ECAI 2016
G.A. Kaminka et al. (Eds.)

doi:10.3233/978-1-61499-672-9-252

A Uniform Account of Realizability in Abstract Argumentation

Thomas Linsbichler[1] and **Jörg Pührer**[2] and **Hannes Strass**[2]

Abstract. We introduce a general framework for analyzing realizability in abstract dialectical frameworks (ADFs) and various of its subclasses. In particular, the framework applies to Dung argumentation frameworks, SETAFs by Nielsen and Parsons, and bipolar ADFs. We present a uniform characterization method for the admissible, complete, preferred and model/stable semantics. We employ this method to devise an algorithm that decides realizability for the mentioned formalisms and semantics; moreover the algorithm allows for constructing a desired knowledge base whenever one exists. The algorithm is built in a modular way and thus easily extensible to new formalisms and semantics. We have implemented our approach in answer set programming, and used the implementation to obtain several novel results on the relative expressiveness of the abovementioned formalisms.

1 Introduction

The abstract argumentation frameworks (AFs) introduced by Dung [9] have garnered increasing attention in the recent past. In his seminal paper, Dung showed how an abstract notion of argument (seen as an atomic entity) and the notion of individual attacks between arguments together could reconstruct several established KR formalisms in argumentative terms. Despite the generality of those and many more results in the field that was sparked by that paper, researchers also noticed that the restriction to *individual attacks* is often overly limiting, and devised extensions and generalizations of Dung's frameworks: directions included generalizing individual attacks to *collective attacks* [23], leading to so-called SETAFs; others started offering a *support* relation between arguments [8], preferences among arguments [1, 22], or attacks on attacks into arbitrary depth [2]. This is only the tip of an iceberg, for a more comprehensive overview we refer to the work of Brewka, Polberg, and Woltran [5].

One of the most recent and most comprehensive generalizations of AFs has been presented by Brewka and Woltran [6] (and later continued by Brewka et al. [4]) in the form of *abstract dialectical frameworks (ADFs)*. These ADFs offer any type of link between arguments: individual attacks (as in AFs), collective attacks (as in SETAFs), and individual and collective support, to name only a few. This generality is achieved through so-called *acceptance conditions* associated to each statement. Roughly, the meaning of relationships between arguments is not fixed in ADFs, but is specified by the user for each argument in the form of Boolean functions (acceptance functions) on the argument's parents. However, this generality comes with a price: Strass and Wallner [29] found that the complexity of the associated reasoning problems of ADFs is in general higher than in AFs (one level up in the polynomial hierarchy). Fortunately, the subclass of *bipolar ADFs* (defined by Brewka and Woltran [6]) is as complex as AFs (for all considered semantics) while still offering a wide range of modeling capacities [29]. However, there has only been little concerted effort so far to exactly analyze and compare the expressiveness of the abovementioned languages.

This paper is about exactly analyzing means of expression for argumentation formalisms. Instead of motivating expressiveness in natural language and showing examples that some formalisms seem to be able to express but others do not, we tackle the problem in a formal way. We use a precise mathematical definition of expressiveness: a set of interpretations is *realizable* by a formalism under a semantics if and only if there exists a knowledge base of the formalism whose semantics is exactly the given set of interpretations. Studying realizability in AFs has been started by Dunne et al. [11, 10], who analyzed realizability for extension-based semantics, that is, interpretations represented by sets where arguments are either accepted (in the extension set) or not accepted (not in the extension set). While their initial work disregarded arguments that are never accepted, there have been continuations where the existence of such "invisible" arguments is ruled out [3, 20]. Dyrkolbotn [12] began to analyze realizability for labeling-based semantics of AFs, that is, three-valued semantics where arguments can be accepted (mapped to true), rejected (mapped to false) or neither (mapped to unknown). Strass [28] started to analyze the relative expressiveness of two-valued semantics for ADFs (relative with respect to related formalisms). Most recently, Pührer [26] presented precise characterizations of realizability for ADFs under several three-valued semantics, namely admissible, grounded, complete, and preferred. The term "precise characterizations" means that he gave necessary and sufficient conditions for an interpretation set to be ADF-realizable under a semantics.

The present paper continues this line of work by lifting it to a much more general setting. We combine the works of Dunne et al. [10], Pührer [26], and Strass [28] into a unifying framework, and at the same time extend them to formalisms and semantics not considered in the respective papers: we treat several formalisms, namely AFs, SETAFs, and (B)ADFs, while the previous works all used different approaches and techniques. This is possible because all of these formalisms can be seen as subclasses of ADFs that are obtained by suitably restricting the acceptance conditions.

Another important feature of our framework is that we uniformly use three-valued interpretations as the underlying model theory. In particular, this means that arguments cannot be "invisible" any more since the underlying vocabulary of arguments is always implicit in each interpretation. Technically, we always assume a fixed underlying vocabulary and consider our results parametric in that vocabulary. In contrast, for example, Dyrkolbotn [12] presents a construc-

[1] Institute of Information Systems, TU Wien, Vienna, Austria
[2] Computer Science Institute, Leipzig University, Leipzig, Germany

tion for realizability that introduces new arguments into the realizing knowledge base; we do not allow that. While sometimes the introduction of new arguments can make sense, for example if new information becomes available about a domain or a debate, it is not sensible in general, as these new arguments would be purely technical with an unclear dialectical meaning. Moreover, it would lead to a different notion of realizability, where most of the realizability problems would be significantly easier, if not trivial.

The paper proceeds as follows. We begin with recalling and introducing the basis and basics of our work – the formalisms we analyze and the methodology with which we analyze them. Next we introduce our general framework for realizability; the major novelty is our consistent use of so-called characterization functions, firstly introduced by Pührer [26], which we adapt to further semantics. The main workhorse of our approach will be a parametric propagate-and-guess algorithm for deciding whether a given interpretation set is realizable in a formalism under a semantics. We then analyze the relative expressiveness of the considered formalisms, presenting several new results that we obtained using an implementation of our framework. We conclude with a discussion.

2 Preliminaries

We make use of standard mathematical concepts like functions and partially ordered sets. For a function $f : X \to Y$ we denote the *update of f with a pair* $(x, y) \in X \times Y$ by $f|_y^x : X \to Y$ with $z \mapsto y$ if $z = x$, and $z \mapsto f(z)$ otherwise. For a function $f : X \to Y$ and $y \in Y$, its preimage is $f^{-1}(y) = \{x \in X \mid f(x) = y\}$. A *partially ordered set* is a pair $(S, \sqsubseteq)$ with $\sqsubseteq$ a partial order on S. A partially ordered set $(S, \sqsubseteq)$ is a *complete lattice* if and only if every $S' \subseteq S$ has both a greatest lower bound (glb) $\bigsqcap S' \in S$ and a least upper bound (lub) $\bigsqcup S' \in S$. A partially ordered set $(S, \sqsubseteq)$ is a *complete meet-semilattice* iff every non-empty subset $S' \subseteq S$ has a greatest lower bound $\bigsqcap S' \in S$ (the *meet*) and every ascending chain $C \subseteq S$ has a least upper bound $\bigsqcup C \in S$.

Three-Valued Interpretations Let A be a fixed finite set of statements. An *interpretation* is a mapping $v : A \to \{\mathbf{t}, \mathbf{f}, \mathbf{u}\}$ that assigns one of the truth values true ($\mathbf{t}$), false ($\mathbf{f}$) or unknown ($\mathbf{u}$) to each statement. An interpretation is *two-valued* if $v(A) \subseteq \{\mathbf{t}, \mathbf{f}\}$, that is, the truth value $\mathbf{u}$ is not assigned. Two-valued interpretations v can be extended to assign truth values $v(\varphi) \in \{\mathbf{t}, \mathbf{f}\}$ to propositional formulas φ as usual.

The three truth values are partially ordered according to their information content: we have $\mathbf{u} <_i \mathbf{t}$ and $\mathbf{u} <_i \mathbf{f}$ and no other pair in $<_i$, which intuitively means that the classical truth values contain more information than the truth value unknown. As usual, we denote by $\leq_i$ the partial order associated to the strict partial order $<_i$. The pair $(\{\mathbf{t}, \mathbf{f}, \mathbf{u}\}, \leq_i)$ forms a complete meet-semilattice with the information meet operation $\sqcap_i$. This meet can intuitively be interpreted as *consensus* and assigns $\mathbf{t} \sqcap_i \mathbf{t} = \mathbf{t}$, $\mathbf{f} \sqcap_i \mathbf{f} = \mathbf{f}$, and returns $\mathbf{u}$ otherwise.

The information ordering $\leq_i$ extends in a straightforward way to interpretations v_1, v_2 over A in that $v_1 \leq_i v_2$ iff $v_1(a) \leq_i v_2(a)$ for all $a \in A$. We say for two interpretations v_1, v_2 that v_2 *extends* v_1 iff $v_1 \leq_i v_2$. The set $\mathcal{V}$ of all interpretations over A forms a complete meet-semilattice with respect to the information ordering $\leq_i$. The consensus meet operation $\sqcap_i$ of this semilattice is given by $(v_1 \sqcap_i v_2)(a) = v_1(a) \sqcap_i v_2(a)$ for all $a \in A$. The least element of $(\mathcal{V}, \leq_i)$ is the valuation $v_{\mathbf{u}} : A \to \{\mathbf{u}\}$ mapping all statements to unknown – the least informative interpretation. By $\mathcal{V}_2$ we denote the set of two-valued interpretations; they are the $\leq_i$-maximal elements of the meet-semilattice $(\mathcal{V}, \leq_i)$. We denote by $[v]_2$ the set of all two-valued interpretations that extend v. The elements of $[v]_2$ form an $\leq_i$-antichain with greatest lower bound $v = \bigsqcap_i [v]_2$.

Abstract Argumentation Formalisms An *abstract dialectical framework (ADF)* is a tuple $D = (A, L, C)$ where A is a set of statements (representing positions one can take or not take in a debate), $L \subseteq A \times A$ is a set of links (representing dependencies between the positions), $C = \{C_a\}_{a \in A}$ is a collection of functions $C_a : 2^{par(a)} \to \{\mathbf{t}, \mathbf{f}\}$, one for each statement $a \in A$. The function C_a is the *acceptance condition of* a and expresses whether a can be accepted, given the acceptance status of its parents $par(a) = \{b \in S \mid (b, a) \in L\}$. We usually represent each C_a by a propositional formula φ_a over $par(a)$. For the acceptance condition C_a, we take $C_a(M \cap par(a)) = \mathbf{t}$ to hold iff M is a model of φ_a.

Brewka and Woltran [6] introduced a useful subclass of ADFs: an ADF $D = (A, L, C)$ is *bipolar* iff all links in L are supporting or attacking (or both). A link $(b, a) \in L$ is *supporting in* D iff for all $M \subseteq par(a)$, we have that $C_a(M) = \mathbf{t}$ implies $C_a(M \cup \{b\}) = \mathbf{t}$. Symmetrically, a link $(b, a) \in L$ is *attacking in* D iff for all $M \subseteq par(a)$, we have that $C_a(M \cup \{b\}) = \mathbf{t}$ implies $C_a(M) = \mathbf{t}$. Intuitively, a link $(b, a) \in L$ is supporting iff it can never be the case that there is some state of affairs where we accept a and reject b, but after additionally also accepting b do not accept a any more. Symmetrically, a link $(b, a) \in L$ is attacking iff it can never be the case that we reject a and b, but after accepting b also accept a. If a link (b, a) is both supporting and attacking then b has no actual influence on a. (But the link does not violate bipolarity.) We write BADFs as $D = (A, L^+ \cup L^-, C)$ and mean that L^+ contains all supporting links and L^- all attacking links; see also Example 1 below.[3]

The semantics of ADFs can be defined using an operator Γ_D over three-valued interpretations [6, 4]. For an ADF D and a three-valued interpretation v, the interpretation $\Gamma_D(v)$ is given by

$$a \mapsto \bigsqcap_i \{w(\varphi_a) \mid w \in [v]_2\}$$

That is, for each statement a, the operator returns the consensus truth value for its acceptance formula φ_a, where the consensus takes into account all possible two-valued interpretations w that extend the input valuation v. If this v is two-valued, we get $[v]_2 = \{v\}$ and thus $\Gamma_D(v)(a) = v(\varphi_a)$.

The standard semantics of ADFs are now defined as follows. For ADF D, an interpretation $v : A \to \{\mathbf{t}, \mathbf{f}, \mathbf{u}\}$ is

- *admissible* iff $v \leq_i \Gamma_D(v)$;
- *complete* iff $\Gamma_D(v) = v$;
- *preferred* iff it is $\leq_i$-maximal admissible;
- a *two-valued model* iff it is two-valued and $\Gamma_D(v) = v$.

We denote the sets of interpretations that are admissible, complete, preferred, and two-valued models by $adm(D)$, $com(D)$, $prf(D)$ and $mod(D)$, respectively. These definitions are proper generalizations of Dung's notions for AFs: For an AF (A, R), where $R \subseteq A \times A$ is the attack relation, the *ADF associated to* (A, R) is $D_{(A,R)} = (A, R, C)$ with $C = \{\varphi_a\}_{a \in A}$ and $\varphi_a = \bigwedge_{b:(b,a) \in R} \neg b$ for all $a \in A$. AFs inherit their semantics from the definitions for ADFs [4, Theorems 2 and 4]. In particular, an interpretation is *stable* for an AF (A, R) if and only if it is a two-valued model of $D_{(A,R)}$.

[3] Other than a part of the name, there is no relationship of bipolar ADFs with the bipolar framework of Cayrol and Lagasquie-Schiex [8]; Brewka and Woltran gave a more detailed comparison of the two formalisms [6].

Example 1. Consider the bipolar ADF $D = (A, L^+ \cup L^-, C)$ over vocabulary $A = \{a, b, c\}$ with

$$\varphi_a = b \wedge c, \qquad \varphi_b = \neg a, \qquad \varphi_c = a \vee \neg b$$

whence it follows that $L^+ = \{(b, a), (c, a), (a, c)\}$ and $L^- = \{(a, b), (b, c)\}$. (The types of links can be read off the polarities of the statements in the acceptance formulas [28, Theorem 1]; statements occurring only positively are supporting, those that occur only negatively are attacking.) Intuitively, the acceptance condition φ_a is a group support: a can only be accepted if *both* b and c are accepted. For b, we have an individual attack just like in standard AFs: b is attacked by a, and therefore only be accepted if a is not accepted. The acceptance condition of c consists of a support by a that overpowers an attack by b; in other words, to be able to accept c, the support from a must be present or the attack from b must be absent, and if both are present then the support is stronger. (We could have specified that the attack is stronger than the support by writing $\varphi_c = a \wedge \neg b$.) Regarding the semantics of D, we find that $mod(D) = prf(D) = \{v_1\}$ with $v_1 = \{a \mapsto \mathbf{f}, b \mapsto \mathbf{t}, c \mapsto \mathbf{f}\}$. Furthermore, we have $adm(D) = com(D) = prf(D) \cup \{v_2\}$ where $v_2 = \{a \mapsto \mathbf{u}, b \mapsto \mathbf{u}, c \mapsto \mathbf{u}\}$. Intuitively, setting all statements to $\mathbf{u}$ is always admissible; in this case it is also complete because no statement is unconditionally accepted or rejected. The non-trivial interpretation v_1 is a model of the BADF because intuitively: a is rejected since it misses the support of c; b is accepted because the attack from a does not materialize; c is rejected because it misses support from a and at the same time is attacked by b. ■

A SETAF is a pair $S = (A, X)$ where $X \subseteq (2^A \setminus \{\emptyset\}) \times A$ is the (set) attack relation. We define three-valued counterparts of the semantics introduced by Nielsen and Parsons [23], following the same conventions as in three-valued semantics of AFs [7] and argumentation formalisms in general. Given a statement $a \in A$ and an interpretation v we say that a is *acceptable* with respect to v if and only if $\forall (B, a) \in X \exists a' \in B : v(a') = \mathbf{f}$ and a is *unacceptable* with respect to v if and only if $\exists (B, a) \in X \forall a' \in B : v(a') = \mathbf{t}$.

For an interpretation $v : A \to \{\mathbf{t}, \mathbf{f}, \mathbf{u}\}$ it holds that

- $v \in adm(S)$ iff for all $a \in A$, a is acceptable wrt. v if $v(a) = \mathbf{t}$ and a is unacceptable wrt. v if $v(a) = \mathbf{f}$;
- $v \in com(S)$ iff for all $a \in A$, a is acceptable wrt. v iff $v(a) = \mathbf{t}$ and a is unacceptable wrt. v iff $v(a) = \mathbf{f}$;
- $v \in prf(S)$ iff v is $\leq_i$-maximal admissible; and
- $v \in mod(S)$ iff $v \in adm(F)$ and $\nexists a \in A : v(a) = \mathbf{u}$.

For a SETAF $S = (A, X)$ the corresponding ADF D_S has acceptance formula $\varphi_a = \bigwedge_{(B,a) \in X} \bigvee_{a' \in B} \neg a'$ for each statement $a \in A$.

Proposition 1. *For any SETAF $S = (A, X)$ it holds that $\sigma(S) = \sigma(D_S)$, where $\sigma \in \{adm, com, prf, mod\}$.*

Proof. Given interpretation v and statement a, it holds that $\Gamma_{D_S}(v)(a) = \mathbf{t}$ iff $\forall w \in [v]_2 : w(a) = \mathbf{t}$ iff $\forall (B, a) \in X \; \exists a' \in B : v(a') = \mathbf{f}$ iff a is acceptable wrt. v and $\Gamma_{D_S}(v)(a) = \mathbf{f}$ iff $\forall w \in [v]_2 : w(a) = \mathbf{f}$ iff $\exists (B, a) \in X \; \forall a' \in B : v(a') = \mathbf{t}$ iff a is unacceptable wrt. v. Hence $\sigma(S) = \sigma(D_S)$ for $\sigma \in \{adm, com, prf, mod\}$. □

Realizability A set $V \subseteq \mathcal{V}$ of interpretations is *realizable* in a formalism $\mathcal{F}$ under a semantics σ if and only if there exists a knowledge base $\mathsf{kb} \in \mathcal{F}$ having exactly $\sigma(\mathsf{kb}) = V$. Pührer [26] characterized realizability for ADFs under various three-valued semantics. We will reuse the central notions for capturing the complete semantics in this work.

Definition 1 (Pührer [26]). Let V be a set of interpretations. A function $f : \mathcal{V}_2 \to \mathcal{V}_2$ is a *com-characterization* of V iff: for each $v \in \mathcal{V}$ we have $v \in V$ iff for each $a \in A$:

- $v(a) \neq \mathbf{u}$ implies $f(v_2)(a) = v(a)$ for all $v_2 \in [v]_2$ and
- $v(a) = \mathbf{u}$ implies $f(v_2')(a) = \mathbf{t}$ and $f(v_2'')(a) = \mathbf{f}$ for some $v_2', v_2'' \in [v]_2$. ▲

Intuitively, a com-characterization f assigns the Boolean value $f(v)(a)$ to a statement a that the acceptance condition of a would have under v in an ADF that has V as its complete semantics. From a function of this kind we can build a corresponding ADF by the following construction. For $f : \mathcal{V}_2 \to \mathcal{V}_2$, we define D_f as the ADF where the acceptance formula for each statement a is given by

$$\varphi_a^f = \bigvee_{\substack{w \in \mathcal{V}_2, \\ f(w)(a) = \mathbf{t}}} \phi_w \quad \text{with} \quad \phi_w = \bigwedge_{w(a') = \mathbf{t}} a' \wedge \bigwedge_{w(a') = \mathbf{f}} \neg a'$$

Observe that for any $v \in \mathcal{V}_2$ we have $v(\phi_w) = \mathbf{t}$ iff $v = w$ by definition. Intuitively, the acceptance condition φ_a^f is constructed such that v is a model of φ_a^f if and only if we find $f(v)(a) = \mathbf{t}$.

Proposition 2 (Pührer [26]). *Let $V \subseteq \mathcal{V}$ be a set of interpretations. (1) For each ADF D with $com(D) = V$, there is a com-characterization f_D for V; (2) for each com-characterization $f : \mathcal{V}_2 \to \mathcal{V}_2$ for V we have $com(D_f) = V$.*

The result shows that V can be realized under complete semantics if and only if there is a com-characterization for V.

3 A General Framework for Realizability

The underlying idea of our framework is that all abstract argumentation formalisms introduced in the previous section can be viewed as subclasses of ADFs. This is clear for ADFs themselves and for BADFs by definition; for (SET)AFs it is fairly easy to see. However, knowing that these formalisms can be recast as ADFs is not enough. To employ this knowledge for realizability, we must precisely characterize the corresponding subclasses in terms of restricting the ADFs' acceptance functions. Fortunately, this is also possible and paves the way for the framework we present in this section. Most importantly, we will make use of the fact that different formalisms and different semantics can be characterized modularly, that is, independently of each other.

Towards a uniform account of realizability for ADFs under different semantics, we start with a new characterization of realizability for ADFs under admissible semantics that is based on a notion similar in spirit to com-characterizations.

Definition 2. Let V be a set of interpretations. A function $f : \mathcal{V}_2 \to \mathcal{V}_2$ is an *adm-characterization* of V iff: for each $v \in \mathcal{V}$ we have $v \in V$ iff for every $a \in A$:

- $v(a) \neq \mathbf{u}$ implies $f(v_2)(a) = v(a)$ for all $v_2 \in [v]_2$. ▲

Similar as for a com-characterization, an adm-characterization f assigns the value $f(v)(a)$ to a statement a that the acceptance condition of a would evaluate to under v in an ADF that has V as its admissible semantics. Note that the only difference to Definition 1 is dropping the second condition related to statements with truth value $\mathbf{u}$. While,

the two conditions in Definition 1 capture the relation $\Gamma_{D_f}(v) = v$, the remaining one in Definition 2 boils down to $v \leq_i \Gamma_{D_f}(v)$ that defines the admissible semantics.

Proposition 3. *Let $V \subseteq \mathcal{V}$ be a set of interpretations. (1) For each ADF D such that $adm(D) = V$, there is an adm-characterization f_D for V; (2) for each adm-characterization $f : \mathcal{V}_2 \to \mathcal{V}_2$ for V we have $adm(D_f) = V$.*

Proof. (1) We define the function $f_D : \mathcal{V}_2 \to \mathcal{V}_2$ as $f_D(v_2)(a) = v_2(\varphi_a)$ for every $v_2 \in \mathcal{V}_2$ and $a \in A$ where φ_a is the acceptance formula of a in D. We will show that f_D is an adm-characterization for $V = adm(D)$. Let v be an interpretation. Consider the case $v \in adm(D)$ and $v(a) \neq u$ for some $a \in A$ and some $v_2 \in [v]_2$. From $v \leq_i \Gamma_D(v)$ we get $v_2(\varphi_a) = v(a)$. By definition of f_D it follows that $f_D(v_2)(a) = v(a)$. Now assume $v \notin adm(D)$ and consequently $v \not\leq_i \Gamma_D(v)$. There must be some $a \in A$ such that $v(a) \neq \mathbf{u}$ and $v(a) \neq \Gamma_D(v)(a)$. Hence, there is some $v_2 \in [v]_2$ with $v_2(\varphi_a) \neq v(a)$ and $f_D(v_2)(a) \neq v(a)$ by definition of f_D. Thus, f_D is an adm-characterization for V.

(2) Observe that for every two-valued interpretation v_2 and every $a \in A$ we have $f(v_2)(a) = v_2(\varphi_a^f)$. ($\subseteq$): Let $v \in adm(D_f)$ be an interpretation and $a \in A$ a statement such that $v(a) \neq \mathbf{u}$. Let v_2 be a two-valued interpretation with $v_2 \in [v]_2$. Since $v \leq_i \Gamma_{D_f}(v)$ we have $v(a) = v_2(\varphi_a^f)$. Therefore, by our observation it must also hold that $f(v_2)(a) = v(a)$. Thus, by Definition 2, $v \in V$. ($\supseteq$): Consider an interpretation v such that $v \notin adm(D_f)$. We show that $v \notin V$. From $v \notin adm(D_f)$ we get $v \not\leq_i \Gamma_{D_f}(v)$. There must be some $a \in A$ such that $v(a) \neq \mathbf{u}$ and $v(a) \neq \Gamma_{D_f}(v)(a)$. Hence, there is some $v_2 \in [v]_2$ with $v_2(\varphi_a^f) \neq v(a)$ and consequently $f(v_2)(a) \neq v(a)$. Thus, by Definition 2 we have $v \notin V$. □

When listing sets of interpretations in examples, for the sake of readability we represent three-valued interpretations by sequences of truth values, tacitly assuming that the underlying vocabulary is given and has an associated total ordering. For example, for the vocabulary $A = \{a, b, c\}$ we represent the interpretation $\{a \mapsto \mathbf{t}, b \mapsto \mathbf{f}, c \mapsto \mathbf{u}\}$ by the sequence $\mathbf{tfu}$.

Example 2. Consider the sets $V_1 = \{\mathbf{uuu}, \mathbf{tff}, \mathbf{ftu}\}$ and $V_2 = \{\mathbf{tff}, \mathbf{ftu}\}$ of interpretations over $A = \{a, b, c\}$. The mapping $f = \{\mathbf{ttt} \mapsto \mathbf{ftt}, \mathbf{ttf} \mapsto \mathbf{tft}, \mathbf{tft} \mapsto \mathbf{ttt}, \mathbf{tff} \mapsto \mathbf{tff}, \mathbf{ftt} \mapsto \mathbf{ftf}, \mathbf{ftf} \mapsto \mathbf{ftt}, \mathbf{fft} \mapsto \mathbf{ttf}, \mathbf{fff} \mapsto \mathbf{ftf}\}$ is an adm-characterization for V_1. Thus, the ADF D_f has V_1 as its admissible interpretations. Indeed, the realizing ADF has the following acceptance conditions:

$$\varphi_a^f \equiv (a \wedge b \wedge \neg c) \vee (a \wedge \neg b) \vee (\neg a \wedge \neg b \wedge c)$$
$$\varphi_b^f \equiv (a \wedge c) \vee (\neg a \wedge b) \vee (\neg a \wedge \neg b \wedge \neg c)$$
$$\varphi_c^f \equiv (a \wedge b) \vee (\neg a \wedge b \wedge \neg c) \vee (\neg b \wedge c)$$

For V_2 no adm-characterization exists because $\mathbf{uuu} \notin V_2$, but the implication of Definition 2 trivially holds for a, b, and c. ■

We have seen that the construction D_f for realizing under complete semantics can also be used for realizing a set V of interpretations under admissible semantics. The only difference is that we here require f to be an adm-characterization instead of a com-characterization for V. Note that admissible semantics can be characterized by properties that are easier to check than existence of an adm-characterization (see the work of Pührer [26]). However, using the same type of characterizations for different semantics allows for a unified approach for checking realizability and constructing a realizing ADF in case one exists.

For realizing under the model semantics, we can likewise present an adjusted version of com-characterizations.

Definition 3. Let $V \subseteq \mathcal{V}$ be a set of interpretations. A function $f : \mathcal{V}_2 \to \mathcal{V}_2$ is a *mod-characterization* of V if and only if: (1) f is defined on V (that is, $V \subseteq \mathcal{V}_2$) and (2) for each $v \in \mathcal{V}_2$, we have $v \in V$ iff $f(v) = v$. ▲

As we can show, there is a one-to-one correspondence between mod-characterizations and ADF realizations.

Proposition 4. *Let $V \subseteq \mathcal{V}$ be a set of interpretations. (1) For each ADF D such that $mod(D) = V$, there is a mod-characterization f_D for V; (2) vice versa, for each mod-characterization $f : \mathcal{V}_2 \to \mathcal{V}_2$ for V we find $mod(D_f) = V$.*

A related result was given by Strass [28, Proposition 10]. The characterization we presented here fits into the general framework of this paper and is directly usable for our realizability algorithm. The next result summarizes how ADF realizability can be captured by different types of characterizations for the semantics we considered so far.

Theorem 5. *Let $V \subseteq \mathcal{V}$ be a set of interpretations and consider $\sigma \in \{adm, com, mod\}$. There is an ADF D such that $\sigma(D) = V$ if and only if there is a σ-characterization for V.*

The preferred semantics of an ADF D is closely related to its admissible semantics as, by definition, the preferred interpretations of D are its $\leq_i$-maximal admissible interpretations. As a consequence we can also describe preferred realizability in terms of adm-characterizations. We use the lattice-theoretic standard notation $\max_{\leq_i} V$ to denote the $\leq_i$-maximal elements of a given set V.

Corollary 6. *Let $V \subseteq \mathcal{V}$ be a set of interpretations. There is an ADF D with $prf(D) = V$ iff there is an adm-characterization for some $V' \subseteq \mathcal{V}$ with $V \subseteq V'$ and $\max_{\leq_i} V' = V$.*

Finally, we give a result on the complexity of deciding realizability for the mentioned formalisms and semantics. We assume here that the representation of an interpretation-set V over vocabulary A has size $\Theta(3^{|A|})$, that is, the size grows asymptotically in the order of $3^{|A|}$. A possible encoding could be a bit string of length $3^{|A|}$ where the presence (or absence) of each $v \in V$ is encoded by a 1 (or 0) at a particular position in the string. There might be specific V with smaller possible representations, but we have no grounds to presume a representation that is exponentially better in the general case.

Proposition 7. *Let $\mathcal{F} \in \{AF, SETAF, BADF, ADF\}$ be a formalism and $\sigma \in \{adm, com, prf, mod\}$ be a semantics. The decision problem "Given a vocabulary A and a set $V \subseteq \mathcal{V}$ of interpretations over A, is there a $\mathsf{kb} \in \mathcal{F}$ such that $\sigma(\mathsf{kb}) = V$?" can be decided in nondeterministic time that is polynomial in the size of V.*

Proof. Roughly, we guess a function $f : \mathcal{V}_2 \to \mathcal{V}_2$ and verify that it is a σ-characterization. Such a function f can be represented in size $O(2^{|A|} \cdot |A|)$, that is, at most polynomial in the input of size $O(3^{|A|})$: the fact that $n \cdot 2^n \in o(3^n) \subseteq O(3^n)$ follows from

$$\lim_{n\to\infty} \frac{n \cdot 2^n}{3^n} = \lim_{n\to\infty} \frac{n}{\left(\frac{3}{2}\right)^n} \overset{*}{=} \lim_{n\to\infty} \frac{1}{\ln \frac{3}{2} \cdot \left(\frac{3}{2}\right)^n} = 0$$

where the starred equality holds by L'Hôpital's rule.

To verify that the guessed f is indeed a σ-characterization, we check (some of) the properties of Definition 1. For $\sigma = com$, this can be done in polynomial time as follows: for each $v \in \mathcal{V}$ and $a \in A$, we

look at the set $[v]_2 \subseteq \mathcal{V}_2$ (which is at most polynomial in the input) and check for the respective witness interpretations (if $v(a) = \mathbf{u}$) or their absence (if $v(a) \neq \mathbf{u}$). For $\sigma = adm$, there are even less conditions to check. For $\sigma = mod$, we compute the set F of fixpoints of f (by going through $\mathcal{V}$ once and checking $f(v) = v$ for each $v \in \mathcal{V}$) and verify that $F = V$. For $\sigma = prf$, we guess the V' (with $V \subseteq V' \subseteq \mathcal{V}$) from Corollary 6 alongside f and verify that f is an adm-characterization for V' and that $\max_{\leq_i} V' = V$. □

3.1 Deciding Realizability: Algorithm 1

Our main algorithm for deciding realizability is a propagate-and-guess algorithm in the spirit of the DPLL algorithm for deciding propositional satisfiability [19]. It is generic with respect to (1) the formalism $\mathcal{F}$ and (2) the semantics σ for which should be realized. To this end, the propagation part of the algorithm is kept exchangeable and will vary depending on formalism and semantics. Roughly, in the propagation step the algorithm uses the desired set V of interpretations to derive certain necessary properties of the realizing knowledge base (line 2). This is the essential part of the algorithm: the derivation rules (*propagators*) used there are based on characterizations of realizability with respect to formalism and semantics. (Propagators will be explained in detail in the next two subsections.) Once propagation of properties has reached a fixed point (line 7), the algorithm checks whether the derived information is sufficient to construct a knowledge base. If so, the knowledge base can be constructed and returned (line 9). Otherwise (no more information can be obtained through propagation and there is not enough information to construct a knowledge base yet), the algorithm guesses another assignment for the characterization (line 11) and calls itself recursively.

The main data structure that Algorithm 1 operates on is a set of triples $(v, a, \mathbf{x})$ consisting of a two-valued interpretation $v \in \mathcal{V}_2$, an atom $a \in A$ and a truth value $\mathbf{x} \in \{\mathbf{t}, \mathbf{f}\}$. This data structure is intended to represent the σ-characterizations introduced in Definitions 1 to 3. There, a σ-characterization is a function $f : \mathcal{V}_2 \to \mathcal{V}_2$ from two-valued interpretations to two-valued interpretations. However, as the algorithm builds the σ-characterization step by step and there might not even be a σ-characterization in the end (because V is not realizable), we use a set F of triples $(v, a, \mathbf{x})$ to be able to represent both partial and incoherent states of affairs. The σ-characterization candidate induced by F is partial if we have that for some v and a, neither $(v, a, \mathbf{t}) \in F$ nor $(v, a, \mathbf{f}) \in F$; likewise, the candidate is incoherent if for some v and a, both $(v, a, \mathbf{t}) \in F$ and $(v, a, \mathbf{f}) \in F$. If F is neither partial nor incoherent, it gives rise to a unique σ-characterization that can be used to construct the knowledge base realizing the desired set of interpretations. The correspondence to the characterization-function is then such that $f(v)(a) = \mathbf{x}$ iff $(v, a, \mathbf{x}) \in F$.

In our presentation of the algorithm we focused on its main features, therefore the guessing step (line 11) is completely "blind". It is possible to use techniques known from constraint satisfaction problems, such as shaving (removing guessing possibilities that directly lead to inconsistency). Finally, we remark that the algorithm can be extended to enumerate all possible realizations of a given interpretation set – by keeping all choice points in the guessing step and thus exhaustively exploring the whole search space.

In the case where the constructed relation F becomes functional at some point, the algorithm returns a realizing knowledge base $kb^{\mathcal{F}}_{\sigma}(F)$. For ADFs, this just means that we denote by f the σ-characterization represented by F and set $kb^{\text{ADF}}_{\sigma}(F) = D_f$. For the remaining formalisms we will introduce the respective constructions in later subsections.

Algorithm 1 $realize(\mathcal{F}, \sigma, V, F)$

Input:
- a formalism $\mathcal{F}$
- a semantics σ for $\mathcal{F}$
- a set V of interpretations $v : A \to \{\mathbf{t}, \mathbf{f}, \mathbf{u}\}$
- a relation $F \subseteq \mathcal{V}_2 \times A \times \{\mathbf{t}, \mathbf{f}\}$, initially empty

Output: a kb $\in \mathcal{F}$ with $\sigma(\mathsf{kb}) = V$ or "no" if none exists

1: **repeat**
2: **set** $F_\Delta := \bigcup_{p \in P^{\mathcal{F}}_{\sigma}} p(V, F) \setminus F$
3: **set** $F := F \cup F_\Delta$
4: **if** $\exists v \in \mathcal{V}_2, \exists a \in A : \{(v, a, \mathbf{t}), (v, a, \mathbf{f})\} \subseteq F$ **then**
5: **return** "no"
6: **end if**
7: **until** $F_\Delta = \emptyset$
8: **if** $\forall v \in \mathcal{V}_2, \forall a \in A, \exists x \in \{\mathbf{t}, \mathbf{f}\} : (v, a, x) \in F$ **then**
9: **return** $kb^{\mathcal{F}}_{\sigma}(F)$
10: **end if**
11: **choose** $v \in \mathcal{V}_2, a \in A$ with $(v, a, \mathbf{t}) \notin F$, $(v, a, \mathbf{f}) \notin F$
12: **if** $realize(\mathcal{F}, \sigma, V, F \cup \{(v, a, \mathbf{t})\}) \neq$ "no" **then**
13: **return** $realize(\mathcal{F}, \sigma, V, F \cup \{(v, a, \mathbf{t})\})$
14: **else**
15: **return** $realize(\mathcal{F}, \sigma, V, F \cup \{(v, a, \mathbf{f})\})$
16: **end if**

The algorithm is parametric in two dimensions, namely with respect to the formalism $\mathcal{F}$ and with respect to the semantics σ. These two aspects come into the algorithm via so-called *propagators*. A propagator is a formalism-specific or semantics-specific set of derivation rules. Given a set V of desired interpretations and a partial σ-characterization F, a propagator p derives new triples $(v, a, \mathbf{x})$ that must necessarily be part of any total σ-characterization f for V such that f extends F. In what follows, we present semantics propagators for admissible, complete and two-valued model (in (SET)AF terms stable) semantics, and formalism propagators for BADFs, AFs, and SETAFs.

3.2 Semantics Propagators

The semantics propagators are defined in Figure 1. They are directly derived from the properties of σ-characterizations presented in Definitions 1 to 3. While the definitions provide exact conditions to check whether a given function is a σ-characterization, the propagators allow us to derive definite values of partial characterizations that are necessary to fulfill the conditions for being a σ-characterization.

For admissible semantics, the condition for a function f to be an adm-characterization of a desired set of interpretations V (cf. Definition 2) can be split into a condition for desired interpretations $v \in V$ and two conditions for undesired interpretations $v \notin V$. Propagator $p^{\in}_{adm}$ derives new triples by considering interpretations $v \in V$. Here, for all two-valued interpretations v_2 that extend v, the value $f(v_2)$ has to be in accordance with v on v's Boolean part, that is, the algorithm adds $(v_2, a, v(a))$ whenever $v(a) \neq \mathbf{u}$. On the other hand, $p^{\notin}_{adm}$ derives new triples for $v \notin V$ in order to ensure that there is a two-valued interpretation v_2 extending v where $f(v_2)$ differs from v on a Boolean value of v. Note that while $p^{\in}_{adm}$ immediately allows us to derive information about F for each desired interpretation $v \in V$, propagator $p^{\notin}_{adm}$ is much weaker in the sense that it only derives a triple of F if there is no other way to meet the conditions for an undesired interpretation. Special treatment is required for the interpretation $v_{\mathbf{u}}$ that maps all statements to $\mathbf{u}$ and is admissible for every

$$p^{\in}_{adm}(V,F) = \{(v_2,a,v(a)) \mid v \in V, v_2 \in [v]_2, v(a) \neq \mathbf{u}\}$$
$$p^{\notin}_{adm}(V,F) = \{(v_2,a,\neg v(a)) \mid v \in \mathcal{V} \setminus V, v_2 \in [v]_2, v(a) \neq \mathbf{u}, \forall b \in A \setminus v^{-1}(\mathbf{u}), \forall v_2' \in [v]_2 : (a,v_2) \neq (b,v_2') \rightarrow (v_2',b,v(b)) \in F\}$$
$$p^{\lightning}_{adm}(V,F) = \{(v,a,\mathbf{t}),(v,a,\mathbf{f}) \mid v \in \mathcal{V}_2, a \in A, v_{\mathbf{u}} \notin V\}$$
$$p^{\in}_{mod}(V,F) = \{(v,a,v(a)) \mid v \in V, a \in A\}$$
$$p^{\notin}_{mod}(V,F) = \{(v,a,\neg v(a)) \mid v \in \mathcal{V}_2 \setminus V, a \in A, \forall c \in A \setminus \{a\} : (v,c,v(c)) \in F\}$$
$$p^{\lightning}_{mod}(V,F) = \{(v,a,\mathbf{t}),(v,a,\mathbf{f}) \mid v \in \mathcal{V}_2, a \in A, V \not\subseteq \mathcal{V}_2\}$$

$$p^{\in,\mathbf{u}}_{com}(V,F) = \{(v_2,a,\neg\mathbf{x}) \mid v \in V, v_2 \in [v]_2, v(a) = \mathbf{u}, \mathbf{x} \in \{\mathbf{t},\mathbf{f}\}, \forall v_2' \in [v]_2 : v_2 \neq v_2' \rightarrow (v_2',a,\mathbf{x}) \in F\}$$
$$p^{\notin,\mathbf{tf}}_{com}(V,F) = \{(v_2,a,\neg v(a)) \mid v \in \mathcal{V} \setminus V, v_2 \in [v]_2, v(a) \neq \mathbf{u}, \forall b \in A \setminus v^{-1}(\mathbf{u}), \forall v_2' \in [v]_2 : (a,v_2) \neq (b,v_2') \rightarrow (v_2',b,v(b)) \in F, \forall b \in v^{-1}(\mathbf{u}), \exists v_2'', v_2''' \in [v]_2 : (v_2'',b,\mathbf{t}),(v_2''',b,\mathbf{f}) \in F\}$$
$$p^{\notin,\mathbf{u}}_{com}(V,F) = \{(v_2,a,\neg\mathbf{x}) \mid v \in \mathcal{V} \setminus V, v_2 \in [v]_2, v(a) = \mathbf{u}, \forall b \in A \setminus v^{-1}(\mathbf{u}), \forall v_2' \in [v]_2 : (v_2,b,v(b)) \in F, \forall b \in v^{-1}(\mathbf{u}) \setminus \{a\} : \exists v_2'', v_2''' \in [v]_2 : (v_2'',b,\mathbf{t}), (v_2''',b,\mathbf{f}) \in F, \forall v_2'''' \in [v]_2 \setminus \{v_2\} : (v_2'''',b,\mathbf{x}) \in F\}$$

Figure 1: *Semantics propagators for the complete ($P^{ADF}_{com} = \{p^{\in,\mathbf{tf}}_{com}, p^{\in,\mathbf{u}}_{com}, p^{\notin,\mathbf{tf}}_{com}, p^{\notin,\mathbf{u}}_{com}\}$ with $p^{\in,\mathbf{tf}}_{com}(V,F) = p^{\in}_{adm}(V,F)$), admissible ($P^{ADF}_{adm} = \{p^{\in}_{adm}, p^{\notin}_{adm}, p^{\lightning}_{adm}\}$), and model semantics ($P^{ADF}_{mod} = \{p^{\in}_{mod}, p^{\notin}_{mod}, p^{\lightning}_{mod}\}$).*

ADF. This is not captured by $p^{\in}_{adm}$ and $p^{\notin}_{adm}$ as these deal only with interpretations that have Boolean mappings. Thus, propagator $p^{\lightning}_{adm}$ serves to check whether $v_{\mathbf{u}} \in V$. If this is not the case, the propagator immediately makes the relation F incoherent and the algorithm correctly answers "no".

For complete semantics and interpretations $v \in V$, propagator $p^{\in,\mathbf{tf}}_{com}$ derives triples just like in the admissible case. Propagator $p^{\in,\mathbf{u}}_{com}$ deals with statements $a \in A$ having $v(a) = \mathbf{u}$ for which there have to be at least two $v_2, v_2' \in [v]_2$ having $f(v_2)(a) = \mathbf{t}$ and $f(v_2')(a) = \mathbf{f}$. Hence $p^{\in,\mathbf{u}}_{com}$ derives triple $(v_2,a,\neg\mathbf{x})$ if for all other $v_2' \in [v]_2$ we find a triple $(v_2',a,\mathbf{x})$. For interpretations $v \notin V$ it must hold that there is some $a \in A$ such that (i) $v(a) \neq \mathbf{u}$ and $f(v_2)(a) \neq v(a)$ for some $v_2 \in [v]_2$ or (ii) $v(a) = \mathbf{u}$ but for all $v_2 \in [v]_2$, $f(v_2)$ assigns the same Boolean truth value $\mathbf{x}$ to a. Now if neither (i) nor (ii) can be fulfilled by any statement $b \in A \setminus \{a\}$ due to the current contents of F, propagators $p^{\notin,\mathbf{tf}}_{com}$ and $p^{\notin,\mathbf{u}}_{com}$ derive triple $(v_2,a,\neg v(a))$ for $v(a) \neq \mathbf{u}$ if needed for a to fulfill (i) and $(v_2,a,\neg\mathbf{x})$ for $v(a) = \mathbf{u}$ if needed for a to fulfill (ii), respectively.

Example 3. Consider the set $V_3 = \{\mathbf{uuu}, \mathbf{fuu}, \mathbf{uuf}, \mathbf{ftf}\}$. First, we consider a run of $realize(\text{ADF}, adm, V_3, \emptyset)$. In the first iteration, propagator $p^{\in}_{adm}$ ensures that F_Δ in line 2 contains $(\mathbf{fff},a,\mathbf{f})$, $(\mathbf{ftf},a,\mathbf{f})$, $(\mathbf{ftf},c,\mathbf{f})$, and $(\mathbf{fff},c,\mathbf{f})$. Based on the latter three tuples and $\mathbf{fuf} \notin V_3$, propagator $p^{\notin}_{adm}$ derives $(\mathbf{fff},a,\mathbf{t})$ in the second iteration which together with $(\mathbf{fff},a,\mathbf{f})$ causes the algorithm to return "no". Consequently, V_3 is not adm-realizable. A run of $realize(\text{ADF}, com, V_3, \emptyset)$ on the other hand returns com-characterization f for V_3 that maps $\mathbf{ttf}$ to $\mathbf{tff}$, $\mathbf{ftt}$ to $\mathbf{fft}$, $\mathbf{ftf}$ and $\mathbf{fff}$ to $\mathbf{ftf}$ and all other $v_2 \in \mathcal{V}_2$ to $\mathbf{fff}$. Hence, ADF D_f, given by the acceptance conditions $\varphi^f_a = a \wedge b \wedge \neg c$, $\varphi^f_b = (\neg a \wedge b \wedge \neg c) \vee (\neg a \wedge \neg b \wedge \neg c)$, and $\varphi^f_c = \neg a \wedge b \wedge c$, has V_3 as its complete semantics. ■

Finally, for two-valued model semantics, propagator $p^{\in}_{mod}$ derives new triples by looking at interpretations $v \in V$. For those, we must find $f(v) = v$ in each mod-characterization f by definition. Thus the algorithm adds $(v,a,v(a))$ for each $a \in A$ to the partial characterization F. Propagator $p^{\notin}_{mod}$ looks at interpretations $v \in \mathcal{V}_2 \setminus V$, for which it must hold that $f(v) \neq v$. Thus there must be a statement $a \in A$ with $v(a) \neq f(v)(a)$, which is exactly what this propagator derives whenever it is clear that there is only one statement candidate left. This, in turn, is the case whenever all $b \in A$ with the opposite truth value $\neg v(a)$ and all $c \in A$ with $c \neq a$ cannot coherently become the necessary witness any more. The propagator $p^{\lightning}_{mod}$ checks whether $V \subseteq \mathcal{V}_2$, that is, the desired set of interpretations consists entirely of two-valued interpretations. In that case this propagator makes the relation F incoherent, following a similar strategy as $p^{\lightning}_{adm}$.

Algorithm 2 $realizePrf(\mathcal{F}, V)$

Input: • a formalism $\mathcal{F}$
• a set V of interpretations $v : A \rightarrow \{\mathbf{t},\mathbf{f},\mathbf{u}\}$
Output: Return some $\mathsf{kb} \in \mathcal{F}$ with $prf(\mathsf{kb}) = V$ if one exists or "no" otherwise.

1: **if** $\max_{\leq_i} V \neq V$ **then**
2: **return** "no"
3: **end if**
4: **set** $V^{<_i} := \{v \in \mathcal{V} \mid \exists v' \in V : v <_i v'\}$
5: **set** $X := \emptyset$
6: **repeat**
7: **choose** $V' \subseteq V^{<_i}$ with $V' \notin X$
8: **set** $X := X \cup \{V'\}$
9: **set** $V^{adm} := V \cup V'$
10: **if** $realize(\mathcal{F}, adm, V^{adm}, \emptyset) \neq$ "no" **then**
11: **return** $realize(\mathcal{F}, adm, V^{adm}, \emptyset)$
12: **end if**
13: **until** $\forall V' \subseteq V^{<_i} : V' \in X$
14: **return** "no"

The Special Case of Preferred Semantics Realizing a given set of interpretations V under preferred semantics requires special treatment. We do not have a σ-characterization function for $\sigma = prf$ at hand to directly check realizability of V but have to find some $V' \subseteq \{v \in \mathcal{V} \mid \exists v' \in V : v <_i v'\}$ such that $V \cup V'$ is realizable under admissible semantics (cf. Corollary 6). Algorithm 2 implements this idea by guessing such a V' (line 7) and then using Algorithm 1 to try to realize $V \cup V'$ under admissible semantics (line 11). If $realize$ returns a knowledge base kb realizing $V \cup V'$ under adm we can directly use kb as solution of $realizePrf$ since it holds that $prf(\mathsf{kb}) = V$, given that V is an $\leq_i$-antichain (line 2).

3.3 Formalism Propagators

When constructing an ADF realizing a given set V of interpretations under a semantics σ, the function $kb^{\text{ADF}}_\sigma(F)$ makes use of the σ-characterization given by F in the following way: v is a model of the acceptance condition φ_a if and only if we find $(v,a,\mathbf{t}) \in F$. Now as bipolar ADFs, SETAFs and AFs are all subclasses of ADFs by restricting the acceptance conditions of statements, these restrictions also carry over to the σ-characterizations. The propagators defined

$$p^{\text{SETAF}}(V,F) = \{(v_{\mathbf{f}}, a, \mathbf{t}) \mid a \in A\} \cup \{(w, a, \mathbf{t}) \mid (v, a, \mathbf{t}) \in F, w \in \mathcal{V}_2, w <_t v\} \cup \{(w, a, \mathbf{f}) \mid (v, a, \mathbf{f}) \in F, w \in \mathcal{V}_2, v <_t w\}$$

$$p^{\text{AF}}(V,F) = p^{\text{SETAF}}(V,F) \cup \{(v_1 \sqcup_t v_2, a, \mathbf{t}) \mid (v_1, a, \mathbf{t}) \in F, (v_2, a, \mathbf{t}) \in F\} \qquad L^+ = \left\{(b,a) \;\middle|\; (v, a, \mathbf{f}) \in F, v(b) = \mathbf{f}, (v|^b_{\mathbf{t}}, a, \mathbf{t}) \in F\right\}$$

$$p^{\text{BADF}}(V,F) = \{(v|^b_{\mathbf{t}}, a, \mathbf{x}) \mid (v, a, \mathbf{x}) \in F, (w, a, \neg\mathbf{x}) \in F, w(b) = \mathbf{f}, (w|^b_{\mathbf{t}}, a, \mathbf{x}) \in F\} \qquad L^- = \left\{(b,a) \;\middle|\; (v, a, \mathbf{t}) \in F, v(b) = \mathbf{f}, (v|^b_{\mathbf{t}}, a, \mathbf{f}) \in F\right\}$$

Figure 2: *Formalism propagators. For formalism $\mathcal{F} \in \{AF, SETAF, BADF\}$ and any $\sigma \in \{adm, com, prf, mod\}$, we set the respective propagator for $\mathcal{F}$ to $P^{\mathcal{F}}_\sigma = P^{ADF}_\sigma \cup \{p^{\mathcal{F}}\}$ with $p^{\mathcal{F}}$ as defined above. L^+ and L^- define link polarities for kb^{BADF}_σ.*

in Figure 2 use structural knowledge on the form of acceptance conditions of the respective formalisms to reduce the search space or to induce incoherence of F whenever V is not realizable.

Bipolar ADFs For bipolar ADFs, we use the fact that each of their links must have at least one polarity, that is, must be supporting or attacking. Therefore, if a link is not supporting, it must be attacking, and vice versa. For canonical realization, we obtain the polarities of links, that is, the sets L^+ and L^-, as defined in Figure 2.

AFs To explain the AF propagators, we first need some more definitions. On the two classical truth values, we define the truth ordering $\mathbf{f} <_t \mathbf{t}$, whence the operations $\sqcup_t$ and $\sqcap_t$ with $\mathbf{f} \sqcup_t \mathbf{t} = \mathbf{t}$ and $\mathbf{f} \sqcap_t \mathbf{t} = \mathbf{f}$ result. These operations can be lifted pointwise to two-valued interpretations as usual, i.e., $(v_1 \sqcup_t v_2)(a) = v_1(a) \sqcup_t v_2(a)$ and $(v_1 \sqcap_t v_2)(a) = v_1(a) \sqcap_t v_2(a)$. Again, the reflexive version of $<_t$ is denoted by $\leq_t$. The pair $(\mathcal{V}_2, \leq_t)$ of two-valued interpretations ordered by the truth ordering forms a complete lattice with glb $\sqcap_t$ and lub $\sqcup_t$. This complete lattice has the least element $v_{\mathbf{f}} : A \to \{\mathbf{f}\}$, the interpretation mapping all statements to false, and the greatest element $v_{\mathbf{t}} : A \to \{\mathbf{t}\}$ mapping all statements to true, respectively.

Acceptance conditions of AF-based ADFs have the form of conjunctions of negative literals. In the complete lattice $(\mathcal{V}_2, \leq_t)$, the model sets of AF acceptance conditions correspond to the lattice-theoretic concept of an *ideal*, a subset of $\mathcal{V}_2$ that is downward-closed with respect to $\leq_t$ and upward-closed with respect to $\sqcup_t$. The propagator directly implements these closure properties: application of p^{AF} ensures that when a σ-characterization F that is neither incoherent nor partial is found in line 8 of Algorithm 1, then there is, for each $a \in A$, an interpretation v_a such that $(v_a, a, \mathbf{t}) \in F$ and $v \leq_t v_a$ for each $(v, a, \mathbf{t}) \in F$. Hence v_a is crucial for the acceptance condition, or in AF terms the attacks, of a and we can define $kb^{\text{AF}}_\sigma(F) = (A, \{(b, a) \mid a, b \in A, v_a(b) = \mathbf{f}\})$.

SETAFs The propagator for SETAFs, p^{SETAF}, is a weaker version of that of AFs, since we cannot presume upward-closure with respect to $\sqcup_t$. In SETAF-based ADFs the acceptance formula is in *conjunctive normal form* containing only negative literals. By a transformation preserving logical equivalence we obtain an acceptance condition in *disjunctive normal form*, again with only negative literals; in other words, a *disjunction* of AF acceptance formulas. Thus, the model set of a SETAF acceptance condition is not necessarily an ideal, but a union of ideals. For the canonical realization we can make use of the fact that, for each $a \in A$, the set $V^{\mathbf{t}}_a = \{v \in \mathcal{V}_2 \mid (v, a, \mathbf{t}) \in F\}$ is downward-closed with respect to $\leq_t$, hence the set of models of $\bigvee_{v \in \max_{\leq_t} V^{\mathbf{t}}} \bigwedge_{v(b)=\mathbf{f}} \neg b$ is exactly $V^{\mathbf{t}}_a$. The clauses of its corresponding CNF-formula exactly coincide with the sets of arguments attacking a in $kb^{\text{SETAF}}_\sigma(F)$.

3.4 Correctness

For a lack of space, we could not include a formal proof of soundness and completeness of Algorithm 1, but rather present arguments for termination and correctness.

Termination With each recursive call, the set F can never decrease in size, as the only changes to F are adding the results of propagation in line 3 and adding the guesses in line 11. Also within the until-loop, the set F can never decrease in size; furthermore there is only an overall finite number of triples that can be added to F. Thus at some point we must have $F_\Delta = \emptyset$ and leave the until-loop. Since F always increases in size, at some point it must either become functional or incoherent, whence the algorithm terminates.

Soundness If the algorithm returns $kb^{\mathcal{F}}_\sigma(F)$ as a realizing knowledge base, then according to the condition in line 8 the relation F induced a total function $f : \mathcal{V}_2 \to \mathcal{V}_2$. In particular, because the until-loop must have been run through at least once, there was at least one propagation step (line 2). Since the propagators are defined such that they enforce everything that must hold in a σ-characterization, we conclude that the induced function f indeed is a σ-characterization for V. By construction, we consequently find that $\sigma(kb^{\mathcal{F}}_\sigma(F)) = V$.

Completeness If the algorithm answers "no", then the execution reached line 5. Thus, for the constructed set F, there must have been an interpretation $v \in \mathcal{V}_2$ and a statement $a \in A$ such that $\{(v, a, \mathbf{t}), (v, a, \mathbf{f})\} \subseteq F$, that is, F is incoherent. Since F is initially empty, the only way it could get incoherent is in the propagation step in line 2. (The guessing step cannot create incoherence, since exactly one truth value is guessed for v and a.) However, the propagators are defined such that they infer only assignments (triples) that are necessary for the given F. Consequently, the given interpretation set V is such that either there is no realization within the ADF fragment corresponding to formalism $\mathcal{F}$ (that is, the formalism propagator derived the incoherence) or there is no σ-characterization for V with respect to general ADFs (that is, the semantics propagator derived the incoherence). In any case, V is not σ-realizable for $\mathcal{F}$.

4 Implementation

As Algorithm 1 is based on propagation, guessing, and checking it is perfectly suited for an implementation using answer set programming (ASP) [24, 21] as this allows for exploiting conflict learning strategies and heuristics of modern ASP solvers. Thus, we developed ASP encodings in the `gringo` language [17] for our approach. Similar as the algorithm, our declarative encodings are modular, consisting of a main part responsible for constructing set F and separate encodings for the individual propagators. If one wants, e.g., to compute an AF realization under admissible semantics for a set V of interpretations, an input program encoding V is joined with the main encoding, the propagator encoding for admissible semantics as well as the propagator encoding for AFs. Every answer set of such a program encodes a respective characterization function. Our ASP encoding for preferred semantics is based on the admissible encoding and guesses further interpretations following the essential idea of Algorithm 2. For constructing a knowledge base with the desired semantics, we also provide two ASP encodings that transform the output to an ADF in the syntax of

the DIAMOND tool [14], respectively an AF in ASPARTIX syntax [13, 15]. Both argumentation tools are based on ASP themselves. The encodings for all the semantics and formalisms we covered in the paper can be downloaded from http://www.dbai.tuwien.ac.at/research/project/adf/unreal/.

5 Expressiveness Results

In this section we briefly present some results that we have obtained using our implementation. We first introduce some necessary notation to describe the relative expressiveness of knowledge representation formalisms [18, 28]. For formalisms $\mathcal{F}_1$ and $\mathcal{F}_2$ with semantics σ_1 and σ_2, we say that $\mathcal{F}_2$ under σ_2 is at least as expressive as $\mathcal{F}_1$ under σ_1 and write $\mathcal{F}_1^{\sigma_1} \leq_e \mathcal{F}_2^{\sigma_2}$ if and only if $\Sigma_{\mathcal{F}_1}^{\sigma_1} \subseteq \Sigma_{\mathcal{F}_2}^{\sigma_2}$, where $\Sigma_{\mathcal{F}}^{\sigma} = \{\sigma(\mathsf{kb}) \mid \mathsf{kb} \in \mathcal{F}\}$ is the *signature of $\mathcal{F}$ under σ*. As usual, we define $\mathcal{F}_1 <_e \mathcal{F}_2$ if and only if $F_1 \leq_e \mathcal{F}_2$ and $F_2 \not\leq_e \mathcal{F}_1$.

We now start by considering the signatures of AFs, SETAFs and (B)ADFs for the unary vocabulary $\{a\}$:

$$\begin{aligned}
\Sigma_{\text{AF}}^{adm} &= \Sigma_{\text{SETAF}}^{adm} = \{\{\mathbf{u}\}, \{\mathbf{u}, \mathbf{t}\}\} \\
\Sigma_{\text{AF}}^{com} &= \Sigma_{\text{SETAF}}^{com} = \{\{\mathbf{u}\}, \{\mathbf{t}\}\} \\
\Sigma_{\text{AF}}^{prf} &= \Sigma_{\text{SETAF}}^{prf} = \{\{\mathbf{u}\}, \{\mathbf{t}\}\} \\
\Sigma_{\text{AF}}^{mod} &= \Sigma_{\text{SETAF}}^{mod} = \{\emptyset, \{\mathbf{t}\}\} \\
\Sigma_{\text{ADF}}^{adm} &= \Sigma_{\text{BADF}}^{adm} = \Sigma_{\text{AF}}^{adm} \cup \{\{\mathbf{u}, \mathbf{f}\}, \{\mathbf{u}, \mathbf{t}, \mathbf{f}\}\} \\
\Sigma_{\text{ADF}}^{com} &= \Sigma_{\text{BADF}}^{com} = \Sigma_{\text{AF}}^{com} \cup \{\{\mathbf{f}\}, \{\mathbf{u}, \mathbf{t}, \mathbf{f}\}\} \\
\Sigma_{\text{ADF}}^{prf} &= \Sigma_{\text{BADF}}^{prf} = \Sigma_{\text{AF}}^{prf} \cup \{\{\mathbf{f}\}, \{\mathbf{t}, \mathbf{f}\}\} \\
\Sigma_{\text{ADF}}^{mod} &= \Sigma_{\text{BADF}}^{mod} = \Sigma_{\text{AF}}^{mod} \cup \{\{\mathbf{f}\}, \{\mathbf{t}, \mathbf{f}\}\}
\end{aligned}$$

The following result shows that the expressiveness of the formalisms under consideration is in line with the amount of restrictions they impose on acceptance formulas.

Theorem 8. *For any $\sigma \in \{adm, com, prf, mod\}$:*

1. *AF$^\sigma$ $<_e$ SETAF$^\sigma$.*
2. *SETAF$^\sigma$ $<_e$ BADF$^\sigma$.*
3. *BADF$^\sigma$ $<_e$ ADF$^\sigma$.*

Proof. (1) AF$^\sigma$ $\leq_e$ SETAF$^\sigma$ is clear (by modeling individual attacks via singletons). For SETAF$^\sigma$ $\not\leq_e$ AF$^\sigma$ the witnessing interpretation sets over vocabulary $A = \{a, b, c\}$ are $\{\mathbf{uuu}, \mathbf{ttf}, \mathbf{tft}, \mathbf{ftt}\} \in \Sigma_{\text{SETAF}}^{\sigma} \setminus \Sigma_{\text{AF}}^{\sigma}$ and $\{\mathbf{ttf}, \mathbf{tft}, \mathbf{ftt}\} \in \Sigma_{\text{SETAF}}^{\tau} \setminus \Sigma_{\text{AF}}^{\tau}$ with $\sigma \in \{adm, com\}$ and $\tau \in \{prf, mod\}$. By each pair of arguments of A being $\mathbf{t}$ in at least one model, a realizing AF cannot feature any attack, immediately giving rise to the model $\mathbf{ttt}$. The respective realizing SETAF is given by the attack relation $X = \{(\{a, b\}, c), (\{a, c\}, b), (\{b, c\}, a)\}$.

(2) It is clear that SETAF$^\sigma$ $\leq_e$ BADF$^\sigma$ holds (SETAFs are bipolar since all parents are always attacking). For BADF$^\sigma$ $\not\leq_e$ SETAF$^\sigma$ the respective counterexamples can be read off the signatures above: for $\sigma \in \{adm, com\}$ we find $\{\mathbf{u}, \mathbf{t}, \mathbf{f}\} \in \Sigma_{\text{BADF}}^{\sigma} \setminus \Sigma_{\text{SETAF}}^{\sigma}$ and for $\tau \in \{prf, mod\}$ we find $\{\mathbf{t}, \mathbf{f}\} \in \Sigma_{\text{BADF}}^{\tau} \setminus \Sigma_{\text{SETAF}}^{\tau}$. The realizing bipolar ADF has acceptance condition $\varphi_a = a$.

(3) For $\sigma = mod$ the result is known [28, Theorem 14]; for the remaining semantics the model sets witnessing ADF$^\sigma$ $\not\leq_e$ BADF$^\sigma$ over vocabulary $A = \{a, b\}$ are

$$\begin{aligned}
\{\mathbf{uu}, \mathbf{tu}, \mathbf{tt}, \mathbf{tf}, \mathbf{fu}\} &\in \Sigma_{\text{ADF}}^{adm} \setminus \Sigma_{\text{BADF}}^{adm} \\
\{\mathbf{uu}, \mathbf{tu}, \mathbf{tt}, \mathbf{tf}, \mathbf{fu}\} &\in \Sigma_{\text{ADF}}^{com} \setminus \Sigma_{\text{BADF}}^{com} \\
\{\mathbf{tt}, \mathbf{tf}, \mathbf{fu}\} &\in \Sigma_{\text{ADF}}^{prf} \setminus \Sigma_{\text{BADF}}^{prf}
\end{aligned}$$

A witnessing ADF is given by $\varphi_a = a$ and $\varphi_b = a \leftrightarrow b$. □

Theorem 8 is concerned with the relative expressiveness of the formalisms under consideration, given a certain semantics. Considering different semantics we find that for all formalisms the signatures become incomparable:

Proposition 9. *$\mathcal{F}_1^{\sigma_1} \not\leq_e \mathcal{F}_2^{\sigma_2}$ and $\mathcal{F}_2^{\sigma_2} \not\leq_e \mathcal{F}_1^{\sigma_1}$ for all formalisms $\mathcal{F}_1, \mathcal{F}_2 \in \{AF, SETAF, BADF, ADF\}$ and all semantics $\sigma_1, \sigma_2 \in \{adm, com, prf, mod\}$ with $\sigma_1 \neq \sigma_2$.*

Proof. First, the result for adm and com follows by $\{\mathbf{u}, \mathbf{t}\} \in \Sigma_{\text{AF}}^{adm}$, but $\{\mathbf{u}, \mathbf{t}\} \notin \Sigma_{\text{ADF}}^{com}$ and $\{\mathbf{t}\} \in \Sigma_{\text{AF}}^{com}$, but $\{\mathbf{t}\} \notin \Sigma_{\text{ADF}}^{adm}$. Moreover, taking into account that the set of preferred interpretations (resp. two-valued models) always forms a $\leq_i$-antichain while the set of admissible (resp. complete) interpretations never does, the result follows for $\sigma_1 \in \{adm, com\}$ and $\sigma_2 \in \{prf, mod\}$. Finally, since a $\mathsf{kb} \in \mathcal{F}$ may not have any two-valued models and a preferred interpretation is not necessarily two-valued, the result for prf and mod follows. □

Disregarding the possibility of realizing the empty set of interpretations under the two-valued model semantics, we obtain the following relation for ADFs.

Proposition 10. $(\Sigma_{ADF}^{mod} \setminus \{\emptyset\}) \subseteq \Sigma_{ADF}^{prf}$.

Proof. Consider some $V \in \Sigma_{\text{ADF}}^{mod}$ with $V \neq \emptyset$. Clearly $V \subseteq \mathcal{V}_2$ and by Proposition 4 there is a mod-characterization $f : \mathcal{V}_2 \to \mathcal{V}_2$ for V, that is, $f(v) = v$ iff $v \in V$. Define $f' : \mathcal{V}_2 \to \mathcal{V}_2$ such that $f'(v) = f(v) = v$ for all $v \in V$ and $f'(v)(a) = \neg v(a)$ for all $v \in \mathcal{V} \setminus V$ and $a \in A$. Now it holds that f' is an adm-characterization of $V' = \{v \in \mathcal{V} \mid \forall v_2 \in [v]_2 : v_2 \in V\} \cup \{v_{\mathbf{u}}\}$. Since $\max_{\leq_i} V' = V$ we get that the ADF D with acceptance formula $\varphi_a^{f'}$ for each $a \in A$ has $prf(D) = V$ whence $V \in \Sigma_{\text{ADF}}^{prf}$. □

In contrast, this relation does not hold for AFs, which was shown for extension-based semantics by Linsbichler et al. [20, Theorem 5] and immediately follows for the three-valued case.

6 Discussion

We presented a framework for realizability in which AFs, SETAFs, BADFs and general ADFs can be treated in a uniform way. The centerpiece of our approach is an algorithm for deciding realizability of a given interpretation-set in a formalism under a semantics. The algorithm makes use of so-called propagators, by which it can be adapted to the different formalisms and semantics. We also presented an implementation of our framework in answer set programming and several novel expressiveness results that we obtained using our implementation. In unpublished related work, our colleague Sylwia Polberg studied a wide range of abstract argumentation formalisms, in particular their relationship with ADFs [25]. This can be the basis for including further formalisms into our realizability framework: all that remains to do is figuring out suitable ADF fragments and developing propagators for them, just like we did exemplarily for Nielsen and Parsons' SETAFs. For further future work, several semantics whose realizability is yet unstudied could be added to our framework, for example semantics based on conflict-freeness, like three-valued versions of conflict-free, naive, and stage semantics [27, 16, 29].

Acknowledgements This research was supported by the German Research Foundation (DFG) under project BR 1817/7-1 and the Austrian Science Fund (FWF) under projects I1102, I2854 and P25518.

References

[1] Leila Amgoud and Claudette Cayrol, 'A reasoning model based on the production of acceptable arguments', *Annals of Mathematics and Artificial Intelligence*, **34**(1–3), 197–215, (2002).

[2] Pietro Baroni, Federico Cerutti, Massimiliano Giacomin, and Giovanni Guida, 'AFRA: Argumentation framework with recursive attacks', *International Journal of Approximate Reasoning*, **52**(1), 19–37, (2011).

[3] Ringo Baumann, Wolfgang Dvořák, Thomas Linsbichler, Hannes Strass, and Stefan Woltran, 'Compact argumentation frameworks', in *Proceedings of the 21st European Conference on Artificial Intelligence (ECAI 2014)*, eds., Torsten Schaub, Gerhard Friedrich, and Barry O'Sullivan, volume 263 of *Frontiers in Artificial Intelligence and Applications*, pp. 69–74. IOS Press, (2014).

[4] Gerhard Brewka, Stefan Ellmauthaler, Hannes Strass, Johannes P. Wallner, and Stefan Woltran, 'Abstract Dialectical Frameworks Revisited', in *Proceedings of the 23rd International Joint Conference on Artificial Intelligence (IJCAI 2013)*, ed., Francesca Rossi, pp. 803–809. AAAI Press / IJCAI, (2013).

[5] Gerhard Brewka, Sylwia Polberg, and Stefan Woltran, 'Generalizations of Dung frameworks and their role in formal argumentation', *IEEE Intelligent Systems*, **29**(1), 30–38, (2014). Special Issue on Representation and Reasoning.

[6] Gerhard Brewka and Stefan Woltran, 'Abstract Dialectical Frameworks', in *Proceedings of the 12th International Conference on Principles of Knowledge Representation and Reasoning (KR 2010)*, eds., Fangzhen Lin, Ulrike Sattler, and Mirosław Truszczyński, pp. 102–111. AAAI Press, (2010).

[7] Martin Caminada and Dov Gabbay, 'A logical account of formal argumentation', *Studia Logica*, **93**(2–3), 109–145, (2009).

[8] Claudette Cayrol and Marie-Christine Lagasquie-Schiex, 'On the acceptability of arguments in bipolar argumentation frameworks', in *Proceedings of the 8th European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty (ECSQARU 2005)*, ed., Lluis Godo, volume 3571 of *Lecture Notes in Computer Science*, pp. 378–389. Springer, (2005).

[9] Phan M. Dung, 'On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games', *Artificial Intelligence*, **77**(2), 321–357, (1995).

[10] Paul E. Dunne, Wolfgang Dvořák, Thomas Linsbichler, and Stefan Woltran, 'Characteristics of multiple viewpoints in abstract argumentation', *Artificial Intelligence*, **228**, 153–178, (2015).

[11] Paul E. Dunne, Wolfgang Dvořák, Thomas Linsbichler, and Stefan Woltran, 'Characteristics of multiple viewpoints in abstract argumentation', in *Proceedings of the Fourth Workshop on Dynamics of Knowledge and Belief (DKB 2013)*, eds., Chistoph Beierle and Gabriele Kern-Isberner, pp. 16–30, (2013).

[12] Sjur K. Dyrkolbotn, 'How to Argue for Anything: Enforcing Arbitrary Sets of Labellings using AFs', in *Proceedings of the 14th International Conference on Principles of Knowledge Representation and Reasoning (KR 2014)*, eds., Chitta Baral, Giuseppe De Giacomo, and Thomas Eiter, pp. 626–629. AAAI Press, (2014).

[13] Uwe Egly, Sarah A. Gaggl, and Stefan Woltran, 'Answer-set programming encodings for argumentation frameworks', *Argument & Computation*, **1**(2), 147–177, (2010).

[14] Stefan Ellmauthaler and Hannes Strass, 'The DIAMOND system for computing with abstract dialectical frameworks', in *Proceedings of the Fifth International Conference on Computational Models of Argument (COMMA 2014)*, eds., Simon Parsons, Nir Oren, Chris Reed, and Federico Cerutti, volume 266 of *FAIA*, pp. 233–240. IOS Press, (2014).

[15] Sarah A. Gaggl, Norbert Manthey, Alessandro Ronca, Johannes P. Wallner, and Stefan Woltran, 'Improved answer-set programming encodings for abstract argumentation', *Theory and Practice of Logic Programming*, **15**(4–5), 434–448, (2015).

[16] Sarah A. Gaggl and Hannes Strass, 'Decomposing Abstract Dialectical Frameworks', in *Proceedings of the Fifth International Conference on Computational Models of Argument (COMMA 2014)*, eds., Simon Parsons, Nir Oren, Chris Reed, and Federico Cerutti, volume 266 of *FAIA*, pp. 281–292. IOS Press, (2014).

[17] Martin Gebser, Roland Kaminski, Benjamin Kaufmann, and Torsten Schaub, *Answer Set Solving in Practice*, Morgan and Claypool Publishers, 2012.

[18] Goran Gogic, Henry Kautz, Christos Papadimitriou, and Bart Selman, 'The comparative linguistics of knowledge representation', in *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI 1995)*, pp. 862–869. Morgan Kaufmann, (1995).

[19] Carla P. Gomes, Henry A. Kautz, Ashish Sabharwal, and Bart Selman, 'Satisfiability Solvers', in *Handbook of Knowledge Representation*, eds., Frank van Harmelen, Vladimir Lifschitz, and Bruce W. Porter, volume 3 of *Foundations of Artificial Intelligence*, 89–134, Elsevier, (2008).

[20] Thomas Linsbichler, Christof Spanring, and Stefan Woltran, 'The hidden power of abstract argumentation semantics', in *Theory and Applications of Formal Argumentation – 3rd International Workshop (TAFA 2015), Revised Selected Papers*, eds., Elizabeth Black, Sanjay Modgil, and Nir Oren, volume 9524 of *Lecture Notes in Computer Science*, pp. 146–162. Springer, (2015).

[21] Victor W. Marek and Mirosław Truszczyński, 'Stable models and an alternative logic programming paradigm', in *In The Logic Programming Paradigm: a 25-Year Perspective*, eds., Krzysztof R. Apt, Victor W. Marek, Mirosław Truszczyński, and David S. Warren, 375–398, Springer, (1999).

[22] Sanjay Modgil, 'Reasoning about preferences in argumentation frameworks', *Artificial Intelligence*, **173**(9–10), 901–934, (2009).

[23] Søren Holbech Nielsen and Simon Parsons, 'A generalization of Dung's abstract framework for argumentation: Arguing with sets of attacking arguments', in *Proceedings of the 3rd International Workshop on Argumentation in Multi-Agent Systems (ArgMAS 2006)*, eds., Nicolas Maudet, Simon Parsons, and Iyad Rahwan, volume 4766 of *Lecture Notes in Computer Science*, pp. 54–73. Springer, (2006).

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

[25] Sylwia Polberg. Understanding the Abstract Dialectical Framework (Preliminary Report). Available at `http://arxiv.org/abs/1607.00819`, July 2016.

[26] Jörg Pührer, 'Realizability of Three-Valued Semantics for Abstract Dialectical Frameworks', in *Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI 2015)*, eds., Qiang Yang and Michael Wooldridge, pp. 3171–3177. AAAI Press, (2015).

[27] Hannes Strass, 'Approximating operators and semantics for abstract dialectical frameworks', *Artificial Intelligence*, **205**, 39–70, (December 2013).

[28] Hannes Strass, 'Expressiveness of Two-Valued Semantics for Abstract Dialectical Frameworks', *Journal of Artificial Intelligence Research*, **54**, 193–231, (2015).

[29] Hannes Strass and Johannes P. Wallner, 'Analyzing the Computational Complexity of Abstract Dialectical Frameworks via Approximation Fixpoint Theory', *Artificial Intelligence*, **226**, 34–74, (2015).