553–556. AAAI Press.

Čyras, K., & Toni, F. (2016c). Properties of ABA+ for non-monotonic reasoning. In Kern-Isberner, G., & Wassermann, R. (Eds.), *Proc. NMR*, pp. 25–34.

Delgrande, J. P., Schaub, T., Tompits, H., & Wang, K. (2004). A classification and survey of preference handling approaches in nonmonotonic reasoning. *Computational Intelligence*, *20*(2), 308–334.

Dimopoulos, Y., Nebel, B., & Toni, F. (2002). On the computational complexity of assumption-based argumentation for default reasoning. *Artificial Intelligence*, *141*(1/2), 57–78.

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

Dung, P. M., Kowalski, R. A., & Toni, F. (2006). Dialectic proof procedures for assumption-based, admissible argumentation. *Artificial Intelligence*, *170*(2), 114–159.

Dung, P. M., Kowalski, R. A., & Toni, F. (2009). Assumption-based argumentation. In Rahwan, I., & Simari, G. R. (Eds.), *Argumentation in Artificial Intelligence*, pp. 25–44. Springer.

Dung, P. M., Mancarella, P., & Toni, F. (2007). Computing ideal sceptical argumentation. *Artificial Intelligence*, *171*(10-15), 642–674.

Dung, P. M., Toni, F., & Mancarella, P. (2010). Some design guidelines for practical argumentation systems. In Baroni, P., Cerutti, F., Giacomin, M., & Simari, G. R. (Eds.), *Proc. COMMA*, Vol. 216 of *Frontiers in Artificial Intelligence and Applications*, pp. 183–194. IOS Press.

Dunne, P. E. (2009). The computational complexity of ideal semantics. *Artificial Intelligence*, *173*(18), 1559–1591.

Dvořák, W., & Dunne, P. E. (2018). Computational problems in formal argumentation and their complexity. In Baroni, P., Gabbay, D., Giacomin, M., & van der Torre, L. (Eds.), *Handbook of Formal Argumentation*, chap. 14. College Publications.

Egly, U., Gaggl, S. A., & Woltran, S. (2008). ASPARTIX: implementing argumentation frameworks using answer-set programming. In de la Banda, M. G., & Pontelli, E. (Eds.), *Proc. ICLP*, Vol. 5366 of *Lecture Notes in Computer Science*, pp. 734–738. Springer.

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

Egly, U., & Woltran, S. (2006). Reasoning in argumentation frameworks using quantified Boolean formulas. In Dunne, P. E., & Bench-Capon, T. J. M. (Eds.), *Proc. COMMA*, Vol. 144 of *Frontiers in Artificial Intelligence and Applications*, pp. 133–144. IOS Press.

Fan, X., & Toni, F. (2014). Decision making with assumption-based argumentation. In Black, E., Modgil, S., & Oren, N. (Eds.), *Theory and Applications of Formal Argumentation*, pp. 127–142, Berlin, Heidelberg. Springer Berlin Heidelberg.

Fan, X., & Toni, F. (2016). On the interplay between games, argumentation and dialogues. In Jonker, C. M., Marsella, S., Thangarajah, J., & Tuyls, K. (Eds.), *Proc. AAMAS*, pp. 260–268. ACM.

Fan, X., Toni, F., Mocanu, A., & Williams, M. (2014). Dialogical two-agent decision making with assumption-based argumentation. In Bazzan, A. L. C., Huhns, M. N., Lomuscio, A., & Scerri, P. (Eds.), *Proc. AAMAS*, pp. 533–540. IFAAMAS/ACM.

Gaertner, D., & Toni, F. (2007a). CaSAPI: A system for credulous and sceptical argumentation. In Simari, G. R., & Torroni, P. (Eds.), *Proc. NMR*, pp. 80–95.

Gaertner, D., & Toni, F. (2007b). Computing arguments and attacks in assumption-based argumentation. *IEEE Intelligent Systems*, *22*(6), 24–33.

Gaertner, D., & Toni, F. (2008). Hybrid argumentation and its properties. In Besnard, P., Doutre, S., & Hunter, A. (Eds.), *Proc. COMMA*, Vol. 172 of *Frontiers in Artificial Intelligence and Applications*, pp. 183–195. IOS Press.

Gaggl, S. A., Linsbichler, T., Maratea, M., & Woltran, S. (2020). Design and results of the second international competition on computational models of argumentation. *Artificial Intelligence*, *279*.

Gaggl, S. A., Manthey, N., Ronca, A., Wallner, J. P., & Woltran, S. (2015). Improved answer-set programming encodings for abstract argumentation. *Theory and Practice of Logic Programming*, *15*(4-5), 434–448.

García, A. J., & Simari, G. R. (2004). Defeasible logic programming: An argumentative approach. *Theory and Practice of Logic Programming*, *4*(1-2), 95–138.

García, A. J., & Simari, G. R. (2014). Defeasible logic programming: Delp-servers, contextual queries, and explanations for answers. *Argument & Computation*, *5*(1), 63–88.

García, A. J., & Simari, G. R. (2018). Argumentation based on logic programming. In Baroni, P., Gabbay, D., Giacomin, M., & van der Torre, L. (Eds.), *Handbook of Formal Argumentation*, chap. 8, pp. 409–435. College Publications.

Gebser, M., Harrison, A., Kaminski, R., Lifschitz, V., & Schaub, T. (2015). Abstract gringo. *Theory and Practice of Logic Programming*, *15*(4-5), 449–463.

Gebser, M., Kaminski, R., Kaufmann, B., Ostrowski, M., Schaub, T., & Wanko, P. (2016). Theory solving made easy with Clingo 5. In *Technical Communications of ICLP*, OASICS, pp. 2:1–2:15. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik.

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

Gebser, M., Kaufmann, B., Kaminski, R., Ostrowski, M., Schaub, T., & Schneider, M. T. (2011). Potassco: The Potsdam answer set solving collection. *AI Communications*, *24*(2), 107–124.

Gelfond, M., & Lifschitz, V. (1988). The stable model semantics for logic programming. In *Proc. ICLP/SLP*, pp. 1070–1080. MIT Press.

Gordon, T. F., Prakken, H., & Walton, D. (2007). The Carneades model of argument and burden of proof. *Artificial Intelligence*, *171*(10-15), 875–896.

Heyninck, J. (2019). *Investigations into the logical foundations of defeasible reasoning: an argumentative perspective*. Ph.D. thesis, Ruhr University Bochum, Germany.

Karamlou, A., Čyras, K., & Toni, F. (2019). Complexity results and algorithms for bipolar argumentation. In Elkind, E., Veloso, M., Agmon, N., & Taylor, M. E. (Eds.), *Proc. AAMAS*, pp. 1713–1721. International Foundation for Autonomous Agents and Multiagent Systems.

Lehtonen, T., Wallner, J. P., & Järvisalo, M. (2017). From structured to abstract argumentation: Assumption-based acceptance via AF reasoning. In Antonucci, A., Cholvy, L., & Papini, O. (Eds.), *Proc. ECSQARU*, Vol. 10369 of *Lecture Notes in Computer Science*, pp. 57–68. Springer.

Lehtonen, T., Wallner, J. P., & Järvisalo, M. (2019). Reasoning over assumption-based argumentation frameworks via direct answer set programming encodings. In *Proc. AAAI*, pp. 2938–2945. AAAI Press.

Lehtonen, T., Wallner, J. P., & Järvisalo, M. (2020). An answer set programming approach to argumentative reasoning in the ASPIC+ framework. In Calvanese, D., Erdem, E., & Thielscher, M. (Eds.), *Proc. KR*, pp. 636–646.

Modgil, S., & Prakken, H. (2013). A general account of argumentation with preferences. *Artificial Intelligence*, *195*, 361–397.

Modgil, S., & Prakken, H. (2018). Abstract rule-based argumentation. In Baroni, P., Gabbay, D., Giacomin, M., & van der Torre, L. (Eds.), *Handbook of Formal Argumentation*, chap. 6, pp. 287–364. College Publications.

Niemelä, I. (1999). Logic programs with stable model semantics as a constraint programming paradigm. *Annals of Mathematics and Artificial Intelligence*, *25*(3-4), 241–273.

Nieves, J. C., Cortés, U., & Osorio, M. (2008). Preferred extensions as stable models. *Theory and Practice of Logic Programming*, *8*(4), 527–543.

Niskanen, A., Wallner, J. P., & Järvisalo, M. (2019). Synthesizing argumentation frameworks from examples. *Journal of Artificial Intelligence Research*, *66*, 503–554.

Papadimitriou, C. H. (1994). *Computational complexity*. Addison-Wesley.

Prakken, H. (2010). An abstract framework for argumentation with structured arguments. *Argument & Computation*, *1*(2), 93–124.

Prakken, H., & Sartor, G. (2015). Law and logic: A review from an argumentation perspective. *Artificial Intelligence*, *227*, 214–245.

Sakama, C., & Rienstra, T. (2017). Representing argumentation frameworks in answer set programming. *Fundamenta Informaticae*, *155*(3), 261–292.

Schulz, C., & Toni, F. (2017). Labellings for assumption-based and abstract argumentation. *International Journal of Approximate Reasoning*, *84*, 110–149.

Thang, P. M., & Luong, H. T. (2013). Translating preferred subtheories into structured argumentation. *Journal of Logic and Computation*, *24*(4), 831–849.

Thimm, M. (2014). Tweety: A comprehensive collection of Java libraries for logical aspects of artificial intelligence and knowledge representation. In Baral, C., Giacomo, G. D., & Eiter, T. (Eds.), *Proc. KR*, pp. 528–537. AAAI Press.

Thimm, M. (2017). The Tweety library collection for logical aspects of artificial intelligence and knowledge representation. *Künstliche Intelligenz*, *31*(1), 93–97.

Thimm, M., & Villata, S. (2017). The first international competition on computational models of argumentation: Results and analysis. *Artificial Intelligence*, *252*, 267–294.

Toni, F. (2008). Assumption-based argumentation for epistemic and practical reasoning. In Casanovas, P., Sartor, G., Casellas, N., & Rubino, R. (Eds.), *Computable Models of the Law*, pp. 185–202, Berlin, Heidelberg. Springer Berlin Heidelberg.

Toni, F. (2013). A generalised framework for dispute derivations in assumption-based argumentation. *Artificial Intelligence*, *195*, 1–43.

Toni, F. (2014). A tutorial on assumption-based argumentation. *Argument & Computation*, *5*(1), 89–117.

Toni, F., & Sergot, M. (2011). Argumentation and answer set programming. In Balduccini, M., & Son, T. C. (Eds.), *Logic Programming, Knowledge Representation, and Nonmonotonic Reasoning: Essays Dedicated to Michael Gelfond on the Occasion of His 65th Birthday*, pp. 164–180. Springer.

Valiant, L. G., & Vazirani, V. V. (1986). NP is as easy as detecting unique solutions. *Theoretical Computer Science*, *47*(3), 85–93.

Wakaki, T. (2017a). Assumption-based argumentation equipped with preferences and constraints. In Moral, S., Pivert, O., Sánchez, D., & Marín, N. (Eds.), *Proc. SUM*, Vol. 10564 of *Lecture Notes in Computer Science*, pp. 178–193. Springer.

Wakaki, T. (2017b). Assumption-based argumentation equipped with preferences and its application to decision making, practical reasoning, and epistemic reasoning. *Computational Intelligence*, *33*(4), 706–736.

Wakaki, T., & Nitta, K. (2008). Computing argumentation semantics in answer set programming. In Hattori, H., Kawamura, T., Idé, T., Yokoo, M., & Murakami, Y. (Eds.), *Proc. JSAI, Revised Selected Papers*, Vol. 5447 of *Lecture Notes in Computer Science*, pp. 254–269. Springer.