TPLP **20** (5): 703–718, 2020.
doi:10.1017/S1471068420000253

On the Semantics of Abstract Argumentation Frameworks: A Logic Programming Approach

Gianvincenzo Alfano, Sergio Greco, Francesco Parisi, and Irina Trubitsyna
DIMES Department, University of Calabria, Rende, Italy
(*e-mail:* {g.alfano,greco,fparisi,i.trubitsyna}@dimes.unical.it)

Abstract

Recently there has been an increasing interest in frameworks extending Dung's abstract Argumentation Framework (AF). Popular extensions include bipolar AFs and AFs with recursive attacks and necessary supports. Although the relationships between AF semantics and Partial Stable Models (PSMs) of logic programs has been deeply investigated, this is not the case for more general frameworks extending AF.

In this paper we explore the relationships between AF-based frameworks and PSMs. We show that every AF-based framework Δ can be translated into a logic program P_Δ so that the extensions prescribed by different semantics of Δ coincide with subsets of the PSMs of P_Δ. We provide a logic programming approach that characterizes, in an elegant and uniform way, the semantics of several AF-based frameworks. This result allows also to define the semantics for new AF-based frameworks, such as AFs with recursive attacks and recursive deductive supports.

KEYWORDS: abstract argumentation, argumentation semantics, partial stable models

1 Introduction

Formal argumentation has emerged as one of the important fields in Artificial Intelligence (Bench-Capon and Dunne 2007; Simari and Rahwan 2009). In particular, Dung's abstract Argumentation Framework (AF) is a simple, yet powerful formalism for modelling disputes between two or more agents (Dung 1995). An AF consists of a set of *arguments* and a binary *attack* relation over the set of arguments that specifies the *interactions* between arguments: intuitively, if argument a attacks argument b, then b is acceptable only if a is not. Hence, arguments are abstract entities whose role is entirely determined by the interactions specified by the attack relation.

Dung's framework has been extended in many different ways, including the introduction of new kinds of interactions between arguments and/or attacks. In particular, the class of Bipolar Argumentation Frameworks (BAFs) is an interesting extension of the AF which allows for also modelling the *support* between arguments (Nouioua and Risch 2011; Villata et al. 2012). Further extensions consider second-order interactions (Villata et al. 2012), e.g., attacks to attacks/supports, as well as more general forms of interactions such as recursive AFs where attacks can be recursively attacked (Baroni et al. 2011; Cayrol et al. 2017) and recursive BAFs, where attacks/supports can be recursively attacked/supported (Gottifredi et al. 2018; Cayrol et al. 2018). An overview of the extensions of the Dung's framework is provided at the end of this section.

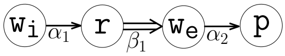


Fig. 1: BAF of Example 1

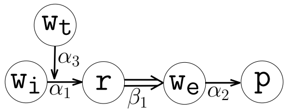


Fig. 2: Recursive BAF of Example 1

Example 1
Consider a scenario for deciding whether to play tennis. Assume we have the following arguments: $\mathtt{w_i}$ (it is windy), $\mathtt{r}$ (it is raining), $\mathtt{w_e}$ (the court is wet), $\mathtt{p}$ (play tennis), and the logical implications: (α_1) if it is windy, then it does not rain, (α_2) if the court is wet, then we do not play tennis, and (β_1) if it is raining, then the court is wet. This situation can be modelled using the BAF shown in Figure 1, where the implications α_1 and α_2 are *attacks* (denoted by $\rightarrow$), and the implication β_1 is a *support* (denoted by $\Rightarrow$).

Now assume that there also exists an argument $\mathtt{w_t}$ (we are in the winter season) that attacks the implication α_1 (in the winter season, implication α_1 cannot be applied). The new scenario can be modeled by the recursive BAF shown in Figure 2 where the new attack is named as α_3. □

Several interpretations of the notion of support have been proposed (Cayrol and Lagasquie-Schiex 2013; Cohen et al. 2014). Intuitively, the way the support is interpreted changes the set of *extensions* (i.e., the set of acceptable elements) of an argumentation framework. For instance, the (unique complete) extension of the BAF shown in Figure 1 is the set $\{\mathtt{w_i}, \mathtt{p}\}$ under the so-called *necessary* interpretation of support, while it is $\{\mathtt{w_i}, \mathtt{w_e}\}$ under the *deductive* interpretation.

Following Dung's approach, the meaning of recursive AF-based frameworks is still given by relying on the concept of extension. However, the extensions of an *AF with Recursive Attacks (AFRA)* (Baroni et al. 2011) and of an *Attack-Support Argumentation Framework (ASAF)* (Cohen et al. 2015; Gottifredi et al. 2018) also include the (names of) attacks and supports that intuitively contribute to determine the set of accepted arguments. Particularly, the acceptability of an attack is related to the acceptability of its source argument: an attack in the AFRA is defeated even when its source argument is defeated. This is not the case for *Recursive AF (RAF)* (Cayrol et al. 2017) and *Recursive AF with Necessities (RAFN)* frameworks (Cayrol et al. 2018), which offer a different semantics for recursive AFs and recursive BAFs with necessary supports, respectively.

Recently there has been an increasing interest in studying the relationships between argumentation frameworks and logic programming (LP). In particular, the semantic equivalence between complete extensions in AF and 3-valued stable models in LP was first established in (Wu et al. 2009). Then, the relationships of LP with AF have been further studied in (Caminada et al. 2015), whereas those with Assumption-Based Argumentation (Bondarenko et al. 1997; Craven and Toni 2016) have been considered in (Caminada and Schulz 2017), and those with Abstract Dialectical Frameworks have been investigated in (Alcântara et al. 2019). Efficient mappings from AF to *Answer Set Programming* (i.e. LP with *Stable Model* semantics (Gelfond and Lifschitz 1988)) have been investigated as well (Sakama and Rienstra 2017; Gaggl et al. 2015). The well-know AF system ASPARTIX is implemented by rewriting the input AF into an ASP program and using an ASP solver to compute extensions. Although the ASPARTIX system allows also to reason on some extensions of AF, such as *Extended AF (EAF)* (Modgil 2009) and AFRA, so far the relationships between LP and frameworks extending AF has not been adequately studied. Thus, in this paper, we investigate these relationships by generalizing the work in (Caminada et al. 2015) and providing relationships between LP and different recently proposed generalizations

https://doi.org/10.1017/S1471068420000253 Published online by Cambridge University Press

of the Dung's framework. As discussed in Section 5, our work is complementary to approaches providing the semantics for an AF-based framework by flattening it into a Dung's framework.

Contributions. The main contributions are as follows:

- We introduce a general approach for characterizing the extensions of different AF-based frameworks under several well-known semantics in terms of Partial Stable Models (PSMs) of logic programs. This is achieved by providing a modular definition of the sets of *defeated* and *acceptable* elements (i.e., arguments, attacks and supports) for each AF-based framework, and by leveraging on the connection between argumentation semantics and subsets of PSMs.
- Our approach is used to define new semantics for AFs with recursive attacks and supports under deductive interpretation of supports, where the status of an attack is considered independently from the status of its source.

Our results can be used $i)$ for better understanding the semantics of several AF-based frameworks, $ii)$ to easily define new semantics for extended frameworks, and $iii)$ to provide additional tools for computing stable semantics using answer set solvers (Gebser et al. 2018) and even other complete-based semantics using classical program rewriting (Janhunen et al. 2006) (see also (Sakama and Rienstra 2017; Gaggl et al. 2015)).

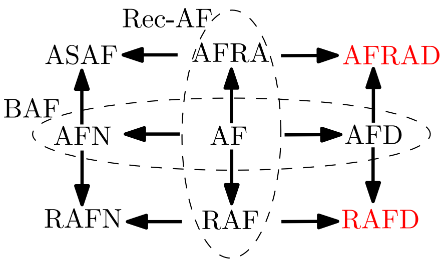
AF-based frameworks. It is important to observe that different frameworks extending AF share the same structure, although they have different semantics. Thus, in the following we distinguish between framework and *class* of frameworks. Two frameworks sharing the same syntax (i.e. the structure) belong to the same (syntactic) class. For instance, BAF is a syntactic class, whereas AFN and AFD are two specific frameworks sharing the same BAF syntax; their semantics differ because they interpret supports in different ways. Regarding the class *Recursive AF (Rec-AF)*, where AFs are extended by allowing recursive attacks, two different frameworks called AFRA and RAF, differing only in the determination of the status of attacks, have been proposed. The frameworks ASAF and RAFN are two different frameworks belonging to the same class, called *Recursive BAF* (Rec-BAF), consisting in the extension of BAF with recursive attacks and supports. The differences between ASAF and RAFN semantics are not in the way they interpret supports (both based on the necessity interpretation), but in a different determination of the status of attacks as they extend AFRA and RAF, respectively.

Figure 3 overviews the frameworks extending AF studied in this paper. Horizontal arrows denote the addition of supports with two different semantics (necessary semantics in the left direction and deductive semantics in the right direction), whereas vertical arrows denote the extension with recursive interactions (i.e., attacks and supports); the two directions denote two different semantics proposed in the literature for determining the acceptance status of attacks. Frameworks AFRAD and RAFD (in red) are novel and generalize some previously proposed frameworks. More specifically, as shown in the figure, AFRAD (resp., RAFD) generalizes AFRA and AFD (resp., RAF and AFD), as the latters are special cases of the formers, respectively. Clearly, frameworks in the corners are the most general ones. However, for the sake of presentation, before considering the most general frameworks, we also analyze the case of BAFs.

2 Preliminaries

We start by recalling abstract argumentation frameworks in increasing order of the number of features they can model. Hereafter, we will use $\mathfrak{F}$ to denote the set of the 9 frameworks shown on

https://doi.org/10.1017/S1471068420000253 Published online by Cambridge University Press



Legend of Acronyms:
AF: abstract Argumentation Framework [Dung,1995]
BAF: Bipolar AF
AFN: AF with Necessities [Nouia and Risch,2011]
AFD: AF with Deductive supports [Villata et al.,2012]
Rec-AF: Recursive-AF
AFRA: AF with Recursive Attacks [Baroni et al.,2011]
RAF: Recursive AF [Cayrol et al.,2017]
Rec-BAF: Recursive-BAF
ASAF: Attack-Support AF [Gottifredi et al.,2018]
RAFN: Recursive AF with Necessities [Cayrol et al.,2018]
AFRAD: AF with Rec. Attacks and Deductive supports
RAFD: Recursive AF with Deductive supports

Fig. 3: AF-based frameworks investigated in the paper.

left-hand side of Figure 3. Moreover, with a little abuse of notation, we will use the same symbol Δ to denote any framework in $\mathfrak{F}$.

2.1 Argumentation Frameworks

An abstract *Argumentation Framework* (AF) is a pair $\langle A, \Omega\rangle$, where A is a set of *arguments* and $\Omega \subseteq A \times A$ is a set of *attacks*. An AF can be seen as a directed graph, whose nodes represent arguments and edges represent attacks; an attack $(a, b) \in \Omega$ from a to b is represented by $a \rightarrow b$.

Different semantics notions have been defined leading to the characterization of collectively acceptable sets of arguments, called *extensions* (Dung 1995). Given an AF $\Delta = \langle A, \Omega\rangle$ and a set $\mathbf{S} \subseteq A$ of arguments, an argument $a \in A$ is said to be $i)$ *defeated* w.r.t. $\mathbf{S}$ iff $\exists b \in \mathbf{S}$ such that $(b, a) \in \Omega$, and $ii)$ *acceptable* w.r.t. $\mathbf{S}$ iff for every argument $b \in A$ with $(b, a) \in \Omega$, there is $c \in \mathbf{S}$ such that $(c, b) \in \Omega$. The sets of defeated and acceptable arguments w.r.t. $\mathbf{S}$ are defined as follows (where Δ is understood):

- $Def(\mathbf{S}) = \{a \in A \mid \exists\, b \in \mathbf{S}\,.\,(b, a) \in \Omega\}$;
- $Acc(\mathbf{S}) = \{a \in A \mid \forall\, b \in A\,.\,(b, a) \in \Omega \;\Rightarrow\; b \in Def(\mathbf{S})\}$.

Given an AF $\langle A, \Omega\rangle$, a set $\mathbf{S} \subseteq A$ of arguments is said to be $i)$ *conflict-free* iff $\mathbf{S} \cap Def(\mathbf{S}) = \emptyset$, and $ii)$ *admissible* iff it is conflict-free and $\mathbf{S} \subseteq Acc(\mathbf{S})$.

Given an AF $\langle A, \Omega\rangle$, a set $\mathbf{S} \subseteq A$ is an *extension* called:

- *complete* iff it is conflict-free and $\mathbf{S} = Acc(\mathbf{S})$;
- *preferred* iff it is a maximal (w.r.t. $\subseteq$) complete extension;
- *stable* iff it is a total preferred extension, i.e. a preferred extension s.t. $\mathbf{S} \cup Def(\mathbf{S}) = A$;
- *semi-stable* iff it is a preferred extension such that $\mathbf{S} \cup Def(\mathbf{S})$ is maximal;
- *grounded* iff it is the smallest (w.r.t. $\subseteq$) complete extension;
- *ideal* iff it is the biggest (w.r.t. $\subseteq$) complete extension contained in every preferred extension.

The set of complete (resp., preferred, stable, semi-stable, grounded, ideal) extensions of a framework Δ will be denoted by $\mathcal{CO}(\Delta)$ (resp., $\mathcal{PR}(\Delta)$, $\mathcal{ST}(\Delta)$, $\mathcal{SST}(\Delta)$, $\mathcal{GR}(\Delta)$, $\mathcal{ID}(\Delta)$).

Example 2

Let $\Delta = \langle A, \Omega\rangle$ be an AF where $A = \{\mathtt{a}, \mathtt{b}, \mathtt{c}, \mathtt{d}\}$ and $\Omega = \{(\mathtt{a}, \mathtt{b}), (\mathtt{b}, \mathtt{a}), (\mathtt{a}, \mathtt{c}), (\mathtt{b}, \mathtt{c}), (\mathtt{c}, \mathtt{d}), (\mathtt{d}, \mathtt{c})\}$. The set of complete extension is $\mathcal{CO}(\Delta) = \{\emptyset, \{\mathtt{d}\}, \{\mathtt{a}, \mathtt{d}\}, \{\mathtt{b}, \mathtt{d}\}\}$. Consequently, $\mathcal{PR}(\Delta) = \mathcal{ST}(\Delta) = \mathcal{SST}(\Delta) = \{\{\mathtt{a}, \mathtt{d}\}, \{\mathtt{b}, \mathtt{d}\}\}$, $\mathcal{GR}(\Delta) = \{\emptyset\}$, $\mathcal{ID}(\Delta) = \{\{\mathtt{d}\}\}$. □

https://doi.org/10.1017/S1471068420000253 Published online by Cambridge University Press

2.2 *Bipolar Argumentation Frameworks*

A *Bipolar Argumentation Framework* (BAF) is a triple $\langle A, \Omega, \Gamma \rangle$, where A is a set of *arguments*, $\Omega \subseteq A \times A$ is a set of *attacks*, and $\Gamma \subseteq A \times A$ is a set of *supports*. A BAF can be represented by a directed graph with two types of edges: *attacks* and *supports*, denoted by $\rightarrow$ and $\Rightarrow$, respectively. A *support path* $a_0 \stackrel{+}{\Rightarrow} a_n$ from argument a_0 to argument a_n is a sequence of n edges $a_{i-1} \Rightarrow a_i$ with $0 < i \leq n$. We use $\Gamma^+ = \{(a,b) \mid a, b \in A \ \wedge a \stackrel{+}{\Rightarrow} b\}$ to denote the set of pairs (a,b) such that there exists a support path from a to b. It is assumed that Γ is acyclic.

Different interpretations of the support relation have been proposed (Simari and Rahwan 2009; Cayrol and Lagasquie-Schiex 2013; Cohen et al. 2014). Given a BAF Δ and an interpretation $\mathcal{I}$ of the support relation, the semantics of Δ w.r.t. $\mathcal{I}$ can be given in terms of an equivalent AF $\Delta_{\mathcal{I}}$, derived from Δ by substituting supports with the so-called *complex* or *extended* attacks. In this paper we consider $\mathcal{I} \in \{d, n\}$, where d and n denote *deductive* and *necessary* interpretation of supports proposed in (Villata et al. 2012) and (Nouioua and Risch 2011), respectively.

AF with Necessities (AFN). An AFN is a BAF where supports are interpreted as necessary. The necessary interpretation of a support $a \Rightarrow b$ is that b is accepted only if a is accepted. Given an AFN $\Delta = \langle A, \Omega, \Gamma \rangle$, there exists an *extended attack* from a to b if there are either:

- an attack $a \rightarrow c$ and a support path $c \stackrel{+}{\Rightarrow} b$ (that we call *supported attack*), or
- a support path $c \stackrel{+}{\Rightarrow} a$ and an attack $c \rightarrow b$ (that we call *mediated attack*).

We denote by $\Delta_n = \langle A, \Omega_n \rangle$ the AF derived from Δ by replacing supports with extended attacks.

AF with Deductive supports (AFD). An AFD is a BAF where supports are interpreted as deductive. The deductive interpretation of a support $a \Rightarrow b$ is that b is accepted whenever a is accepted (and a is defeated whenever b is defeated). Given an AFD $\Delta = \langle A, \Omega, \Gamma \rangle$, there exists a *complex attack* from argument a to argument b if there are either:

- a support path $a \stackrel{+}{\Rightarrow} c$ and an attack $c \rightarrow b$ (*supported attack*), or
- an attack $a \rightarrow c$ and support path $b \stackrel{+}{\Rightarrow} c$ (*mediated attack*).

$\Delta_d = \langle A, \Omega_d \rangle$ denotes the AF derived from Δ by replacing supports with complex attacks.

Given a BAF $\langle A, \Omega, \Gamma \rangle$ with interpretation $\mathcal{I} \in \{n, d\}$ of supports, and a set of arguments $\mathbf{S} \subseteq A$, then $Def(\mathbf{S}) = \{a \in A \mid \exists b \in \mathbf{S}\,.\,(b,a) \in \Omega_{\mathcal{I}}\}$, and $Acc(\mathbf{S}) = \{a \in A \mid \forall b \in A\,.\,(b,a) \in \Omega_{\mathcal{I}} \Rightarrow\ b \in Def(\mathbf{S})\}$.

Example 3

Consider the BAF $\Delta = \langle A, \Omega, \Gamma \rangle$ of Figure 1. Under the necessary interpretation of supports $\Delta_n = \langle A, \Omega_n \rangle$, where $\Omega_n = \{(\mathtt{w_i}, \mathtt{r}), (\mathtt{w_i}, \mathtt{w_e}), (\mathtt{w_e}, \mathtt{p})\}$. Δ_n has a unique complete extension $\{\mathtt{w_i}, \mathtt{p}\}$. Dually, under the deductive interpretation of supports $\Delta_d = \langle A, \Omega_d \rangle$, where $\Omega_d = \{(\mathtt{w_i}, \mathtt{r}), (\mathtt{r}, \mathtt{p}), (\mathtt{w_e}, \mathtt{p})\}$. Δ_d has a unique complete extension $\{\mathtt{w_i}, \mathtt{w_e}\}$. □

2.3 *Recursive Argumentation Frameworks*

A *Recursive Argumentation Framework (Rec-AF)* is a tuple $\langle A, \Sigma, \mathbf{s}, \mathbf{t} \rangle$, where A is a set of arguments, Σ is a set disjunct from A representing attack names, $\mathbf{s}$ (resp., $\mathbf{t}$) is a function from Σ to A (resp., to $(A \cup \Sigma)$) mapping each attack to its source (resp., target). An attack may be recursive as an argument may attack an argument or an attack, and extensions may contain both arguments and attacks. Two different semantics have been proposed in literature.

Recursive AF (RAF). In (Cayrol et al. 2017) a semantic framework for Rec-AF, called *Recursive*

https://doi.org/10.1017/S1471068420000253 Published online by Cambridge University Press

Argumentation Framework, is proposed. The semantics for an RAF is given in terms of *defeated* and *acceptable* sets.

- $Def(\mathbf{S}) = \{X \in A \cup \Sigma \mid \exists\, \alpha \in \Sigma \cap \mathbf{S}\,.\, \mathtt{s}(\alpha) \in A \cap \mathbf{S} \wedge \mathbf{t}(\alpha) = X\}$;
- $Acc(\mathbf{S}) = \{X \in A \cup \Sigma \mid \forall \alpha \in \Sigma\,.\, \mathbf{t}(\alpha) = X \Rightarrow \alpha \in Def(\mathbf{S}) \vee \mathtt{s}(\alpha) \in Def(\mathbf{S})\}$.

The peculiarity of RAF semantics is that an attack is defeated only if it is explicitly attacked and, consequently, can be accepted whenever its source is defeated.

AF with Recursive Attacks (AFRA). Differently from RAF semantics, in an AFRA (Baroni et al. 2011) the status of an attack is also related to the status of its source argument.

Given $X \in A \cup \Sigma$ and $\alpha \in \Sigma$, we say that α *(directly or indirectly) attacks* X (denoted by $\alpha\ \mathtt{def}\ X$) if either $\mathbf{t}(\alpha) = X$ or $\mathbf{t}(\alpha) = \mathtt{s}(X)$. Given an AFRA $\langle A, \Sigma, \mathtt{s}, \mathbf{t}\rangle$[1] and a set $\mathbf{S} \subseteq A \cup \Sigma$ of arguments and attacks, the *defeated* and *acceptable* sets are:

- $Def(\mathbf{S}) = \{X \in A \cup \Sigma \mid \exists\, \alpha \in \Sigma \cap \mathbf{S}\,.\, \alpha\ \mathtt{def}\ X\}$;
- $Acc(\mathbf{S}) = \{X \in A \cup \Sigma \mid \forall\, \alpha \in \Sigma\,.\, \alpha\ \mathtt{def}\ X \Rightarrow \alpha \in Def(\mathbf{S})\}$.

The idea behind AFRA semantics is that whenever an argument a is defeated, every attack starting from a is (indirectly) defeated as well.

The notions of *conflict-free*, *admissible sets*, and the different types of extensions can be defined in a standard way (see Section 2.1) by considering $\mathbf{S} \subseteq A \cup \Sigma$ and by using the new definitions of defeated and acceptable sets reported above.

Example 4
Let $\Delta = \langle A, \Sigma, \mathtt{s}, \mathbf{t}\rangle$ be an Rec-AF, where $A = \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$, $\Sigma = \{\alpha_1, \alpha_2\}$, $\mathtt{s} = \{\alpha_1/\mathtt{a}, \alpha_2/\mathtt{b}\}$, $\mathbf{t} = \{\alpha_1/\mathtt{b}, \alpha_2/\mathtt{c}\}\rangle$ where $\alpha/y \in \mathtt{s}$ (resp., $\beta/y \in \mathbf{t}$) denotes that $\mathtt{s}(\alpha) = y$ (resp., $\mathbf{t}(\beta) = y$). Considering the set $\mathbf{S} = \{\mathtt{a}, \alpha_1\}$, under the AFRA (resp., RAF) semantics we have that $Def(\mathbf{S}) = \{\mathtt{b}, \alpha_2\}$ (resp., $Def(\mathbf{S}) = \{\mathtt{b}\}$), and there exists a unique complete extension $\{\mathtt{a}, \mathtt{c}, \alpha_1\}$ (resp., $\{\mathtt{a}, \mathtt{c}, \alpha_1, \alpha_2\}$). □

It has been shown that RAF and AFRA semantics may differ only in the status of attacks, and extensions under RAF semantics could be derived from extensions under AFRA semantics and vice versa (Cayrol et al. 2017).

2.4 Recursive Bipolar Argumentation Frameworks with Necessities

By combining the concepts of both bipolarity and recursive interactions, more general argumentation frameworks have been defined.

A *Recursive Bipolar Argumentation Framework (Rec-BAF)* is a tuple $\langle A, \Sigma, \Pi, \mathtt{s}, \mathbf{t}\rangle$, where A is a set of arguments, Σ is a set of attack names, Π is a set of necessary support names, $\mathtt{s}$ (resp., $\mathbf{t}$) is a function from $\Sigma \cup \Pi$ to A (resp., to $A \cup \Sigma \cup \Pi$) mapping each attack/support to its source (resp., target). In the following, given a set Φ such that either $\Phi \subseteq \Sigma$ or $\Phi \subseteq \Pi$, we denote by $i)$ $\Phi^* = \{(\mathtt{s}(\gamma), \mathbf{t}(\gamma)) \mid \gamma \in \Phi\}$ the set of pairs connected by an attack/support edge, and $ii)$ Φ^+ the transitive closure of Φ. It is assumed that Π^* is acyclic.

[1] For the sake of presentation, we consider a slight generalization of AFRA, where attack names are first-class citizens, allowing to also represent more than one attack from the same source to the same target. In the original work an AFRA is a tuple $\langle A, \Omega\rangle$ where A is a set of arguments and Ω is a set of attacks $\Omega : A \to (A \cup \Omega)$ (Baroni et al. 2011).

Two different semantics have been defined under necessary interpretation of supports.

Recursive AF with Necessities (RAFN). The *Recursive Argumentation Framework with Necessities* has been proposed in (Cayrol et al. 2018). The semantics combines the RAF interpretation of attacks with that of BAF under the necessity interpretation of supports (i.e., AFN). Here we consider a simplified version where supports have a single source and the support relation is acyclic. Formally, given an RAFN $\langle A, \Sigma, \Pi, \mathbf{s}, \mathbf{t}\rangle$, $X \in (A \cup \Sigma \cup \Pi)$, $a \in A$, and $\mathbf{S} \subseteq A \cup \Sigma \cup \Pi$, we say that argument a *recursively attacks* X *given* $\mathbf{S}$ (denoted as $a \ \mathtt{att}_{\mathbf{S}} \ X$) if either $(a, X) \in (\Sigma \cap \mathbf{S})^*$ or there exists $b \in A$ such that $(a, b) \in (\Sigma \cap \mathbf{S})^*$ and $(b, X) \in (\Pi \cap \mathbf{S})^+$.

For any RAFN Δ and $\mathbf{S} \subseteq A \cup \Sigma \cup \Pi$, the *defeated* and *acceptable* sets (given $\mathbf{S}$) are:

- $Def(\mathbf{S}) = \{X \in A \cup \Sigma \cup \Pi \mid \exists b \in A \cap \mathbf{S} \,.\, b \ \mathtt{att}_{\mathbf{S}} \ X\}$;
- $Acc(\mathbf{S}) = \{X \in A \cup \Sigma \cup \Pi \mid \forall b \in A \,.\, b \ \mathtt{att}_{\mathbf{S}} \ X \Rightarrow b \in Def(\mathbf{S})\}$.

Attack-Support AF (ASAF). The *Attack-Support Argumentation Framework (ASAF)* has been proposed in (Cohen et al. 2015; Gottifredi et al. 2018). The semantics combines the AFRA interpretation of attacks with that of BAF under the necessary interpretation of supports (i.e., AFN). For the sake of presentation, we consider a slight generalization of ASAF, where attack and support names are first-class citizens, giving the possibility to represent multiple attacks and supports from the same source to the same target.[2]

Formally, given an ASAF $\langle A, \Sigma, \Pi, \mathbf{s}, \mathbf{t}\rangle$, $X \in (A \cup \Sigma \cup \Pi)$, $\alpha \in \Sigma$, and $\mathbf{S} \subseteq A \cup \Sigma \cup \Pi$, we say that $i)$ α *(directly or indirectly) attacks* X (denoted by $\alpha \ \mathtt{def} \ X$) if either $\mathbf{t}(\alpha) = X$ or $\mathbf{t}(\alpha) = \mathbf{s}(X)$, and $ii)$ α *extendedly defeats* X *given* $\mathbf{S}$ (denoted as $\alpha \ \mathtt{def}_{\mathbf{S}} \ X$) if either $\alpha \ \mathtt{def} \ X$ or there exists $b \in A$ such that $\mathbf{t}(\alpha) = b$ and either $(b, X) \in (\Pi \cap \mathbf{S})^+$ or $(b, \mathbf{s}(X)) \in (\Pi \cap \mathbf{S})^+$. For any ASAF Δ and $\mathbf{S} \subseteq A \cup \Sigma \cup \Pi$, the *defeated* and *acceptable* sets (given $\mathbf{S}$) are:

- $Def(\mathbf{S}) = \{X \in A \cup \Sigma \cup \Pi \mid \exists\, \alpha \in \Sigma \cap \mathbf{S} \,.\, \alpha \ \mathtt{def}_{\mathbf{S}} \ X\}$;
- $Acc(\mathbf{S}) = \{X \in A \cup \Sigma \cup \Pi \mid \forall \alpha \in \Sigma \,.\, \alpha \ \mathtt{def}_{\mathbf{S}} \ X \Rightarrow \alpha \in Def(\mathbf{S})\}$.

Again, the notions of *conflict-free*, *admissible sets*, and the different types of extensions can be defined in a standard way (see Section 2.1) by considering $\mathbf{S} \subseteq A \cup \Sigma \cup \Pi$ and by using the definitions of defeated and acceptable sets reported above.

Note that for AFs with high-order interactions the mapping to AF is not trivial, as in the case of BAF, because extensions also contain attacks and supports. In particular, an equivalent AF for an ASAF can be obtained by translating it into an AFN (Cohen et al. 2015) that in turns can be translated into an AF (Nouioua and Risch 2011) (see also (Gottifredi et al. 2018)).

Example 5

Consider the Rec-BAF Δ with necessary supports of Figure 2. Under both ASAF and RAFN semantics $\mathcal{CO}(\Delta) = \{\{\mathtt{w_i}, \mathtt{r}, \mathtt{w_e}, \mathtt{w_t}, \alpha_2, \alpha_3, \beta_1\}\}$. Consider now the Rec-BAF Δ' obtained by adding to Δ an argument $\mathtt{s}$ attacking argument $\mathtt{w_t}$ with attack α_4. Under the ASAF semantics Δ' has a unique complete extension $\{\mathtt{w_i}, \mathtt{s}, \mathtt{p}, \alpha_1, \alpha_4, \beta_1\}$; note that attacks α_2 and α_3 are not part of the extension as their sources (i.e., $\mathtt{w_e}$ and $\mathtt{w_t}$, respectively) are defeated. Differently, $\{\mathtt{w_i}, \mathtt{s}, \mathtt{p}, \alpha_1, \alpha_2, \alpha_3, \alpha_4, \beta_1\}$ is the only complete extension of Δ' under the RAFN semantics. □

Analogous to the case of Rec-AFs, ASAF and RAFN semantics may differ only in the status of attacks. Moreover, for each semantics, the RAFN extensions can be derived from the corresponding ASAF extensions and vice versa.

[2] In the original work (Cohen et al. 2015; Gottifredi et al. 2018) an ASAF is a tuple $\langle A, \Omega, \Gamma\rangle$ where A is a set of arguments, Ω is a set of attacks $\Omega : A \rightarrow (A \cup \Omega)$, and Γ is a set of supports $\Gamma : A \rightarrow (A \cup \Gamma)$.

2.5 Partial Stable Models

We summarize the basic concepts which underly the notion of PSMs (Saccà and Zaniolo 1990).

A (normal, logic) program is a set of rules of the form $A \leftarrow B_1 \wedge \cdots \wedge B_n$, with $n \geq 0$, where A is an atom, called head, and $B_1 \wedge \cdots \wedge B_n$ is a conjunction of literals, called body. We consider programs without function symbols. Given a program P, $ground(P)$ denotes the set of all ground instances of the rules in P. The Herbrand Base of a program P, i.e. the set of all ground atoms which can be constructed using predicate and constant symbols occurring in P, is denoted by B_P, whereas $\neg B_P$ denotes the set $\{\neg A \mid A \in B_P\}$. Analogously, for any set $S \subseteq B_P \cup \neg B_P$, $\neg S$ denotes the set $\{\neg A \mid A \in S\}$, where $\neg\neg A = A$. Given $I \subseteq B_P \cup \neg B_P$, $pos(I)$ (resp., $neg(I)$) stands for $I \cap B_P$ (resp., $\neg I \cap B_P$). I is *consistent* if $pos(I) \cap \neg neg(I) = \emptyset$, otherwise I is *inconsistent*.

Given a program P, $I \subseteq B_P \cup \neg B_P$ is an *interpretation* of P if I is consistent. Also, I is *total* if $pos(I) \cup neg(I) = B_P$, *partial* otherwise. A partial interpretation M of a program P is a *partial model* of P if for each $\neg A \in M$ every rule in $ground(P)$ having as head A contains at least one body literal B such that $\neg B \in M$. Given a program P and a partial model M, the positive instantiation of P w.r.t. M, denoted by P^M, is obtained from $ground(P)$ by deleting: (a) each rule containing a negative literal $\neg A$ such that $A \in pos(M)$; (b) each rule containing a literal B such that neither B nor $\neg B$ is in M; (c) all the negative literals in the remaining rules. Clearly, all the rules in P are definite clauses and hence the minimal Herbrand model of P can be obtained as the least fixpoint of its immediate consequence operator T_{P^M}, denoted by $T^{\omega}_{P^M}(\emptyset)$. For any partial model M of a logic program P, $T^{\omega}_{P^M}(\emptyset) \subseteq M$ (Saccà and Zaniolo 1990).

Let P be a program and M a partial model for P. Then M is (a) *founded* if $T^{\omega}_{P^M}(\emptyset) = pos(M)$; (b) *stable* if it is founded and it is not a proper subset of any other founded model. The set of partial stable models of a logic program P, denoted by $\mathcal{PM}(P)$, define a meet semi-lattice. The *well-founded* model (denoted by $\mathcal{WF}(P)$) and the *maximal-stable* models $\mathcal{MS}(P)$[3], are defined by considering $\subseteq$-minimal and $\subseteq$-maximal elements. The set of (total) *stable* models (denoted by $\mathcal{SM}(P)$) is obtained by considering the maximal-stable models which are total, whereas the *least-undefined* models (denoted by $\mathcal{LM}(P)$) are obtained by considering the maximal-stable models with a $\subseteq$-minimal set of undefined atoms (i.e., atoms which are neither true or false). The *max-deterministic* model (denoted by $\mathcal{MD}(P)$) is the $\subseteq$-maximal PSM contained in every maximal-stable model (Saccà 1997; Greco and Saccà 1999).

Example 6

Consider the program P consisting of the following four rules $\{\mathtt{a} \leftarrow \neg\mathtt{b};\ \mathtt{b} \leftarrow \neg\mathtt{a};\ \mathtt{c} \leftarrow \neg\mathtt{a} \wedge \neg\mathtt{b} \wedge \neg\mathtt{d};\ \mathtt{d} \leftarrow \neg\mathtt{c}\}$. The set of partial stable models of P is $\mathcal{PS}(P) = \{\ \emptyset,\ \{\neg\mathtt{c},\mathtt{d}\},\ \{\mathtt{a},\neg\mathtt{b},\neg\mathtt{c},\mathtt{d}\},\ \{\neg\mathtt{a},\mathtt{b},\neg\mathtt{c},\mathtt{d}\}\ \}$. Consequently, $\mathcal{WF}(\Delta) = \{\ \emptyset\ \}$, $\mathcal{MD}(\Delta) = \{\ \{\neg\mathtt{c},\mathtt{d}\}\ \}$, $\mathcal{MS} = \mathcal{ST}(\Delta) = \mathcal{LS}(\Delta) = \{\{\mathtt{a},\neg\mathtt{b},\neg\mathtt{c},\mathtt{d}\},\{\neg\mathtt{a},\mathtt{b},\neg\mathtt{c},\mathtt{d}\}\}$. □

Propositional Programs. Given a set of symbols $\Lambda = \{a_1, ..., a_n\}$, a *(propositional) program* over Λ is a set of $|\Lambda|$ rules $a_i \leftarrow body_i$ $(1 \leq i \leq n)$, where every $body_i$ is a propositional formula defined over Λ. The semantics of a propositional program P, defined over a given alphabet Λ, is given in terms of the set $\mathcal{PS}(P)$ of its Partial Stable Models (PSMs) that are obtained as follows: *i*) P is first rewritten into a set of standard (ground) logic rules P', whose bodies contain

[3] Corresponding to the *preferred extensions* of (Dung 1991).

https://doi.org/10.1017/S1471068420000253 Published online by Cambridge University Press

conjunction of literals (even by adding fresh symbols to the alphabet)[4]; *ii*) next, the set of PSMs of P' is computed; *iii*) finally, fresh literals added to Λ in the first step are deleted from the models. It is worth noting that for propositional programs we can assume as Herbrand Base the set of (ground) atoms occurring in the program.

3 A Logic Programming Approach

In this section we present a new way to define the semantics of AF-based frameworks by considering propositional programs and partial stable models. In order to compare extensions E of a given framework Δ (containing acceptable elements) with PSMs of a given program P (containing true and false atoms), we denote as $\widehat{E} = E \cup \{\neg a \mid a \in Def(E)\}$ the *completion* of E. Moreover, for a collection of extensions $\mathbf{E}$, $\widehat{\mathbf{E}}$ denotes the set $\{\widehat{E} \mid E \in \mathbf{E}\}$.

Observe also that for any framework Δ and complete extension E for Δ, elements not occurring in $E \cup Def(E)$ are said to be undecided (or undefined), whereas for any program P and PSM M for P, atoms not occurring in $pos(M) \cup neg(M)$ are said to be undefined. Thus, to compare complete extensions and PSMs it is sufficient to consider the completion of extensions.

The next proposition states the relationship between the argumentation frameworks (e.g. AF, BAF, Rec-AF, etc.) and logic programs with partial stable models.

Proposition 1

For any framework $\Delta \in \mathfrak{F}$ and a propositional program P, whenever $\widehat{\mathcal{CO}(\Delta)} = \mathcal{PS}(P)$ it holds that $\widehat{\mathcal{PR}(\Delta)} = \mathcal{MS}(P)$, $\widehat{\mathcal{ST}(\Delta)} = \mathcal{ST}(P)$, $\widehat{\mathcal{SST}(\Delta)} = \mathcal{LM}(P)$, $\widehat{\mathcal{GR}(\Delta)} = \mathcal{WF}(P)$, and $\widehat{\mathcal{ID}(\Delta)} = \mathcal{MD}(P)$.[5]

The result of Proposition 1 derives from the fact that preferred, stable, semi-stable, grounded, and ideal extensions are defined by selecting a subset of the complete extensions satisfying given criteria (see Section 2). On the other side, the maximal, stable, least-undefined, well-founded, and max-deterministic (partial) stable models are obtained by selecting a subset of the PSMs satisfying criteria coinciding with those used to restrict the set of complete extensions.

Given a framework Δ and an extension E, for any element a which could occur in some extension of Δ, the truth value $v_E(a)$, or simply $v(a)$ whenever E is understood, is equal to `true` if $a \in E$, `false` if $a \in Def(E)$, `undec` (*undecided*) otherwise. Hereafter, we assume that `false` $<$ `undec` $<$ `true` and $\neg$`undec` $=$ `undec`.

The strict relationship between the semantics of AFs (given in terms of subset of complete extensions) and the semantics of logic programs (given in terms of subset of PSMs) has been shown in (Wu et al. 2009; Caminada et al. 2015). The relationship is based on the observation that the meaning of an attack $a \to b$ is that the condition $v(b) \leq \neg v(a)$ must hold. On the other side, the satisfaction of a logical rule $a \leftarrow b_1, ..., b_n$ implies that $v(a) \geq min\{v(b_1), ..., v(b_n), \texttt{true}\}$.

Definition 1

Given an AF $\Delta = \langle A, \Omega \rangle$, we denote as $P_\Delta = \{a \leftarrow \bigwedge_{(b,a)\in\Omega} \neg b \mid a \in A\}$ the propositional program derived from Δ.

[4] A rule $a \leftarrow (b \vee c) \wedge (d \vee e)$ is rewritten as $a \leftarrow \neg a_1 \wedge \neg a_2$, $a_1 \leftarrow \neg b \wedge \neg c$ and $a_2 \leftarrow \neg d \wedge \neg e$.

[5] For the novel frameworks $\Delta \in$ {AFRAD, RAFD}, the set $\mathcal{CO}(\Delta)$ of the complete extensions, and the sets of extensions prescribed by the other semantics, are defined in Section 4.

https://doi.org/10.1017/S1471068420000253 Published online by Cambridge University Press

The semantics of an AF Δ can be obtained by considering PSMs of the logic program P_Δ. Particularly, for any AF Δ, $\widehat{\mathcal{CO}(\Delta)} = \mathcal{PS}(P_\Delta)$. Therefore, a natural question is: *Can we also model semantics defined for frameworks extending AF by means of PSMs of logic programs?* The answer is *Yes* and we shall investigate this relationship in the rest of the paper.

Although for a BAF Δ with deductive (resp., necessary) supports this could be carried out by considering the program P_{Δ_d} (resp., P_{Δ_n}), where Δ_d (resp., Δ_n) is the AF obtained from Δ by substituting supports with complex (resp., extended) attacks, we propose a general method that can be applied to all the discussed frameworks, and even to new frameworks (see Section 4).

In order to model frameworks extending Dung's framework by logic programs under PSM semantics, we provide new definitions of defeated and acceptable sets that, for a given set $\mathbf{S}$, will be denoted by $\textsc{Def}(\mathbf{S})$ and $\textsc{Acc}(\mathbf{S})$, respectively. These definitions will be used to derive rules in P_Δ. For AFs we have that for every set $\mathbf{S} \subseteq A$, $\textsc{Def}(\mathbf{S}) = Def(\mathbf{S})$ and $\textsc{Acc}(\mathbf{S}) = Acc(\mathbf{S})$.

3.1 Bipolar AFs

To extend the above result to more general frameworks containing supports (i.e. BAFs and recursive BAFs), we need to separately consider different interpretations of supports.

AFN. The necessary interpretation of supports means that whenever there is a support $a \Rightarrow b$, the condition $v(b) \le v(a)$ must hold. Thus, defeated and acceptable sets can be defined as follows.

Definition 2
For any AFN $\langle A, \Omega, \Gamma\rangle$ and set of arguments $\mathbf{S} \subseteq A$,

•$\textsc{Def}(\mathbf{S}) = \{a \in A \mid (\exists b \in \mathbf{S} \,.\, (b,a) \in \Omega) \ \vee \ (\exists c \in \textsc{Def}(\mathbf{S}) \,.\, (c,a) \in \Gamma)\}$;
•$\textsc{Acc}(\mathbf{S}) = \{a \in A \mid (\forall b \in A \,.\, (b,a) \in \Omega \Rightarrow b \in \textsc{Def}(\mathbf{S})) \wedge (\forall c \in A \,.\, (c,a) \in \Gamma \Rightarrow c \in \textsc{Acc}(\mathbf{S}))\}$.

It is worth noting that $\textsc{Def}(\mathbf{S})$ and $\textsc{Acc}(\mathbf{S})$ are defined recursively, and that in general they may differ from $Def(\mathbf{S})$ and $Acc(\mathbf{S})$, respectively, as shown in the following example.

Example 7
Let $\langle\{\mathtt{a},\mathtt{b},\mathtt{c},\mathtt{d}\}, \{(\mathtt{b},\mathtt{c}),(\mathtt{c},\mathtt{d})\}, \{(\mathtt{b},\mathtt{a})\}\rangle$ be an AFN. Then, $Def(\{\mathtt{a}\}) = \{\mathtt{c}\}$ and $Acc(\{\mathtt{a}\}) = \{\mathtt{a},\mathtt{b},\mathtt{d}\}$, whereas $\textsc{Def}(\{\mathtt{a}\}) = \emptyset$ and $\textsc{Acc}(\{\mathtt{a}\}) = \{\mathtt{a},\mathtt{b}\}$. On the other hand $Def(\{\mathtt{a},\mathtt{b},\mathtt{d}\}) = \textsc{Def}(\{\mathtt{a},\mathtt{b},\mathtt{d}\}) = \{\mathtt{c}\}$ and $Acc(\{\mathtt{a},\mathtt{b},\mathtt{d}\}) = \textsc{Acc}(\{\mathtt{a},\mathtt{b},\mathtt{d}\}) = \{\mathtt{a},\mathtt{b},\mathtt{d}\}$. □

Theorem 1
Given an AFN Δ and an extension $\mathbf{S} \in \mathcal{CO}(\Delta)$, then $Def(\mathbf{S})$=$\textsc{Def}(\mathbf{S})$ and $Acc(\mathbf{S})$=$\textsc{Acc}(\mathbf{S})$.

Theorem 1 states that in order to define the semantics for an AFN Δ we can use acceptable sets $\mathbf{S} = \textsc{Acc}(\mathbf{S})$. This is captured by the following definition, that shows how to derive a propositional program from an AFN.

Definition 3
Given an AFN $\Delta = \langle A, \Omega, \Gamma\rangle$, then $P_\Delta = \{a \leftarrow (\bigwedge_{(b,a)\in\Omega} \neg b \ \wedge \ \bigwedge_{(c,a)\in\Gamma} c) \mid a \in A\}$ denotes the propositional program derived from Δ.

Theorem 2
For any AFN Δ, $\widehat{\mathcal{CO}(\Delta)} = \mathcal{PS}(P_\Delta) = \mathcal{PS}(P_{\Delta_n})$.

The previous theorem states that the set of complete extensions of an AFN Δ coincides with the set of PSMs of the derived logic program P_Δ. Consequently the set of PSMs of P_Δ and P_{Δ_n}, derived from the AF Δ_n, also coincide. Moreover, using Proposition 1, also the others argumentation semantics turns out to be characterized in terms of subsets of PSMs.

Example 8
Consider the AFN Δ of Figure 1. Then, the propositional program derived from Δ is $P_\Delta = \{(\mathtt{w_i} \leftarrow), (\mathtt{r} \leftarrow \neg\mathtt{w_i}), (\mathtt{w_e} \leftarrow \mathtt{r}), (\mathtt{p} \leftarrow \neg\mathtt{w_e})\}$, and $P_{\Delta_n} = \{(\mathtt{w_i} \leftarrow), (\mathtt{r} \leftarrow \neg\mathtt{w_i}), (\mathtt{w_e} \leftarrow \neg\mathtt{w_i}), (\mathtt{p} \leftarrow \neg\mathtt{w_e})\}$. Clearly, $\widehat{\mathcal{CO}(\Delta)} = \mathcal{PS}(P_\Delta) = \mathcal{PS}(P_{\Delta_n}) = \{\{\mathtt{w_i}, \neg\mathtt{r}, \neg\mathtt{w_e}, \mathtt{p}\}\}$. □

AFD. The deductive interpretation of supports means that whenever there is a support $a \Rightarrow b$, the condition $v(a) \le v(b)$ must hold. Thus, defeated and acceptable sets can be defined as follows.

Definition 4
For any AFD $\Delta = \langle A, \Omega, \Gamma \rangle$ and set of arguments $\mathbf{S} \subseteq A$,
- $\textsc{Def}(\mathbf{S}) = \{a \in A \mid (\exists\, b \in \mathbf{S}\,.\,(b,a) \in \Omega) \;\vee\; (\exists\, c \in \textsc{Def}(\mathbf{S})\,.\,(a,c) \in \Gamma)\}$;
- $\textsc{Acc}(\mathbf{S}) = \{a \in A \mid (\forall\, b \in A\,.\,(b,a) \in \Omega \Rightarrow b \in \textsc{Def}(\mathbf{S})) \wedge (\forall c \in A\,.(a,c) \in \Gamma \Rightarrow c \in \textsc{Acc}(\mathbf{S}))\}$.

Theorem 3
Given an AFD Δ and an extension $\mathbf{S} \in \mathcal{CO}(\Delta)$, then $Def(\mathbf{S})$=$\textsc{Def}(\mathbf{S})$ and $Acc(\mathbf{S})$=$\textsc{Acc}(\mathbf{S})$.

We derive a program P_Δ from a given AFD Δ as follows.

Definition 5
Given an AFD $\Delta = \langle A, \Omega, \Gamma \rangle$, then $P_\Delta = \{a \leftarrow (\bigwedge_{(b,a)\in\Omega} \neg b \;\wedge\; \bigwedge_{(a,c)\in\Gamma} c) \mid a \in A\}$ denotes the propositional program derived from Δ.

Similarly to what done earlier, results stating the relationships between AFD semantics and partial stable models can be obtained.

Theorem 4
For any AFD Δ, $\widehat{\mathcal{CO}(\Delta)} = \mathcal{PS}(P_\Delta) = \mathcal{PS}(P_{\Delta_d})$.

3.2 *Recursive BAFs with Necessary Supports*

In this section we study the relationship between partial stable models and the semantics of Rec-BAFs. Particularly, we first present results for RAFN semantics, and then we discuss results for the ASAF framework. We remand to the next section the presentation of two novel semantics for recursive bipolar AFs with deductive interpretation of supports.

RAFN. We next provide the definitions of defeated and acceptable sets for an RAFN.

Definition 6
For any RAFN $\langle A, \Sigma, \Pi, \mathbf{s}, \mathbf{t} \rangle$ and set $\mathbf{S} \subseteq A \cup \Sigma \cup \Pi$, we have that:
- $\textsc{Def}(\mathbf{S}) = \{X \in A \cup \Sigma \cup \Pi \mid (\exists \alpha \in \Sigma \cap \mathbf{S}\,.\,\mathbf{s}(\alpha) \in \mathbf{S} \wedge \mathbf{t}(\alpha) = X) \;\vee$ $(\exists\, \beta \in \Pi \cap \mathbf{S}\,.\,\mathbf{s}(\beta) \in \textsc{Def}(\mathbf{S}) \wedge \mathbf{t}(\beta) = X)\}$;
- $\textsc{Acc}(\mathbf{S}) = \{X \in A \cup \Sigma \cup \Pi \mid (\forall \alpha \in \Sigma\,.\,\mathbf{t}(\alpha) = X \Rightarrow (\alpha \in \textsc{Def}(\mathbf{S}) \vee \mathbf{s}(\alpha) \in \textsc{Def}(\mathbf{S}))) \wedge$ $(\forall \beta \in \Pi\,.\,\mathbf{t}(\beta) = X \Rightarrow (\beta \in \textsc{Def}(\mathbf{S}) \vee \mathbf{s}(\beta) \in \textsc{Acc}(\mathbf{S})))\}$.

The following theorem allows to easily derive the propositional program for any RAFN, by directly looking at the set $\textsc{Acc}(\mathbf{S})$ of acceptable elements.

Theorem 5
Given an RAFN Δ and an extension $\mathbf{S} \in \mathcal{CO}(\Delta)$, then $Def(\mathbf{S})$=$\textsc{Def}(\mathbf{S})$ and $Acc(\mathbf{S})$=$\textsc{Acc}(\mathbf{S})$.

https://doi.org/10.1017/S1471068420000253 Published online by Cambridge University Press

Definition 7
Given an RAFN $\Delta = \langle A, \Sigma, \Pi, \mathbf{s}, \mathbf{t}\rangle$, then P_Δ (the propositional program derived from Δ) contains, for each $X \in A \cup \Sigma \cup \Pi$, a rule

$$X \leftarrow \bigwedge_{\alpha\in\Sigma\wedge\mathbf{t}(\alpha)=X} (\neg\alpha \vee \neg\mathbf{s}(\alpha)) \wedge \bigwedge_{\beta\in\Pi\wedge\mathbf{t}(\beta)=X} (\neg\beta \vee \mathbf{s}(\beta)).$$

The set of complete extensions of an RAFN Δ coincides with the set of PSMs of P_Δ.

Theorem 6
For any RAFN Δ, $\widehat{\mathcal{CO}(\Delta)} = \mathcal{PS}(P_\Delta)$.

Previous results also apply to restricted frameworks such as RAF, where $\Pi = \emptyset$, and AFN, where $\mathbf{t} : \Sigma \to A$.

ASAF. We next provide definitions of defeated and acceptable sets for an ASAF.

Definition 8
Given an ASAF $\langle A, \Sigma, \Pi, \mathbf{s}, \mathbf{t}\rangle$ and a set $\mathbf{S} \subseteq A \cup \Sigma \cup \Pi$, we define:

- $\text{DEF}(\mathbf{S}) = \{X \in A \cup \Sigma \cup \Pi \mid (X \in \Sigma \wedge \mathbf{s}(X) \in \text{DEF}(\mathbf{S})) \vee (\exists \alpha \in \Sigma \cap \mathbf{S}\,.\,\mathbf{t}(\alpha) = X) \vee (\exists \beta \in \Pi \cap \mathbf{S}\,.\,\mathbf{t}(\beta) = X \;\wedge\; \mathbf{s}(\beta) \in \text{DEF}(\mathbf{S}))\}$;
- $\text{ACC}(\mathbf{S}) = \{X \in A\cup\Sigma\cup\Pi \mid (X \in \Sigma \Rightarrow \mathbf{s}(X) \in \text{ACC}(\mathbf{S})) \wedge (\forall \alpha \in \Sigma\,.\,\mathbf{t}(\alpha)=X \Rightarrow \alpha \in \text{DEF}(\mathbf{S})) \wedge (\forall \beta \in \Pi\,.\,\mathbf{t}(\beta)=X \Rightarrow (\beta \in \text{DEF}(\mathbf{S}) \;\vee\; \mathbf{s}(\beta) \in \text{ACC}(\mathbf{S})))\}$.

The acceptable elements of an ASAF can be computed by using the previous definition.

Theorem 7
Given an ASAF Δ and an extension $\mathbf{S} \in \mathcal{CO}(\Delta)$, then $Def(\mathbf{S})$=$\text{DEF}(\mathbf{S})$ and $Acc(\mathbf{S})$=$\text{ACC}(\mathbf{S})$.

By exploiting the result of Theorem 7 now define the propositional program for an ASAF Δ, which is easily derived by looking at the new definition of acceptable elements (i.e., $\text{ACC}(\mathbf{S})$).

Definition 9
For any ASAF $\Delta = \langle A, \Sigma, \Pi, \mathbf{s}, \mathbf{t}\rangle$, P_Δ (the propositional program derived from Δ) contains, for each $X \in A \cup \Sigma \cup \Pi$, a rule of the form

$$X \leftarrow \varphi(X) \wedge \bigwedge_{\alpha\in\Sigma\wedge\mathbf{t}(\alpha)=X} \neg\alpha \wedge \bigwedge_{\beta\in\Pi\wedge\mathbf{t}(\beta)=X} (\neg\beta \vee \mathbf{s}(\beta)) \text{ where } \varphi(X) = \begin{cases} \mathbf{s}(X) \text{ if } X \in \Sigma \\ \texttt{true} \text{ otherwise.} \end{cases}$$

Theorem 8
For any ASAF Δ, $\widehat{\mathcal{CO}(\Delta)} = \mathcal{PS}(P_\Delta)$.

Similarly to the case of RAFN, the above results also apply to restricted frameworks such as AFRA, where $\Pi = \emptyset$, and AFN, where $\mathbf{t} : \Sigma \to A$.

Example 9
Consider the Rec-BAF Δ' of Example 5 derived from the Rec-BAF of Figure 2 by adding an argument $\mathtt{s}$ attacking argument $\mathtt{w_t}$ through α_4, under the necessary interpretation of supports. The propositional program under the RAFN semantics is $P_{\Delta'} = \{(\mathtt{w_i} \leftarrow), (\mathtt{r} \leftarrow \neg\alpha_1 \vee \neg\mathtt{w_i}), (\mathtt{w_e} \leftarrow \neg\beta_1 \vee \mathtt{r}), (\mathtt{p} \leftarrow \neg\alpha_2 \vee \neg\mathtt{w_e}), (\mathtt{w_t} \leftarrow \neg\alpha_4 \vee \neg\mathtt{s}), (\alpha_1 \leftarrow \neg\alpha_3 \vee \neg\mathtt{w_t}), (\mathtt{s} \leftarrow), (\alpha_2 \leftarrow), (\alpha_3 \leftarrow), (\alpha_4 \leftarrow), (\beta_1 \leftarrow)\}$, whose set of partial stable model is $M_1 = \mathcal{PS}(P_{\Delta'}) = \{\{\mathtt{s}, \mathtt{w_i}, \neg\mathtt{r}, \neg\mathtt{w_e}, \neg\mathtt{w_t}, \mathtt{p}, \beta_1, \alpha_1, \alpha_2, \alpha_3, \alpha_4\}\}$.

Analogously, the propositional program for Δ' under the ASAF semantics is $P_{\Delta'} = \{(\mathtt{w_i} \leftarrow), (\mathtt{r} \leftarrow \neg\alpha_1), (\mathtt{w_e} \leftarrow \neg\beta_1 \vee \mathtt{r}), (\mathtt{p} \leftarrow \neg\alpha_2), (\mathtt{w_t} \leftarrow \neg\alpha_4), (\alpha_1 \leftarrow \neg\alpha_3 \wedge \mathtt{w_i}), (\mathtt{s} \leftarrow), (\alpha_2 \leftarrow \mathtt{w_e}), (\alpha_3 \leftarrow \mathtt{w_t}), (\alpha_4 \leftarrow \mathtt{s}), (\beta_1 \leftarrow)\}$, whose set of partial stable model is $M_2 = \mathcal{PS}(P_{\Delta'}) = \{\{\mathtt{s}, \mathtt{w_i}, \neg\mathtt{r}, \neg\mathtt{w_e}, \neg\mathtt{w_t}, \mathtt{p}, \beta_1, \alpha_1, \neg\alpha_2, \neg\alpha_3, \alpha_4\}\}$, which differs from M_1 in the status of α_2 and α_3. □

https://doi.org/10.1017/S1471068420000253 Published online by Cambridge University Press

4 Recursive BAFs with Deductive Supports

In this section we study two new frameworks both belonging to the Rec-BAF class and both extending AFD by allowing recursive attacks and deductive supports. The first one, called *Recursive Argumentation Framework with Deductive supports (RAFD)*, extends RAF, whereas the second one, called *Argumentation Framework with Recursive Attacks and Deductive supports (AFRAD)*, extends AFRA. It is again assumed that Π is acyclic and $\Sigma \cap \Pi = \emptyset$.

As we shall define the semantics by defining directly the sets $\textsc{Def}(\mathbf{S})$ and $\textsc{Acc}(\mathbf{S})$, differently from the previous section, we do not have any results regarding the equivalence between the sets $Acc(\mathbf{S})$ and $\textsc{Acc}(\mathbf{S})$ for $\mathbf{S} = \textsc{Acc}(\mathbf{S})$.

RAFD. As usual, we first define the sets of defeated and acceptable elements, and then the propositional logic program for an RAFD.

Definition 10
For any RAFD $\langle A, \Sigma, \Pi, \mathbf{s}, \mathbf{t}\rangle$ and set $\mathbf{S} \subseteq A \cup \Sigma \cup \Pi$, we have that:

- $\textsc{Def}(\mathbf{S}) = \{X \in A \cup \Sigma \cup \Pi \mid (\exists \alpha \in \Sigma \cap \mathbf{S} \,.\, \mathbf{t}(\alpha) = X \wedge \mathbf{s}(\alpha) \in \mathbf{S}) \vee$
$(\exists \beta \in \Pi \cap \mathbf{S} \,.\, \mathbf{s}(\beta) = X \wedge \mathbf{t}(\beta) \in \textsc{Def}(\mathbf{S}))\ \}$;
- $\textsc{Acc}(\mathbf{S}) = \{X \in A \cup \Sigma \cup \Pi \mid (\forall \alpha \in \Sigma \,.\, \mathbf{t}(\alpha) = X \Rightarrow (\alpha \in \textsc{Def}(\mathbf{S}) \vee \mathbf{s}(\alpha) \in \textsc{Def}(\mathbf{S}))) \wedge$
$(\forall \beta \in \Pi \,.\, \mathbf{s}(\beta) = X \ \Rightarrow (\beta \in \textsc{Def}(\mathbf{S}) \vee \mathbf{t}(\beta) \in \textsc{Acc}(\mathbf{S})))\ \}$.

The sets of extensions prescribed by the different semantics are based on the defeated and acceptable sets defined above. That is, given an RAFD $\Delta = \langle A, \Sigma, \Pi, \mathbf{s}, \mathbf{t}\rangle$, a set $\mathbf{S} \subseteq A \cup \Sigma \cup \Pi$ of elements is a complete extension of Δ iff it is conflict-free (i.e., $\mathbf{S} \cap \textsc{Def}(\mathbf{S}) = \emptyset$) and $\mathbf{S} = \textsc{Acc}(\mathbf{S})$. As done for the other frameworks, we use $\mathcal{CO}(\Delta)$ to denote the set of complete extensions of Δ. Moreover, the set of preferred (resp., stable, semi-stable, grounded, ideal) extensions is defined in the standard way (see Section 2.1) by using again $\textsc{Def}(\mathbf{S})$ and $\textsc{Acc}(\mathbf{S})$.

Using the definition of $\textsc{Acc}(\mathbf{S})$, we define the propositional program for an RAFD Δ.

Definition 11
Given an RAFD $\Delta = \langle A, \Sigma, \Pi, \mathbf{s}, \mathbf{t}\rangle$, then P_Δ (the propositional program derived from Δ) contains, for each $X \in A \cup \Sigma \cup \Pi$, a rule of the form

$$X \leftarrow \bigwedge_{\alpha \in \Sigma \wedge \mathbf{t}(\alpha) = X} (\neg\alpha \vee \neg\mathbf{s}(\alpha)) \wedge \bigwedge_{\beta \in \Pi \wedge \mathbf{s}(\beta) = X} (\neg\beta \vee \mathbf{t}(\beta)).$$

Theorem 9
For any RAFD Δ, $\widehat{\mathcal{CO}(\Delta)} = \mathcal{PS}(P_\Delta)$.

Thus, as expected, the semantics of an RAFD Δ can be carried out by using the PSMs of P_Δ.

AFRAD. The following definition formalizes defeated and acceptable sets for an AFRAD.

Definition 12
Given an AFRAD $\langle A, \Sigma, \Pi, \mathbf{s}, \mathbf{t}\rangle$ and a set $\mathbf{S} \subseteq A \cup \Sigma \cup \Pi$, we have that

- $\textsc{Def}(\mathbf{S}) = \{X \in A \cup \Sigma \cup \Pi \mid (X \in \Sigma \wedge \mathbf{s}(X) \in \textsc{Def}(\mathbf{S})) \vee (\exists \alpha \in \Sigma \cap \mathbf{S} \,.\, \mathbf{t}(\alpha) = X) \vee$
$(\exists \beta \in \Pi \cap \mathbf{S} \,.\, \mathbf{s}(\beta) = X \ \wedge \ \mathbf{t}(\beta) \in \textsc{Def}(\mathbf{S}))\}$;
- $\textsc{Acc}(\mathbf{S}) = \{X \in A \cup \Sigma \cup \Pi \mid (X \in \Sigma \Rightarrow \mathbf{s}(X) \in \textsc{Acc}(\mathbf{S}))) \wedge (\forall \alpha \in \Sigma \,.\, \mathbf{t}(\alpha) = X \Rightarrow \alpha \in \textsc{Def}(\mathbf{S}))$
$\wedge (\forall \beta \in \Pi \,.\, \mathbf{s}(\beta) = X \Rightarrow (\beta \in \textsc{Def}(\mathbf{S}) \vee \mathbf{t}(\beta) \in \textsc{Acc}(\mathbf{S}))))\}$.

Similarly to what done for RAFDs, the set $\mathcal{CO}(\Delta)$ of complete extensions of an AFRAD Δ, and the sets of extensions prescribed by the other semantics, are defined by using the defeated and acceptable sets defined above.

https://doi.org/10.1017/S1471068420000253 Published online by Cambridge University Press

Definition 13
For any AFRAD $\Delta = \langle A, \Sigma, \Pi, \mathtt{s}, \mathtt{t}\rangle$, P_Δ (the propositional program derived from Δ) contains, for each $X \in A \cup \Sigma \cup \Pi$, a rule of the form

$$X \leftarrow \varphi(X) \wedge \bigwedge_{\alpha\in\Sigma\wedge \mathtt{t}(\alpha)=X} \neg\alpha \wedge \bigwedge_{\beta\in\Pi\wedge \mathtt{s}(\beta)=X} (\neg\beta \vee \mathtt{t}(\beta)) \text{ where } \varphi(X) = \begin{cases} \mathtt{s}(X) \text{ if } X \in \Sigma \\ \mathtt{true} \text{ otherwise.} \end{cases}$$

Theorem 10
For any AFRAD Δ, $\widehat{\mathcal{CO}(\Delta)} = \mathcal{PS}(P_\Delta)$.

Example 10
Consider the Rec-BAF Δ' of Example 5 and assume that supports are interpreted as deductive. The propositional program under the RAFD semantics is $P_{\Delta'} = \{(\mathtt{w_i} \leftarrow), (\mathtt{r} \leftarrow (\neg\alpha_1 \vee \neg\mathtt{w_i}) \wedge (\neg\beta_1 \vee \mathtt{w_e})), (\mathtt{w_e} \leftarrow), (\mathtt{p} \leftarrow \neg\alpha_2 \vee \neg\mathtt{w_e}), (\mathtt{w_t} \leftarrow \neg\alpha_4 \vee \neg\mathtt{s}), (\mathtt{s} \leftarrow), (\alpha_1 \leftarrow \neg\alpha_3 \vee \neg\mathtt{w_t}), (\alpha_2 \leftarrow), (\alpha_3 \leftarrow), (\alpha_4 \leftarrow), (\beta_1 \leftarrow)\}$, whose set of partial stable model is $M_1 = \mathcal{PS}(P_{\Delta'}) = \{\{\mathtt{s}, \mathtt{w_i}, \neg\mathtt{r}, \mathtt{w_e}, \neg\mathtt{w_t}, \neg\mathtt{p}, \beta_1, \alpha_1, \alpha_2, \alpha_3, \alpha_4\}\}$. Analogously, the propositional program for Δ' under the AFRAD semantics is $P_{\Delta'} = \{(\mathtt{w_i} \leftarrow), (\mathtt{r} \leftarrow \neg\alpha_1 \wedge (\neg\beta_1 \vee \mathtt{w_e})), (\mathtt{w_e} \leftarrow), (\mathtt{p} \leftarrow \neg\alpha_2), (\mathtt{w_t} \leftarrow \neg\alpha_4), (\mathtt{s} \leftarrow), (\alpha_1 \leftarrow \mathtt{w_i} \wedge \neg\alpha_3), (\alpha_2 \leftarrow \mathtt{w_e}), (\alpha_3 \leftarrow \mathtt{w_t}), (\alpha_4 \leftarrow \mathtt{s}), (\beta_1 \leftarrow)\}$, whose set of partial stable model is $M_2 = \mathcal{PS}(P_{\Delta'}) = \{\{\mathtt{s}, \mathtt{w_i}, \neg\mathtt{r}, \mathtt{w_e}, \neg\mathtt{w_t}, \neg\mathtt{p}, \alpha_1, \neg\alpha_2, \neg\alpha_3, \alpha_4, \beta_1\}\}$. Observe that the RAFD (resp., AFRAD) program differs from the RAFN (resp., ASAF) program only in rules having as head arguments $\mathtt{r}$ and $\mathtt{w_e}$. □

5 Discussion and Future Work

By exploring the connection between formal argumentation and logic programming, we have proposed a simple but general logical framework which is able to capture, in a systematic and succinct way, the different features of several AF-based frameworks under different argumentation semantics and interpretation of the support relation. The proposed approach can be used for better understanding the semantics of extended AF frameworks (sometimes a bit involved), and is flexible enough for encouraging the study of other extensions.

As pointed out in Section 1, our work is complementary to approaches providing the semantics for an AF-based framework by using meta-argumentation, that is, by relying on a translation from a given AF-based framework to an AF (Cohen et al. 2015). In this regard, we observe that meta-argumentation approaches have the drawback of making a bit difficult understanding the original meaning of arguments and interactions once translated into the resulting meta-AF. In fact, those approaches rely on translations that generally require adding several meta-arguments and meta-attacks to the resulting meta-AF in order to model the original interactions.

Concerning approaches that provide the semantics of argumentation frameworks by LPs (Caminada et al. 2015), we observe that a logic program for an AF-based framework can be obtained by first flattening the given framework into a meta-AF and then converting it into a logic program. The so-obtained program contains the translation of meta-arguments and meta-attacks that make the program much more verbose and difficult to understand (because not straightly derived from the given extended AF framework) in our opinion, compared with the direct translation we proposed. For instance, given an ASAF $\Delta = \langle A, \Sigma, \Pi, \mathtt{s}, \mathtt{t}\rangle$, the propositional program P_Δ directly obtained from Δ has a number of rules equal to $|A| + |\Sigma| + |\Pi|$, while the program $P_{\Delta'}$ obtained considering the translation from Δ to an AFN and then to meta-AF Δ' consists of $|A|+|\Sigma|+3|\Pi|$

https://doi.org/10.1017/S1471068420000253 Published online by Cambridge University Press

rules, of which $2|\Pi|$ rules define new (meta-)arguments (examples of the arguments introduced, which correspond to rules of $P_{\Delta'}$, can be found in (Gottifredi et al. 2018)). In addition, the size of body's rules may also increase for $P_{\Delta'}$ since the number of extended/complex attacks that need to be added may be relevant in some cases. Finally, the models of $P_{\Delta'}$ contain literals corresponding to meta-arguments having no meaning w.r.t. extensions of the extended AF Δ.

In brief, the program that we directly obtain from a given AF-based framework is more concise and easy to understand with respect to that obtained by (possibly several stages of) translations to AF. Moreover, the proposed approach uniformly deals with several AF-based frameworks, including RAFN and the novel frameworks RAFD and AFRAD for which a translation to AF has not been defined. Nevertheless, we believe that our approach is also complementary to approaches using intermediate translations to AF in order to define an LP for an extended AF.

Furthermore, our approach can also be used to provide additional tools for computing complete extensions using answer set solvers (Gebser et al. 2018) and classical program rewriting (Janhunen et al. 2006; Sakama and Rienstra 2017; Gaggl et al. 2015). In particular, we plan to experimentally compare the following LP approaches for the computation of extensions of AF-based frameworks Δ: (i) using the propositional program P_Δ directly obtained from Δ; and (ii) using the propositional program $P_{\Delta'}$ obtained from Δ by transforming it to an AF Δ' (possibly through different transformations, involving different intermediate argumentation frameworks).

Other extensions of the Dung's framework not explicitly discussed in this paper are also captured by our technique as they are special cases of some of those studied in this paper. This is the case of *Extended AF (EAF)* and *hierarchical EAF*, which extend AF by allowing second order and stratified attacks, respectively (Modgil 2009), that are special cases of recursive attacks.

Future work will be also devoted to further generalize our logical approach in order to deal also with AF-based framework considering probabilities (Fazzinga et al. 2015), weights (Bistarelli et al. 2018), and preferences (Amgoud and Vesic 2011; Modgil 2009), and frameworks considering supports with multiple sources (Cayrol et al. 2018). Finally, we plan to investigate incremental techniques tailored at using our approach to compute extensions of dynamic AF-based frameworks, where the sets of arguments and interactions change over the time (Greco and Parisi 2016; Alfano et al. 2017; Alfano et al. 2018; Alfano et al. 2020).

References

ALCÂNTARA, J. F. L., SÁ, S., AND GUADARRAMA, J. C. A. 2019. On the equivalence between abstract dialectical frameworks and logic programs. *TPLP 19,* 5-6, 941–956.

ALFANO, G., COHEN, A., GOTTIFREDI, S., GRECO, S., PARISI, F., AND SIMARI, G. R. 2020. Dynamics in abstract argumentation frameworks with recursive attack and support relations. In *ECAI (To appear).*

ALFANO, G., GRECO, S., AND PARISI, F. 2017. Efficient computation of extensions for dynamic abstract argumentation frameworks: An incremental approach. In *IJCAI*. 49–55.

ALFANO, G., GRECO, S., AND PARISI, F. 2018. A meta-argumentation approach for the efficient computation of stable and preferred extensions in dynamic bipolar argumentation frameworks. *Intelligenza Artificiale 12,* 2, 193–211.

AMGOUD, L. AND VESIC, S. 2011. A new approach for preference-based argumentation frameworks. *Ann. Math. Artif. Intell. 63,* 2, 149–183.

BARONI, P., CERUTTI, F., GIACOMIN, M., AND GUIDA, G. 2011. AFRA: Argumentation Framework with Recursive Attacks. *IJAR 52,* 1, 19–37.

BENCH-CAPON, T. AND DUNNE, P. E. 2007. Argumentation in artificial intelligence. *AI 171*, 619 – 641.

BISTARELLI, S., ROSSI, F., AND SANTINI, F. 2018. A novel weighted defence and its relaxation in abstract argumentation. *IJAR 92*, 66–86.

Bondarenko, A., Dung, P. M., Kowalski, R. A., and Toni, F. 1997. An abstract, argumentation-theoretic approach to default reasoning. *AI 93*, 63–101.

Caminada, M., Sá, S., Alcântara, J. F. L., and Dvorák, W. 2015. On the equivalence between logic programming semantics and argumentation semantics. *IJAR 58*, 87–111.

Caminada, M. and Schulz, C. 2017. On the equivalence between assumption-based argumentation and logic programming. *JAIR 60*, 779–825.

Cayrol, C., Fandinno, J., del Cerro, L. F., and Lagasquie-Schiex, M. 2017. Valid attacks in argumentation frameworks with recursive attacks. In *Proc. of COMMONSENSE*.

Cayrol, C., Fandinno, J., del Cerro, L. F., and Lagasquie-Schiex, M. 2018. Structure-based semantics of argumentation frameworks with higher-order attacks and supports. In *COMMA*. 29–36.

Cayrol, C. and Lagasquie-Schiex, M. 2013. Bipolarity in argumentation graphs: Towards a better understanding. *IJAR 54,* 7, 876–899.

Cohen, A., Gottifredi, S., Garcia, A. J., and Simari, G. R. 2014. A survey of different approaches to support in argumentation systems. *The Know. Eng. Rev. 29,* 5, 513–550.

Cohen, A., Gottifredi, S., Garcia, A. J., and Simari, G. R. 2015. An approach to abstract argumentation with recursive attack and support. *J. Appl. Log. 13,* 4, 509–533.

Craven, R. and Toni, F. 2016. Argument graphs and assumption-based argumentation. *AI 233*, 1–59.

Dung, P. M. 1991. Negations as hypotheses: An abductive foundation for logic programming. In *ICLP*. 3–17.

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

Fazzinga, B., Flesca, S., and Parisi, F. 2015. On the complexity of probabilistic abstract argumentation frameworks. *TOCL 16,* 3, 22:1–22:39.

Gaggl, S. A., Manthey, N., Ronca, A., Wallner, J. P., and Woltran, S. 2015. Improved answer-set programming encodings for abstract argumentation. *TPLP 15,* 4-5, 434–448.

Gebser, M., Leone, N., Maratea, M., Perri, S., Ricca, F., and Schaub, T. 2018. Evaluation techniques and systems for answer set programming: a survey. In *IJCAI*. 5450–5456.

Gelfond, M. and Lifschitz, V. 1988. The stable model semantics for logic programming. In *ICSLP*. 1070–1080.

Gottifredi, S., Cohen, A., Garcia, A. J., and Simari, G. R. 2018. Characterizing acceptability semantics of argumentation frameworks with recursive attack and support relations. *AI 262*, 336–368.

Greco, S. and Parisi, F. 2016. Incremental computation of deterministic extensions for dynamic argumentation frameworks. In *JELIA*. 288–304.

Greco, S. and Saccà, D. 1999. Complexity and expressive power of deterministic semantics for datalog¬. *Inf. Comput. 153,* 1, 81–98.

Janhunen, T., Niemelä, I., Seipel, D., Simons, P., and You, J.-H. 2006. Unfolding partiality and disjunctions in stable model semantics. *ACM Trans. Comput. Logic 7,* 1.

Modgil, S. 2009. Reasoning about preferences in argumentation frameworks. *AI 173,* 9-10, 901–934.

Nouioua, F. and Risch, V. 2011. Argumentation frameworks with necessities. In *SUM*. 163–176.

Saccà, D. 1997. The expressive powers of stable models for bound and unbound DATALOG queries. *J. Comput. Syst. Sci. 54,* 3, 441–464.

Saccà, D. and Zaniolo, C. 1990. Stable models and non-determinism in logic programs with negation. In *PODS*. 205–217.

Sakama, C. and Rienstra, T. 2017. Representing argumentation frameworks in answer set programming. *Fundam. Inform. 155,* 3, 261–292.

Simari, G. R. and Rahwan, I., Eds. 2009. *Argumentation in Artificial Intelligence*.

Villata, S., Boella, G., Gabbay, D. M., and van der Torre, L. W. N. 2012. Modelling defeasible and prioritized support in bipolar argumentation. *AMAI 66,* 1-4, 163–197.

Wu, Y., Caminada, M., and Gabbay, D. M. 2009. Complete extensions in argumentation coincide with 3-valued stable models in logic programming. *Studia Logica 93,* 2-3, 383–403.