Abstract Dialectical Frameworks Revisited*

Gerhard Brewka and **Stefan Ellmauthaler** and **Hannes Strass**
Institute of Computer Science, Leipzig University

Johannes Peter Wallner and **Stefan Woltran**
Institute of Information Systems, Vienna University of Technology

Abstract

We present various new concepts and results related to abstract dialectical frameworks (ADFs), a powerful generalization of Dung's argumentation frameworks (AFs). In particular, we show how the existing definitions of stable and preferred semantics which are restricted to the subcase of so-called bipolar ADFs can be improved and generalized to arbitrary frameworks. Furthermore, we introduce preference handling methods for ADFs, allowing for both reasoning with and about preferences. Finally, we present an implementation based on an encoding in answer set programming.

1 Introduction

Dung's abstract argumentation frameworks (AFs) [Dung, 1995] are widely used in argumentation for handling conflicts among (abstract) arguments. One criticism often advanced against abstract argumentation frameworks is that only one form of interaction between atomic arguments is permitted: specifically that an argument *attacks* another. Brewka and Woltran [2010] have proposed a new model – Abstract Dialectical Frameworks (ADFs) – in which more general forms of argument interaction are captured. For instance, ADFs allow to express that an argument supports another one, that two arguments – none of which is strong enough individually – may jointly attack a third one, what the effects of combining attacking and supporting arguments are, and the like.

To achieve this, each argument, x, is associated with an *acceptance condition*, C_x, which is some propositional function whose truth status is determined by the corresponding status of those arguments y with (y, x) being a link in the ADF. Dung's AFs are recovered by setting as acceptance condition for each argument the function $\bigwedge_{y:(y,x)} \neg y$, i.e. x is accepted if none of its parents is. It is this concept of associating individual acceptance conditions with arguments that provides ADFs with a rich expressive capability.

The question arises what the role of ADFs in argumentation is, specifically whether they should be viewed as a knowledge representation (KR) formalism or as an abstraction tool. The abstraction view is best exemplified by the use of AFs within ASPIC [Prakken, 2010]. ASPIC's translational approach allows to use expressive KR languages. Knowledge bases in these languages generate arguments and attack relations; in other words, they are translated into an AF which, via any of the standard Dung semantics, provides the original knowledge base with an argumentation theoretic semantics.

It is natural to ask where ADFs stand in this game. First we want to emphasize that the distinction between KR language and abstraction tool is not crisp: each KR language is abstract to a certain extent and disregards aspects irrelevant for its purpose. ADFs are certainly less abstract than AFs, as the latter abstract from everything but attacks among arguments whereas the former abstract from everything but acceptance conditions, a more general notion. Still, we do not consider ADFs primarily as a KR tool. We rather see them as convenient alternatives to AFs as target languages in the translational approach. Since AFs are a special case of ADFs, translations to ADFs are obviously not more difficult than those to AFs. However, since the additional expressiveness of ADFs brings them closer to rich KR languages, many translations will in fact become easier. Borrowing terminology from software engineering, the term *argumentation middleware* appropriately characterizes what we see in ADFs.

Here are just two examples witnessing the usefulness of ADFs. For one, Brewka and Gordon [2010] have shown how the Carneades formalism [Gordon *et al.*, 2007] can be reconstructed and generalized using ADFs.[1] For another, ADFs also allow to express attacks from *sets* of arguments as proposed by Nielsen and Parsons [2006]: an attack from a set B to a is expressed by setting the acceptance function of a to $\neg(\bigwedge_{b \in B} b)$. An alternative approach to deal with such situations is meta-argumentation, see e.g. [Boella *et al.*, 2009]. Here additional (artificial) arguments are added together with certain gadgets to capture the functioning of e.g. set-attacks within AFs. While this methodology allows to stay within the simple framework of AFs, it comes with the price that the additional arguments require special treatment, in particular when the AF is further processed. ADFs circumvent the addition of artificial arguments, yet staying in the abstract domain.

*This research has been supported by DFG (projects BR 1817/7-1 and FOR 1513) and FWF (projects I1102 and P25518-N23).

[1] Van Gijzel and Prakken [2011] have shown that Carneades can also be captured within Dung's formalism. AFs obtained from their translation are cycle-free, thus always induce a unique extension.

In [Brewka and Woltran, 2010], the standard Dung semantics of grounded, preferred and stable extensions are generalized to ADFs, the latter two to a restricted type of ADFs called bipolar. As has been recently recognized [Ellmauthaler, 2012; Strass, 2013], some examples are not handled as intended in these ADF semantics (see also Example 1 in Section 3). In fact, the work by Strass [2013] proposes new variants of these semantics using operator-based definitions on lattices in the spirit of [Denecker *et al.*, 2004].

In this work, we present new concepts and results which substantially increase the range of applicability of ADFs.

1. We introduce new definitions of preferred and stable semantics which avoid the mentioned unintended results and moreover cover arbitrary ADFs, not only bipolar ones. Technically, the new definitions rest on a shift from two-valued to three-valued interpretations. They are based on an operator already defined in [Brewka and Woltran, 2010] and are conceptually simpler than the proposal in [Strass, 2013].

2. As a second contribution, we provide a complexity analysis for the proposed semantics. This not only gives a clear picture of the effects our generalizations have on computational issues, and how they extend expressibility as compared to Dung semantics. It also provides the basis for an implementation of ADFs in terms of answer set programming (ASP), an issue we will address as well.

3. Finally, to underline the modelling capacities of ADFs we also address the problem of preferential reasoning in abstract argumentation [Amgoud and Cayrol, 1998; Bench-Capon, 2003]. We introduce the concept of a prioritized ADF where certain nodes are used to represent (dynamic) preferences (an approach also followed in [Modgil, 2009]) and show that prioritized ADFs can easily be compiled to standard ADFs.

The paper proceeds as follows. After providing the necessary background regarding ADFs and three-valued interpretations in Section 2, Section 3 introduces the new definitions of preferred and stable semantics for ADFs. The complexity of the new semantics (together with some strengthening of previous results) is then given in Section 4. The new preference handling methods which allow us to capture both static and dynamic preferences are discussed in Section 5. Finally, an implementation of ADFs based on ASP is briefly presented in Section 6. A discussion of related work concludes the paper.

2 Background

An abstract dialectical framework (ADF) is a directed graph whose nodes represent statements or positions which can be accepted or not. The links represent dependencies: the status of a node s only depends on the status of its parents (denoted $par(s)$), that is, the nodes with a direct link to s. In addition, each node s has an associated acceptance condition C_s specifying the exact conditions under which s is accepted. C_s is a function assigning to each subset of $par(s)$ one of the truth values $\mathbf{t}$, $\mathbf{f}$.[2] Intuitively, if for some $R \subseteq par(s)$ we have $C_s(R) = \mathbf{t}$, then s will be accepted provided the nodes in R are accepted and those in $par(s) \setminus R$ are not accepted.

[2]In the original paper *in* and *out* were used. We prefer truth values here as they allow us to apply standard logical terminology.

Definition 1. An *abstract dialectical framework* is a tuple $D = (S, L, C)$ where

- S is a set of statements (positions, nodes),
- $L \subseteq S \times S$ is a set of links,
- $C = \{C_s\}_{s \in S}$ is a set of total functions $C_s : 2^{par(s)} \rightarrow \{\mathbf{t}, \mathbf{f}\}$, one for each statement s. C_s is called acceptance condition of s.

In many cases it is convenient to represent acceptance conditions as propositional formulas. For this reason we will frequently use a logical representation of ADFs (S, L, C) where C is a collection $\{\varphi_s\}_{s \in S}$ of propositional formulas.

Moreover, unless specified differently we will tacitly assume that the acceptance formulas specify the parents a node depends on implicitly. It is then not necessary to give the links in the graph explicitly. We thus can represent an ADF D as a tuple (S, C) where S and C are as above and L is implicitly given as $(a, b) \in L$ iff a appears in φ_b.[3]

The different semantics of ADFs over statements S are based on the notion of a model. A two-valued interpretation v – a mapping from statements to the truth values true and false – is a *two-valued model* (*model*, if clear from the context) of an ADF (S, C) whenever for all statements $s \in S$ we have $v(s) = v(\varphi_s)$, that is, v maps exactly those statements to true whose acceptance conditions are satisfied under v. Our analysis in this paper will be based on a straightforward generalization of two-valued interpretations for ADFs to Kleene's strong three-valued logic [Kleene, 1952].[4] A three-valued interpretation is a mapping $v : S \rightarrow \{\mathbf{t}, \mathbf{f}, \mathbf{u}\}$ that assigns one of the truth values true ($\mathbf{t}$), false ($\mathbf{f}$) or unknown ($\mathbf{u}$) to each statement. Interpretations can easily be extended to assign truth values to propositional formulas over the statements: negation switches $\mathbf{t}$ and $\mathbf{f}$, and leaves $\mathbf{u}$ unchanged; a conjunction is $\mathbf{t}$ if both conjuncts are $\mathbf{t}$, it is $\mathbf{f}$ if some conjunct is $\mathbf{f}$ and it is $\mathbf{u}$ otherwise; disjunction is dual. It is also straightforward to generalize the notion of a model: a three-valued interpretation is a model whenever for all statements $s \in S$ we have $v(s) \neq \mathbf{u}$ implies $v(s) = v(\varphi_s)$.

The three truth values are partially ordered by $\leq_i$ 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. The pair $(\{\mathbf{t}, \mathbf{f}, \mathbf{u}\}, \leq_i)$ forms a complete meet-semilattice[5] with the meet operation $\sqcap$. This meet can be read as *consensus* and assigns $\mathbf{t} \sqcap \mathbf{t} = \mathbf{t}$, $\mathbf{f} \sqcap \mathbf{f} = \mathbf{f}$, and returns $\mathbf{u}$ otherwise.

The information ordering $\leq_i$ extends in a straightforward way to valuations v_1, v_2 over S in that $v_1 \leq_i v_2$ iff

[3]When presenting examples we will use a notation where acceptance conditions are written in square brackets behind nodes, e.g. c $[\neg(a \wedge b)]$ denotes a node c which is jointly attacked by nodes a and b, that is, each attacker alone is insufficient to defeat c.

[4]A comparable treatment for AFs was given by the labellings of [Caminada, 2006]. We use standard notation and terminology from mathematical logic.

[5]A complete meet-semilattice is such that every non-empty finite subset has a greatest lower bound, the meet; and every nonempty directed subset has a least upper bound. A subset is directed if any two of its elements have an upper bound in the set.

$v_1(s) \leq_i v_2(s)$ for all $s \in S$. The set of all three-valued interpretations over S forms a complete meet-semilattice with respect to the information ordering $\leq_i$. The consensus meet operation $\sqcap$ of this semilattice is given by $(v_1 \sqcap v_2)(s) = v_1(s) \sqcap v_2(s)$ for all $s \in S$. The least element of this semilattice is the valuation mapping all statements to unknown – the least informative interpretation.

Obviously, a three-valued interpretation v is two-valued if all statements are mapped to either true or false. The two-valued interpretations are the $\leq_i$-maximal elements of the meet-semilattice. For a three-valued interpretation v, we say that a two-valued interpretation w *extends* v iff $v \leq_i w$. This means that all statements mapped to $\mathbf{u}$ by v are mapped to $\mathbf{t}$ or $\mathbf{f}$ by w. 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 $v = \bigsqcap [v]_2$.

Before introducing the new semantics we recall the grounded semantics as defined in [Brewka and Woltran, 2010].[6] This semantics – which we are not going to change – is actually based on an operator Γ_D over three-valued interpretations. Brewka and Woltran's definition is equivalent to the following one: For an ADF D and a three-valued interpretation v, the interpretation $\Gamma_D(v)$ is given by

$$s \mapsto \bigsqcap \{w(\varphi_s) \mid w \in [v]_2\}$$

That is, for each statement s, the operator returns the consensus truth value for its acceptance formula φ_s, 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 $[v]_2 = \{v\}$, thus $\Gamma_D(v)(s) = v(\varphi_s)$ and v is a two-valued model for D iff $\Gamma_D(v) = v$.

The grounded model of an ADF D is now defined as the least fixpoint of Γ_D. This fixpoint is in general three-valued; it always exists since the operator Γ_D is $\leq_i$-monotone, as shown in [Brewka and Woltran, 2010]. Thus the grounded semantics can be seen as the greatest possible consensus between all acceptable ways of interpreting the ADF at hand.

3 Semantics

As we will see the operator Γ_D already provides the right basis for defining admissible, complete and preferred semantics for arbitrary ADFs.

Recall that for an admissible AF interpretation v we have the following for all $s \in S$: (1) if v maps s to true, then all its attackers must be mapped to false; (2) if v maps s to false, then some attacker of s must be mapped to true. This can be generalised to ADFs in the following way: a three-valued interpretation is admissible iff it does not make an unjustified commitment that the operator Γ_D will subsequently revoke.

Definition 2. A three-valued interpretation v for an ADF D is *admissible* iff $v \leq_i \Gamma_D(v)$.

That is, admissible interpretations may contain "at most as much" as is imperative by the consensus over all acceptance functions. Note that admissible interpretations (as well as the special cases complete and preferred interpretations to be defined now) are actually three-valued models. For this reason we will also call them admissible (complete, preferred) models frequently.

For complete models we stipulate that a model assigns to a statement *exactly* those truth values that equal their consensus over all two-valued interpretations that are at least as informative.

Definition 3. A three-valued interpretation v for an ADF D is *complete* iff $\Gamma_D(v) = v$.

It immediately follows from this definition that the grounded semantics is always a complete model, and each complete model is admissible. With our generalization of the admissible semantics at hand, we next define preferred models.

Definition 4. A three-valued interpretation v for D is *preferred* iff it is $\leq_i$-maximal admissible.

As it is the case for AFs, for ADFs we have that all preferred models are complete. Moreover, the set of all complete models forms a complete meet-semilattice with the information ordering $\leq_i$ and we can prove the following result, which is a generalization of Theorem 25 in [Dung, 1995].

Theorem 1. *Let D be an* ADF.

1. *Each preferred model is a complete model, but not vice versa.*
2. *The grounded model is the $\leq_i$-least complete model.*
3. *The complete models of an* ADF *form a complete meet-semilattice with respect to $\leq_i$.*

Proof. 1. If v is preferred, then $v \leq_i \Gamma_D(v)$. We have to show that $\Gamma_D(v) = v$. Assume to the contrary that $\Gamma_D(v) \not\leq_i v$, then $v <_i \Gamma_D(v)$. Since Γ_D is $\leq_i$-monotone, we get $\Gamma_D(v) \leq_i \Gamma_D(\Gamma_D(v))$, and $\Gamma_D(v)$ is admissible in contradiction to v being $\leq_i$-maximal admissible. Thus $\Gamma_D(v) \leq_i v$ and v is complete.

As a counterexample in the opposite direction, consider the ADF in Example 2. It has two complete models – its grounded model and the single two-valued model. Only the latter is $\leq_i$-maximal.

2. The grounded model is the $\leq_i$-least fixpoint of Γ_D and thus the $\leq_i$-least complete model.
3. Let S be the set of statements in D and define F as the set of all fixpoints of Γ_D. It is clear that the grounded model of D is the least element of F. Now let $E \subseteq F$ be finite and non-empty. We have to show that E has a greatest lower bound in F. Let e be the greatest lower bound of E in S. The set $\{v : S \to \{\mathbf{t}, \mathbf{f}, \mathbf{u}\} \mid v \leq_i e\}$ forms a complete lattice in which Γ_D possesses a greatest fixpoint which is the greatest lower bound of E in F. Now let E additionally be directed. We have to show that it has a least upper bound in F. Let e' be the least upper bound of E in S. The set $\{v : S \to \{\mathbf{t}, \mathbf{f}, \mathbf{u}\} \mid e' \leq_i v\}$ forms a complete meet-semilattice where Γ_D possesses a least fixpoint which is the least upper bound of E in F. □

We can also show that our definitions are indeed proper generalizations of Dung's notions for AFs.

[6] The semantics was called well-founded there. The term grounded is more in line with standard terminology in argumentation.

Definition 5. For an AF $F = (A, R)$, define the ADF *associated to* F as $D_F = (A, R, C)$ with $C = \{\varphi_a\}_{a \in A}$ and $\varphi_a = \bigwedge_{b:(b,a) \in R} \neg b$ for $a \in A$. For an interpretation v, the set $E_v = \{s \in S \mid v(s) = \mathbf{t}\}$ defines the unique *extension* associated with v.

Theorem 2. *Let F be an* AF *and D_F its associated* ADF. *An extension is admissible, complete, preferred, grounded for F iff it is admissible, complete, preferred, grounded for D_F.*

Next, we show that the well-known relationships between Dung semantics carry over to our generalizations. We also include the concept two-valued models. As we will see below in Example 2, two-valued models are not suitable to capture stable models in general, but they coincide for ADFs associated to an AF (see the forthcoming Theorem 4).

Theorem 3. *Let D be an* ADF. *The following inclusions hold:*

$$val_2(D) \subseteq pref(D) \subseteq com(D) \subseteq adm(D),$$

where $val_2(D)$, $pref(D)$, $com(D)$ and $adm(D)$ denote the sets of two-valued models, preferred models, complete models and admissible interpretations of D, respectively.

Now we generalize (and also correct) the definition of stable models to arbitrary ADFs.

Definition 6. Let $D = (S, L, C)$ be an ADF with $C = \{\varphi_s\}_{s \in S}$. A two-valued model v of D is a *stable model* of D iff E_v equals the grounded extension of the reduced ADF $D^v = (E_v, L^v, C^v)$, where $L^v = L \cap (E_v \times E_v)$ and for $s \in E_v$ we set $\varphi_s^v = \varphi_s[b/\mathbf{f} : v(b) = \mathbf{f}]$.

In the reduct, in each acceptance formula we replace statements $b \in S$ that v maps to false by their truth value. The rest of the definition straightforwardly expresses the intuition underlying stable models: if all statements the model v takes to be false are indeed false, we must find a constructive proof for all statements the model takes to be true.

Example 1. Consider the ADF D given by

$$a\ [\mathbf{t}], \quad b\ [\neg a \vee c], \quad c\ [b].$$

In words, a is always accepted, a attacks b, and the links between b and c are support links. According to the original definition in [Brewka and Woltran, 2010], $\{a, b, c\}$ is the single stable model, violating the basic intuition that all elements of a stable model should have a non-cyclic justification: here b is accepted because c is and vice versa.

It is easy to see that according to our new definition, $M_1 = \{a, b, c\}$ is not stable. The reduced ADF is identical to the original one, and its grounded semantics leaves b and c undefined. On the other hand, $M_2 = \{a\}$ is stable, as intended: the reduced ADF consists of $a\ [\mathbf{t}]$ only, and its grounded semantics evaluates a to true.

Example 2. Consider the ADF D given by

$$a\ [c], \quad b\ [c], \quad c\ [a \leftrightarrow b].$$

The only two-valued model is $v : S \to \{\mathbf{t}\}$. Since c is true because a and b are and vice versa, the model contains unintended cyclic support and thus should not be stable. Indeed, for the reduct we get $D^v = D$. Let us compute the grounded semantics of D. We start with interpretation $w : S \to \{\mathbf{u}\}$. Since none of the acceptance formulas is a tautology, w is already a fixpoint and thus the grounded model of D. Hence v is not a stable model and D has no stable models, just as intended. Since v is a minimal model of D the example illustrates that in Definition 6 we actually need the grounded semantics; requiring E_v to agree with some minimal extension of the reduct is insufficient.

We can also show that our stable models are a proper generalization of Dung's stable extensions.

Theorem 4. *Let $F = (A, R)$ be an* AF *and D_F its associated* ADF. *For any interpretation v for A, the following are equivalent:*

(A) E_v is a stable extension of F,

(B) v is a stable model of D_F,

(C) v is a two-valued model of D_F.

Note that for AF-based ADFs, there is no distinction between models and stable models. The intuitive explanation for this is that stable semantics breaks cyclic supports, which cannot arise in AFs because they cannot express support.

4 Complexity

For this section, let $D = (S, L, C)$ be an ADF where C is represented by a collection $\{\varphi_s\}_{s \in S}$ of propositional formulas. As our first important complexity result, we show that although expressiveness increases in the step from AFs to ADFs, the complexity of deciding existence of a two-valued model (stable extension in AF terminology) stays the same.

Proposition 5. *Deciding whether $D = (S, L, C)$ has a two-valued model is* NP*-complete.*

Proof. For membership, we can guess an interpretation $v : S \to \{\mathbf{t}, \mathbf{f}\}$ and in polynomial time check whether $v(s) = v(\varphi_s)$ for each $s \in S$.

For hardness, let ψ be a propositional formula over a vocabulary P. We construct an ADF D_ψ that has a model iff ψ is satisfiable. Set $S = P \cup \{s\}$, $L = S \times S$ and for the acceptance formulas set $\varphi_p = p$ for each $p \in P$ and $\varphi_s = \neg s \wedge \neg \psi$. We have to show that D_ψ has a model iff ψ is satisfiable. If ψ is satisfiable, there exists a satisfying valuation v for P. Then $v(\neg\psi) = \mathbf{f}$ and $v(s) = \mathbf{f}$ and v is a model for D_ψ. Now let ψ be unsatisfiable and assume that D_ψ has a model v. Obviously $v(\neg\psi) = \mathbf{t}$ and thus $v(s) = v(\varphi_s) = v(\neg s \wedge \neg\psi) = v(\neg s)$, contradiction. □

Note that deciding whether D has a three-valued model is trivial, since the grounded model always exists. However, verifying that a given interpretation is the grounded model is not trivial. The proof of the next result is quite laborious, but both membership and hardness parts work by reducing to and from the DP-complete problem of checking whether a propositional formula ϕ is satisfiable and a formula ψ is valid.

Theorem 6. *Verifying that a three-valued interpretation v is the grounded model of an* ADF *D is* DP*-complete.*

Based on this result and its proof we can show the same complexity bounds for arbitrary fixpoints of Γ_D, that is, for the complete models of an abstract dialectical framework.

Corollary 7. *Verifying that a three-valued interpretation v is complete in an* ADF D *is* DP*-complete.*

Checking that a two-valued interpretation is a stable model for D now essentially boils down to computing the grounded semantics of the reduct.

Proposition 8. *Verifying that a two-valued interpretation v is a stable model for D is in* DP.

Proof sketch. Verifying that v is a model of D and constructing the reduct D^v can be done in polynomial time; checking that E_v is the grounded extension is in DP by Theorem 6. □

This immediately leads to the containment part of the next result. For its hardness part, we can adapt the proof of Theorem 6.12 in [Denecker *et al.*, 2004].

Theorem 9. *Deciding whether D has a stable model is Σ_2^{P}-complete.*

We can also show that the additional expressiveness of ADFs in comparison to AFs is computationally significant in the case of admissible semantics.

Proposition 10. *Verifying that a three-valued interpretation v is admissible in an* ADF D *is* coNP*-complete.*

Proof. For membership consider the co-problem, i.e. verify that v is not admissible in D. Guess a statement $s \in S$ such that $v(s) \in \{\mathbf{t}, \mathbf{f}\}$ and a two-valued interpretation v' that extends v. Check that v is not admissible, i.e. $v'(\varphi_s) \neq v(s)$.

For hardness we provide a reduction from the problem if a given propositional formula ϕ over vocabulary P is valid. Construct an ADF D with statements $P \cup \{a\}$, where $a \notin P$ and $\varphi_s = s$ if $s \in P$ and $\varphi_a = \phi$. Further construct a three-valued interpretation v with $v(s) = \mathbf{u}$ for $s \in P$ and $v(a) = \mathbf{t}$. We show that v is admissible iff ϕ is valid. The set of two-valued extensions v' of v, if restricted to P, equals the set of possible two-valued interpretations of ϕ. Hence if ϕ is valid, v will be admissible, since then all two-valued interpretations are models of ϕ and likewise for all extensions v' of v we have $v'(\varphi_a) = \mathbf{t}$. Similarly if ϕ is not valid then the consensus of the extensions of v will evaluate a to $\mathbf{u}$ or $\mathbf{f}$, since there is an interpretation that falsifies ϕ and hence an extension v' of v with $v'(\varphi_a) = \mathbf{f}$, and v is not admissible. □

5 Preferences in ADFs

Since ADFs completely specify – via their acceptance conditions – in what situations a node is to be accepted, there is strictly speaking no room for adding preferences. However, preferences provide a convenient alternative to defining acceptance conditions for each node individually, in particular in the restricted case where each link is either attacking or supporting.[7] [Brewka and Woltran, 2010] already contained a short discussion of preferences defined over the links in an ADF. Here we are interested in preferences over the nodes. This is more in line with existing approaches to handle preferences and values [Amgoud and Cayrol, 1998; Bench-Capon, 2003] in Dung frameworks. In fact, what we are aiming for is an approach that generalizes the one from Amgoud and Cayrol.

Definition 7. A prioritized ADF (PADF) is a tuple $A = (S, L^+, L^-, >)$ where S is the set of nodes, L^+ and L^- are subsets of $S \times S$, the supporting and attacking links, and $>$ is a strict partial order (irreflexive, transitive, antisymmetric) on S representing preferences among the nodes.

Here $(a, b) \in >$ (alternatively: $a > b$) states that a is preferred to b. We define the semantics of prioritized ADFs via a translation to standard ADFs: A translates to $trans(A) = (S, L^+ \cup L^-, C)$, where for each node $n \in S$ the acceptance condition C_n is defined as: $C_n(M) = \mathbf{t}$ iff for each $a \in M$ such that $(a, n) \in L^-$ and not $n > a$ we have: for some $b \in M$, $(b, n) \in L^+$ and $b > a$. Intuitively, an attacker does not succeed if the attacked node is more preferred or if there is a more preferred supporting node.

Example 3. Assume the parents of statement g are a, b, c, d, (a, g) is a supporting link, (b, g), (c, g) and (d, g) are attacking. Moreover, let $g > d$ and $a > c$. The acceptance condition for g is obtained as a conjunction of implications, one for each attacker which is not strictly less preferred than g. The left side of the implication consists of the attacker, the right side is the disjunction of those supporting nodes which are more preferred than the attacker. In the example we obtain $\varphi_g = (b \rightarrow \mathbf{f}) \wedge (c \rightarrow a)$ or, equivalently $\neg b \wedge (c \rightarrow a)$.

This handles the special case of prioritized AFs exactly like in [Amgoud and Cayrol, 1998].

Proposition 11. *Let $F = (S, Att, >)$ be a* PAF *in the sense of [Amgoud and Cayrol, 1998]. E is a stable (preferred, grounded) extension of F iff E is a stable (preferred, grounded) extension of the* PADF $A = (S, \emptyset, Att, >)$.

Often preferences, rather than being given in advance, are a matter of debate themselves, and whether node a is preferred over node b may dynamically depend on what else is accepted. We now show how this can be modeled in ADFs. To do so we have to deviate somewhat from the abstract view underlying ADFs (and also Dung AFs) that the nodes themselves are atomic entities whose meaning is not further analyzed. Here we will assume that some of the nodes represent preference information.[8]

In a nutshell, we handle dynamic preferences as follows. We first guess a (stable, preferred, grounded) extension M. Some nodes in M will carry preference information. We extract this information and check whether M can be reconstructed under the preference information, thus verifying that the preferences represented in the model itself were taken into account adequately.

Definition 8. An abstract dialectical framework with dynamic preferences (DADF) is a tuple $A = (S, L^+, L^-, P)$ where S is the set of nodes, L^+ and L^- are subsets of $S \times S$, the supporting and attacking links, and $P : S \rightarrow S \times S$ is a partial function.

[7] Similarly [Brewka and Woltran, 2010] showed how ADFs can be specified by assigning weights to links and by using these weights together with proof standards for generating the actual acceptance conditions.

[8] This is similar in spirit to the dynamic treatment of preferences via *arguments attacking links* in [Modgil, 2009].

The function P assigns preference information to some of the nodes in S. If $P(a) = (b, c)$ then node a carries the information that b is preferred over c. For a set of nodes $M \subseteq S$ we use $>_M$ to denote the smallest strict partial order on S containing the set $\{(b, c) \mid P(a) = (b, c) \text{ for some } a \in M\}$. Note that $>_M$ may be undefined, e.g. if M contains two nodes with conflicting preference information. The semantics of DADFs is now defined as follows:

Definition 9. Let $D = (S, L^+, L^-, P)$ be a DADF. E is a (stable, preferred, grounded) extension of D iff $>_E$ is defined and E is a (stable, preferred, grounded) extension of the PADF $D_E = (S, L^+, L^-, >_E)$.

This has some similarity with the treatment of dynamic preferences in default logic [Brewka, 1994], where it is checked whether a default logic extension E can be constructed in a way compatible with the preference information contained in E. For ADFs corresponding to Dung AFs this provides an alternative to Modgil's more demanding extended AFs [Modgil, 2009].

Example 4. (after [Brewka, 1994]). Assume we have two conflicting arguments s_1, s_2 about whether a contract is perfected. Argument s_3 says s_1 is to be preferred as it is based on federal law. Argument s_4 to the contrary states s_2 should be preferred as it is more recent. The judge decides (s_5) to support s_3. It is known (s_6) that the judge's support has preference over s_4. We obtain the DADF D with $S = \{s_1, \ldots, s_6\}$, $L^+ = \{(s_5, s_3)\}$, $L^- = \{(s_1, s_2), (s_2, s_1), (s_3, s_4), (s_4, s_3)\}$, $P(s_3) = s_1 > s_2$, $P(s_4) = s_2 > s_1$, and $P(s_6) = s_5 > s_4$.

For $M = \{s_1, s_3, s_5, s_6\}$, we get $s_1 >_M s_2$ and $s_5 >_M s_4$. We must check whether M is obtained as an extension of the PADF $D_M = (S, L^+, L^-, >_M)$. To do so we translate D_M to the ADF:

$$s_1\ [\mathbf{t}], \quad s_2\ [\neg s_1], \quad s_3\ [s_4 \rightarrow s_5],$$
$$s_4\ [\neg s_3], \quad s_5\ [\mathbf{t}], \quad s_6\ [\mathbf{t}]$$

M is the grounded, preferred and stable extension of D_M and thus a corresponding extension of D. D does not have any other extensions.

Static and dynamic preferences can be combined easily. One needs both a preference ordering $>$ and a function P. The ordering $>_E$ used to verify that E can be reconstructed accordingly now is simply the smallest strict partial order containing both the preferences in E and those in $>$.

6 ASP-based implementation

We have implemented a system called DIAMOND (DIAlectical MOdels eNcoDing)[9] to compute different ADF models as introduced in this paper. The implementation is based on the answer set programming (ASP) paradigm and utilizes the Potassco collection [Gebser *et al.*, 2007]. In [Ellmauthaler and Wallner, 2012] an ASP-based software system for the semantical notions in [Brewka and Woltran, 2010] was presented. This system is restricted to bipolar ADFs and suffers from the deficiencies of the original ADF semantics.

More detailed comparisons of our system and the other one will be presented elsewhere. However, preliminary tests showed that DIAMOND is significantly faster when the semantics agree, albeit the systems are hard to compare as our system is purely written for clasp, while the other one utilizes claspD together with optimization techniques.

DIAMOND computes two-valued, stable, grounded, complete, admissible and preferred models. To represent the problem in an adequate way, we guess possible models and check whether their properties are satisfied. For the stable, grounded, and complete models the operator Γ_D has been implemented and an iteration technique is used to find the fixpoints. Similar techniques were used in an ASP tool for Dung's AFs [Egly *et al.*, 2010].

7 Related work and conclusions

Denecker *et al.* [2004] describe a general method for deriving approximations of operators associated with knowledge representation formalisms. As it turns out, the operator Γ_D defined by [Brewka and Woltran, 2010] is the most precise (*ultimate*) approximation of the notion of a two-valued model for an ADF.[10] Denecker *et al.* [2004] proceed to study several ultimate semantics for logic programming; for example, our grounded semantics would be called the ultimate Kripke-Kleene semantics in their approach. They also define ultimate stable models for logic programs – these are similar to our stable models but in effect require the construction of two reducts with different fixpoint computations. It may well be that ADFs constitute a formalism where ultimate approximation arises naturally. A further study is left for future work.

Strass [2013] has the objective of locating abstract dialectical frameworks within the realm of nonmonotonic formalisms, such as propositional logic programs and Dung AFs. In the course of that work, he also discusses several operator-based semantics for ADFs. However, the operator Strass uses is manually defined and less precise than the original operator from [Brewka and Woltran, 2010] which we use in this work. Recall further that we view ADFs as abstraction tools rather than KR languages and our objective is to increase their range of applicability in this regard. That said, it remains an important objective of future work to study the relationship between the operators of Brewka and Woltran [2010] and Strass [2013] and the respective semantics defined through them.

The shift from a two-valued to a three-valued perspective in argumentation can be attributed to Caminada [2006] who introduced three-valued *labellings* (assignments of in, out or $undec$ to each argument) as an alternative to Dung's extension based definitions. Our ADF generalizations of AF extension based semantics also extend to labellings: it is clear that labellings and three-valued interpretations are interchangeable.

In his equational approach Gabbay [2012] goes even further, admitting arbitrary values between 0 and 1 in the context of Dung-style argumentation frameworks. It is a topic of further research whether a similar generalization to continuous values is beneficial in the context of ADFs as well.

[9] The system is available for download and experimentation at `http://www.informatik.uni-leipzig.de/~ellmau/diamond`.

[10] According to personal communication this was conjectured by Mirosław Truszczyński. It was later proven in [Strass, 2013].

References

[Amgoud and Cayrol, 1998] Leila Amgoud and Claudette Cayrol. On the acceptability of arguments in preference-based argumentation. In *Proc. UAI'98*, pages 1–7. Morgan Kaufmann, 1998.

[Bench-Capon, 2003] Trevor J. M. Bench-Capon. Persuasion in practical argument using value-based argumentation frameworks. *J. Log. Comput.*, 13(3):429–448, 2003.

[Boella *et al.*, 2009] Guido Boella, Dov M. Gabbay, Leendert van der Torre, and Serena Villata. Meta-argumentation modelling I: Methodology and techniques. *Studia Logica*, 93(2-3):297–355, 2009.

[Brewka and Gordon, 2010] Gerhard Brewka and Thomas F. Gordon. Carneades and abstract dialectical frameworks: A reconstruction. In *Proc. COMMA'10*, pages 3–12. IOS Press, 2010.

[Brewka and Woltran, 2010] Gerhard Brewka and Stefan Woltran. Abstract dialectical frameworks. In *Proc. KR'10*, pages 102–111. AAAI Press, 2010.

[Brewka, 1994] Gerhard Brewka. Reasoning about priorities in default logic. In *Proc. AAAI'94*, pages 940–945, 1994.

[Caminada, 2006] Martin W. A. Caminada. On the issue of reinstatement in argumentation. In *Proc. JELIA'06*, volume 4160 of *Lecture Notes in Computer Science*, pages 111–123. Springer, 2006.

[Denecker *et al.*, 2004] Marc Denecker, Victor W. Marek, and Mirosław Truszczyński. Ultimate approximation and its application in nonmonotonic knowledge representation systems. *Information and Computation*, 192(1):84–121, 2004.

[Dung, 1995] Phan Minh Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artif. Intell.*, 77(2):321–358, 1995.

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

[Ellmauthaler and Wallner, 2012] Stefan Ellmauthaler and Johannes P. Wallner. Evaluating abstract dialectical frameworks with ASP. In *Proc. COMMA'12*, volume 245, pages 505–506. IOS Press, 2012.

[Ellmauthaler, 2012] Stefan Ellmauthaler. Abstract dialectical frameworks: Properties, complexity, and implementation. Master's thesis, TU Vienna, 2012.

[Gabbay, 2012] Dov M. Gabbay. Equational approach to argumentation networks. *Argument and Computation*, 3(2-3):87–142, 2012.

[Gebser *et al.*, 2007] Martin Gebser, Benjamin Kaufmann, André Neumann, and Torsten Schaub. Conflict-driven answer set solving. In *Proc. IJCAI*, pages 386–392. AAAI Press/The MIT Press, 2007.

[Gordon *et al.*, 2007] Thomas F. Gordon, Henry Prakken, and Douglas Walton. The Carneades model of argument and burden of proof. *Artif. Intell.*, 171(10-15):875–896, 2007.

[Kleene, 1952] Stephen C. Kleene. *Introduction to Metamathematics*. D. Van Nostrand Company, Princeton, New Jersey, 1952.

[Modgil, 2009] Sanjay Modgil. Reasoning about preferences in argumentation frameworks. *Artif. Intell.*, 173(9-10):901–934, 2009.

[Nielsen and Parsons, 2006] Søren H. Nielsen and Simon Parsons. A generalization of Dung's abstract framework for argumentation: Arguing with sets of attacking arguments. In *Argumentation in Multi-Agent Systems*, volume 4766 of *LNCS*, pages 54–73. Springer, 2006.

[Prakken, 2010] Henry Prakken. An abstract framework for argumentation with structured arguments. *Argument and Computation*, 1:93–124, 2010.

[Strass, 2013] Hannes Strass. Approximating operators and semantics for abstract dialectical frameworks. Technical Report 1, Institute of Computer Science, Leipzig University, January 2013.

[Van Gijzel and Prakken, 2011] Bas Van Gijzel and Henry Prakken. Relating Carneades with abstract argumentation. In *Proc. IJCAI'11*, pages 1113–1119. AAAI Press, 2011.