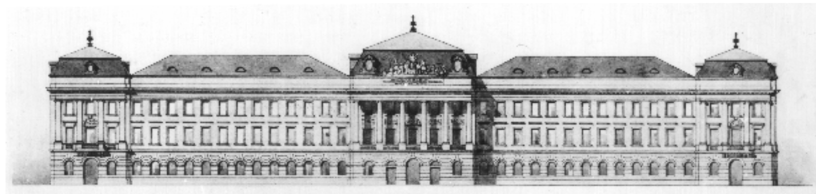
dbai

TECHNICAL REPORT



INSTITUT FÜR INFORMATIONSSYSTEME

ABTEILUNG DATENBANKEN UND ARTIFICIAL INTELLIGENCE

Revisiting Extension–Based Semantics of Abstract Dialectical Frameworks

DBAI-TR-2015-88

Sylwia Polberg

Institut für Informationssysteme
Abteilung Datenbanken und
Artificial Intelligence
Technische Universität Wien
Favoritenstr. 9
A-1040 Vienna, Austria
Tel: +43-1-58801-18403
Fax: +43-1-58801-18493
sekret@dbai.tuwien.ac.at
www.dbai.tuwien.ac.at

DBAI TECHNICAL REPORT
2015

TECHNISCHE UNIVERSITÄT WIEN
Vienna University of Technology

DBAI TECHNICAL REPORT
DBAI TECHNICAL REPORT DBAI-TR-2015-88, 2015

Revisiting Extension–Based Semantics of Abstract Dialectical Frameworks

Sylwia Polberg [1]

Abstract. This report is meant to serve as a new version of our previous work [35] and contains all the proofs and analysis done there. We introduce a new subfamily of the ca–semantics and the acyclic grounded semantics. We also introduce a number of new properties and complete the analysis of the relation between labeling–based and extension–based semantics. Finally, we correct various minor issues and clarify some explanations.
One of the most prominent tools for abstract argumentation is the Dung's framework, AF for short. Although powerful, AFs have their shortcomings, which led to development of numerous enrichments. Among the most general ones are the abstract dialectical frameworks, also known as the ADFs. They make use of the so–called acceptance conditions to represent arbitrary relations. This level of abstraction brings not only new challenges, but also requires addressing existing problems in the field. One of the most controversial issues, recognized not only in argumentation, concerns the support or positive dependency cycles. In this paper we introduce a new method to ensure acyclicity of arguments and present a family of extension–based semantics built on it, along with their classification w.r.t. cycles. We provide ADF versions of the properties known from the Dung setting, provide sufficient requirements for the semantics to coincide and compare them with the labeling–based semantics.

[1]Institute for Information Systems 184/2, Technische Universität Wien, Favoritenstrasse 9-11, 1040 Vienna, Austria. E-mail: polberg@dbai.tuwien.ac.at

Acknowledgements: The author is funded by the Vienna PhD School of Informatics and project I1102 supported by the Austrian Science Fund FWF.

Contents

1 Introduction

This report is meant to serve as a new version of our previous work [35–37] and contains all the proofs and analysis done there. We introduce a new subfamily of the ca–semantics and the acyclic grounded semantics. We also prove a number of new properties and complete the analysis of the relation between labeling–based and extension–based semantics. Finally, we correct various minor issues and clarify some explanations.

Over the last years, argumentation has become an influential subfield of artificial intelligence, with applications ranging from legal reasoning [8] or dialogues and persuasion [29, 40] to medicine [26, 27] or eGovernment [2]. Till today, various formalisms and classifications of types of argumentation have been created [41]. One of them is *abstract argumentation*, which has become especially popular thanks to the research of Phan Minh Dung [23]. Although the framework he has developed is quite powerful, it has certain shortcomings, which inspired a search for more general models. Throughout the years, many different argumentation frameworks were created, ranging from the ones employing various measures of arguments or relations strengths and preferences [1, 7, 22, 30] to ones that focus on researching new types of interactions between the framework elements [4, 18, 31, 32, 34]. An overview of available structures can be found in [12]. One of the most general enrichments of the latter type are the abstract dialectical frameworks, ADFs for short [13]. Instead of extending the Dung's frameworks with elements representing new types of relations each time it is needed, they make use of so–called acceptance conditions to express arbitrary interactions between the arguments. However, a framework cannot be considered a suitable argumentation tool without properly developed semantics.

The semantics of a framework are meant to represent what is considered rational. We may require the chosen opinion to be e.g. consistent, defensible, providing counterarguments for what we cannot accept and so on. Given many of the advanced semantics, such as grounded or complete, we can observe that they return the same results when faced with simple, tree–like frameworks [23]. The differences between them become more visible when we work with the more complicated cases.On various occasions examples were found for which none of the available semantics returned satisfactory answers. This gave rise to new approaches, each trying to tackle this issue. For example, for handling indirect attacks and defenses we have prudent and careful semantics [20,21]; for the problem of even and odd attack cycles we can resort to some of the SCC–recursive semantics [6]; while for treatment of self attackers, sustainable and tolerant semantics were developed [9]. Introducing a new type of relation adds to these issues, but also creates complications of its own.

The support relation raises a number of questions, however, the most controversial problem concerns the support cycles and is handled differently from formalism to formalism. Among the best known structures are the Bipolar Argumentation Frameworks (BAFs) [18], Argumentation Frameworks with Necessities (AFNs) [32] and Evidential Argumentation Systems (EASs) [34]. While the latter two discard support cycles, BAFs do not make such restrictions and in general, neither do ADFs [11,13]. This variety is not an error in any of the structures. First of all, in a more advanced setting, a standard Dung semantics can be extended in several ways. Moreover, since one can find arguments both for and against any of the cycle treatments, lack of consensus as to what approach is the best should not be surprising.

Many properties of the available semantics can be seen as "inside" ones, i.e. "what can I consider rational?". On the other hand, some can be understood as on the "outside", e.g. "what can be considered a valid attacker, what should I defend from?". Various examples of such behavior exist even in the Dung setting. An admissible extension defends against all possible attacks in the framework. We can then restrict this by saying that self–attackers are not rational, and thus limit the set of arguments we have to defend the extension from. If we now add support, we can again define admissibility in the basic manner. However, one often demands that the extensions are free from support cycles and that we only defend from arguments not taking part in them. From this perspective semantics can be seen as a two–person discussion, describing what "I can claim" and "what my opponent can claim". This is also the point of view that we follow in this paper and it will serve as a basis for the classification of our semantics. Please note that this sort of dialogue perspective can already be found in argumentation [24,28], although it is used in a slightly different context.

Various extension–based semantics for ADFs have already been proposed in the original paper [13]. Unfortunately, some of them were defined only for a particular ADF subclass called bipolar and were suitable for certain types of situations. Therefore, only three of them – conflict–free, model and grounded – remain. The research in [11, 42] resulted in establishing a family of semantics we can qualify as labeling–based. Although they resolve the problems of the initial formulations, they have their own drawbacks. They are described in terms of e.g. fixpoints of a three–valued characteristic operator, which is based on the consensus of acceptance conditions. In this formulation, it is not always visible at the first glance how defense and other notions known from the Dung setting behave in ADFs. Moreover, verifying an existing interpretation rather than constructing one from some initial data can result in an argument affecting his own status in face of self–dependencies, which is not always a desirable property when a framework can express support. Finally, shifting from two–valued to three–valued setting is more than just a structural change. While in the extension–based semantics we often aim to accept as many arguments as the rationality allows, in labeling setting knowing that something is true is equally important to knowing it is false. Thus, one makes use of information maximality rather than subset maximality, which in a bipolar setting creates differences not present in AFs. Although we find this method to be suitable for the labeling intuitions, we are missing semantics that would still let us focus on the argument's acceptance.

The aim of this paper is to introduce a family of extension–based semantics and to specialize them to handle the problem of support cycles, as it seems to be the biggest difference between the approaches of the current frameworks that allow positive relations. Consequently, we present methods for ensuring acyclicity in ADFs. Furthermore, a classification of our sub–semantics in the inside–outside fashion that we have described before is introduced. We also recall our previous research on admissibility in [39] and show how it fits into our system. Our results also include which known properties, such as Fundamental Lemma, carry over from the Dung framework. Furthermore, we introduce a subclass of ADFs for which our system collapses, i.e. all sub–semantics of a given type produce the same answers. Finally we provide an analysis of the similarities and differences between the extension and labeling–based semantics.

The report is structured as follows. In Section 2, we provide a short recap on AFs, BAFs, AFNs, EASs and ADFs. Section 3 will focus on the basic concepts underlying the semantics of ADFs. We will then recall the labeling–based semantics in Section 4 and introduce the new extension–based ones in Section 5, including an analysis of their properties in Section 6. We close the paper with a comparison of both of the approaches.

2 Argumentation Frameworks

2.1 Dung's Argumentation Frameworks

Let us start from the basics: the abstract argumentation framework by Dung [23].

Definition 2.1. A **Dung's abstract argumentation framework** (AF for short) is a pair (A, R), where A is a set of **arguments** and $R \subseteq A \times A$ represents the **attack** relation.

AFs can be simply represented as directed graphs. We will now briefly recall the available semantics, for more details we refer the reader to [3].

Definition 2.2. Let $F = (A, R)$ be a Dung's framework. An argument $a \in A$ is **defended** by a set E in F[1], if for each $b \in A$ s.t. $(b, a) \in R$, there exists $c \in E$ s.t. $(c, b) \in R$. A set $E \subseteq A$ is:

- **conflict–free** in F iff for each $a, b \in E$, $(a, b) \notin R$.
- **naive** in F iff it is maximal w.r.t. set inclusion conflict–free in F.
- **admissible** in F iff it is conflict–free in F and defends all of its members.
- **preferred** in F iff it is maximal w.r.t. set inclusion admissible in F.
- **complete** in F iff it is admissible in F and all arguments defended by it are contained in it.
- **stable** in F iff it is conflict–free in F and for each $a \in A \setminus E$ there exists an argument $b \in E$ s.t. $(b, a) \in R$.

The stable semantics is somewhat different than the rest in the sense that depending on the given framework, it might not produce any extensions. This problem is addressed with maximizing the amount of arguments covered by the extension [15]:

Definition 2.3. Let $F = (A, R)$ be a Dung's framework and $E \subseteq A$ a set of arguments. The set of arguments attacked by E is $E^+ = \{a \mid \exists b \in E \text{ s.t. } (b, a) \in R\}$. The set $E^+ \cup E$ is the **range** of E. A conflict–free set $E \subseteq A$ is stable in F iff $E^+ = A \setminus E$. A complete extension $E \subseteq A$ is **semi–stable** in F iff its range is maximal w.r.t. set inclusion.

We close the list with the grounded semantics. It basically represents the knowledge that we can only build from the initial (i.e. unattacked) arguments, i.e. starting with an empty set we first include the initial arguments, then add all elements defended by the set and continue until nothing more is added. The formal definition is given by the means of the characteristic function of AF:

[1] Defense is often substituted with acceptability: say that a is acceptable w.r.t. E if E defends a.

Definition 2.4. Let $F = (A, R)$ be a Dung's framework. The **characteristic function** $\mathcal{F}_F : 2^A \rightarrow 2^A$ of F is defined as: $\mathcal{F}_F(E) = \{a \mid a$ is defended by E in $F\}$. The **grounded extension** of F is the least fixed point of $\mathcal{F}_F$.

Furthermore, other semantics can also be described in terms of the characteristic function; for example, a conflict–free set E is admissible iff $E \subseteq \mathcal{F}_F(E)$ and complete iff $E = \mathcal{F}_F(E)$.

Please note there is also an alternative way to compute the grounded extension:

Proposition 2.5. *Let $F = (A, R)$ be a Dung's framework. The unique* ***grounded extension*** *of F is defined as the outcome E of the following "algorithm". Let us start with $E = \emptyset$:*

1. *put each argument $a \in A$ which is not attacked in F into E; if no such argument exists, return E.*

2. *remove from F all (new) arguments in E and all arguments attacked by them (together with all adjacent attacks) and continue with Step 1.*

What we have described above forms a family of so–called extension–based semantics. We now continue with the labeling–based ones, which are thoroughly explained in [14].

Definition 2.6. Let $F = (A, R)$ be a Dung's framework. A three–valued labeling is a total function $Lab : A \rightarrow \{in, out, undec\}$[2]. An in–labeled argument is **legally in** iff all its attackers are labeled out. An out–labeled argument is **legally out** iff at least one its attacker is labeled in. An $undec$–labeled argument is **legally undec** iff not all of its attackers are labeled out and it does not have an attacker that is labelled in.

By $in(Lab), out(Lab)$ and $undec(Lab)$ we will denote the arguments mapped respectively to in, out and $undec$ by Lab. We will also write a labeling as a triple (I, O, U), where $I = in(Lab)$, $O = out(Lab)$ and $U = undec(Lab)$.

Definition 2.7. Let $F = (A, R)$ be a Dung's framework and Lab a three–valued labeling on A. Lab is:

- **admissible** in F iff each in–labeled argument is legally in and each out–labeled argument is legally out.

- **complete** in F if it is admissible and every $undec$–labeled argument is legally $undec$.

- **preferred** in F if it is complete and the set of arguments labeled in is maximal w.r.t. set inclusion.

- **grounded** in F if it is complete and the set of arguments labeled in is minimal w.r.t. set inclusion.

[2]Sometimes the $\mathbf{t}, \mathbf{f}$ and $\mathbf{u}$ notation is also used.

- **semi–stable** in F if it is complete and the set of elements mapped to $undec$ is minimal w.r.t. set inclusion.
- **stable** in F if it is complete and the set of elements mapped to $undec$ is empty.

The correspondence between the labeling–based and extension–based has already been studied in [3, 14].

Theorem 2.8. *Let $F = (A, R)$ be a Dung's framework and $E \subseteq A$ be a σ–extension of F, where $\sigma \in \{$admissible, complete, grounded, preferred, stable, semi–stable$\}$. Then $(E, E^+, A \setminus (E \cup E^+))$ is a σ–labeling of F.*

Let Lab be a σ–labeling of F, where $\sigma \in \{$admissible, complete, grounded, preferred, stable, semi–stable $\}$. Then $in(Lab)$ is a σ–extension of F.

Remark. Depending on the semantics, there can be more than one labeling corresponding to a given extension. Let E^- be the set of arguments that attack E. Obviously, E defends its members iff $E^- \subseteq E^+$. Therefore, for a labeling to be admissible it suffices that the set of out arguments contains E^-; on the other hand, due to legality it cannot map more than E^+. This gives us a certain freedom in assignments. On the other hand, for example stable semantics possesses a one to one correspondence between the labelings and extensions.

Finally, we would like to recall several important lemmas and theorems from the original paper on AFs [23]. The so–called Fundamental Lemma is as follows:

Lemma 2.9. ***Dung's Fundamental Lemma*** *Let $F = (A, R)$ be a Dung's framework, E an admissible extension of F and $a, b \in A$ arguments that are defended by E in F. Then the set $E' = E \cup \{a\}$ is admissible in F and b is defended by E' in F.*

We can now recall some relations between the existing semantics.

Theorem 2.10. *Let $F = (A, R)$ be a Dung's framework. Every stable extension of F is a preferred extension, but not vice versa.*

Theorem 2.11. *Let $F = (A, R)$ be a Dung's framework. The following holds:*

1. *The set of all admissible sets of F form a complete partial order w.r.t. set inclusion.*
2. *For each admissible set E of F, there exists a preferred extension E' of F s.t. $E \subseteq E'$.*
3. *F possesses at least one preferred extension.*

Theorem 2.12. *Let $F = (A, R)$ be a Dung's framework. The following holds:*

1. *Every preferred extension of F is a complete extension, but not vice versa.*
2. *The grounded extension of F is the least w.r.t. set inclusion complete extension.*

3. The complete extensions of F form a complete semilattice w.r.t. set inclusion. [3]

Example 1. Consider the Dung framework $F = (A, R)$ with $A = \{a, b, c, d, e\}$ and the attack relation $R = \{(a, b), (c, b), (c, d), (d, c), (d, e), (e, e)\}$, as depicted in Figure 1. It has eight conflict–free extensions in total, namely $\{a, c\}$,$\{a, d\}$, $\{b, d\}$, $\{a\}$, $\{b\}$, $\{c\}$, $\{d\}$ and $\emptyset$. As b is attacked by an unattacked argument, it cannot be defended against it and will not be in any admissible extension. From this $\{a, c\}$, $\{a, d\}$ and $\{a\}$ are complete. We end up with two preferred extensions, $\{a, c\}$ and $\{a, d\}$. However, only $\{a, d\}$ is stable, and $\{a\}$ is the grounded extension.

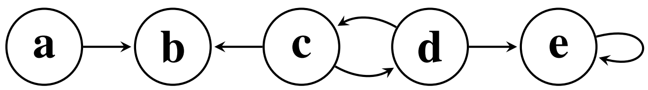


Figure 1: Sample Dung framework

2.2 Argumentation Frameworks with Support

Although the Dung's framework is a powerful tool, it has its shortcomings. Having only a binary attack at hand limits what can be modeled naturally, and what requires additional modifications which can make the representation of a problem and verifying the answer more complicated. Not surprisingly, this framework has been generalized in various ways in order to address its deficiencies (an overview can be found in [19]). In the context of this report, the enrichments that permit new types of relations are the most interesting.

Although many studies focused on developing the attack relation, with time it was acknowledged that a positive interaction between arguments beyond defense also needs to be expressed. Initially, there was hope that since Dung's framework has one abstract attack, one type of support would be sufficient [16]. However, various arguments and examples against this claim have been given, and more specialized forms of support have been researched. Currently the most recognized frameworks following the Dung representation are the Bipolar Argumentation Framework BAF [18], Argumentation Framework with Necessities AFN [32] and Evidential Argumentation System EAS [34]. The approaches towards modeling support can be classified in two ways. First of all we have the BAF style, more in line with meta–argumentation, where we can create coalition arguments or, depending on the type of positive relation that is used, we derive advanced conflicts and evaluate the resulting framework in a Dung manner. Although this study does not discuss certain problems of a bipolar setting such as support cycles, it provides a valuable insight into the consequences of using positive relations. The other approach, more visible in AFNs and EASs, treats support as a fully valued interaction and adapts semantics in an appropriate manner, rather than trying to translate the structure back into the Dung setting. We will briefly recall these frameworks and although their translation into ADFs is a matter of ongoing work and not a topic we want to discuss in this report, the differences between the frameworks will further exemplify the directions of the semantics we have taken in ADFs.

[3] A partial order $(A, \leq)$ is a complete semilattice iff each nonempty subset of A has a glb and each increasing sequence of A has a lub.

2.2.1 Bipolar Argumentation Frameworks

The original bipolar argumentation framework BAF [16] studied a relation we will refer to as **abstract support**:

Definition 2.13. A **bipolar argumentation framework** is a tuple (A, R, S), where A is a set of **arguments**, $R \subseteq A \times A$ represents the **attack** relation and $S \subseteq A \times A$ the **support**. It is also assumed that $R \cap S = \emptyset$[4].

The biggest difference between this abstract relation and any other interpretation of support, or even conflict, is the fact that it did not affect the acceptability of an argument. By this, we understand that an argument did not require any form of support and was able to stand "on its own". The positive interaction was used to derive additional indirect forms of conflict, which were later used to enhance the semantics from the Dung setting. The first developed type was the "supported attack". Later, in [17] the *secondary attack* was also introduced (first referred to as diverted).

Definition 2.14. Let $BF = (A, R, S)$ be a BAF. An argument $a \in A$ **support attacks** argument $b \in A$, if there exists some argument $c \in A$ s.t. there is a sequence of supports from a to c (i.e. $aS...Sc$) and cRb. a **secondary attacks** b if there is some argument c s.t. $cS...Sb$ and aRc.

These additional notions are now used to form stronger version of known semantics. Please note that the definition of defense is the same as in the Dung setting (i.e. requires direct attack).

Definition 2.15. Let (A, R, S) be a BAF. A set of arguments $E \subseteq A$ is **+conflict–free** iff $\nexists a, b \in E$ s.t. a (directly or indirectly) attacks b. E is **safe** iff $\nexists b \in A$ s.t. b is at the same time (directly or indirectly) attacked by E and either there is a sequence of supports from an element of E to b, or $b \in E$. E is closed under S iff $\forall b \in E, a \in A$, if bSa then $a \in E$. Then E is:

- **d–admissible** in BF iff it is +conflict–free and defends all its elements
- **s–admissible** in BF iff it is safe and defends all its elements
- **c–admissible** in BF iff it is +conflict–free, closed for S and defends all its elements
- **d–/s–/c–preferred** in BF iff it is maximal w.r.t. set inclusion d–/s–/c–admissible
- **stable** in BF iff it is +conflict–free and $\forall b \notin E$, b is (directly or indirectly) attacked by E.

The weak dependency between an argument and its supporter led to the development of more specific interpretations, most notably the deductive, necessary and evidential support. The first one remained in the BAF setting, while the latter two were developed in different frameworks. We say that an argument a *deductively supports* b if acceptance of a implies the acceptance of b [10] and not acceptance of b implies non acceptance of a. Although originally used rather for coalitions and meta–argumentation purposes, it is also studied in a standard setting in [18]. The deductive behavior of support in BAFs is achieved by introducing another type of indirect conflict, namely the *mediated attack*. Further study also motivated the *super–mediated attack*.

[4] This requirement is dropped in later works [18].

Definition 2.16. Let $BF = (A, R, S)$ be a Dung's framework and $a, b, c \in A$. There is a **mediated attack** from a to b iff there is some argument c s.t. there is a sequence of supports from b to c and aRc. There is a **super–mediated attack** from a to b iff there is some argument c s.t. a direct or supported attacks c and b supports c.

Finally, it is easy to see that BAFs do not make any special acyclicity assumptions as to the support relation[5]. Thus, cyclic arguments are considered valid attackers that can be used both by us and by the opponent.

Example 2. Let $(\{a, b, c, d, e, f\}, \{(b, a), (c, d)\}, \{(b, c), (c, b), (d, f), (e, d)\})$ be the BAF depicted in Figure 2. We use normal arrow to denote attack, dashed for support and red ones for indirect attacks. We have in total four supported attacks in our framework. First of all, since c supports b which attacks a, c support attacks a. In a similar manner, b support attacks d. However, as both c and b are indirectly self–supporters, technically speaking (b, a) and (c, d) are also support attacks. We have one supported attack (c, f) and a single mediated one (c, e). Finally, there are two super–mediated attacks. Since the definition subsumes mediated attacks, the first one is again (c, e), though obtained in two ways; first due to direct attack, the other due to the supported one (c, d). As b support attacks d, we have that b super–mediated attacks e.

We can observe that $\{b, d\}$ is +conflict–free, safe and closed under support. Moreover, since none of its elements is attacked, it follows easily that the set is d–/s–/c–admissible. Sets $\{b\}$ and $\{c\}$ are also +conflict–free and safe, however, they are not closed under support. Consequently, they will be d– and s–admissible, but not c–admissible. Set $\{f\}$ is also d–/s–/c–admissible. It is +conflict–free, safe and closed for support. Moreover, since defense only considers direct attacks, there is no argument f should defend from. Similarly, $\{e\}$ is d– and s–admissible. Since it is not closed for support, it cannot be c–admissible. Finally, $\{d, e\}$ is not admissible at all, as it cannot defend d from c.

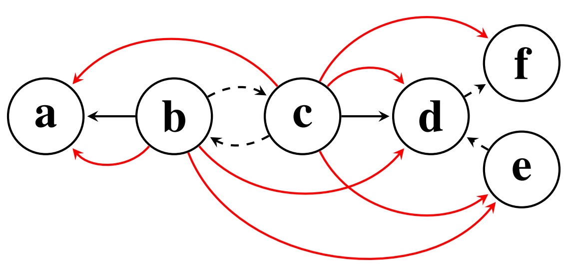


Figure 2: Sample BAF

2.2.2 Argumentation Frameworks with Necessities

The necessary support in its binary form was first developed in [33]. We say that an argument a *necessary supports* b if we need to assume a in order to accept b. The developed semantics were

[5]Only in the case of stable semantics the framework is assumed to be acyclic

built around the supported and secondary attacks and discarded any support cycles. However, they not always returned intended results. Therefore, we would like to focus on the more recent formulation that was presented in [32], this time with a set form of support.

Definition 2.17. An **argumentation framework with necessities** is a tuple (A, R, N), where A is the set of **arguments**, $R \subseteq A \times A$ represents (binary) **attacks**, and $N \subseteq (2^A \setminus \emptyset) \times A$ is the **necessity relation**.

Given a set $E \subseteq A$ and an argument a, ENa should be read as "at least one element of E needs to be present in order to accept a". Thus, we can consider an arbitrary set of arguments to sufficiently support a iff every set supporting a through N has at least one element in common with it. The AFN semantics are built around the notions of coherence:

Definition 2.18. Let $FN = (A, R, N)$ be an AFN and $E \subseteq A$ a set of arguments. E is **coherent** iff every $b \in E$ is powerful, i.e. there exists a sequence $a_0, .., a_n$ of distinct elements of E s.t $a_n = b$, there is no $C \subseteq A$ s.t. CNa_0, and finally for $1 \leq i \leq n$ it holds that for every set $C \subseteq A$ if CNa_i, then $C \cap \{a_0, ..., a_{i-1}\} \neq \emptyset$. A coherent set E is **strongly coherent** iff it is conflict–free.

Although it may look a bit complicated at first, the definition of coherence grasps the intuition that we need to provide sufficient acyclic support for the arguments we want to accept. Defense in AFNs is understood as the ability to provide support and to counter the attacks from any coherent set. Using these notions, the AFN semantics are built in a way corresponding to Dung semantics.

Definition 2.19. Let $FN = (A, R, N)$ be an AFN. A set of arguments $E \subseteq A$ **defends** a, if $E \cup \{a\}$ is coherent and for every $c \in A$, if cRa then for every coherent set $C \subseteq A$ containing c, ERC. The set of arguments **deactivated** by E is defined as $E^+ = \{a \,|\, ERA \textit{ or there is } B \subseteq A \textit{ s.t. } BNa \textit{ and } B \cap E = \emptyset\}$. The set E is:

- **admissible** in FN iff it is strongly coherent and defends all of its arguments.
- **preferred** in FN iff it is maximal w.r.t. set inclusion admissible.
- **complete** in FN iff it is admissible and contains any argument it defends.
- **stable** in FN iff it is complete and $E^+ = A \setminus E$.

It is easy to see that, through the notion of coherency, AFNs discard cyclic arguments both on the "inside" and the "outside". This means we cannot accept them in an extension and they are not considered as valid attackers.

Example 3. Consider an AFN $(\{a, b, c, d, e, f\}, \{(a, e), (d, b), (e, c), (f, d)\}, \{(\{b, c\}, a), (\{f\}, f)\})$ depicted in Figure 3. The coherent sets include $\emptyset, \{a, b\}, \{a, c\}, \{b\}, \{c\}, \{d\}, \{e\}$ and any of their combinations. We can observe that f does not appear in any of them - it does not possess a powerful sequence in the framework. The strongly coherent sets are $\emptyset, \{b\}, \{c\}, \{d\}, \{e\}$, $\{a, b\}, \{a, c\}$, $\{b, c\}, \{b, e\}, \{c, d\}, \{d, e\}$, $\{a, b, c\}$ and $\{a, c, d\}$. $\emptyset$ is trivially admissible. So is $\{d\}$, due to the fact that its only attacker does not posses a coherent set. However, $\{e\}$ is not admissible; it does not attack one of the coherent sets of a, namely $\{a, b\}$. On the other $\{d, e\}$ is already admissible. Due to the fact that no coherent argument can attack d, no strongly coherent set containing b will be admissible. The two final extensions are $\{a, c\}$ and $\{a, c, d\}$; although c is supporting a and a attacks e, the indirect conflict between c and a is not enough to consider c as defending itself in the AFN terms. The sets $\{d\}$, $\{d, e\}$ and $\{a, c, d\}$ are our complete extensions, with the first one being grounded and the latter two being preferred. In this case, both $\{d, e\}$ and $\{a, c, d\}$ are stable.

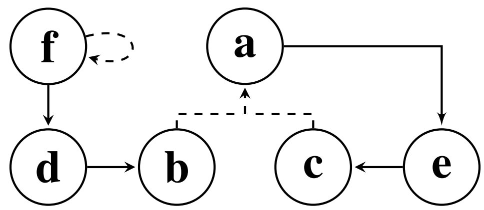


Figure 3: Sample AFN

2.2.3 Evidential Argumentation Systems

The last type of support we will consider here is the the *evidential support* [34]. It distinguishes between the standard and *prima facie* arguments. The latter are the only ones that are valid without any support. Every other argument that we want to accept needs to be supported by at least one prima facie argument, be it directly or not. While the acyclicity in the necessary support required us to trace back to either an attacker or an initial argument, the evidential support restricts this even further by allowing us to go back to only a subgroup of the initial arguments, marked as prima facie.

Definition 2.20. An **evidential argumentation system** (EAS) is a tuple (A, R, E) where A is a set of **arguments**, $R \subseteq (2^A \setminus \emptyset) \times A$ is the **attack relation**, and $E \subseteq (2^A \setminus \emptyset) \times A$ is the **evidential support**. We assume that $\nexists x \in 2^A, y \in A$ s.t. xRy and xEy. The prima facie arguments are represented with a single one $\eta \in A$ referred to as environment or evidence. Consequently, $\nexists (x, y) \in R$ where $\eta \in x$; and $\nexists x$ where $(x, \eta) \in R$ or $(x, \eta) \in E$.

Although both EAS and AFN use group support, the way the E and N relations are read is not the same. In the AFN case, a set of arguments sufficiently supported an argument $a \in A$ if every set supporting a through N had at least one element in common with it. In the EAS case, its quite the opposite – we would say that the set sufficiently supported a if it fully contained at least one set supporting a through E. The idea that the valid arguments (and attackers) need to trace back to the environment is captured with the notions of e–support and e–supported attack, which can be formulated in the recursive or sequence–based manner [38].

Definition 2.21. Let $ES = (A, R, E)$ be an EAS. An argument $a \in A$ has **evidential support** (e–support) from a set $X \subseteq A$ iff:

1. $a = \eta$; or

2. There is a non-empty $T \subseteq X$ such that TEa and $\forall x \in T$, x has evidential support from $X \backslash \{a\}$

An argument a is minimally evidentially supported by (or has minimal evidential support from) a set X if there is no set X' such that $X' \subset X$ and a is evidentially supported by X'.

Remark. Note that by this definition η has evidential support from any set.

Definition 2.22. Let $ES = (A, R, E)$ be an EAS. Given a set of arguments $X \subseteq A$, an **evidential sequence** for an argument $a \in X$ is a sequence of distinct elements of X $(a_0, .., a_n)$ s.t. $a_n = a$, $a_0 = \eta$, and if $n > 0$, then $\forall_{i=1}^{n}$ there exists a nonempty $T \subseteq \{a_0, ..., a_{i-1}\}$ s.t. TEa_i.

Theorem 2.23. *Let $ES = (A, R, E)$ be an EAS, $X \subseteq A$ be a set of arguments and $a \in A$. a is e–supported by X iff there exists an evidential sequence for a on $X \cup \{a\}$.*

Definition 2.24. Let $ES = (A, R, E)$ be an EAS. A set $X \subseteq A$ carries out an **evidence supported attack** (e–supported attack) on an argument $a \in A$ iff $(X', a) \in R$ where $X' \subseteq X$, and for all $x \in X'$, x has evidential support from X.

We can now continue with EAS semantics.

Definition 2.25. Let $ES = (A, R, E)$ be an EAS. An argument $a \in A$ is **acceptable** with respect to a set of arguments $X \subseteq A$ iff

- a is evidentially supported by X; and
- given a minimal e–supported attack by a set $T \subseteq A$ against a, it is the case that X carries out an e–supported attack against a member of T.

Definition 2.26. Let $ES = (A, R, E)$ be an EAS. A set of arguments $X \subseteq A$ is:

- **self–supporting** in ES iff all arguments in X are e–supported by X.
- **conflict–free** in ES iff there is no $a \in X$ and $X' \subseteq X$ such that $X'Ra$.
- **admissible** in ES iff it is conflict–free and all elements of X are acceptable w.r.t. X.
- **preferred** in ES iff it is maximal w.r.t. set inclusion admissible.
- **complete** in ES iff it is admissible and all arguments acceptable w.r.t. X are in X.
- **stable** in ES iff it is conflict–free, self–supporting, and for any argument a e–supported by A where $a \notin X$, X e–support attacks either a or every set of arguments minimally e–supporting a.

Definition 2.27. Let $ES = (A, R, E)$ be a finitiary EAS. The **characteristic function** $F_{ES} : 2^A \rightarrow 2^A$ of ES is defined as: $F_{ES}(X) = \{a \mid a \text{ is acceptable w.r.t. } X \text{ in } ES\}$. The **grounded extension** of a ES is the least fixed point of F_{ES}.

From the fact that every valid argument needs to be grounded in the environment it clearly results that EAS semantics are acyclic both on the inside and outside. In a certain sense this requirement is even stronger than in AFNs, as one is allowed to come back to only a single special argument rather than any initial one.

Example 4. Let $(\{\eta, a, b, c, d, e, f\}, \ \{(\{b\}, a), (\{b\}, c), (\{c\}, b), (\{c\}, d), (\{d\}, f), \ (\{f\}, f)\}, \{(\{\eta\}, b), (\{\eta\}, c), (\{\eta\}, d), (\{\eta\}, f), (\{d\}, e)\})$ be the EAS depicted in Figure 4. The admissible extensions are $\emptyset, \{\eta\}, \{\eta, b\}, \{\eta, c\}, \{\eta, b, d\}$ and $\{\eta, b, d, e\}$, with $\{\eta\}, \{\eta, c\}$ and $\{\eta, b, d, e\}$ being the complete ones. Obviously, the latter two are preferred. However, only $\{\eta, b, d, e\}$ is stable. Since a is not a valid argument (it is not e–supported in the framework), we do not have to attack it. Although $\{\eta, c\}$ attacks b and d (and by this, also e), it is not in any way in conflict with f. The grounded extension is just $\{\eta\}$.

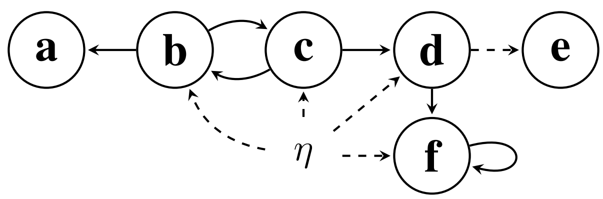


Figure 4: Sample EAS

2.3 Abstract Dialectical Frameworks

Abstract dialectical frameworks have been defined in [13] and till today various results as to their semantics, instantiation and complexity have already been published in [11, 39, 42–44]. The main goal of ADFs is to be able to express arbitrary relations and avoid the need of extending AFs by a new relation sets each time they are needed. This is achieved by the means of the so–called acceptance conditions. They define what sets of arguments related to a given argument should be present for it to be accepted or rejected.

Definition 2.28. An **abstract dialectical framework** (ADF) as a tuple (A, L, C), where A is a set of abstract **arguments** (nodes, statements), $L \subseteq S \times S$ is a set of **links** (edges) and $C = \{C_s\}_{s \in S}$ is a set of **acceptance conditions**, one condition per each argument.

Originally, the acceptance conditions were defined in terms of functions:

Definition 2.29. Let $par(s)$ denote the set of **parents** of an argument s; it consists of those $p \in A$ for which *(p,s)* $\in L$. Then an **acceptance condition** is given by a total function $C_s : 2^{par(s)} \rightarrow \{in, out\}$.

Within ADFs, we distinguish a particular subclass called bipolar. It is particularly valuable due to the fact bipolar ADFs, BADFs for short, appear to be of lower complexity than general ones [44]. We will recall them in order to motivate some of our naming choices in this paper.

Definition 2.30. Let $D = (A, L, C)$ be an ADF. A link $(r, s) \in L$ is:

- supporting iff for no $R \subseteq par(s)$ we have that $C_s(R) = in$ and $C_s(R \cup \{r\}) = out$
- attacking iff for no $R \subseteq par(s)$ we have that $C_s(R) = out$ and $C_s(R \cup \{r\}) = in$

An ADF is bipolar iff it contains only links that are supporting or attacking.

Remark. Please note that links can be both attacking and supporting (in which they are also often called redundant), or neither – ADFs are able to express more than attack and support.

Alternatively, one can also represent the acceptance conditions by propositional formulas over arguments instead of "boolean" functions [25]. Please note that links represent just connections between arguments, the burden of saying what is the nature of this connection falls to the acceptance conditions. Moreover, the parents of an argument can be easily extracted from the conditions and since we will not need BADFs through the rest of the paper, from now on we will use the shortened notation $D = (A, C)$. In order to introduce our new semantics, we need to explain some basic notions first.

3 Building Blocks of ADF Semantics

In this section we will introduce the concepts on which the semantics of ADFs are built. While the recap on interpretations will be relevant both to extension and labeling–based semantics, further sections will be required mostly for the former and for understanding the relation between the two approaches.

3.1 Interpretations

Interpretations will be equally important both in labeling and extension–based semantics. While in the first case interpretations will be returned instead of sets of arguments, in the latter they will be used to store accepted and rejected arguments in order to determine their acceptability.

A two (three–valued) interpretation is simply a mapping that assigns truth values (respectively $\{\mathbf{t}, \mathbf{f}\}$ and $\{\mathbf{t}, \mathbf{f}, \mathbf{u}\}$) to arguments. We will be making use both of partial (i.e. defined only for a subset of A) and full ones. The truth values can be compared with respect to truth ordering, i.e. $\mathbf{f} \leq_t \mathbf{u} \leq_t \mathbf{t}$, or precision (information) ordering: $\mathbf{u} \leq_i \mathbf{t}$ and $\mathbf{u} \leq_i \mathbf{f}$. The latter will be used in the context of labeling semantics. The pair $(\{\mathbf{t}, \mathbf{f}, \mathbf{u}\}, \leq_i)$ forms a complete meet–semilattice with the meet operation $\sqcap$ assigning values in the following way: $\mathbf{t} \sqcap \mathbf{t} = \mathbf{t}$, $\mathbf{f} \sqcap \mathbf{f} = \mathbf{f}$ and $\mathbf{u}$ in all other cases. It can naturally be extended to interpretations: given two interpretations v and v' on A, we say that v' contains more information, denoted $v \leq_i v'$, iff $\forall_{s \in A}\, v(s) \leq_i v'(s)$. Similar follows for the meet operation. In case v is three and v' two–valued, we say that v' extends v. This means that elements mapped originally to $\mathbf{u}$ are now assigned either $\mathbf{t}$ or $\mathbf{f}$. The set of all two–valued interpretations extending v is denoted $[v]_2$.

Example 5. Let $v = \{a : \mathbf{t}, b : \mathbf{t}, c : \mathbf{f}, d : \mathbf{u})$ be a three–valued interpretation. We have two extending interpretations, namely $v' = \{a : \mathbf{t}, b : \mathbf{t}, c : \mathbf{f}, d : \mathbf{t})$ and $v'' = \{a : \mathbf{t}, b : \mathbf{t}, c : \mathbf{f}, d : \mathbf{f})$. Clearly, it holds that $v \leq_i v'$ and $v \leq_i v''$. However, v' and v'' are incomparable w.r.t. $\leq_i$.

Let now $w = \{a : \mathbf{f}, b : \mathbf{f}, c : \mathbf{f}, d : \mathbf{t})$ be another three–valued interpretation. The meet of v and w gives us a new interpretation $w' = \{a : \mathbf{u}, b : \mathbf{u}, c : \mathbf{f}, d : \mathbf{u})$: as the assignments of a, b and d differ between v and w, the resulting value is $\mathbf{u}$. On the other hand, c is in both cases $\mathbf{f}$ and thus retains its value.

We will use v^x to denote a set of arguments mapped to x by v, where x a given truth–value.

3.2 Decisiveness

The notion of decisiveness is a key concept in our extension–based semantics for abstract dialectical frameworks. It can also be used to describe the behavior of the operator of labeling–based semantics, which will become more apparent when comparing the two approaches (see Section 7). While our idea uses interpretations, please note that the set form of decisiveness was also present in the original paper [13] in order to define the grounded semantics. We will compare the two versions at the end of this section.

Let us assume an ADF $D = (A, C)$. Given an acceptance condition C_s for some argument $s \in A$ and an interpretation v, we define a shorthand $v(C_s)$ as $C_s(v^{\mathbf{t}} \cap par(s))$. For a given propositional formula φ and an interpretation v defined over all of the atoms of the formula, $v(\varphi)$ will just stand for the value of the formula under v. However, apart from knowing the "current" value of an acceptance condition for some interpretation, we would also like to know if this interpretation is "final". By this we understand that no new information will cause the value to change. For example, given a condition $\varphi_s = a \land \neg b$ for some argument s dependent on a and b, knowing that b is true is enough to map φ_s to out in a way that no matter the value of a, it will always stay out. In order to verify whether our interpretation is decisive for some argument, we will explore how the interpretations "filling in" the missing values evaluate the argument's condition. We will refer to them as completions:

Definition 3.1. Let $D = (A, C)$ be an ADF, $E \subseteq A$ a set of arguments and v a two–valued interpretation defined on E. A **completion** of v to a set Z where $E \subseteq Z$, is an interpretation v' defined on Z in a way that $\forall a \in E\ v(a) = v'(a)$. v' is a $\mathbf{t}/\mathbf{f}$ completion of v iff all arguments in $Z \setminus E$ are mapped respectively to $\mathbf{t}/\mathbf{f}$.

Remark. We would like to draw the attention to the similarity between the concepts of completion and extending interpretation. Basically, given a three–valued interpretation v defined over A, the set $[v]_2$ corresponds precisely to the set of completions to A of the two–valued part of v. However, if we used the notion of an extension instead of a completion in a two–valued setting, it could be easily mistaken for the extension understood as set of arguments, not as an interpretation. Therefore, we will use our notation to avoid such collisions.

Definition 3.2. Let $D = (A, C)$ be an ADF, $E \subseteq A$ a set of arguments and v a two–valued interpretation defined on E. v is **decisive** for an argument $s \in A$ iff for any two completions $v_{par(s)}$

and $v'_{par(s)}$ of v to $E \cup par(s)$, it holds that $v_{par(s)}(C_s) = v'_{par(s)}(C_s)$. s is **decisively out/in** w.r.t. v if v is decisive and all of its completions evaluate C_s to respectively out, in.

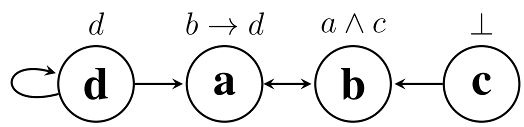


Figure 5: Sample ADF

Example 6. Let $(\{a, b, c, d\}, \{C_a : b \rightarrow d, C_b : a \wedge c, C_c : \bot, C_d : d\})$ be the ADF depicted in Figure 5. Example of a decisively in interpretation for a is $v = \{b : \mathbf{f}\}$. It simply means that knowing that b is false, no matter the value of d, the implication is always true and thus the acceptance condition is satisfied. From the more technical side, it is the same as checking that both completions of v to $\{b, d\}$, namely $\{b : \mathbf{f}, d : \mathbf{t}\}$ and $\{b : \mathbf{f}, d : \mathbf{f}\}$, satisfy the condition. Example of a decisively out interpretation for b is $v' = \{c : \mathbf{f}\}$. Again, it suffices to falsify one element of a conjunction to know that the whole formula will evaluate to false.

Remark. Please note that the existence of an interpretation that satisfies the acceptance condition of an argument a (i.e. there is a set of parents s.t. condition is in) implies the existence of a decisively in interpretation for a and vice versa. Moreover, if an argument is decisively out/in w.r.t. an interpretation, it holds that its acceptance condition is out/in. It basically results from the definition of a completion and decisiveness. Finally, if an argument is decisively in/out w.r.t. some interpretation, then it is decisively out w.r.t. any of its completions, not necessarily the ones that are defined for all parents.

Please note that although decisiveness in the interpretation form is more convenient for our purposes, the set version of this idea was already developed in the original paper [13] for the grounded semantics. Thus, one can choose between the representations depending on which one is more suitable. The set of arguments that were decisively in w.r.t. some set of accepted (G) and rejected (B) arguments was retrieved via the acc function. Similarly, reb produced a set of decisively out elements:

Definition 3.3. Let $D = (A, C)$ be an ADF and $G, B \subseteq A$ sets of arguments. Then $acc(G, B) = \{r \in A \mid G \subseteq A' \subseteq (A \backslash B) \rightarrow C_r(A' \cap par(s)) = in\}$ and $reb(G, B) = \{r \in A \mid G \subseteq A' \subseteq (A \backslash B) \rightarrow C_r(A' \cap par(s)) = out\}$.

We will now show that the set and interpretation approaches represent the same concept. Since we are interested in extensions, i.e. single status assignments to arguments, we can assume that $G \cap B = \emptyset$. Then we have that an argument $r \in A$ is in $acc(G, B)$, if for all possible subsets of arguments that contain the accepted ones (G) and not including any of the rejected ones (thus they can only be from $A \setminus B$) the acceptance condition is met. This is precisely checking if an argument is decisively in w.r.t. an interpretation v, where $v^{\mathbf{t}} = G$ and $v^{\mathbf{f}} = B$. Clearly, $reb(G, B)$ is just finding arguments that are decisively out w.r.t. v. We will come back to this representation when recalling the ADF grounded semantics.

3.3 Evaluations

Acceptance conditions tell us on what other arguments a given argument depends. We can see if they need to be accepted or rejected for the condition to be in our out and derive a range of decisively in interpretation based on it. We can then focus on the arguments in the condition and investigate them in a similar manner and continue this process until we have a full picture telling us when, how, and if at all, the arguments can be accepted or rejected, if they can be derived from initial arguments, include cyclic dependencies and so on. To this end, we introduce the notions of positive dependency functions and evaluations.

Let us assume an ADF (A, C). Given an argument $s \in A$ and $x \in \{in, out\}$, by $min\_dec(x, s)$ we will the denote the set of minimal two–valued interpretations that are decisively x for s. By minimal we understand that both $v^{\mathbf{t}}$ and $v^{\mathbf{f}}$ are minimal w.r.t. set inclusion.

Definition 3.4. Let $D = (A, C)$ be an ADF and $E \subseteq A$ a set of arguments. A **positive dependency function** on E is a function pd assigning every argument $a \in E$ an interpretation $v \in min\_dec(in, a)$ s.t. $v^t \subseteq E$ or $\mathcal{N}$ for null iff no such interpretation can be found. The function is **sound** iff no argument is mapped to $\mathcal{N}$. pd is maximally sound on E iff it is a sound function on $E' \subseteq E$ and there is no sound positive dependency function pd' on E'', where $E' \subset E'' \subseteq E$, s.t. $\forall a \in E', pd(a) = pd'(a)$.

Definition 3.5. Let $D = (A, C)$ be an ADF, $X \subseteq A$ and pd_E^D a maximally sound positive dependency function of X defined over $E \subseteq X$. A **standard positive dependency evaluation** for an argument $e \in E$ in D based on pd_E^D is a pair (F, B), where $F \subseteq E$ is a set of arguments s.t. $e \in F$, and $\forall a \in F$, $pd_E^D(a)^{\mathbf{t}} \subseteq F$, and $B = \bigcup_{a \in F} pd_E^D(a)^{\mathbf{f}}$.

We will refer to F as the **pd–set** of the evaluation and to B as the **blocking set** of the evaluation.

Example 7. Let $(\{a, b, c, d, e\}, \{C_a : \bot, C_b : a \wedge c, C_c : d \wedge \neg e, C_d : d, C_e : \top\})$ be the ADF depicted in Figure 6. The argument a has no standard evaluation, as it possesses no decisively in interpretation to start with. Although the argument b has a decisively in interpretation $\{a : \mathbf{t}, c : \mathbf{t}\}$, it depends on a and thus there does not exist a sound pd–function from which we could construct an evaluation for b. For d we have a simple evaluation $(\{d\}, \emptyset)$, and based on it an evaluation $(\{c, d\}, \{e\})$ for c. Finally, e as an initial argument has a trivial evaluation $(\{e\}, \emptyset)$.

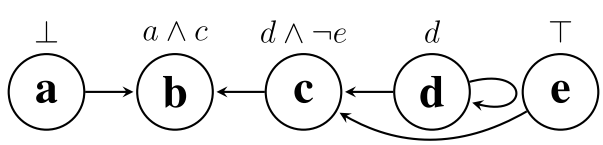


Figure 6: Sample ADF

While standard evaluations are already quite useful, we will also be interested in the more specialized types, dealing with the issue of so–called support cycles. In our case we will refer to them as positive dependency cycles, in order not to confuse them with certain definitions of support (evidential, necessary etc.) studied in other generalizations of the Dung's framework or with the support and attack links of bipolar abstract dialectical frameworks (see Section 2).

The informal understanding of a cycle is simply whether acceptance of an argument depends on this argument. A natural way to analyze this situation would be to "track" the evaluation of a given argument, e.g. in order to accept a we need to accept b, to accept b we need to accept c and so on. This simple case becomes more complicated when disjunction is introduced. We then receive a number of such "paths", some of them ending with cycles, some not. Moreover, they might be conflicting with each other, and we can have a situation where all acyclic evaluations are attacked and a cycle is forced. Our idea is to "unwind" the arguments and construct their evaluation paths, while still keeping in mind the arguments that they are in conflict with.

First of all, we will consider the partially acyclic evaluations. They can be seen as refinement of the standard ones, where the arguments are separated in to two groups; one that can be ordered into a sequence s.t. each argument depends only on the predecessors, and the other for which it is not possible, thus serving as a container for the cycles.

Definition 3.6. Let $D = (A, C)$ be an ADF, $X \subseteq A$ and pd_E^D a maximally sound positive dependency function of X defined over $E \subseteq X$.

A **partially acyclic positive dependency evaluation** based on pd_E^D for an argument $x \in E$ is a triple $(F, (a_0, ..., a_n), B)$, where $F \cap \{a_0, ..., a_n\} = \emptyset$, $(a_0, ..., a_n)$ is a sequence of distinct elements of E satisfying the requirements:

- if the sequence is non–empty, then $a_n = x$; otherwise, $x \in F$
- $\forall_{i=1}^{n}$, $pd_E^D(a_i)^{\mathbf{t}} \subseteq F \cup \{a_0, ..., a_{i-1}\}$, $pd_E^D(a_0)^{\mathbf{t}} \subseteq F$
- $\forall a \in F$, $pd_E^D(a)^{\mathbf{t}} \subseteq F$
- $\forall a \in F, \exists b \in F$ s.t. $a \in pd_E^D(b)$

Finally, $B = \bigcup_{a \in F} pd_E^D(a)^{\mathbf{f}} \cup \bigcup_{i=0}^{n} pd_E^D(a_i)^{\mathbf{f}}$, The sequence part of the evaluation will be referred to as the **pd–sequence**.

We can now introduce the last type of evaluations: the acyclic ones, being a subclass of partially acyclic.

Definition 3.7. Let $D = (A, C)$ be an ADF, $X \subseteq A$ and pd_E^D a maximally sound positive dependency function of X defined over $E \subseteq X$. A partially acyclic evaluation $(F, (a_0, ..., a_n), B)$ for an argument $x \in E$ is an **acyclic positive dependency evaluation** for x iff $F = \emptyset$.

Example 8. Let us come back to the framework $(\{a, b, c, d, e\}, \{C_a : \bot, C_b : a \wedge c, C_c : d \wedge \neg e, C_d : d, C_e : \top\})$ from Example 7 and Figure 6. The standard evaluation for e was $(\{e\}, \emptyset)$. Since e does not depend on any other argument, it can be easily moved into the pd–sequence and the partially acyclic representation of the standard evaluation is $(\emptyset, (e), \emptyset)$. This evaluation also happens to be acyclic. Although the evaluation for d looks similar, we can observe that the argument depends on itself, and thus the pd–sequence will be empty. The partial representation is thus $(\{d\}, (), \emptyset)$.

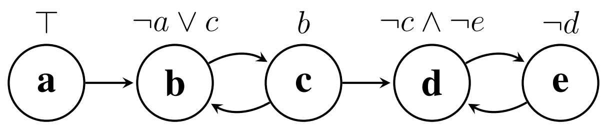


Figure 7: Sample ADF

Finally, let us look at the evaluation for c. The evaluation $(\{c, d\}, (), \emptyset)$ would not satisfy the partially acyclic requirements, since no argument in the pd–set depends on c. Consequently, we can "push" c into the sequence and obtain the evaluation $(\{d\}, (c), \{e\})$, which clearly shows where the actual cycle occurs. Neither c nor d possess acyclic evaluations.

We will use the shortened notation $((a_0, ..., a_n), B)$ in order to denote the acyclic evaluations.

We will say a standard evaluation (F, B) based on pd^D_E can be made acyclic for an argument $e \in F$ and w.r.t. pd^D_E iff there exists a way to order the elements of F into a sequence satisfying the pd–sequence requirements. It is also easy to see that any evaluation can be transformed into a standard one by joining the pd–set and the pd–sequence into a single pd–set.

We will say that an argument a is **pd–acyclic** on some set of arguments E iff there exist a pd–function on E and a corresponding acyclic pd–evaluation for a. Furthermore, we will simply write that an argument has an acyclic pd–evaluation on E if there is some pd–function on E from which we can produce the evaluation.

There are two ways we can "attack" an evaluation. Either we accept an argument that needs to be rejected in order for the evaluation to hold (i.e. it is in the blocking set), or we are able to discard an argument from the pd–sequence or the pd–set. This leads to the following, more abstract formulation: [6]

Definition 3.8. Let $D = (A, C)$ be an ADF and $(F, (a_0, ..., a_n), B)$ a partially acyclic evaluation on a set $E \subseteq A$ for an argument $a \in E$. A two–valued interpretation v defined on a subset of A **blocks** $(F, (a_0, ..., a_n), B)$ iff $\exists b \in B$ s.t. $v(b) = \mathbf{t}$ or $\exists x \in \{a_0, ..., a_n\} \cup F$ s.t. $v(x) = \mathbf{f}$.

Remark. An evaluation can be self–blocking, i.e. some members of the pd–sequence or the pd–set are present in the blocking set. Although an evaluation like that will never be accepted in an extension, it can make a difference in what we consider a valid attacker.

The idea of a pd–evaluation, especially an acyclic one, is strongly related to the concept of powerful and evidential sequences from AFNs and EASs (see Sections 2.2.2 and 2.2.3). The difference lies in the fact that in these frameworks, blocking an evaluation and attacking an argument are in precise correspondence. Since ADFs can also handle relations beyond attack and support, blocking a sequence might not always break conflict–freeness, and hence the blocking set needs to be stored. Let us now show this on an example:

[6] Since every standard evaluation can be made partially acyclic and every acyclic evaluation is also a partial one, we will only present the most general definition.

Example 9. Let $(\{a, b, c, d, e\}, \{C_a : \top, C_b : \neg a \lor c, C_c : b, C_d : \neg c \land \neg e, C_e : \neg d\})$ be the ADF in Figure 7. For argument b there exist two minimal decisively in interpretations: $v_1 = \{a : \mathbf{f}\}$ and $v_2 = \{c : \mathbf{t}\}$. The interpretations for a and c are respectively $w_1 = \{\}$ and $z_1 = \{b : \mathbf{t}\}$. Therefore, on $\{a, b, c\}$ we have two pd–functions, namely $pd_1 = \{a : w_1, b : v_1, c : z_1\}$ and $pd_2 = \{a : w_1, b : v_2, c : z_1\}$. They result in one acyclic evaluation for a: $((a), \emptyset)$, one for b: $((b), \{a\})$ and one for c: $((b, c), \{a\})$. Let us analyze the set $E = \{a, b, c\}$. We can see that accepting a "forces" a cycle between b and c. The acceptance conditions of all arguments are satisfied, thus this simple check is not enough to verify if a cycle occurs. If we checked only if the members of the pd–sequences are accepted, we would also get the wrong answer. Only looking at the whole evaluations shows us that b and c are both blocked by a. Although b and c are technically pd–acyclic in E, we see that their evaluations are in fact blocked and this type of conflict needs to be taken into account by the semantics.

We will close this section with a discussion on minimal interpretations and evaluations. First of all, the usage of only minimal decisively in interpretations in the construction of evaluations is purely our design choice. Allowing every type of interpretation would not affect our semantics; our interest lies in whether an evaluation of a given type exists or if all are blocked. Consequently, from an existing one we can always "remove" unnecessary elements from the blocking set in order to trim it to minimal interpretations. And if all evaluations are blocked, then so are the ones constructed with the minimal interpretations. Our choice to consider only minimal interpretations was motivated by practical and aesthetic reasons. Removing the minimality constraint caused an increase in the number of evaluations that could not be managed by our simple implementation used for testing. Moreover, we wanted the interpretations to reflect actual dependencies in the framework, which without minimality was not possible – an interpretation can, for example, be mapped for more arguments than just parents of a given argument.

In what follows we will also introduce the concept of minimal evaluations. This comes from the fact that not every evaluation may be of interest to us. For example, it may contain redundant elements on which the argument of interest does not really depend, or they may unnecessarily long. Let us consider an example.

Example 10. Let $(\{a, b, c, d\}, \{C_a : b \lor c, C_b : c, C_c : \top, C_d : \neg b \lor c\})$ be the ADF depicted in Figure 8. Let us focus on argument a and the following three acyclic evaluations for it: $((c, a), \emptyset)$, $((c, b, a), \emptyset)$ and $((d, c, a), \{b\})$. We can observe that a depends on c, but it can be reached either directly or though b. Although the "longer" part is perfectly fine, it can be seen as somewhat redundant due to the presence of the direct one. On the other hand, the $((d, c, a), \{b\})$ evaluation contains data useless for a – it includes the analysis of argument d, which is not related to a at all.

Let us now consider argument d and its evaluations. It possesses an acyclic one $((d), \{b\})$, which can also be changed into standard, and a purely standard one $(\{d\}, \emptyset)$. While the first one is created with the decisively in interpretation $\{b : \mathbf{f}\}$, the other with $\{d : \mathbf{t}\}$. If we were to consider minimal evaluations based only on subset relations between pd–sets and blocking sets, we can observe that the standard evaluation corresponding to $((d), \{b\})$ would have been "lost". Thus, in this approach a minimal evaluation of one type may not necessarily be a minimal one of another type. While it does not create problems if we are trying to answer the question if all

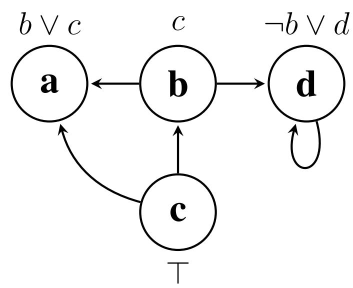


Figure 8: Sample ADF

standard evaluations of an argument are blocked, it can make a difference if we distinguish between types of evaluations. Consequently, a safe approach to minimality should take the pd–function into account.

We close this section by introducing the concept of minimal evaluations.

Definition 3.9. Let $D = (A, C)$ be an ADF and pd_E^D a positive dependency function on a set $E \subseteq A$. Let $a \in E$ and (F, B) a standard evaluation for $a \in E$. (F, B) is a minimal standard evaluation for a w.r.t. pd_E^D if there is no other standard evaluation (F', B') for a based on pd_E^D s.t. $F' \subseteq F$ and $B' \subseteq B$.

Let (G, B) be a pd–acyclic evaluation for $a \in E$ based on pd_E^D. (G, B) is a minimal pd–acyclic evaluation for a w.r.t. pd_E^D if there is no other pd–acyclic evaluation (G', B') for a based on pd_E^D s.t. $B' \subseteq B$ and G' is a subsequence of G.

Let (F, G, B) be a partially pd–acyclic evaluation for $a \in E$ based on pd_E^D. (F, G, B) is a partially minimal pd–acyclic evaluation for a w.r.t. pd_E^D if there is no other partially pd–acyclic evaluation (F', G', B') for a based on pd_E^D s.t. $B' \subseteq B$, $F' \subseteq F$ and G' is a subsequence of G.

3.4 Range

Just like in the Dung's framework, the concept of range and the E^+ set also appears in ADFs. The original definition from [35] required the notion of conflict–freeness. We will recall it here and later show that with the use of evaluations, we can drop the conflict–freeness assumption. For more explanations and examples concerning this semantics, please refer to Section 5.2.

Definition 3.10. Let $D = (A, C)$ be an ADF. A set of arguments $E \subseteq A$ is a **conflict–free extension** of D if for all $s \in E$ we have $C_s(E \cap par(s)) = in$. E is a **pd–acyclic conflict–free extension** of D iff for every argument $a \in E$, there exists an unblocked pd–acyclic evaluation on E w.r.t. v^E.

3.4.1 Standard Range

The basic concept of range is based on decisive outing. We start with arguments we can accept and then look for ones that are decisively outed by our choice. Since discarding one argument can also discard another that depends on it via a chain reaction, we repeat this search until no further arguments can be found.

Definition 3.11. Let $D = (A, C)$ be an ADF, $E \subseteq A$ a conflict–free extension of D and v_E a partial two–valued interpretation built as follows:

1. let $M = E$ and for every $a \in E$ set $v_E(a) = \mathbf{t}$;
2. for every argument $b \in A \setminus M$ that is decisively out w.r.t. v_E, set $v_E(b) = \mathbf{f}$ and add b to M;
3. now repeat the previous step until there are no new elements added to M.

By E^+ we understand the set of arguments $v_E^{\mathbf{f}}$ and we will refer to it as the **discarded set**. v_E now forms a **range interpretation** of E, where the usual range is denoted as E^R and equals $E \cup E^+$[7].

We can also redefine this notion by the use of standard evaluations, which limits the algorithm to a single iteration. Moreover, it allows us to find arguments decisively outed by a set of arguments without the conflict–freeness assumption.

Lemma 3.12. *Let $D = (A, C)$ be an ADF, $E \subseteq A$ a set of arguments and $X = \{a \in A \mid$ for every standard dependency evaluation (F, B) for a in D, $B \cap E \neq \emptyset\}$. If E is conflict–free, then $X = E^+$.*

Proof. First of all, let us notice that if E is conflict–free, then $E \cap X = \emptyset$. Since E is conflict–free, then for every argument $a \in E$ we can create a trivial decisively in interpretation which maps $E \cap par(a)$ to $\mathbf{t}$ and $par(a) \setminus E$ to $\mathbf{f}$. Thus, it is easy to see that the false part of the interpretation does not contain arguments appearing in E and that the positive part is fully in E. We can thus trivially construct an evaluation for any argument in the set with a blocking set disjoint from E. Consequently, no $a \in E$ will appear in X and we can produce an interpretation v_X s.t. elements of E are mapped to $\mathbf{t}$ and elements from X are mapped to $\mathbf{f}$.

Let us now assume that there is an argument which is in X, but not in E^+. This means that although every evaluation of an argument is blocked through the blocking set by E, the argument is not decisively out w.r.t. v_E. If an argument $a \in A$ is not decisively out, then there exists a completion v of v_E s.t. $C_a(v^{\mathbf{t}} \cap par(a)) = in$). It is easy to see that we can construct a trivial decisively in interpretation for a by mapping to $\mathbf{f}$ all arguments that are not yet assigned a value by v. Thus, a minimal one v_{min} also exists and is not "prevented" by v_E. Moreover, elements from $v_{min}^{\mathbf{t}}$ are not falsified by v_E and thus they could not have been decisively out w.r.t. it. As a result, we can find completions of v_E that satisfy their conditions and construct possible minimal decisively in interpretations for them, or use the ones we have already at hand in case given arguments have already been considered before or are contained in E. We can continue in this manner until our collection of interpretations produces a standard dependency evaluation which by construction has a blocking set disjoint with E. We reach a contradiction and a could not have been in X. Therefore, if an argument is in X, it is in E^+.

[7]It can be equivalently seen as $v_E^{\mathbf{t}} \cup v_E^{\mathbf{f}}$ or simply as the set of arguments for which v_E is defined.

Let us now assume that there is an argument $a \in A$ which is in E^+, but not in X. This means that even though a is decisively out w.r.t. v_E, there is an evaluation (F, B) s.t. $a \in F$ and $E \cap B = \emptyset$. Let us go through the range interpretation construction from Definition 3.11 and start with an interpretation v mapping and only elements of E to $\mathbf{t}$. Assume that a is already decisively out w.r.t. v. This means that either a possesses no set of arguments for which its condition is in, or there exists a minimal subset $E' \subseteq E \cap par(a)$ s.t. $\forall E'' \subseteq par(a), E' \subseteq E''$, $C_a(E'') = out$. If the first scenario is true, then obviously there is no decisively in interpretation for a and no evaluation. We thus reach a contradiction. If it is the latter, then every minimally decisively in interpretation for a has to falsify at least one argument in E''; otherwise, we can create its completion that will out the condition of a. Thus, any standard evaluation for a will have at least one member of its blocking set that is also in E. We reach a contradiction.

Let us thus assume that a was not decisively outed in the first iteration and let us continue with the second one. We have the interpretation v which maps to true members of E and for every argument mapped to false, either it does not possess a decisively in interpretation or every such interpretation has an argument mapped to $\mathbf{f}$ which is in E. Assume that a is decisively outed in this step. Since it did not happen in the first one, it means that there existed at least one completion of the initial v that evaluated the condition of a to in. Thus, we can still create at least one minimal decisively in interpretation for a related to these completions. Since no member mapped to false by any of these interpretations was contained in E, then it must have been the case that the v updated after the first iteration maps to $\mathbf{f}$ arguments that were assigned $\mathbf{t}$ by these decisively in interpretations. However, from this follows that a standard evaluation built for a with any of these interpretations contains arguments that we have outed in the previous step, whose false parts of the interpretations were not disjoint from E. Consequently, the blocking set of the evaluation would also not be disjoint with E. We reach a contradiction.

We can repeat this procedure until we reach the point in iteration that decisively outs a and come to the conclusion that a could not have possessed an evaluation (F, B) s.t. $B \cap E = \emptyset$. Thus, whatever argument is in E^+, it is also in X.

□

Example 11. Let us consider the framework $(\{a, b, c, d, e\}, \{C_a : a \wedge \neg b, C_b : a, C_c : \neg b, C_d : \neg a, C_e : d, C_f : f\})$ depicted in Figure 9 and focus on the conflict–free set $\{a\}$. We will now compute its standard range. First of all, the interpretation $v = \{a : \mathbf{t}\}$ decisively outs d. We update v and now have $\{a : \mathbf{t}, d : \mathbf{f}\}$. Our new interpretation now decisively outs e and we can extend it to $\{a : \mathbf{t}, d : \mathbf{f}, e : \mathbf{f}\}$. No further arguments can be falsified, as for both b and c the conditions are in w.r.t. $\{a\}$ and even though the condition of f is for now out, a completion of v mapping f to $\mathbf{t}$ can make it in. Let us now compute the standard range in the evaluation manner. For b we have an evaluation $(\{a, b\}, \{b\})$, for c $(\{c\}, \{b\})$, for d $(\{d\}, \{a\})$, $(\{d, e\}, \{a\})$ for e and finally $(\{f\}, \emptyset)$ for f. We can observe that only the evaluations for d and e are blocked by $\{a\}$.

3.4.2 Acyclic Range

The notions of the discarded set and the range are quite strong in the sense that they require an explicit "attack" on arguments that take part in dependency cycles. This is not always a desirable

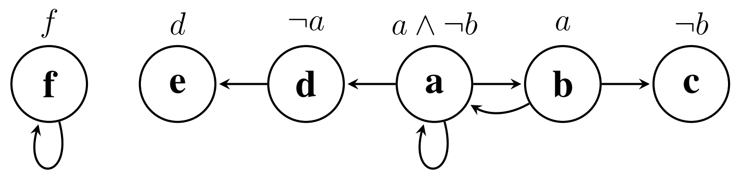


Figure 9: Sample ADF

property. Depending on the approach we might not treat cyclic arguments as valid and hence want them "out of the way".

Definition 3.13. Deprecated Let $D = (A, C)$ be an ADF, $E \subseteq A$ a conflict–free extension of D and v^a_E a partial two–valued interpretation built as follows:

1. Let $M = E$. For every $a \in M$ set $v^a_E(a) = \mathbf{t}$.
2. For every argument $b \in A \setminus M$ s.t. every acyclic pd–evaluation of b in A is blocked by v^a_E, set $v^a_E(b) = \mathbf{f}$ and add b to M.
3. Repeat the previous step until there are no new elements added to M.

By E^{a+} we understand the set of arguments mapped to **f** by v^a_E and refer to it as **acyclic discarded set**. We refer to v^a_E as **acyclic range interpretation** of E.

Lemma 3.14. *Let $D = (A, C)$ be an ADF, $E \subseteq A$ be a pd–acyclic conflict–free extension of D, v^a_E its acyclic range interpretation and $a \in A$ an argument s.t. it has at least one pd–acyclic evaluation $((a_0, ..., a_n), B)$ on A. The interpretation v^a_E blocks the evaluation iff $E \cap B \neq \emptyset$.*

Proof. If $E \cap B \neq \emptyset$, then the evaluation $((a_0, ..., a_n), B)$ for a is blocked by the definition. Let us thus focus on the other way around. Assume that v^a_E blocks $((a_0, ..., a_n), B)$, but $E \cap B = \emptyset$. This means that the blocking can occur only by falsifying a member of the pd–sequence. Let a_j be the first member of the pd–sequence falsified by the v^a_E and let v and v' be the stages of building the range interpretation before and after a_j is falsified. Assume that $j = 0$. By the pd–sequence requirements, the decisively in interpretation v_0 with which a_0 entered the evaluation consists only of false assignments. Thus, a_0 possesses a trivial pd–acyclic evaluation $((a_0), v^{\mathbf{f}}_0)$. For v' to map a_0 to false, v has to block all of its evaluation, including the trivial one. Since v does not map a_0 to false, it has to be the case that $v^{\mathbf{f}}_0 \cap v^{\mathbf{t}} \neq \emptyset$. This contradicts our assumption and thus it cannot be the case that $j = 0$. Let us continue with $j = 1$. The interpretation v_1 with which it entered the evaluation has a positive part consisting of at most a_0. We can create an evaluation $((a_0, a_1), v^{\mathbf{f}}_0 \cup v^{\mathbf{f}}_1)$ for a_1. Since a_0 is not falsified by v^a_E and a_1 is not yet falsified by v, it cannot be the case that v' falsified v_1 without a member of the blocking set being accepted. Again we reach a contradiction and conclude that it cannot be the case that $j = 1$. We can continue reasoning in this way until we reach a_n and we can thus conclude it could not have been falsified by the v^a_E unless a member of the blocking set was accepted. Thus, if an argument possessing an acyclic evaluation is falsified by range, then it cannot be the case that the range does not map to true at least one member of the blocking set. □

The analysis above brings us to a conclusion that the algorithm from the original definition of the acyclic range in fact terminates after first iteration. Consequently, we can rephrase it in the following way, similar to Lemma 3.12:

Lemma 3.15. *Let $D = (A, C)$ be an ADF, $E \subseteq A$ a set of arguments and $X = \{a \in A \mid$ for every pd–acyclic dependency evaluation (F, B) for a, $B \cap E \neq \emptyset\}$. If E is pd–acyclic conflict–free, then $X \cap E = \emptyset$. If E is conflict–free, then $X \setminus E = E^+$.*

Proof. If E is pd–acyclic conflict–free, then every argument $a \in E$ will have a pd–acyclic evaluation on E s.t. $E \cap B = \emptyset$. Consequently, no such a will qualify for X and $X \cap E = \emptyset$.

The equality between $X \setminus E$ and E^+ follows easily from Lemma 3.14.

□

Example 12. Let us come back to the framework $(\{a, b, c, d, e\}, \{C_a : a \wedge \neg b, C_b : a, C_c : \neg b, C_d : \neg a, C_e : d, C_f : f\})$ depicted in Figure 9 and described in Example 11. The standard range of set $\{a\}$ was $v = \{a : \mathbf{t}, d : \mathbf{f}, e : \mathbf{f}\}$. The evaluations for e and d can be made acyclic, and as their blocking sets contain a, it is easy to see that both of the arguments will also be falsified in the acyclic range. Since f possesses no acyclic evaluation, it will also be in the discarded set. Finally, the evaluation $(\{a, b\}, \{b\})$ for b cannot be made acyclic and the argument will be falsified for the same reason as f.

3.4.3 Partially Acyclic Range

The last type of range we will consider, the partially acyclic one, will be used in one family of our semantics. It can be seen as a certain middle ground between the standard and acyclic range. We discard the arguments if we block all of its pd–acyclic evaluations, unless it is based on a "cycle" that we are ready to accept. The value of this approach will become more apparent in Section 5.5.4.

Definition 3.16. Let $D = (A, C)$ be an ADF and $E \subseteq A$ a set of arguments. The **partially acyclic discarded set** of E is $E^{p+} = \{a \in A \mid$ for every pd–acyclic evaluation (F, B) for a, $B \cap E \neq \emptyset$ and there is no partially acyclic evaluation (F', G', B') for a s.t. $F' \subseteq E$ and $B' \cap E = \emptyset\}$.

Lemma 3.17. *Let $D = (A, C)$ be an ADF, $E \subseteq A$ a set of arguments and E^{p+} its partially acyclic discarded set. If E is conflict–free in D, then $E \cap E^{p+} = \emptyset$.*

Proof. Let a be an arbitrary argument in E. Since $a \in E$ and E is conflict–free, it is easy to see that we can construct a standard evaluation (F, B) for a on E s.t. $F \subseteq E$ and $B \cap E = \emptyset$. We can transform this evaluation into a partially acyclic one (F', G', B), where $F' \cup G' = F$. Clearly, $F' \subseteq E$ and $B \cap E = \emptyset$. Consequently, a could not have been in the partially acyclic discarded set and it follows that $E \cap E^{p+} = \emptyset$. □

Definition 3.18. Let $D = (A, C)$ be an ADF, $E \subseteq A$ a conflict–free extension of D and E^{p+} its partially acyclic discarded set. The **partially acyclic range** of E in the interpretation v_E^p mapping to $\mathbf{t}$ all and only arguments in E and mapping to $\mathbf{f}$ all and only arguments in E^{p+}.

Example 13. Let us come back to the framework $(\{a, b, c, d, e\}, \{C_a : a \wedge \neg b, C_b : a, C_c : \neg b, C_d : \neg a, C_e : d, C_f : f\})$ depicted in Figure 9 and described in Examples 11 and 12 The standard range of set $\{a\}$ was $v = \{a : \mathbf{t}, d : \mathbf{f}, e : \mathbf{f}\}$, while the acyclic one was $w = \{a : \mathbf{t}, b : \mathbf{f}, d : \mathbf{f}, e : \mathbf{f}, f : \mathbf{f}\}$. In the partially acyclic case, the arguments d, e and f will also be mapped to $\mathbf{f}$ by the range. However, even though argument b does not possess an acyclic evaluation, the partial representation $(\{a\}, (b), \{b\})$ of the standard one $(\{a, b\}, \{b\})$ has its pd–set contained in $\{a\}$. Consequently, the argument does not meet the partially acyclic range requirements.

4 Labeling–Based Semantics of ADFs

The two approaches towards labeling–based semantics of ADFs were developed in [11, 42]. They are based on the notion of a characteristic operator. While in the Dung setting the operator worked with sets, here three valued interpretations are used.

Definition 4.1. Let V_S be the set of all three–valued interpretations defined on S, s an argument in S and v an interpretation in V_S. The **three–valued characteristic operator** of D is a function $\Gamma_D : V_S \to V_S$ s.t. $\Gamma_D(v) = v'$ with $v'(s) = \bigsqcap_{w \in [v]_2} C_s(par(s) \cap w^{\mathbf{t}})$.

Remark. This operator working on three–valued interpretations is a more sophisticated version of the operator introduced in the original paper [13] and recalled in Section 5.4. This will become more visible when we describe the behavior of Γ_D in terms of decisiveness in Section 7.

Recall that verifying the value of an acceptance condition under a set of extensions of a three–valued interpretation $[v]_2$ is just like testing its value against the completions of the two–valued part of v. Thus, an argument that is assigned $\mathbf{t}$ by the $\Gamma_D(v)$ is decisively in w.r.t. the two–valued sub–interpretation of v. Similarly, one that is mapped to $\mathbf{f}$ is decisively out.

Remark. It is easy to see that in a certain sense this operator allows self–justification and self–falsification. Take for example a self–supporter; if we generate an interpretation in which it is false then, obviously, it will remain false. Same follows if we assume it to be true. This results from the fact that the operator functions on interpretations defined on all arguments, thus allowing a self–dependent argument to affect its status. The same is true if we consider bigger positive dependency cycles.

The labeling–based semantics are now as follows:

Definition 4.2. Let v be a three–valued interpretation for D and Γ_D its characteristic operator. We say that v is:

- **three–valued model** iff for all $s \in S$ we have that $v(s) \neq \mathbf{u}$ implies that $v(s) = v(\varphi_s)$;
- **admissible** iff $v \leq_i \Gamma_D(v)$;

- **complete** iff $v = \Gamma_D(v)$;
- **preferred** iff it is $\leq_i$–maximal admissible; and
- **grounded** iff it is the least fixpoint of Γ_D.

The stable semantics is a slightly different case. Although formally we receive a set, not an interpretation, this makes no difference for stability. As nothing is left undecided, there is a one–to–one correspondence between the extensions and labelings. The current state of the art definition, presented in [11, 42] is based on the grounded semantics:

Definition 4.3. Let M be a model of D and $D^M = (M, L^M, C^M)$ a reduct of D, where $L^M = L \cap (M \times M)$ and for $m \in M$ we set $C_m^M = \varphi_m[b/\mathbf{f} : b \notin M]$. Let gv be the grounded model of D^M. Model M is **stable** iff $M = gv^{\mathbf{t}}$.

Finally, the labeling based semantics preserve the properties known from the Dung's framework [11]:

Theorem 4.4. *Let $D = (A, C)$ be an ADF. The following holds:*

- *Each preferred labeling is a complete labeling, but not vice versa.*
- *The grounded model is the $\leq_i$–least complete labeling.*
- *The complete labelings of D form a complete meet–semilattice w.r.t. $\leq_i$.*

Example 14. We will now show the extensions of all of the semantics and their sub–semantics on an example. Let $(\{a, b, c, d\}, \{C_a : \neg b, C_b : \neg a, C_c : b \wedge \neg d, C_d : d\})$ be an ADF, as depicted in Figure 10. Its possible labelings are visible in Table 1. As there are over twenty possible three–valued models, we will not list them.

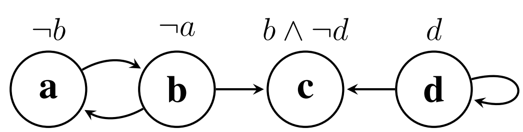


Figure 10: Sample ADF

5 Extension–Based Semantics of Abstract Dialectical Frameworks

In this section we will recall the existing extension–based ADF semantics and introduce new ones. Although various semantics for ADFs have already been defined in the original paper [13], only

Table 1: Labelings of the ADF from Figure 10.

ADM	$\{a:\mathbf{f},b:\mathbf{t},c:\mathbf{u},d:\mathbf{t}\},\{a:\mathbf{u},b:\mathbf{u},c:\mathbf{f},d:\mathbf{t}\},\{a:\mathbf{u},b:\mathbf{u},c:\mathbf{u},d:\mathbf{f}\},$ $\{a:\mathbf{f},b:\mathbf{t},c:\mathbf{f},d:\mathbf{t}\},\{a:\mathbf{f},b:\mathbf{t},c:\mathbf{t},d:\mathbf{f}\},\{a:\mathbf{t},b:\mathbf{f},c:\mathbf{u},d:\mathbf{u}\},$ $\{a:\mathbf{u},b:\mathbf{u},c:\mathbf{u},d:\mathbf{t}\},\{a:\mathbf{t},b:\mathbf{f},c:\mathbf{f},d:\mathbf{t}\},\{a:\mathbf{t},b:\mathbf{f},c:\mathbf{f},d:\mathbf{f}\},$ $\{a:\mathbf{f},b:\mathbf{t},c:\mathbf{u},d:\mathbf{u}\},\{a:\mathbf{t},b:\mathbf{f},c:\mathbf{u},d:\mathbf{t}\},\{a:\mathbf{f},b:\mathbf{t},c:\mathbf{u},d:\mathbf{f}\},$ $\{a:\mathbf{t},b:\mathbf{f},c:\mathbf{f},d:\mathbf{u}\},\{a:\mathbf{t},b:\mathbf{f},c:\mathbf{u},d:\mathbf{f}\},\{a:\mathbf{u},b:\mathbf{u},c:\mathbf{u},d:\mathbf{u}\}$
COMP	$\{a:\mathbf{t},b:\mathbf{f},c:\mathbf{f},d:\mathbf{f}\},\{a:\mathbf{f},b:\mathbf{t},c:\mathbf{u},d:\mathbf{u}\},\{a:\mathbf{f},b:\mathbf{t},c:\mathbf{f},d:\mathbf{t}\},$ $\{a:\mathbf{t},b:\mathbf{f},c:\mathbf{f},d:\mathbf{u}\},\{a:\mathbf{u},b:\mathbf{u},c:\mathbf{u},d:\mathbf{f}\},\{a:\mathbf{f},b:\mathbf{t},c:\mathbf{t},d:\mathbf{f}\},$ $\{a:\mathbf{t},b:\mathbf{f},c:\mathbf{f},d:\mathbf{t}\},\{a:\mathbf{u},b:\mathbf{u},c:\mathbf{f},d:\mathbf{t}\},\{a:\mathbf{u},b:\mathbf{u},c:\mathbf{u},d:\mathbf{u}\}$
PREF	$\{a:\mathbf{f},b:\mathbf{t},c:\mathbf{f},d:\mathbf{t}\},\{a:\mathbf{f},b:\mathbf{t},c:\mathbf{t},d:\mathbf{f}\},\{a:\mathbf{t},b:\mathbf{f},c:\mathbf{f},d:\mathbf{f}\},$ $\{a:\mathbf{t},b:\mathbf{f},c:\mathbf{f},d:\mathbf{t}\}$
STB	$\{a:\mathbf{t},b:\mathbf{f},c:\mathbf{f},d:\mathbf{f}\},\{a:\mathbf{f},b:\mathbf{t},c:\mathbf{t},d:\mathbf{f}\}$
GRD	$\{a:\mathbf{u},b:\mathbf{u},c:\mathbf{u},d:\mathbf{u}\}$

three of them – conflict–free, model and grounded (initially referred to as well–founded) – are still used (issues with the other formulations can be found in [11, 39, 42]). Moreover, the treatment of cycles and their handling by the semantics was not sufficiently developed. In this section we will address all of those issues. Before we continue, let us first motivate our choice on how to treat cycles. As we have shown in Section 2.2, the opinions on support cycles differ between the available frameworks. There is no consensus as to how they should be treated, as we can find examples both for and against their validity. Therefore, we would like to explore the possible approaches in the context of ADFs by developing appropriate semantics. In the next subsection, we will explain our classification system. We will then proceed with describing the surviving semantics and their new, acyclic versions. Afterwards we will introduce the families of admissible, preferred and complete semantics. The analysis of their properties and how the existing semantics fit into our system will be given in Section 6.

5.1 Classification

The classification of the sub–semantics that we will adopt in this paper is as follows. Bearing in mind the intuition we have presented in the introduction, appropriate semantics will receive an $xy-$ prefix, where $x, y \in \{a, c\}$. It will denote whether we demand acyclicity - a - or not - c - on the "inside" (x) and on the "outside" (y). As the conflict–free (and naive) semantics focus only on what we can accept, we will drop the prefixing in this case. Although the model, stable and grounded ones fit into our classification (more details can be found in Section 6), they have a sufficiently unique naming and further annotations are not necessary. We are thus left with admissible, preferred and complete. The BAF approach follows the idea that we can accept arguments that are not acyclic in our opinion and we allow our opponent to do the same. The ADF semantics we have developed in [39] also shares this view. Therefore, they will receive the $cc-$ prefix. On the other hand, AFN and EAS semantics do not permit cycles both in extensions and attackers. Consequently, the semantics following this line of reasoning will be prefixed with $aa-$.

We believe that also a non–uniform approach can be suitable in certain situations. By "non–uniform" we mean not accepting cyclic arguments, but still treating them as valid attackers and so on (i.e. $ca-$ and $ac-$). Imagine a case with a suspect, prosecutor and a jury. The suspect can utter a self–supporting argument such as "I'm telling the truth!", which expressed properly can convince the jury and raise doubt. The prosecutor has to disprove the suspects claim with sufficient evidence and a clear, acyclic chain of reasoning. Depending on whom we identify with, the requirements shift and hence we can have semantics that allow cycles on the "inside", but not on the "outside", and vice versa. Following this line of thought we introduce both uniform and non–uniform sub–semantics when required.

Remark. Please note that such non–uniform approaches can also be found in logic programming, one can for example compare the supported and stable models.

5.2 Conflict–free and Naive Semantics

In the Dung setting, conflict–freeness meant that the elements of an extension could not attack one another. This is also the common interpretation in various other AF generalizations, including the bipolar ones such as AFNs and EASs [32, 34]. Providing an argument with the required support is then a separate condition. In ADFs, where we lose the set representation of relations in favor of abstraction, not including "attackers" and accepting "supporters" is combined into one notion. This represents the intuition of "arguments that can stand together" presented in [5].

Definition 5.1. Let $D = (A, C)$ be an ADF. A set of arguments $E \subseteq A$ is a **conflict–free extension** of D if for all $s \in E$, $C_s(E \cap par(s)) = in$.

The acyclic version of conflict–freeness is a bit more than just a pd–acyclic set; we have to make sure that the evaluation is unblocked.

Definition 5.2. Let $D = (A, C)$ be an ADF. A conflict–free extension $E \subseteq A$ of D is a **pd–acyclic conflict–free extension** of D iff for every argument $a \in E$, there exists an unblocked pd–acyclic evaluation on E w.r.t. v^E.

Remark. As we are dealing with a conflict– free extension, all the arguments of a given pd–sequence are naturally $\mathbf{t}$ both in v_E and v_E^a. Therefore, in order to ensure that an evaluation is unblocked it suffices to check whether $E \cap B = \emptyset$. Consequently, in this case it does not matter w.r.t. which version of range we are verifying the evaluations.

Definition 5.3. Let $D = (A, C)$ be an ADF. The **naive** and **pd–acyclic naive** extensions of D are respectively maximal w.r.t. set inclusion conflict–free and pd–acyclic conflict–free extensions of D.

Example 15. Let us now look at the ADF $(\{a, b, c\}, \{C_a : \neg c \vee b, C_b : a, C_c : c\})$ depicted in Figure 11. The conflict–free extensions are $\emptyset$, $\{a\}$, $\{c\}$, $\{a, b\}$ and $\{a, b, c\}$. Since there exists no acyclic evaluation for c, it cannot appear in any pd–acyclic conflict–free extension. Thus, only $\emptyset$, $\{a\}$ and $\{a, b\}$ qualify for acyclic type. The naive and pd–acyclic naive extensions are respectively $\{a, b, c\}$ and $\{a, b\}$.

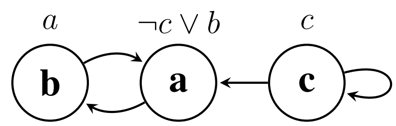


Figure 11: Sample ADF

5.3 Model and Stable Semantics

The concept of a model basically follows the intuition that if something can be accepted, it should be accepted:

Definition 5.4. Let $D = (A, C)$ be an ADF. A conflict–free extension $E \subseteq A$ of D is a **model** of D if $\forall\ s \in A,\ \ C_s(E \cap par(s)) = in$ implies $s \in E$.

Although this definition is simple, several of its properties should be explained. First of all, verifying whether a condition of an argument s is met does not check the effect of accepting s on E, thus it can happen that E is conflict–free, but $E \cup \{s\}$ is not. Let $(\{a, b\}, \{C_a : \neg a, C_b : \neg a\})$ be a simple ADF in which a is attacking both a and b. The set $\{b\}$ is conflict–free and the condition of a w.r.t. $\{b\}$ is in. Thus, $\{b\}$ is not a model. However, the set $\{a, b\}$ is no longer conflict–free; the conditions of both arguments evaluate to out. Since $\{a\}$ is trivially not conflict–free and the conditions of a and b are in w.r.t. $\emptyset$, we obtain no model for this framework. Consequently, we can see that model semantics is not universally defined.

On the other hand, a condition of an argument may be out not just due to the presence of undesired argument, but also due to the absence of needed ones. Consequently, an argument may be out w.r.t. a set of arguments E, but in w.r.t. E' s.t. $E \subset E'$. Thus, especially in the presence of positive dependency cycles, model extensions might be comparable w.r.t. $\subseteq$.

Finally, we would like to make a note concerning the arguments that are not included in a model. Informally speaking, there might be three reasons for them to be out. They are either inconsistent (i.e. the condition never mapped any set to in), are "attacked" by the set, or they cannot be accepted as at least one argument necessary for their acceptance was missing. The last case is especially is interesting; lack of support means two things – either we were able to trace back to an inconsistent or attacked argument, or we reached a positive dependency cycle. Looking at the model semantics from the "defense" perspective, we are either able to attack (or cut off the support of) our attacker, or the attacker is not valid due to a positive dependency cycle. This description clearly follows the idea of $ca-$ semantics; as we will show in Lemma 6.14 in Section 6, this is indeed the case.

The model semantics was used as a mean to obtain the stable models. The main idea was to make sure that the model is "acyclic". Unfortunately, the used reduction method was not adequate, as shown and fixed in [11]. The new method used the grounded reduct to ensure acyclicity. However, we will show that we can get the same result by enforcing pd–acyclic conflict–freeness of the models.

Theorem 5.5. *Let $D = (A, C)$ be an ADF and $E \subseteq A$ a set of arguments. E is pd–acyclic conflict–free in D iff it is the grounded extension of the reduct $D^E = (E, C^E)$ of D w.r.t. E.*

Proof. Let us first show that if E is pd–acyclic conflict–free in D, then it is grounded in D^E. Let E' the grounded extension of D^E and let a be an argument in E. Since E is pd–acyclic conflict–free in D, then E possesses a pd–acyclic evaluation $((a_0, ..., a_n), B)$ on E s.t. $B \cap E = \emptyset$. The decisively in interpretation v_{a_0} with which a_0 entered the evaluation consists only of false mappings. Thus, $C_{a_0}(\emptyset) = in$ and for any set of arguments $X \subseteq par(a_0)$ s.t. $C_{a_0}(B) = out$, $\exists b \in B$ s.t. $v_{a_0}(b) = \mathbf{f}$. As $\{a_0, ..., a_n\} \subseteq E$ and $E \cap B = \emptyset$, it follows that arguments from B (and thus from $v^{\mathbf{f}}_{a_0}$) do not appear in D^E. By the definition of the reduct it thus follows that for every subset X of parents of a_0 in D^E, $C^E_{a_0}(X) = in$. Consequently, a_0 will be mapped to $\mathbf{t}$ in the first iteration of the grounded algorithm from Proposition 5.9 and will appear in the grounded extension E' of D^E. Let us now focus on a_1 and its decisively in interpretation v_{a_1} with which it entered the evaluation. By the pd–sequence requirements, $v^{\mathbf{t}}_{a_1} \subseteq \{a_0\}$. If it is empty, we can repeat the analysis from a_0 and conclude it has to be contained in E'. If it is not, then by a similar construction it follows that for any subset X of parents of a_1 in D^E, if $a_0 \in X$ then $C^E_{a_1}(X) = in$. Consequently, a_1 will be mapped to $\mathbf{t}$ in the second iteration of the grounded algorithm from Proposition 5.9 and thus will appear in the grounded extension E' of D^E. We can repeat this procedure till we reach a_n and do it for any pd–acyclic evaluation on E for which the blocking set is disjoint from E. Thus, $E \subseteq E'$ and as $E' \subseteq E$ by the definition of D^E, the first direction of our proof is done.

Let us now show that if E is grounded in D^E, then it is pd–acyclic conflict–free in D. Let a be an argument in E. By the construction of the reduct, it follows that if $C^E_a(E \cap par(a)) = in$ in D^E, then $C_a(E \cap par(a)) = in$ in D. Thus, E is conflict–free in D. Let $((a_0, ..., a_n), B)$ be a pd–acyclic evaluation for a on E in D^E s.t. $E \cap B = \emptyset$. By the analysis above and the construction of the reduct it we can observe that if v_i is the decisively in interpretation with which an argument a_i in the pd–sequence entered the evaluation in D^E, then a_i possesses a decisively in interpretation v'_i in D s.t. ${v_i'}^{\mathbf{t}} = v^{\mathbf{t}}_i$ and $v^{\mathbf{f}}_i \subseteq {v_i'}^{\mathbf{f}}$. Since the false mappings present in v'_i but not in v_i only concern the arguments not present in D^E, the interpretation does not falsify any arguments in E. Thus, the original evaluation $((a_0, ..., a_n), B)$ for a on E in D^E can be transformed into an evaluation $((a_0, ..., a_n), B')$ for a on E in D s.t. $B \subseteq B'$ and $B' \cap E = \emptyset$. Hence, there exists an unblocked evaluation on E in D for an arbitrary $a \in E$ and E is conflict–free in D. We can thus conclude that E is pd–acyclic conflict–free in D. □

Straightforwardly from the above follows the redefinition of stability with pd–acyclicity:

Theorem 5.6. *Let $D = (A, C)$ be an ADF. A model $E \subseteq A$ of D is a* ***stable extension*** *of D iff it is pd–acyclic conflict–free.*

Finally, we can also obtain stable extensions in a manner similar to Dung's:

Lemma 5.7. *A set $E \subseteq A$ is stable iff it is a pd–acyclic conflict–free extension s.t. $E^{a+} = A \setminus E$.*

Proof. The fact that if E is stable extension, then it is pd–acyclic conflict–free and $E^{a+} = A \setminus E$ follows from the definition of stability and Lemma 6.3. Let us now assume that E is pd–acyclic conflict–free and $E^{a+} = A \setminus E$. In order to show that it is stable, we need to prove it is E is also a model. By Proposition 6.1, we know that every $a \in E^{a+}$ is decisively out w.r.t. v_E^a. Consequently, $C_a(E \cap par(a)) = out$. Thus, model requirements are satisfied and E is stable. □

Example 16. Let us come back to the ADF $(\{a, b, c\}, \{C_a : \neg c \vee b, C_b : a, C_c : c\})$ depicted in Figure 11 and described in Example 15. The conflict–free extensions were $\emptyset, \{a\}, \{c\}, \{a, b\}$ and $\{a, b, c\}$. The first two are not models: in the first case $C_a(\emptyset) = in$, while in the latter $C_b(\{a\}) = in$. Hence, the model condition is not satisfied. We can also observe that the models $\{c\}$ and $\{a, b\}$ are subsets of $\{a, b, c\}$. Thus, model extension are not necessarily incomparable w.r.t. $\subseteq$. Recall that $\emptyset, \{a\}$ and $\{a, b\}$ were the pd–acyclic conflict–free extensions. The only one that is also a model is $\{a, b\}$ and thus we obtain our single stable extension.

5.4 Grounded and Acyclic Grounded Semantics

The basic grounded semantics was already introduced in the original paper on ADFs [13]. Just like in the Dung setting, it preserves the unique–status property and is defined in the terms of a special operator:

Definition 5.8. Let $D = (A, C)$ be an ADF. Let $\Gamma'_D(G, B) = (acc(G, B), reb(G, B))$, where $acc(G, B) = \{r \in A \mid G \subseteq A' \subseteq (A \backslash B) \to C_r(A' \cap par(s)) = in\}$ and $reb(G, B) = \{r \in A \mid G \subseteq A' \subseteq (A \backslash B) \to C_r(A' \cap par(s)) = out\}$. Then E is the **grounded model** of D iff for some $E' \subseteq A$, (E, E') is the least fix–point of Γ'_D.

As we have explained in Section 3, acc and reb are nothing more than the means of retrieving decisively in and out arguments via a set representation. We are now interested in the least fixpoint of the operator, which as noted in [13] can be reached by iterating Γ'_D starting with $(G, B) = (\emptyset, \emptyset)$. It is easy to see that at all steps $G \cap B = \emptyset$: as the sets are initially disjoint, we can see it as an interpretation, and clearly no argument can be at the same time decisively in and out w.r.t. this interpretation. Therefore, we propose an alternative way to compute the grounded extension, in line with Proposition 2.5:

Proposition 5.9. *Let $D = (A, C)$ be an ADF and v an empty interpretation. For every argument $a \in A$ that is decisively in w.r.t. v, set $v(a) = \mathbf{t}$ and for every argument $b \in A$ that is decisively out w.r.t. v, set $v(b) = \mathbf{f}$. Repeat the procedure until no further assignments can be done. The* ***grounded extension*** *of D is then $v^{\mathbf{t}}$.*

We can observe that when it comes to "inside", the grounded semantics follows the acyclic approach. The arguments that are accepted first clearly have a decisively in interpretation without $\mathbf{t}$ mappings and every new iteration accepts arguments that positively depend at most on them. Thus, we can easily construct pd–acyclic evaluations that are "defended" by the interpretation. On the other hand, rejecting arguments follows the standard approach, thus grounded semantics can be classified as a member of the ac–family.

However, this is not the only way to look at the grounded semantics. Although the accepted arguments will always be acyclic, the rejection may not necessarily be standard. The semantics of frameworks such as AFNs and EASs (see Section 2) clearly point to the aa–approach. Consequently, we will introduce the acyclic grounded semantics, where the acyclicity in the name points to the "outside" restrictions:

Definition 5.10. Let $D = (A, C)$ be an ADF and v an empty interpretation. For every argument $a \in A$ that is decisively in w.r.t. v, set $v(a) = \mathbf{t}$. For every argument $b \in A$ s.t. all of its pd–acyclic evaluations are blocked by v, set $v(b) = \mathbf{f}$. Repeat the procedure until no further assignments can be done. The **acyclic grounded extension** of D is then $v^{\mathbf{t}}$.

We will now show possible grounded extensions on an example.

Example 17. Let us come back again to the ADF $(\{a, b, c\}, \{C_a : \neg c \vee b, C_b : a, C_c : c\})$ depicted in Figure 11. We will now try to find its grounded extension. Let v be an empty interpretation. The only argument that has a satisfied acceptance condition, and thus the chance to be decisively in, is a. However, it is easy to see that if we accept c, the condition is outed. Hence, we obtain no decisiveness in this case. Since b and c are both out, we can check if they have a chance to be decisively out. Again, condition of b can be met if we accept a, and condition of c if we accept c; as v does not define the status of a and c, we obtain no decisiveness again. Thus, $\emptyset$ is the grounded extension. Let us now try to find the acyclic grounded extension for this framework. Let v be an empty interpretation. As c does not possess any pd–acyclic evaluations, it is trivially mapped to $\mathbf{f}$. Falsifying c allows us to accept a, and based on a we can assume b. Consequently, our acyclic grounded extension will be $\{a, b\}$, not $\emptyset$ like in the standard case.

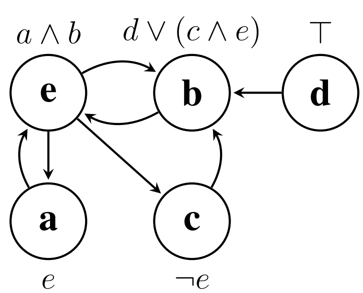


Figure 12: Sample ADF

Let us now look at the ADF $(\{a, b, c, d, e\}, \{C_a : e, C_b : d \vee (c \wedge e), C_c : \neg e, C_d : \top, C_e : a \wedge b\})$ depicted in Figure 12. Assume an empty interpretation v. It is easy to see that only d is decisively in w.r.t. v and that there are no decisively out arguments. However, now that we have $d : \mathbf{t}$ assignment, b can be also decisively assumed. Again, no decisive outing occurs, and next round returns us no new assignments. Thus, the grounded extension is $\{b, d\}$. When it comes to acyclic case, we can again trivially accept d. However, since a and e have no pd–acyclic evaluations, they are mapped to $\mathbf{f}$. By accepting d we can assume b, and from the rejection of e follows c. Consequently, our acyclic grounded extension will be $\{b, d, c\}$ and contains the standard one $\{b, d\}$.

5.5 Admissible, Preferred and Complete Semantics

In this section we will focus on admissible, preferred and complete semantics. We will describe them family after family. What is important to understand is the fact that even though there are significant differences between the approaches, the core concept remains the same – admissibility representing a defensible stand, preferred extensions being maximal admissible, and complete accepting whatever they defend. By replacing defense with decisiveness w.r.t. range, we basically obtain the ADF semantics. The differences lie in which range should be chosen, and if acyclicity of the extension is also desired.

5.5.1 CC Family

The basic admissible semantics was developed in [39]. It basically followed the intuition that we need to be able to discard any counterarguments of our opponent and made no acyclicity assumptions:

Definition 5.11. Deprecated: Let $D = (A, C)$ be an ADF. A conflict–free extension $E \subseteq A$ is **admissible** in D iff for any nonempty $F \subseteq A \setminus E$, if there exists an $a \in E$ s.t. $C_e(par(e) \cap (F \cup E)) = out$ then $F \cap E^+ \neq \emptyset$.[8]

The new simplified version of the previous formulation, taking into account our classification, is now as follows:

Definition 5.12. Let $D = (A, C)$ be an ADF. A conflict–extension $E \subseteq A$ is **cc–admissible** in D iff every argument in E is decisively in w.r.t. to the standard range interpretation v_E.

Definition 5.13. Let $D = (A, C)$ be an ADF. A cc–admissible extension $E \subseteq A$ is **cc–complete** in D iff every argument in E that is decisively in w.r.t. the range interpretation v_E is in E.

Definition 5.14. Let $D = (A, C)$ be an ADF. A set of arguments is a **cc–preferred** extension of D iff it is a maximal w.r.t. set inclusion cc–admissible extension of D.

Example 18. Let us assume an ADF $(\{a, b, c, d\}, \{C_a : \top, C_b : c \vee \neg a, C_c : b \vee \neg d, C_d : d\})$, as depicted in Figure 13. We can observe that even though the set $\{c\}$ is conflict–free, it is not cc–admissible. Its discarded set is empty and accepting d clearly outs the condition of c. However, $\{b, c\}$ is cc–admissible; although both a and d are not discarded, they do not affect the conditions of b and c further. Further cc–admissible sets include $\emptyset$, $\{a\}$, $\{d\}$, $\{a, d\}$, $\{a, b, c\}$, $\{b, c, d\}$ and $\{a, b, c, d\}$. The last one is clearly cc–preferred. The cc–complete extensions are $\{a\}$, $\{a, d\}$, $\{a, b, c\}$ and $\{a, b, c, d\}$.

[8]The new formulation is equivalent to this one and we see it as more elegant. However, we would like to recall this version to avoid confusion for readers familiar with our previous works.

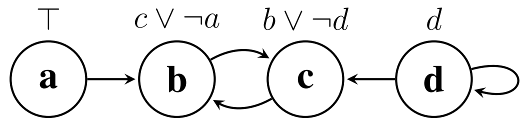


Figure 13: Sample ADF

5.5.2 AA Family

Let us now consider the opposite of cc–semantics, namely the aa–family. Although the pd–acyclicity and the usage of acyclic range are rather natural, recall that with semantics acyclic on the inside we had to deal with the "second" level of conflict visible in the blocking sets of acyclic evaluations. This gives rise to another level of "defense", where not only we check if arguments are decisively in w.r.t. range, but also need to protect their evaluations. Please consult Example 19 to see that decisiveness alone is not sufficient.

Definition 5.15. Let $D = (A, C)$ be an ADF. A pd–acyclic conflict–free extension $E \subseteq A$ of D is **aa–admissible** in D iff every argument in E is decisively in w.r.t. acyclic range interpretation v_E^a and for every member of the extension there exists an acyclic pd–evaluation $((a_0, ..., a_n), B)$ on E s.t. all members of B are mapped to **f** v_E^a.

Remark. We have recalled the definition as it was originally given in [35]. However, please note that there is a certain redundancy in it. It is true that it is not the case that if an argument in an extension is decisively in w.r.t. range, then its acyclic evaluation is "protected" by the range. On the other hand, a protected evaluation does imply decisiveness. This comes from the fact that evaluations are built with decisively in interpretations. Consequently, since their $\mathbf{t}$ mapping are contained in the extension and the $\mathbf{f}$ ones are in the discarded set, then the range is a completion for these interpretations, which obviously means that the arguments are decisively in w.r.t. it.

Definition 5.16. Let $D = (A, C)$ be an ADF. An aa–admissible extension $E \subseteq A$ of D is **aa–complete** in D iff every argument in A that is decisively in w.r.t. the acyclic range interpretation v_E^a is in E[9].

Definition 5.17. Let $D = (A, C)$ be an ADF. A set of arguments is **aa–preferred** in D iff it is a maximal w.r.t. set inclusion aa–admissible extension of D.

The following example shows that decisiveness encapsulates defense of an argument, but not necessarily of its evaluation:

Example 19. Let us recall the framework $(\{a, b, c, d\}, \{C_a : \top, C_b : c \vee \neg a, C_c : b \vee \neg d, C_d : d\})$ from Figure 13 and Example 18. Argument a has a trivial acyclic evaluations $((a), \emptyset)$, while d only a standard one $(\{d\}, \emptyset)$. As for b and c, we have a standard evaluation $(\{b, c\}, \emptyset)$ that cannot be made acyclic, and the acyclic ones $((b), \{a\})$, $((c), \{d\})$, $((b, c), \{a\})$ and $((c, b), \{d\})$.

[9]Please consult Lemma 6.7 to see that no further "defense" of acyclicity is required.

We can observe that $\{c\}$ is a pd–acyclic conflict–free extension. Its acyclic range interpretation is $v = \{c : \mathbf{t}, d : \mathbf{f}\}$, since d does not possess any acyclic evaluations to start with. c is decisively in w.r.t. v and its acyclic evaluation $((c), \{d\})$ is protected by the range. Thus, it is an aa–admissible extension, even though it was not cc–admissible. Further aa–admissible extensions include $\emptyset, \{a\}, \{a, c\}, \{b, c\}$ and $\{a, b, c\}$. However, due to the fact that d is any acyclic discarded set, only $\{a, b, c\}$ is aa–complete. It is also the only aa–preferred extension.

Let us now assume a slightly modified ADF $(\{a, b, c, d\}, \{C_a : \top, C_b : c \vee \neg a, C_c : b \vee \neg d, C_d : \top\})$. The evaluations are just like before, with the exception of d, which now has a trivial acyclic one $((d), \emptyset)$. We can observe that $\{c\}$ is a pd–acyclic conflict–free extension. This time its range interpretation is $v = \{c : \mathbf{t}\}$ (both standard and acyclic) and we can observe that c is not decisively in w.r.t. v; a completion $v' = \{b : \mathbf{f}, c : \mathbf{t}, d : \mathbf{t}\}$ falsifies the acceptance condition. Thus, $\{c\}$ cannot be aa–admissible. Let us now look at the set $\{b, c\}$. Again, it is pd–acyclic conflict–free; its range is simply $v = \{b : \mathbf{t}, c : \mathbf{t}\}$. Both arguments are decisively in w.r.t. v; whether we utter a, d or both, it will not change the outcomes of the acceptance conditions. However, if our opponent uses $\{a, d\}$, the arguments are still able to stand only due to a cyclic dependency expressed by the standard evaluation $(\{b, c\}, \emptyset)$. In other words, all of their acyclic evaluations are blocked by v. Consequently, $\{b, c\}$ is not aa–admissible and in this case only $\emptyset, \{a\}, \{d\}$ and $\{a, d\}$ are our aa–admissible extensions, again with the last one being aa–complete and aa–preferred.

5.5.3 AC Family

Definition 5.18. Let $D = (A, C)$ be an ADF. A pd–acyclic conflict–free extension $E \subseteq A$ of D is **ac–admissible** in D iff every argument in E is decisively in w.r.t. standard range interpretation v_E and for every member of the extension there exists an acyclic pd–evaluation $((a_0, ..., a_n), B)$ on E s.t. all members of B are mapped to $\mathbf{f}$ by v_E.

Definition 5.19. Let $D = (A, C)$ be an ADF. A ac–admissible extension $E \subseteq A$ of D is **ac–complete** in D iff every argument in A that is decisively in w.r.t. the standard range interpretation v_E is in E [10].

Definition 5.20. Let $D = (A, C)$ be an ADF. A set of arguments is **ac–preferred** in D iff it is a maximal w.r.t. set inclusion ac–admissible extension of D.

Let us now look at an example to highlight the differences between the ac and aa–approach.

Example 20. Let us come back to the framework $(\{a, b, c, d\}, \{C_a : \top, C_b : c \vee \neg a, C_c : b \vee \neg d, C_d : d\})$ from Examples 18 and 19. Recall that d has a standard evaluation $(\{d\}, \emptyset)$ and no acyclic ones. The acyclic discarded set of $\{b, c\}$ is $\{d\}$, while the standard one remains empty. Consequently, even though $\{b, c\}$ is aa–admissible, it is not ac–admissible, as the acyclic evaluations of the arguments are not defended by the standard range. All in all, we have only have two ac–admissible extensions: $\emptyset$ and $\{a\}$, with $\{a\}$ being the ac–complete and ac–preferred one.

[10] Please consult Lemma 6.7 to see that no further "defense" of acyclicity is required.

5.5.4 CA Family

Let us come to the last family of semantics, the ca– type, which in many ways is very different from others. Since acyclicity is required on the "outside", but not on the "inside", this is the only approach in which an argument we can accept may not be considered a valid attacker. This leads to a loss or weakening of various properties, such as fundamental lemma or semilattice structure of the complete extensions, as we will see in Section 6. Moreover, this difference between acceptance and validity causes yet another controversy, namely how should we treat arguments that are in principle not acyclic, but follow from the arguments we have included in our extension. Let us look at a very simple example:

Example 21. Consider an ADF $(\{a, b\}, \{C_a : a \wedge \neg b, C_b : a\})$. We can observe that a has no pd–acyclic evaluations, only a standard one $(\{a\}, \{b\})$. Similar follows for b, it has a single evaluation $(\{a, b\}, \{b\})$, which is also self–blocking. Its partially acyclic representation is $(\{a\}, (b), \{b\})$. We can observe that the partially acyclic discarded sets of $\{a\}$ is empty, while the acyclic one is $\{b\}$. Thus, in the first case a would by no means be decisively in w.r.t. the appropriate range of $\{a\}$. However, in the acyclic case it is perfectly fine. We can also observe that even though b is in the acyclic discarded set, it is not decisively out w.r.t. the acyclic range of $\{a\}$. Thus, depending on whether we discard all cycles or only the ones not coming from the extension, we would get an answer that $\{a\}$ is and is not ca–admissible.

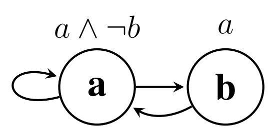


Figure 14: Sample ADF

Bearing this in mind, we will introduce two ca–subfamilies: the first one ca_1 discarding cyclic arguments independently of their origin, which was already considered in the previous version of this report [35], and ca_2, which makes an exception for those that follow from the arguments in our extension.

Definition 5.21. Let $D = (A, C)$ be an ADF. A conflict–free extension $E \subseteq A$ of D is **ca_1–admissible** in D iff every argument in E is decisively in w.r.t. the acyclic range interpretation v_E^a.

Definition 5.22. Let $D = (A, C)$ be an ADF. A ca_1–admissible extension $E \subseteq A$ of D is **ca_1–complete** in D iff every argument $s \in A \setminus E^{a+}$ that is decisively in w.r.t. the acyclic range interpretation v_E^a is in E.

Definition 5.23. Let $D = (A, C)$ be an ADF. A set of arguments is **ca_1–preferred** in D iff it is a maximal w.r.t. set inclusion ca_1–admissible extensions of D.

Definition 5.24. Let $D = (A, C)$ be an ADF. A conflict–free extension $E \subseteq A$ of D is **ca$_2$–admissible** in D iff every argument in E is decisively in w.r.t. the partially acyclic range interpretation v^p_E.

Definition 5.25. Let $D = (A, C)$ be an ADF. A ca$_2$–admissible extension $E \subseteq A$ of D is **ca$_2$–complete** in D iff every argument $s \in A$ that is decisively in w.r.t. the partially acyclic range interpretation v^p_E is in E.

Definition 5.26. Let $D = (A, C)$ be an ADF. A set of arguments is **ca$_2$–preferred** in D iff it is a maximal w.r.t. set inclusion ca$_2$–admissible extension of D.

The most important difference between the ca$_1$ and ca$_2$ approaches lies in the behavior of the arguments falsified by the range interpretation. In the first case, an argument discarded by the acyclic range interpretation can be in fact decisively in w.r.t. the interpretation unless the accepted arguments are acyclic. A case of such behavior was already visible in Example 21. The ca$_2$ approach is free from this problem, as shown in Proposition 6.1.

Example 22. Let us consider a small modification of the framework from Example 21. Let $(\{a, b, c\}, \{C_a : a \wedge \neg b, C_b : a, C_c : \neg b\})$ be the framework depicted in Figure 15. Just like before, a has no pd–acyclic evaluations, only a standard one $(\{a\}, \{b\})$. b has a single standard evaluation $(\{a, b\}, \{b\})$, which is also self–blocking. Its partially acyclic representation is $(\{a\}, (b), \{b\})$. Finally, c has an acyclic evaluation $((c), \{b\})$.

The $\emptyset$ extension is trivially both ca$_1$ and ca$_2$–admissible. Its partially acyclic and acyclic discarded sets coincide and are just $\{a, b\}$. The set $\{c\}$ has the same discarded set, thus making it also a ca$_1$ and ca$_2$–admissible extension. When we consider the set $\{a\}$, its partially acyclic and acyclic discarded sets are respectively $\emptyset$ and $\{b\}$. Thus, while $\{a\}$ is is ca$_1$–admissible, it is not ca$_2$–admissible. Similar follows for the set $\{a, c\}$. As for the complete extensions, $\{c\}$ is both ca$_1$ and ca$_2$–complete, while $\{a, c\}$ is only ca$_1$–complete.

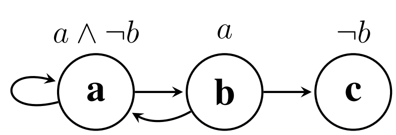


Figure 15: Sample ADF

Although in this case every ca$_2$–complete extension was also ca$_1$–complete, note this in general does not have to be the case. Let us consider a slight modification of our framework, in which the conjunction in the condition of a is replaced with a disjunction: $(\{a, b, c\}, \{C_a : a \vee \neg b, C_b : a, C_c : \neg b\})$. This makes $\{a\}$ both ca$_1$ and ca$_2$–admissible, however, the discarded sets differ. The acyclic one is $\{b\}$, while the partial one is empty. This affects the decisiveness of c and as a result $\{a\}$ is ca$_2$, but not ca$_1$–complete.

We close this section presenting our new semantics with a final example that compares the extensions of all the families.

Example 23. Let us recall the ADF $(\{a,b,c,d,e\},\{C_a : e, C_b : d \vee (c \wedge e), C_c : \neg e, C_d : \top, C_e : a \wedge b\})$ depicted in Figure 12. $\emptyset, \{c\}, \{d\}, \{b,d\}, \{c,d\}, \{b,c,d\}$ and $\{a,b,d,e\}$ are the conflict–free extensions, with the acyclic ones being $\emptyset, \{c\}, \{d\}, \{b,d\}, \{c,d\}$ and $\{b,c,d\}$.

The obvious cc–admissible extensions are $\emptyset, \{d\}$ and $\{b,d\}$ (follows from the discussion on the grounded extension in Example 17). The presence of d makes b acceptable independently of what happens to c and e, thus we do not have to analyze the conflict between them in this context. The last cc–admissible extension is $\{a,b,d,e\}$ and again, since d is present, the conflict can be disregarded. This is also the only cc–admissible extension that is not ac–admissible.

Let us now move to semantics acyclic on the "outside", starting with the aa approach. The cc–admissible extensions $\emptyset$, $\{d\}$ and $\{b,d\}$ are also aa–admissible. However, we can observe a cyclic positive dependency between a and b and $\{a,b,d,e\}$ cannot be aa–admissible. Since we only have to defend against acyclic attackers, $\{c\}$, $\{b,c,d\}$ and $\{c,d\}$ are additional aa extensions. Finally, all of those sets, including $\{a,b,d,e\}$, are ca$_1$ and ca$_2$–admissible.

The extension $\{b,d\}$ will be cc and ac–complete, but not aa, ca$_1$ and ca$_2$–complete as a and e will be automatically in the acyclic range. On the other hand, $\{b,c,d\}$ will be aa, ca$_1$ and ca$_2$–complete, but not cc and ac–complete. Finally, $\{a,b,d,e\}$ will be ca$_1$, ca$_2$ and cc–complete.

The preferred extensions are $\{a,b,d,e\}$ for the cc aproach, $\{b,d\}$ for ac, $\{b,c,d\}$ for aa and finally $\{b,c,d\}$ and $\{a,b,d,e\}$ for the ca$_1$ and ca$_2$ types.

6 Properties of Extension–Based Semantics

Various properties can be proved for our semantics and sub–semantics, obviously the study we provide here will not cover all of them. However, we will show how all sub–semantics of a given type relate one to another as well as recall the lemmas and theorems from the original paper on AFs [23]. Before we continue, we will make a note on some basic properties of the range interpretations:

Proposition 6.1. *Let $D = (A, C)$ be an ADF, $E \subseteq A$ a standard and $S \subseteq A$ a pd–acyclic conflict–free extension of D, with v_E, v_E^p, v_E^a, v_S, v_S^p and v_S^a as their corresponding standard, partially acyclic and acyclic range interpretations. Let $s \in A$ be an argument. The following holds:*

1. *If $v_E(s) = \mathbf{f}$, then s is decisively out w.r.t. v_E. Same holds or v_E^p, but not for v_E^a.*
2. *If $v_S(s) = \mathbf{f}$, then s is decisively out w.r.t. v_S. Same holds for v_E^p and v_E^a.*
3. *If $v_E(s) = \mathbf{f}$, then $C_s(E \cap par(s)) = out$. Same holds or v_E^p, but not for v_E^a.*
4. *If $v_S(s) = \mathbf{f}$, then $C_s(S \cap par(s)) = out$. Same holds for v_E^p and v_E^a.*

Proof.

1. The property w.r.t. v_E holds by definition. As for the partially acyclic range, let us assume it is not the case. This means there exists a completion v of v_E^p to a set E', where $E \subseteq E' \subseteq A$, s.t. $C_s(v^{\mathbf{t}} \cap par(s)) = in$. The $\mathbf{f}$ completion of v to A is a trivial decisively in interpretation for s. Consequently, we can also find a subinterpretation of v that is minimal for s. Let a be an arbitrary argument in $v^{\mathbf{t}}$. Obviously, a is not falsified by v_E^p and thus $a \neq s$. Hence, a either has a pd–acyclic evaluation that is not blocked by v_E^p, or a partially acyclic one s.t. its unblocked and the pd–set is in E - from this also follows that none of $a's$ depends on s. We can recombine such evaluations for $a's$ (see the proof of Theorem 5.5 for recombination approach) and extend them with the minimal interpretation contained in v in order to create one for s that is unblocked by v_E^p and is either acyclic, or partially acyclic with a pd–set in E. Thus, s could not have been falsified by the range in the first place and we reach a contradiction. Consequently, an argument falsified by the partially acyclic range is decisively out w.r.t. this range. The fact that it does not hold for the acyclic version can be already noted in Example 21.

2. Since every pd–acyclic conflict–free extension is also just conflict–free, this property w.r.t. v_E holds by definition and for v_E^p from the point above. Let us thus focus on v_S^a. Assume that $v_S^a(s) = \mathbf{f}$, but s is not decisively out w.r.t. v_S^a. This means there exists a completion v of v_S^a to a set S', where $S \subseteq S' \subseteq A$, s.t. $C_s(v^{\mathbf{t}} \cap par(s)) = in$. The $\mathbf{f}$ completion of v to A is a trivial decisively in interpretation for s. Consequently, we can also find a subinterpretation of v that is minimal for s. Let a be an arbitrary argument in $v^{\mathbf{t}}$. Obviously, a is not falsified by v_E^a and thus $a \neq s$. Hence, a has a pd–acyclic evaluation that is not blocked by v_E^a. We can recombine such evaluations for $a's$ (see the proof of Theorem 5.5 for recombination approach) and extend them with the minimal interpretation contained in v in order to create an acyclic evaluation for s that will be unblocked by v_E^p. Thus, s could not have been falsified by the range in the first place and we reach a contradiction. Consequently, an argument falsified by the acyclic range is decisively out w.r.t. this range.

3. Follows easily from the first point and the definition of decisiveness.

4. Follows easily from the second point and the definition of decisiveness.

□

We now know how decisiveness relates to being falsified in the range. In the next two lemmas we will make the relation between different types of discarded sets clear. We will also show two cases on which partially acyclic and acyclic discarded sets coincide.

Lemma 6.2. *Let $D = (A, C)$ be an ADF and $E \subseteq A$ a conflict–free extension of D. Then $E^+ \subseteq E^{p+} \subseteq E^{a+}$. If E is pd–acyclic conflict–free, then $E^{p+} = E^{a+}$.*

Proof. Let $a \in E^+$. By Lemma 3.12 it follows that for every standard evaluation (F, B) for a on A, $B \cap E \neq \emptyset$. Consequently, also every partially acyclic and acyclic (if exists) evaluation for a is blocked through the blocking set by E. Thus, whatever is in standard discarded range, is also in the partially acyclic one and $E^+ \subseteq E^{p+}$.

Let $a \in E^{p+}$. By definition, this means that every pd–acyclic evaluation of this argument is blocked through the blocking set and it has no unblocked partially acyclic evaluation with a pd–set in E. Consequently, by Lemmas 3.15 and 3.17 it follows that $a \in E^{a+}$. Therefore, $E^{p+} \subseteq E^{a+}$.

Let us now assume that E is pd–acyclic conflict–free and let $a \in E^{a+}$. Let us assume that $a \notin E^{p+}$. Since all acyclic evaluations of a are blocked, it must be the case it has an unblocked partially acyclic evaluation with a pd–set in E. However, since E is pd–acyclic conflict–free, we can create another evaluation for a by substituting the pd–set argument interpretation assignment with the ones that satisfy the E requirements. The resulting evaluation will obviously be unblocked and acyclic. Consequently, a could not have been in E^{a+} in the first place and we reach a contradiction. Thus, since $E^{p+} \subseteq E^{a+}$ for conflict–free sets and $E^{a+} \subseteq E^{p+}$ for pd–acyclic conflict–free, we can conclude that for pd–acyclic conflict–free sets $E^{p+} = E^{a+}$.

□

Lemma 6.3. *Let $D = (A, C)$ be an ADF and $E \subseteq A$ a model. Then $E^{a+} = E^{p+} = A \setminus E$.*

Proof. Let us first focus on the acyclic case, i.e. $E^{a+} = A \setminus E$; assume the equality does not hold and there exists an argument $a \in A \setminus E$ that is not in E^{a+}. By Lemma 3.15 this means it has a pd–acyclic evaluation $((a_0, ..., a_n), B)$ on A s.t. $B \cap E = \emptyset$. Let us consider a_0 and its associated decisively in interpretation v_0. By the pd–sequence requirements, it holds that $v_0^{\mathbf{t}} = \emptyset$. Thus, $C_{a_0}(\emptyset) = in$ and since $E \cap v_0^{\mathbf{f}} = \emptyset$, we can conclude that $C_{a_0}(E \cap par(a_0)) = in$. If $a_0 \notin E$, then E is not a model and we reach a contradiction. Let us thus assume $a_0 \in E$ and focus on a_1 and its associated decisively in interpretation v_1. By the pd–sequence requirements, it holds that $C_{a_1}(\{a_0\} \cap par(a_1)) = in$ and since $E \cap v_1^{\mathbf{f}} = \emptyset$, we can conclude that $C_{a_1}(E \cap par(a_1)) = in$. Thus again, if $a_1 \notin E$, then E is not a model and we reach a contradiction. We can continue in this manner until we reach $a_n = a$ and conclude that if E is a model, then it has to be the case that $E^{a+} = A \setminus E$.

Let us now consider the partially acyclic case. Assume that it does not hold and there is an argument $a \in A \setminus E$ that is not in E^{p+}. This means a either has a pd–acyclic evaluation $((a_0, ..., a_n), B)$ on A s.t. $B \cap E = \emptyset$ or a partially acyclic one $(F, (a_0, ..., a_n), B)$ on A s.t. $B \cap E = \emptyset$ and $F \subseteq E$. If it is the first case, then we can refer to the proof above and see that E could not have been a model. If it is the latter, then we can observe that since $F \subseteq E$, $E \cap v_0^{\mathbf{f}} = \emptyset$ and $v_0^{\mathbf{t}} \subseteq F$, it has to be the case that $C_{a_0}(E \cap par(a_0)) = in$. Thus, either $a_0 \in E$ or E is not a model. We can again repeat this procedure until we reach our argument and conclude that if E is a model, then it has to be the case that $E^{p+} = A \setminus E$. □

We close this section with the completion property of different types of range interpretations, which will be useful in a number of proofs.

Lemma 6.4. *Let $D = (A, C)$ be an ADF and E and E' two conflict–free extensions s.t. $E \subseteq E'$. It follows that $v_{E'}$ is a completion of v_E to some set $A' \subseteq A$.*

Let E and E' be two pd–acyclic conflict–free extensions s.t. $E \subseteq E'$. It follows that $v^a_{E'}$ is a completion of v^a_E to some set $A' \subseteq A$ and that $v^p_{E'}$ is a completion of v^p_E to some set $A'' \subseteq A$.

Proof. Let us start with the conflict–free case. Since $E \subseteq E'$, then $v^{\mathbf{t}}_E \subseteq v^{\mathbf{t}}_{E'}$. The fact that $v^{\mathbf{f}}_E \subseteq v^{\mathbf{f}}_{E'}$ follows easily from Lemma 3.12. Since E is not disjoint from the blocking set of any standard evaluation of an argument in $v^{\mathbf{f}}_E$, then neither is E'. It is also easy to see that no argument in E' could have been in the discarded set of E. Thus, $v_{E'}$ is a completion of v_E.

Let us now focus on the pd–acyclic conflict–free case. The proof for the acyclic range follows similarly to the one above by the use of Lemma 3.15. Since the acyclic and partially acyclic ranges coincide for pd–acyclic conflict–free sets by Lemma 6.2, the property carries over. □

Remark. The completion property does not hold when we consider conflict–free sets and acyclic or partially acyclic ranges – an argument can land in a discarded set due to having its acyclic and partially acyclic evaluations blocked, but may still possess a standard evaluation with which it can enter a different conflict–free set. This is again the ca–misbehavior that we have already discussed before.

6.1 Admissible and Preferred Semantics

Let us now show the relations between the introduced admissible sub–semantics.

Lemma 6.5. *Let $D = (A, C)$ be an ADF. The following holds:*

1. *Every ac–admissible extension of D is cc–admissible in D*
2. *Every ac–admissible extension of D is aa–admissible in D*
3. *Every aa–admissible extension of D is ca$_2$–admissible in D*
4. *Every cc–admissible extension of D is ca$_2$–admissible in D*
5. *Every ca$_2$–admissible extension of D is ca$_1$–admissible in D*
6. *Not every ca$_1$–admissible extension of D is ca$_2$–admissible in D*

Proof.

1. Follows from the definition of ac and cc–admissible semantics.
2. Follows from the definition of ac and aa–admissible semantics and Lemma 6.2.
3. By Lemma 6.2, the partially acyclic and acyclic discarded set coincide for pd–acyclic conflict–free sets. From this and the definitions of the aa and ca$_2$–admissible extensions the property follows.
4. Follows from the definition of cc and ca$_2$–admissible semantics and Lemma 6.2.
5. Follows from the definition of ca$_1$ and ca$_2$–admissible semantics and Lemma 6.2.

6. See Example 22.

□

Remark. The restrictions we put on the "inside" and "outside" affect the number of extensions we receive. The less we have on the inside, the more we can say. The more we have on the outside, the less our opponent is allowed to utter against us. Thus, not surprisingly, the ac approach can be seen as the most strict, while ca families admit the most.

However, the is–a relation between the extensions cannot be assumed in the case of the preferred sub–semantics. Although a given admissible extension can belong to many subsemantics, it does not mean that their maximal elements are the same. Thus, we can only derive some inclusion relation, as depicted in Figure 16.

Lemma 6.6. *Let $D = (A, C)$ be an ADF. Let xy and x'y' be two admissible sub–semantics, where $x, x', y, y' \in \{a, c\}$, s.t. every xy–admissible extension is also x'y'–admissible (see Lemma 6.5). Then every xy–preferred extension of D is contained in some x'y'–preferred extension of D.*

Proof Idea. The reasoning behind it is rather simple;. Given $x, x', y, y' \in \{a, c\}$, if xy–admissible extensions are x'y'–admissible, then also xy–preferred extensions are x'y'–admissible. Taking the maximal x'y'–admissible extensions, hence x'y'–preferred ones, ensures that every xy–preferred one is contained in at least one chosen set.

Before we continue with further analysis, we first have to show that our admissible sub–semantics satisfy the Fundamental Lemma. However, in the case of ca_1–admissibility, we can only assume a weaker version.

Lemma 6.7. ***CC/AC/AA Fundamental Lemma**: Let $D = (A, C)$ be an ADF, E a cc(ac)–admissible extension of D, v_E its range interpretation and $a, b \in A$ two arguments decisively in w.r.t. v_E. Then $E' = E \cup \{a\}$ is cc(ac)–admissible in D and b is decisively in w.r.t. v'_E.*

Let E be an aa-admissible extension of D, v^a_E its acyclic range interpretation and $a, b \in A$ two arguments decisively in w.r.t. v^a_E. Then $E' = E \cup \{a\}$ is aa–admissible in D and b is decisively in w.r.t. v'^a_E.

Proof. Let us assume that $a, b \notin E$ – if it is not the case, then the proofs are trivial.

Let us start with the cc case. First of all, it follows from Proposition 6.1 that neither a nor b could have been mapped to **f** by v_E. Thus, $v_{E'}$ is a completion of v_E and whatever was decisively in w.r.t. v_E must remain this way w.r.t. $v_{E'}$. Consequently, all arguments in E' satisfy the cc–admissibility criterion. The same follows for b – if it was decisively in w.r.t. v_E, then it is also this way w.r.t. $v_{E'}$.

Let us move on to the ac approach. Again, by Proposition 6.1 we know that neither a nor b could have been mapped to **f** by v_E. Thus, $v_{E'}$ is a completion of v_E and whatever was decisively in w.r.t. v_E must remain this way w.r.t. $v_{E'}$. Thus, E' is trivially conflict–free and b is decisively in w.r.t. v'_E. Since every argument in E has a pd–acyclic evaluation s.t. the blocking set is falsified by v_E, then it is easy to see that these evaluations hold in E' as well. What remains to be shown is that there is a pd–acyclic evaluation for a on E' satisfying the ac–admissibility requirements. As a is decisively in w.r.t. v_E, it means that there exists a minimal decisively in interpretation v' for a s.t. v_E (and thus $v_{E'}$) is its completion. The true part of this interpretation will depend only on arguments in E, and following similar lines of reasoning as in the proof of Theorem 5.5 we can recombine the pd–acyclic evaluations of these arguments that satisfy the ac–admissibility criterion into one and extend it with v'. The resulting evaluation will clearly have a blocking set falsified by $v_{E'}$ and satisfy the ac–admissibility requirements. Thus, E' is ac–admissible in D.

Finally, we have the aa–admissible case. By Proposition 6.1 we know that neither a nor b could have been mapped to **f** by v_E^a. Thus, $v_{E'}^a$ is a completion of v_E^a. Therefore, all arguments in E, a and b remain decisively in w.r.t. $v_{E'}^a$. What remains to be showed is that E' preserves its acyclicity and defends the evaluations, which follows the reasoning we presented above in the ac case. □

Lemma 6.8. ***Weak CA*$_1$ *Fundamental Lemma***: *Let $D = (A, C)$ be an ADF, $E \subseteq A$ a ca$_1$–admissible extension, v_E^a its acyclic range interpretation and $a, b \in A \setminus E^{a+}$ arguments decisively in w.r.t. v_E. Then $E' = E \cup \{a\}$ is ca$_1$–admissible in D, b is decisively in w.r.t. $v_{E'}$, but it is not necessarily in $A \setminus E'^{a+}$.*

Proof Sketch. As E may contain cycles, it can be the case that a, b are decisively in w.r.t. v_E^a and at the same time $v_E^a(a) = \mathbf{f}$ and $v_E^a(b) = \mathbf{f}$. Therefore, we only take into account such arguments a, b that are not discarded. As a result, v'_E is a completion of v_E and we can use the proof of the cc part of the Fundamental Lemma (i.e. Lemma 6.7).

Although b will still be decisively in w.r.t. $v_{E'}$, it might be the case that it is in the acyclic discarded set of E'. Let us consider an ADF $(\{a, b, c\}, \{C_a = a, C_b = a \vee \neq c, C_c = \top\})$. The set $\{a\}$ is ca$_1$–admissible and its acyclic discarded set is empty - b still possesses an unblocked pd–acyclic evaluation $((b), \{c\})$. Both c and b are decisively in w.r.t. our extension, however, acceptance of c blocks the pd–acyclic evaluation of b. Thus, b is still decisively in w.r.t. $\{a, c\}$, but it is also in the acyclic discarded set.

Lemma 6.9. ***CA*$_2$ *Fundamental Lemma*** *Let $D = (A, C)$ be an ADF, $E \subseteq A$ an ca$_2$–admissible extension of D, v_E^p its partially acyclic range interpretation and $a, b \in A$ two arguments decisively in w.r.t. v_E^p. Then $E' = E \cup \{a\}$ is ca$_2$–admissible in D and b is decisively in w.r.t. v'^p_E.*

Proof. It follows from Proposition 6.1 that neither a nor b could have been mapped to **f** by v_E^p. Thus, $v_{E'^p}$ is a completion of v_E^p and whatever was decisively in w.r.t. v_E^p must remain this way w.r.t. $v_{E'^p}$. Consequently, all arguments in E' are decisively in w.r.t. $v_{E'^p}$ and ca$_2$–admissibility follows. The same follows for b – if it was decisively in w.r.t. v_E^p, then it is also this way w.r.t. $v_{E'^p}$. □

This leads us to the last result, reflect the Theorem 2.11 from the Dung setting:

Theorem 6.10. *Let $D = (A, C)$ be an ADF and $x \in \{cc, ac, aa\}$. The following holds:*

1. *The sets of all x–admissible extensions of D forms a complete partial order w.r.t. set inclusion* [11]

[11] There are many definitions of complete partial orders. We will assume that a partial order $(A, \leq)$ is a complete iff it has a least element and each of its directed subsets has a lub.

2. *D possesses at least one x–preferred extension.*

3. *For each x–admissible set E of D, there exists an x–preferred extension E' of D s.t. $E \subseteq E'$.*

Proof.

1. We can observe that cc, ac and aa–admissible extensions form partial orders w.r.t. set inclusion. Let P be the collection of admissible extensions of a given type. Since $\emptyset$ is an admissible extension of any type and for any framework, we have a least element. Let $L \subseteq P$ be a nonempty directed subset of P and let $E = \bigcup L$ denote the union of all extensions in L. We will show that E is an admissible extension of a given type. By directness of L it follows that for any two extension $E_1, E_2 \in L$ there exists another one $E_3 \in L$ s.t. $E_1 \subseteq E_3$ and $E_2 \subseteq E_3$. By Lemma 6.4 we can thus conclude that it cannot be the case that there is an argument mapped to $\mathbf{t}$ by the range of E_1 or E_2 and mapped to $\mathbf{f}$ by the range of E_2 or E_1. Thus, it holds that for no extension E' in L there is an argument in E that is mapped to $\mathbf{f}$ by the range of E'. Since $E' \subseteq E$, an interpretation mapping elements of E to $\mathbf{t}$ and everything else to $\mathbf{f}$ is a completion of the range of E'. Thus, by decisiveness and admissibility of E' it follows that all arguments in E' have a satisfied acceptance condition w.r.t. E. Therefore, we can conclude that E is conflict–free. If we focus on the aa and ac case, we can observe that all arguments in a given extension E' had an evaluation whose blocking set was falsified by the range. Consequently, the blocking set of such an evaluation will be disjoint from E and as $E' \subseteq E$, we can conclude that all arguments in E have an unblocked pd–acyclic evaluation on the set. Thus, E is pd–acyclic conflict–free. We can now again use Lemma 6.4 to show that the range of E is a completion of E'. Consequently, decisiveness and "defense" of evaluations carries over and E is an admissible extension of a given type. Therefore, it is an upper bound for L. Let Z be another upper bound for L and assume it is not the case that $E \subseteq Z$. First of all, it cannot be the case that $Z \subset E$; since E is a union of all extensions in L, removal of any of its arguments would render it not an upper bound for L. Let us thus assume that Z and E are incomparable. However, it would again mean that Z does not contain one argument that E has. Consequently, there is an argument in some extension in L that is not in Z and Z cannot be an upper bound. We can thus conclude that E is the least upper bound for L and that P is a complete partial order.

2. Let P be the collection of admissible extensions of a given type. It forms a partial order w.r.t. set inclusion. In a way similar to the one above, we can show that every chain in P has a least upper bound – the lack of "collisions" between ranges is the only important thing. Therefore, by applying Kuratowski-Zorn Lemma, P contains at least one maximal element, which in our case is a preferred extension of a given type.

3. Let E be an admissible extension of a given type, P the set of all admissible extensions of this type and L the collection of all admissible extensions E'' of this type s.t. $E \subseteq E''$. It follows that E will be the least element of L. Moreover, L will form a partial order w.r.t. set inclusion, and by using the reasoning from the points above we can show that it is a complete partial order and that it contains at least one maximal element F. F will also be

a maximal element of P. If it were not the case, then there would exist an extension F' s.t. $F \subset F'$, which would obviously also be in L and thus contradicting maximality of F. Consequently, we have a maximal admissible extension of a given type, which by definition is also preferred, and that by construction contains E.

□

6.2 Complete and Grounded Semantics

We can now analyze the complete sub–semantics. Not surprisingly, the correspondence between the extensions depends on the "outside", i.e. w.r.t. which range interpretation the decisiveness of arguments is evaluated. In other words, arguments that are decisively in w.r.t. the acyclic range interpretation might not necessarily be decisively in w.r.t. the standard one. Hence, although every ac–admissible extension is aa–admissible, not every ac–complete extension is aa–complete. It can already be observed in Example 23. We can observe similar results in the case of cc and ca–complete semantics. Thus, we are left only with the following properties, depicted in Figure 16.

Lemma 6.11. *Let $D = (A, C)$ be an ADF. It holds that:*

1. *Every ac–complete extension of D is cc–complete in D*
2. *Every aa–complete extension of D is ca$_1$–complete in D.*
3. *Every aa–complete extension of D is ca$_2$–complete in D.*
4. *Not every ca$_1$–complete extension of D is ca$_2$–complete in D and vice versa.*

Proof.

1. Let E be an arbitrary ac–admissible extension. By Lemma 6.5, it is also cc–admissible. If E is ac–complete, but not cc–complete, it would mean that at the same time all arguments decisively in w.r.t. v_E are in E and there are is an argument decisively in w.r.t. v_E but not in E. We reach a contradiction.
2. Let E be an arbitrary aa–admissible extension. By Lemma 6.5, it is also ca$_1$–admissible. If all arguments in A that are decisively in w.r.t. v_E^a are in E, then of course so are the ones contained in $A \setminus E^{a+}$. Thus, the ca$_1$–completeness criterion is satisfied.
3. By Lemma 6.2, the partially acyclic and acyclic discarded sets coincide for pd–acyclic conflict–free sets. Consequently, every aa–complete extension meets ca$_2$–complete requirements.
4. See Example 22.

□

We can now continue with an ADF version of Theorem 2.12 from the Dung setting. Unfortunately, it will not carry over to all of our semantics:

Theorem 6.12. *Let $D = (A, C)$ be an ADF. The following holds:*

1. *Every xy–preferred extension of D is an xy–complete extension of D for $x, y \in \{a, c\}$, but not vice versa.*

2. *The grounded extension of D is the least w.r.t. set inclusion ac and cc–complete extension of D.*

3. *The acyclic grounded extension of D is the least w.r.t. set inclusion aa–complete extension of D and a minimal ca$_1$ and ca$_2$–complete extension of D.*

4. *The cc, ac and aa–complete extensions of D form complete meet–semilattices w.r.t. set inclusion.*

Proof.

1. Let us first show the cc/ac/aa/ca$_2$ case. Assume an xy–preferred extension $E \subseteq A$ of D is not xy–complete. This means that there exists some argument $a \in A$ s.t. a is decisively in w.r.t. v_E/v_E^a/v_E^p (depending on the case) but is not in E. By the Fundamental Lemmas 6.7 and 6.9, $E \cup \{a\}$ is cc/ac/aa/ca$_2$ admissible. Obviously $E \subset E \cup \{a\}$, which means E could not have been preferred in the first place. We reach a contradiction. A similar reasoning follows for the ca$_1$–case, just with $a \in A \setminus E^{a+}$.

2. Let $E \subseteq A$ be the grounded extension of D. The fact that it is ac–complete follows easily from the construction presented in Proposition 5.9. It is thus also cc–complete by Lemma 6.11. The fact that it is also the least extension is a result of Lemma 6.4. Let $E' \subseteq A$ be an arbitrary ac/cc–complete extension. It is obviously conflict–free, and so is $\emptyset$. Let v be the standard range of $\emptyset$. The range of E' is a completion of v by Lemma 6.4 and whatever is decisively in/out w.r.t. v, is decisively in/out w.r.t. $v_{E'}$. Moreover, it is also decisively in/out w.r.t. v_E by the construction in Proposition 5.9. Let us extend v with the mappings for the arguments that are decisive w.r.t. it. Obviously, the set represented by the $\mathbf{t}$ mappings is conflict–free. By the fact that E' is ac/cc–complete, it has to contain every argument decisively in w.r.t. its range. Moreover, by definition the range will also map to $\mathbf{f}$ every argument decisively out w.r.t. it. Thus, it has to be the case that $v_{E'}$ is again a completion of v, an by the grounded construction also of v_E. We can continue in the same manner until we cannot extend v further and conclude that $v^{\mathbf{t}} \subseteq E'$. Since by Proposition 5.9 we have also recreated the grounded construction, $v^{\mathbf{t}} = E$. Thus, the grounded extension is ac/cc–complete and is contained in arbitrary ac/cc–complete extensions. Therefore, it is the least ac/cc–complete extension.

3. The proof that the acyclic grounded extension is the least aa–complete extension follows similarly to the proof above by the use of Definition 5.10 and Lemma 6.4. However, in the case of ca_1/ca_2–complete extensions it can only be minimal.

   Assume that E is not a minimal ca_1–complete extension. It means there exists another ca_1–complete extension E' s.t. $E' \subset E$. Let us iterate through the definition of the acyclic grounded semantics and start with $M = \emptyset$ and its acyclic discarded set M^{a+}. Obviously, $M \subseteq E'$. What we need to show is that $M^{a+} \subseteq E'^{a+}$. Since every argument in E' is in E, it means it possesses at least one pd–acyclic evaluation on A with a blocking set disjoint from E'. Thus, $E' \cap M^{a+} = \emptyset$ by Lemma 3.15. By the same lemma it is easy to see that any argument in M^{a+} will be in E'^{a+}. Therefore, at this step $v^a_{E'}$ is a completion of v^a_M and whatever is decisively in w.r.t. the latter, is decisively in w.r.t. the former. Let $a \in A$ be an argument decisively in w.r.t. M. Since at this point M is trivially aa–admissible, then by the AA Fundamental Lemma 6.7 $M \cup \{a\}$ will also be aa–admissible. Thus, a possesses a pd–acyclic evaluation on A that, due to decisiveness and the range–completion relation, cannot not blocked by $v^a_{E'}$. Hence, it has to be the case that $a \in E'$. Let us thus add all such a into M and repeat the procedure. Again, it cannot be the case that an argument in E' is in M^{a+}, as every argument in E has a pd–acyclic evaluation on A with a blocking set disjoint from E and $M \subseteq E'$. Since any evaluation blocked by v^a_M will also be blocked by $v^a_{E'}$, we now can conclude that $v^a_{E'}$ is again a completion of v^a_M. We can repeat the procedure above to extend M and show that it still is the case that $M \subseteq E'$. We can continue in this manner until M cannot grow further and conclude that $M \subseteq E'$. Since we have also recreated the acyclic grounded extension this way, it follows that $M = E$. Therefore, $E \subseteq E' \subset E$, which is a clear contradiction. Consequently, the acyclic grounded extension is a minimal ca_1–complete extension.

   Now, assume that E is not a minimal ca_2–complete extension and that there exists a ca_2–complete one E' s.t. $E' \subset E$. Let us again iterate through the definition of the acyclic grounded semantics and start with $M = \emptyset$ and its acyclic discarded set M^{a+}. Obviously, $M \subseteq E'$. What we need to show is that $M^{a+} \subseteq E'^{p+}$. Every argument in E' is in E, thus it possesses a pd–acyclic evaluation on A with a blocking set disjoint from E'. Consequently, it does no qualify for M^{a+} by Lemma 3.15. Let us now assume that there is an argument $a \in A$ which is in M^{a+}, but not in E'^{p+}. This means it has to possess a partially acyclic evaluation with a blocking set disjoint from E' and pd–set in E'. However, since every argument in E' is in E and thus has a pd–acyclic evaluation unblocked by E', we can transform the partially acyclic evaluation into an acyclic one by replacing the pd–set with a recombined pd–acyclic evaluation. The blocking set of the resulting evaluation will obviously be disjoint from E', and thus from M as well. Consequently, by Lemma 3.15 a could not have been in M^{a+} in the first place. We can thus conclude that $M^{a+} \subseteq E'^{p+}$ and that $v^{p+}_{E'}$ is a completion of v^a_M. From this point on we can proceed in a manner similar to the ca_1 proof and arrive at the conclusion that the acyclic grounded extension is a minimal ca_2–complete one.

4. We will first focus on the glb property. Let C^{cc} be the set of all cc–complete extensions and $S \subseteq C^{cc}$ a nonempty subset. Let $L = \{E \mid E \in C^{cc}$ and for every $E' \in S$, $E \subseteq E'\}$

be the set of all cc–complete extensions that are contained by all extensions in S. By the points above, the grounded extension is obviously in L and the set cannot be empty. Let $M = \bigcup L$ be the union of all extensions in L. We will show it is cc–complete. By Lemma 6.4 and the fact that all extensions in L have a "bigger" one in S, we can observe that for no two extensions in L it can be the case that an argument accepted in one is mapped to $\mathbf{f}$ by the range of the other. Thus, a decisively in interpretation for an argument a in some extension in L that was sufficient for cc–completeness has a $\mathbf{t}$ part contained in M and a $\mathbf{f}$ part disjoint from it. Thus, M is conflict–free. Again, by Lemma 6.4 it follows that the standard range of M is a completion for the standard ranges of extensions in L. From this also follows straightforwardly that M is cc–admissible. If M is not cc–complete, it means there is an argument $a \in A \setminus M$ that is decisively in w.r.t. its range. By Lemma 6.4 it means that a is also decisively in w.r.t. the range of all extension in L. Thus, either one of them is not cc–complete, or it is contained in all of them and by construction it also has to be in M. Therefore, M is cc–complete. Since for any extension $Z \in L$, $Z \subseteq M$ and for any extension $Z' \in S$, $M \subseteq Z$, it follows that it is the greatest lower bound of S. In a similar way we can prove that every nonempty subset of ac and aa–complete extensions also has a greatest lower bound (one can also refer to proof of Theorem 6.10 for further details).

We will now focus on the lub property. Let C^{cc} be the set of all cc–complete extensions and S an increasing sequence of C^{cc}. Any two elements of the sequence are comparable w.r.t. $\subseteq$. Let E be the union of all complete extensions in S. We will prove that E is cc–admissible in D. First of all, since the sequence is increasing, we can observe by Lemma 6.4 that no argument mapped to $\mathbf{f}$ by the range of one extension will be accepted in a different extension. Thus, E is conflict–free. Moreover, again by Lemma 6.4 we can observe that the range interpretation of E will be a completion of the range interpretation of any extension in S. Thus, all arguments in E will be decisively in w.r.t. the range and we can conclude that E is cc–admissible. By Theorem 6.10, for every cc–admissible extension there exists at least one cc–preferred extension, and thus a cc–complete one containing it. Therefore, our sequence will have at least one upper bound. Let U be the collection of all such upper bounds, i.e. cc–complete extensions that contain all of the sets in S. In a manner similar to above, we can find a cc–complete extension G that is a glb for U. Moreover, since by construction G contains all extensions in S and is thus and upper bound for it, and at the same it is contained by all extension in U, i.e. by all other upper bounds for S, it is a least upper bound for S. The proof for ac and aa–complete extensions follows similarly.

$\square$

Proposition 6.13. *There exists and ADF $D = (A, C)$ s.t. :*

1. *The grounded extension of D is neither an aa, ca$_1$ nor a ca$_2$–complete extension of D.*
2. *The ca$_1$ and ca$_2$–complete extensions of D do not form complete meet–semilattices w.r.t. set inclusion.*

Proof.

1. It is a result of the fact that these sub–semantics do not treat cyclic attackers as valid, while the grounded semantics does not make this assumption. Let $(\{a, b\}, \{C_a : \neg b, C_b : b\})$ be a simple ADF where a is attacked by a self–supporting argument b. The aa–complete extension would be $\{a\}$ and the ca$_1$/ca$_2$–complete ones would be $\{a\}$ and $\{b\}$, while the grounded one would be simply $\emptyset$.

2. Please consider a simple ADF $(\{a, b\}, \{C_a : \neg b, C_b : b\})$. The ca$_1$/ca$_2$–complete extensions are $\{a\}$ and $\{b\}$. The lower bound of $\{\{a\}, \{b\}\}$ is $\emptyset$ and it is not a ca$_1$/ca$_2$–complete extension.

□

6.3 Model and Stable Semantics

The relations between the semantics presented in [23] also carry on to some of the specializations and are shown in Figure 16. In this section we will focus on how model and stable semantics relate to various preferred and complete ones.

Lemma 6.14. *Let $D = (A, C)$ be an ADF. Every model of D is ca$_1$ and ca$_2$–complete in D, but not necessarily ca$_1$ or ca$_2$–preferred in D.*

Proof. Let E be a model. By Lemma 6.3, we know that v^a_E/v^p_E is defined for every argument in A. Hence, it is its own single completion and all accepted arguments are decisively in w.r.t. it. Consequently, ca$_1$/ca$_2$–admissibility requirements are satisfied. Since by Lemma 6.3, $E^{a+} = A \backslash E$, there is no argument left to check for decisiveness in the ca$_1$ case and ca$_1$ completeness follows easily. By the same Lemma $E^{p+} = A \setminus E$, and from Proposition 6.1 it follows that every argument in $A \setminus E$ is decisively out w.r.t. to the partial acyclic range of E. Thus, no argument not in E can be decisively in w.r.t. the range and ca$_2$–completeness follows as well.

Since model semantics can produce extensions that are comparable w.r.t. set inclusion, then it is not surprising that a model might not be a ca$_1$/ca$_2$–preferred extension. It is already visible in Example 16: models were sets $\{c\}, \{a, b\}$ and $\{a, b, c\}$, with only the last one being ca$_1$/ca$_2$–preferred. □

Lemma 6.15. *Let $D = (A, C)$ be an ADF. Every stable extension of D is an aa–preferred in D, but not vice versa. It is not necessarily a cc, ac, ca$_1$ or ca$_2$–preferred extension.*

Proof. Let E be a stable extension of D. First of all, it is by definition a pd–acyclic conflict–free model. Thus, by Lemma 6.3 $E^{a+} = A \setminus E$. From this it follows easily that all arguments in E are decisively in w.r.t. v^a_E. Moreover, by pd–acyclic conflict–freeness every argument $a \in E$ has a pd–acyclic evaluation $((a_0, ..., a_n), B)$ on E s.t. $B \cap E = \emptyset$. Thus, $B \subseteq E^{a+}$ and thus every argument has a pd–acyclic evaluation with a blocking set falsified by the range. Consequently, all aa–admissibility criterion are satisfied. Since every argument not in the extension is in the acyclic discarded set and is by Proposition 6.1 decisively out w.r.t. the acyclic range, aa–completeness follows easily. Let us assume that E is not aa–preferred. This means there exists another aa–complete extension $E' \subseteq A$ of D s.t. $E \subset E'$. However, by Lemma 3.15 and the fact that $E^{a+} = A \setminus E$, it means that for any argument in $E' \setminus E$, its every pd–acyclic evaluation $((a_0, ..., a_n), B)$ on A is blocked through the blocking set by E. Thus, E' cannot possibly satisfy the aa–admissibility criterion in the first place. We reach a contradiction and can thus conclude that every stable extension of D is aa–preferred in D.

Concerning the other preferred sub–semantics, we will refer to some of the examples we have already analyzed. Recall the framework $(\{a, b, c\}, \{C_a : \neg c \vee b, C_b : a, C_c : c\})$ and Example 16. The stable extension is $\{a, b\}$, while $\{a, b, c\}$ is ca_1/ca_2–preferred. $\{a, b\}$ is not even ac–admissible, let alone ac–preferred. In the Example 23 we considered a framework $(\{a, b, c, d, e\}, \{C_a : e, C_b : d \vee (c \wedge e), C_c : \neg e, C_d : \top, C_e : a \wedge b\})$. Its cc–preferred extension is $\{a, b, d, e\}$, while the stable is $\{b, c, d\}$. □

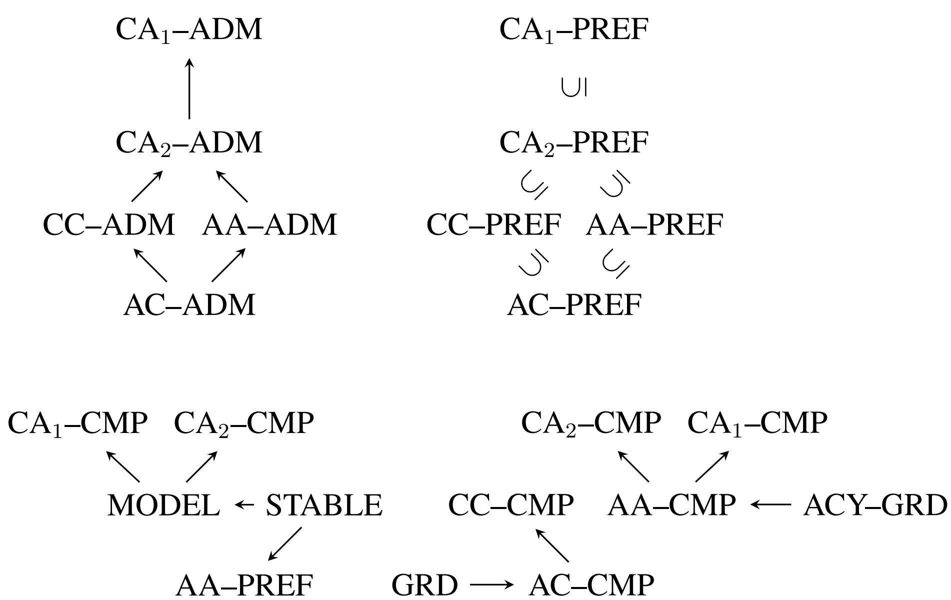


Figure 16: The relations between given extension–based sub–semantics. $x \rightarrow y$ should be read as extensions of x are extensions of y. $x \subseteq y$ should be read as any extension of x is contained in some extension of y.

6.4 Coincidence of Families: the AADF$^+$ Class

Last, but not least, we will describe a subclass of ADFs for which our classification system collapses. By this we understand that all xy–subsemantics of a given semantics coincide, e.g. every aa–admissible extension is cc–admissible and so on. We will refer to the frameworks in this subclass as the positive dependency acyclic abstract dialectical frameworks and denote them as AADF$^+$s.

Definition 6.16. Let $D = (A, C)$ be an ADF. D is an AADF$^+$ iff every standard evaluation for any argument $a \in A$ in D can be made acyclic.

Example 24. Recall the framework $(\{a, b, c\}, \{C_a : a \wedge \neg b, C_b : a, C_c : \neg b\})$ depicted in Figure 15 and described in Example 22. The arguments a and b have standard evaluations $(\{a\}, \{b\})$ and $(\{a, b\}, \{b\})$. None of these can be made acyclic. Consequently, the framework is not an AADF$^+$.

Let us now consider the framework $(\{a, b, c, d, e\}, \{C_a : \top, C_b : \neg a \vee c, C_c : b, C_d : \neg c \wedge \neg e, C_e : \neg d\})$ from Figure 7 and Example 9. Every argument has an acyclic evaluation. However, arguments b and c also possess a standard one $(\{b, c\}, \emptyset)$, which cannot be made acyclic. Thus, this framework is again not an AADF$^+$.

Let $(\{a, b, c, d, e\}, \{C_a : \top, C_b : \neg a \wedge \neg c, C_c : \neg d, C_d : \neg c, C_e : \neg d \wedge \neg e\})$ be the ADF representation of the AF from Figure 1 and Example 1. Every argument has a single minimal decisively in interpretation that maps to $\mathbf{f}$ its parents and has no $\mathbf{t}$ mappings. Thus, it is easy to see every standard evaluation in this framework can be made acyclic and we are dealing with a AADF$^+$.

Theorem 6.17. *Let $D = (A, C)$ be an AADF$^+$. The following holds:*

1. *Every conflict–free extension is pd–acyclic conflict–free*
2. *Every naive extension is pd–acyclic naive*
3. *Every model is stable*
4. *Given a conflict–free set of arguments $E \subseteq A$, $E^+ = E^{p+} = E^{a+}$*
5. *The aa, cc, ac, ca$_1$ and ca$_2$–admissible extensions coincide*
6. *The aa, cc, ac, ca$_1$ and ca$_2$–complete extensions coincide*
7. *The aa, cc, ac, ca$_1$ and ca$_2$–preferred extensions coincide*
8. *Every grounded extension is acyclic grounded and vice versa*

Proof.

1. Let $E \subseteq A$ be a conflict–free extension. Every argument in the set has a satisfied acceptance condition and thus has a trivial decisively in interpretation that maps to $\mathbf{t}$ the elements of E and to $\mathbf{f}$ everything else. By extracting such minimal interpretations we can easily construct an unblocked standard evaluation representing this extension; it will also be an evaluation for any argument in E. Since every evaluation is pd–acyclic, then every $a \in E$ an unblocked pd–acyclic evaluation on E. Consequently, E is pd–acyclic conflict–free.
2. Since pd–acyclic and standard conflict–free extensions coincide, then naturally so do naive
3. Since every conflict–free extension is pd–acyclic conflict–free, then every model is pd–acyclic conflict–free and thus stable

4. By Lemma 6.2 we already know that $E^+ \subseteq E^{p+} \subseteq E^{a+}$. What remains to be shown is that $E^{a+} \subseteq E^+$. Since E is conflict–free, then by the first point it is also pd–acyclic conflict–free, and by Lemma 3.15 for every pd–acyclic evaluation (F, B) of an argument $a \in E^{a+}$, $B \cap E \neq \emptyset$. Due the fact that every standard evaluation can be made acyclic, it means that every standard evaluation of a is blocked through the blocking set by E. Thus, by Lemma 3.12, a is in E^+. Therefore, $E^+ = E^{p+} = E^{a+}$.

5. The coincidence of ca_1 and ca_2–admissible extensions follows easily from the points above. Let E be a ca_2–admissible extension of D. Since every argument is decisively in w.r.t. v^p_E, then for every argument there is a minimal decisively in interpretation with the $\mathbf{t}$ part in E and $\mathbf{f}$ part in E^{p+}. Thus, by a construction similar to the one in the first point, it follows that every argument in E has a standard evaluation s.t. its blocking set is falsified by the partially acyclic range. Since E is pd–acyclic conflict–free, every evaluation can be made acyclic and the discarded sets coincide, it follows that E is pd–acyclic conflict–free, every argument in E is decisively in w.r.t. standard range and has an acyclic evaluation with a blocking set falsified by it. Thus, ac–admissibility conditions are met and by Lemma 6.5, aa/cc/ac/ca_1 and ca_2–admissible extensions coincide.

6. The coincidence of aa/cc/ac/ca_2–complete extensions follows easily from the points above. Let E be a ca_1–complete extension. By the first point, it is also pd–acyclic conflict–free. Thus, by Proposition 6.1 every argument in the acyclic discarded set of E is decisively out w.r.t. the acyclic range. Consequently, if there is no argument in $A \setminus (E^{a+} \cup E)$ that is decisively in w.r.t. the acyclic range of E, then there is none in $A \setminus E$ either. Thus, the the discarded set exclusion in the definition of ca_1–complete semantics becomes redundant and the extensions coincide with aa–complete ones. Therefore, all complete subsemantics produce the same extensions.

7. Since all admissible extensions coincide, so do the preferred.

8. As all complete extensions coincide, then by Lemma 6.12 so do grounded and acyclic grounded.

□

Since both AADF$^+$s and BADFs (bipolar ADFs) deal, this way or the other, with support, it is natural to ask if there is any relation between the two subclasses. The answer is, that there are AADF$^+$s that are not BADFs and vice versa. The subclasses are different, though not disjoint. There exist frameworks that are both; for example, every Dung–style ADF will be both bipolar and positive dependency acyclic.

Example 25. Let us go back to the frameworks in Example 24. The first one $(\{a, b, c\}, \{C_a : a \wedge \neg b, C_b : a, C_c : \neg b\})$ was not an AADF$^+$, however, it is a BADF. The links from a to a and b are supporting and the ones from b to a and c are attacking. Similar holds for the ADF $(\{a, b, c, d, e\}, \{C_a : \top, C_b : \neg a \vee c, C_c : b, C_d : \neg c \wedge \neg e, C_e : \neg d\})$.

The Dung–style ADF $(\{a,b,c,d,e\},\{C_a:\top,C_b:\neg a\wedge\neg c,C_c:\neg d,C_d:\neg c,C_e:\neg d\wedge\neg e\})$ considered at the end of the example was an AADF$^+$. Since every link is attacking in the ADF sense, the framework is also a BADF.

Finally, let us look at a very simple framework $(\{a,b,c\},\{C_a : \top, C_b : \top, C_c : a\oplus b\})$, where a and b are initial arguments and the condition of c is a xor between the two. a and b have trivial acyclic evaluations $((a),\emptyset)$ and $((b),\emptyset)$. For c, there are two main options: $((a,c),\{b\})$ and $((b,c),\{a\})$. These are all the basic evaluations we can get; it is thus easy to see that the framework is an AADF$^+$. However, it is not a BADF; the links from a and b to c cannot be described as supporting or attacking, only dependent.

7 Comparison of Extensions and Labelings

In this section we will compare the new extension–based semantics with the existing labeling–based ones. While, whenever possible, we will show how to construct appropriate extensions or labelings, let us introduce a very simple notion of correspondence:

Definition 7.1. Let $D=(A,C)$ be an ADF, v a three–valued interpretation over A and $E\subseteq A$ a set of arguments. v and E correspond iff $v^{\mathbf{t}}=E$.

By the abuse of notation we will also use the notion of a $\mathbf{u}$–completion, which should be understood as a three–valued interpretation that assigns $\mathbf{u}$ to the "missing" mappings of a given two–valued interpretation.

Although there will be a single set of arguments corresponding to a given interpretation, a set can have many corresponding interpretations. Moreover, even though every interpretation there will exist a corresponding set and vice versa, it might not be an extension or labeling under a given semantics. In this case we will say that there is no corresponding σ–extension or labeling, where σ is an arbitrary semantics.

We will now introduce analyze two frameworks that will serve us as a proof of lack of certain relations between some semantics.

Example 26. Let us consider another simple framework $(\{a,b\},\{C_a : a, C_b : b\})$ depicted in Figure 17. Its cc,ca$_1$ and ca$_2$–complete extensions are $\emptyset$, $\{a\}$, $\{b\}$ and $\{a,b\}$, while the aa and ac one is just $\emptyset$. Thus, we obtain one preferred cc,ca$_1$ and ca$_2$–preferred extension $\{a,b\}$ and a single aa and ac–preferred one – $\emptyset$.

The complete labelings for this framework are $\{a:\mathbf{u},b:\mathbf{u}\}$, $\{a:\mathbf{u},b:\mathbf{f}\}$, $\{a:\mathbf{f},b:\mathbf{u}\}$, $\{a:\mathbf{f},b:\mathbf{f}\}$, $\{a:\mathbf{t},b:\mathbf{u}\}$, $\{a:\mathbf{t},b:\mathbf{f}\}$, $\{a:\mathbf{u},b:\mathbf{t}\}$, $\{a:\mathbf{f},b:\mathbf{t}\}$ and finally $\{a:\mathbf{t},b:\mathbf{t}\}$. The first four correspond to $\emptyset$, then both $\{a\}$ and $\{b\}$ have two labelings, and finally we receive $\{a,b\}$. In this case, our results in compliance with the cc, ca$_1$ and ca$_2$–complete extensions. The preferred labelings are $\{a:\mathbf{f},b:\mathbf{f}\}$,$\{a:\mathbf{t},b:\mathbf{f}\}$, $\{a:\mathbf{f},b:\mathbf{t}\}$ and $\{a:\mathbf{t},b:\mathbf{t}\}$, again producing the sets $\emptyset$, $\{a\}$, $\{b\}$ and $\{a,b\}$. We can observe that $\{a\}$ and $\{b\}$ are not preferred extensions of any family.

Example 27. Let us consider a simple framework $(\{a,b,c,d\},\{C_a : \neg c, C_b : \neg d, C_c : c, C_d : d\})$ depicted in Figure 18. Its extensions and labelings will be listed in Tables 2 and 3.

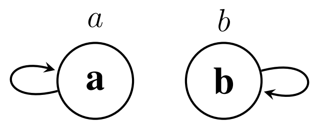


Figure 17: Sample ADF

Although there are many admissible labelings, in the end they produce the following sets: $\emptyset, \{a\}, \{b\}, \{c\}, \{d\}, \{a,b\}, \{a,d\}, \{b,c\}$ and $\{c,d\}$. Similar follows for the complete labelings. The preferred ones correspond to $\{a,b\}, \{a,d\}, \{b,c\}$ and $\{c,d\}$.

We can observe that in our example, every admissible labeling will produce a ca_1 and ca_2–admissible extension and vice versa. However, even though every aa, cc and ac–admissible extension will have a corresponding labeling, it does not hold in the other direction. Although every complete extension of a given type will have a corresponding complete labeling, the sets $\{a\}$ and $\{b\}$ produced by some of the complete labelings are not complete extension in any of the families. Finally, we can see that the ac–preferred extension $\emptyset$ has no corresponding preferred labeling.

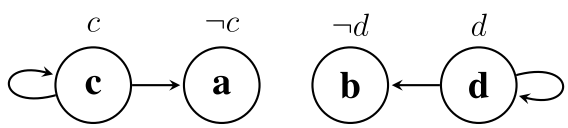


Figure 18: Sample ADF

Table 2: Extensions of the ADF from Figure 18.

ADM	$CA_{1,2}$	$\emptyset, \{a\}, \{b\}, \{c\}, \{d\}, \{a,b\}, \{a,d\}, \{b,c\}, \{c,d\}$
	CC	$\emptyset$, ~~$\{a\}, \{b\}$~~, $\{c\}, \{d\}$, ~~$\{a,b\}, \{a,d\}, \{b,c\}$~~, $\{c,d\}$
	AA	$\emptyset, \{a\}, \{b\}$, ~~$\{c\}, \{d\}$~~, $\{a,b\}$, ~~$\{a,d\}, \{b,c\}, \{c,d\}$~~
	AC	$\emptyset$, ~~$\{a\}, \{b\}, \{c\}, \{d\}, \{a,b\}, \{a,d\}, \{b,c\}, \{c,d\}$~~
COMP	$CA_{1,2}$	~~$\emptyset, \{a\}, \{b\}, \{c\}, \{d\}$~~, $\{a,b\}, \{a,d\}, \{b,c\}, \{c,d\}$
	CC	$\emptyset$, ~~$\{a\}, \{b\}$~~, $\{c\}, \{d\}$, ~~$\{a,b\}, \{a,d\}, \{b,c\}$~~, $\{c,d\}$
	AA	~~$\emptyset, \{a\}, \{b\}, \{c\}, \{d\}$~~, $\{a,b\}$, ~~$\{a,d\}, \{b,c\}, \{c,d\}$~~
	AC	$\emptyset$, ~~$\{a\}, \{b\}, \{c\}, \{d\}, \{a,b\}, \{a,d\}, \{b,c\}, \{c,d\}$~~
PREF	$CA_{1,2}$	~~$\emptyset, \{a\}, \{b\}, \{c\}, \{d\}$~~, $\{a,b\}, \{a,d\}, \{b,c\}, \{c,d\}$
	CC	~~$\emptyset, \{a\}, \{b\}, \{c\}, \{d\}, \{a,b\}, \{a,d\}, \{b,c\}$~~, $\{c,d\}$
	AA	~~$\emptyset, \{a\}, \{b\}, \{c\}, \{d\}$~~, $\{a,b\}$, ~~$\{a,d\}, \{b,c\}, \{c,d\}$~~
	AC	$\emptyset$, ~~$\{a\}, \{b\}, \{c\}, \{d\}, \{a,b\}, \{a,d\}, \{b,c\}, \{c,d\}$~~

7.1 Conflict–free Extensions and Three–Valued Models

We will start by relating conflict–freeness and three–valued models. Please note that the intuitions of two–valued and three–valued models are completely different and should not be confused – it is

Table 3: Labelings of the ADF from Figure 18.

ADM	$\{a:\mathbf{u},b:\mathbf{u},c:\mathbf{u},d:\mathbf{u}\},\{a:\mathbf{u},b:\mathbf{u},c:\mathbf{f},d:\mathbf{u}\},\{a:\mathbf{u},b:\mathbf{u},c:\mathbf{u},d:\mathbf{f}\},$ $\{a:\mathbf{u},b:\mathbf{u},c:\mathbf{f},d:\mathbf{f}\},\{a:\mathbf{u},b:\mathbf{u},c:\mathbf{t},d:\mathbf{u}\},\{a:\mathbf{u},b:\mathbf{u},c:\mathbf{u},d:\mathbf{t}\},$ $\{a:\mathbf{u},b:\mathbf{u},c:\mathbf{t},d:\mathbf{t}\},\{a:\mathbf{u},b:\mathbf{u},c:\mathbf{f},d:\mathbf{t}\},\{a:\mathbf{u},b:\mathbf{u},c:\mathbf{t},d:\mathbf{f}\},$ $\{a:\mathbf{u},b:\mathbf{f},c:\mathbf{u},d:\mathbf{t}\},\{a:\mathbf{u},b:\mathbf{f},c:\mathbf{f},d:\mathbf{t}\},\{a:\mathbf{u},b:\mathbf{f},c:\mathbf{t},d:\mathbf{t}\},$ $\{a:\mathbf{u},b:\mathbf{t},c:\mathbf{u},d:\mathbf{f}\},\{a:\mathbf{u},b:\mathbf{t},c:\mathbf{f},d:\mathbf{f}\},\{a:\mathbf{u},b:\mathbf{t},c:\mathbf{t},d:\mathbf{f}\},$ $\{a:\mathbf{f},b:\mathbf{u},c:\mathbf{t},d:\mathbf{u}\},\{a:\mathbf{f},b:\mathbf{u},c:\mathbf{t},d:\mathbf{f}\},\{a:\mathbf{f},b:\mathbf{u},c:\mathbf{t},d:\mathbf{t}\},$ $\{a:\mathbf{f},b:\mathbf{f},c:\mathbf{t},d:\mathbf{t}\},\{a:\mathbf{f},b:\mathbf{t},c:\mathbf{t},d:\mathbf{f}\},\{a:\mathbf{t},b:\mathbf{u},c:\mathbf{f},d:\mathbf{u}\},$ $\{a:\mathbf{t},b:\mathbf{u},c:\mathbf{f},d:\mathbf{f}\},\{a:\mathbf{t},b:\mathbf{u},c:\mathbf{f},d:\mathbf{t}\},\{a:\mathbf{t},b:\mathbf{f},c:\mathbf{f},d:\mathbf{t}\},$ $\{a:\mathbf{t},b:\mathbf{t},c:\mathbf{f},d:\mathbf{f}\}$
COMP	$\{a:\mathbf{u},b:\mathbf{u},c:\mathbf{u},d:\mathbf{u}\},\{a:\mathbf{f},b:\mathbf{u},c:\mathbf{t},d:\mathbf{u}\},\{a:\mathbf{u},b:\mathbf{f},c:\mathbf{u},d:\mathbf{t}\},$ $\{a:\mathbf{u},b:\mathbf{t},c:\mathbf{u},d:\mathbf{f}\},\{a:\mathbf{t},b:\mathbf{u},c:\mathbf{f},d:\mathbf{u}\},\{a:\mathbf{t},b:\mathbf{t},c:\mathbf{f},d:\mathbf{f}\},$ $\{a:\mathbf{t},b:\mathbf{f},c:\mathbf{f},d:\mathbf{t}\},\{a:\mathbf{f},b:\mathbf{t},c:\mathbf{t},d:\mathbf{f}\},\{a:\mathbf{f},b:\mathbf{f},c:\mathbf{t},d:\mathbf{t}\},$
PREF	$\{a:\mathbf{t},b:\mathbf{t},c:\mathbf{f},d:\mathbf{f}\},\{a:\mathbf{t},b:\mathbf{f},c:\mathbf{f},d:\mathbf{t}\},\{a:\mathbf{f},b:\mathbf{t},c:\mathbf{t},d:\mathbf{f}\},$ $\{a:\mathbf{f},b:\mathbf{f},c:\mathbf{t},d:\mathbf{t}\},$

just the naming that is somewhat unfortunate. Let us start with the positive results in the labeling direction:

Theorem 7.2. *Let $D = (A, C)$ be an ADF, $E \subseteq A$ a conflict–free and $S \subseteq A$ a pd–acyclic conflict–free extension of D. The $\mathbf{u}$–completions of v_E, v_E^p, v_S, v_S^p and v_S^a to A are three–valued models of D.*

However, not every range produces a three–valued model:

Theorem 7.3. *Let $D = (A, C)$ be an ADF, $E \subseteq A$ a conflict–free and v_E^a its acyclic range interpretation. The $\mathbf{u}$–completion of v_E^a might not be a three–valued model of D.*

Both of the theorems follow straightforwardly from the definition of conflict–freeness and Proposition 6.1. The three–valued models will always produce conflict–free extensions; by the first result it is easy to see that they do not have to be pd–acyclic.

Theorem 7.4. *Let $D = (A, C)$ be an ADF and v be a three–valued model of D. Then $v^{\mathbf{t}}$ is a conflict–free extension of D.*

Proof. Since v is a three–valued model, then for every $s \in A$ mapped to $\mathbf{t}$, $v(C_s) = in$. Since $v(C_s) = C_s(v^{\mathbf{t}} \cap par(s))$, conflict–freeness follows straightforwardly. □

7.2 Admissible Semantics

We can now continue with the admissible semantics. Firstly, we will tie the notion of decisiveness to admissibility, following the comparison of completions and extending interpretations that we have presented in Section 3.

Theorem 7.5. *Let $D = (A, C)$ be an ADF, v be a three–valued interpretation on A and v' its (maximal) two–valued sub–interpretation. v is admissible iff all arguments mapped to $\mathbf{t}$ are decisively in w.r.t. v' and all arguments mapped to $\mathbf{f}$ are decisively out w.r.t. v'.*

Proof. Assume v is admissible, but there exists an argument $s \in S$ mapped to $\mathbf{t}$ that is not decisively in w.r.t. v' or it is mapped to $\mathbf{f}$ and is not decisively out w.r.t. v'. This means there exists a completion v'_S of v' to S s.t. $C_s(par(s) \cap v'^{\mathbf{t}}_S)$ is out in the first or in in the latter case. Obviously, v'_S is also an extending interpretation of v, i.e. $v'_S \in [v]_2$. However, if this interpretation evaluated the condition of s to out/in, then obviously the operator could not have assigned s $\mathbf{t}$/$\mathbf{f}$ and we reach a contradiction.

Now assume a two–valued interpretation v' such as all arguments mapped to $\mathbf{t}$/$\mathbf{f}$ are decisively in/out, but its $\mathbf{u}$–completion v is not admissible. This means that $v \not\leq_i \Gamma_D(v)$. Consequently, there exists an argument s mapped to $\mathbf{t}$/$\mathbf{f}$ by v' that is assigned respectively $\mathbf{f}$ or $\mathbf{u}$/$\mathbf{t}$ or $\mathbf{u}$. This means that all/some extensions of the interpretation evaluate the condition of s to out/in. Obviously, it means that all/some completions of v' evaluated the condition of s to out/in. Therefore, the initial assignment could not have been decisive and we reach a contradiction. □

However, please note that this theorem does not imply that admissible extensions and labelings "perfectly" coincide. In labelings, we guess an interpretation, and thus assign initial values to arguments that we want to verify later. If they are self–dependent, it of course affects the outcome. In the extension based approaches, we distinguish whether this dependency is permitted. For example, in the ac–approach the accepted arguments cannot take part in support cycles, thus self–justification is not permitted. On the other hand, the iteratively built standard discarded set does not permit self–falsification. Therefore, most of the approaches will have a corresponding labeling, but the labelings will produce extensions not limited to cc, aa or ac families.

Theorem 7.6. *Let $D = (A, C)$ be an ADF. The following holds:*

1. *Let E be a cc–admissible extension of D. Then the $\mathbf{u}$–completion of v_E is an admissible labeling of D.*

2. *Let E be an ac–admissible extension of D. Then the $\mathbf{u}$–completion of v_E is an admissible labeling of D.*

3. *Let E be an aa–admissible extension of D. Then the $\mathbf{u}$–completion of v^a_E is an admissible labeling of D.*

4. *Let E be a ca$_2$–admissible extension of D. Then the $\mathbf{u}$–completion of v^p_E is an admissible labeling of D.*

5. *Let v be an admissible labeling of D. Then $v^{\mathbf{t}}$ is a ca$_1$ and ca$_2$–admissible extension of D.*

Proof.

1. The proof was provided already in [39]. However, it also straightforwardly follows from the definition of cc–admissibility, Proposition 6.1 and Theorem 7.5.

2. By Proposition 6.1 and the definition of ac–admissibility, everything mapped to $\mathbf{t}$ by v_E is decisively in w.r.t. it and everything mapped to $\mathbf{f}$ by v_E is decisively out w.r.t. it. Hence, by Theorem 7.5 the $\mathbf{u}$–completion of v_E is an admissible labeling of D.

3. By Proposition 6.1 and the definition of aa–admissibility, everything mapped to $\mathbf{t}$ by v_E^a is decisively in w.r.t. it and everything mapped to $\mathbf{f}$ by v_E^a is decisively out w.r.t. it. Hence, by Theorem 7.5 the $\mathbf{u}$–completion of v_E^a is an admissible labeling of D.

4. By Proposition 6.1 and the definition of ca$_2$–admissibility, everything mapped to $\mathbf{t}$ by v_E^p is decisively in w.r.t. it and everything mapped to $\mathbf{f}$ by v_E^p is decisively out w.r.t. it. Hence, by Theorem 7.5 the $\mathbf{u}$–completion of v_E^p is an admissible labeling of D.

5. Let v be an admissible labeling, w its (maximal) two–values subinterpretation and $E = v^{\mathbf{t}}$. We know that v is a three–valued model; thus, by Theorem 7.4 E is at least conflict–free. Let us assume it is not ca$_2$–admissible. This means there exists an argument $a \in E$ and a completion v' of v_E^p s.t. $v'(C_a) = out$. Since at the same time a is decisively in w.r.t. w and $w^{\mathbf{t}} = v^{\mathbf{t}}$, then it follows that $E^{p+} \subset w^{\mathbf{f}}$. Let $x \in w^{\mathbf{f}} \setminus E^{p+}$. Since x is not in the discarded set, it means there exists a pd–acyclic evaluation $((a_0, ..., a_n), B)$ for x on A s.t. $B \cap E = \emptyset$ or a partially acyclic evaluation $(F, (b_0, ..., b_n), B')$ for x on A s.t. $B' \cap E = \emptyset$ and $F \subseteq E$.

   Let us focus on the first case and consider a_0 and its associated decisively in interpretation v_0. Since $v_0^{\mathbf{t}} = \emptyset$ and $v_0^{\mathbf{f}} \cap E = \emptyset$, then it follows that $C_{a_0}(E \cap par(a_0)) = in$. Consequently, a_0 cannot be decisively out w.r.t. neither v_E^p nor w and thus is not mapped to $\mathbf{f}$ in any of the interpretations due to Proposition 6.1 and Theorem 7.5. Let us now focus on a_1 and its associated decisively in interpretation v_1. Since $v_1^{\mathbf{t}} \subseteq \{a_0\}$, $v_1^{\mathbf{f}} \cap E = \emptyset$ and a_0 is not falsified by neither v_E^p nor w, then there is a completion of both that is decisively in for a_1 and thus it could not have been falsified by either of them again. We can continue going up the sequence until we reach $a_n = x$ and the conclusion that it could not have been falsified by neither v_E^p nor w. We reach a contradiction with the assumption that $w(x) = \mathbf{f}$ and conclude that v could not have been an admissible labeling of D.

   Let us now focus on the other case and consider b_0 and its associated decisively in interpretation z_0. By the evaluation requirements and the fact that $F \subseteq E$, it holds that $z_0^{\mathbf{t}} \subseteq E$. As $z_0^{\mathbf{f}} \cap E = \emptyset$, then it follows that $C_{b_0}(E \cap par(b_0)) = in$. Consequently, b_0 cannot be decisively out w.r.t. neither v_E^p nor w and thus is not mapped to $\mathbf{f}$ in any of the interpretations due to Proposition 6.1 and Theorem 7.5. From this point on we can repeat the procedure above and arrive at the same conclusion that v could not have been an admissible labeling of D.

   We have thus proved that every admissible labeling has a corresponding ca$_2$–extension. Since every ca$_2$–admissible extension is ca$_1$ admissible by Lemma 6.5, then every labeling has also a corresponding ca$_1$–admissible extension.

□

Remark. Please note that although every admissible labeling has a corresponding ca$_2$–admissible extension and vice versa, just like in the Dung's framework it does not need to be a one–to–one relation. The $\mathbf{u}$–completion of a given range produces only one of many admissible labelings that have common $\mathbf{t}$ mappings.

The only family that produces labelings that might not be admissible is the ca$_1$ family:

Theorem 7.7. *Let $D = (A, C)$ be an ADF and E a ca$_1$–admissible extension of D. There might be no admissible labeling of D corresponding to E.*

Proof. Recall the framework in Example 21, where $\{a\}$ was a ca$_1$–admissible extension. The only three–valued model that maps a to $\mathbf{t}$ is $\{a : \mathbf{t}, b : \mathbf{u}\}$ and it is not an admissible labeling. □

7.3 Preferred Semantics

Let us now consider the preferred semantics. Unfortunately, due to the differences between two–valued and three–valued approaches and the fact that one follows subset maximality, while the other information precision, we fail to receive an exact correspondence between the results. By this we mean that given a framework there can exist an (arbitrary) preferred extension without a labeling counterpart and a labeling without an appropriate extension, even though certain inclusions relation can be derived. Our positive results are thus quite limited:

Theorem 7.8. *Let $D = (A, C)$ be an ADF, $E \subseteq A$ a ca$_2$–preferred extension and v a preferred labeling of D. The $\mathbf{u}$–completion of v_E^p to A is a preferred labeling of D. The set $v^{\mathbf{t}}$ is a ca$_2$–complete extension, but not necessarily a aa, ac, cc or ca$_1$–complete one. It also might not be an aa, ac, cc, ca$_1$ or ca$_2$–preferred extension of D.*

Proof. We will first analyze the extension case. Let v be the $\mathbf{u}$–completion of v_E^p. By Theorem 7.6 we know that v is at least an admissible labeling. Let us assume it is not a preferred one. This means there exists an admissible labeling v' of D s.t. $v \leq_i v'$ and $v \neq v'$. Consequently, $v^{\mathbf{t}} \subset v'^{\mathbf{t}}$ or $v^{\mathbf{f}} \subset v'^{\mathbf{f}}$. If it is the first case, then by Theorem 7.6 it means that $v'^{\mathbf{t}}$ is a ca$_2$–admissible extension and thus E could not have been ca$_2$–preferred. Let us thus assume that $v^{\mathbf{f}} \subset v'^{\mathbf{f}}$ and let w be the maximal two–valued subinterpretation of v'. This means there exists an argument $a \in v'^{\mathbf{f}} \setminus v^{\mathbf{f}}$ that is decisively out w.r.t. w, but has a pd–acyclic evaluation $((a_0, ..., a_n), B)$ not blocked by v_E^p or a partially acyclic one $(F, (a_0, ..., a_n), B)$ s.t. it is not blocked by v_E^p and $F \subseteq E$.

Let us focus on the acyclic evaluation and first consider a_0 and its corresponding decisively in interpretation v_0 with which it entered the evaluation. We know that $v_0^{\mathbf{t}} = \emptyset$. Consequently, if a_0 were to be decisively out w.r.t. w, it would have to be the case that $w^{\mathbf{t}} \cap v_0^{\mathbf{f}} \neq \emptyset$. Since $w^{\mathbf{t}} = E$ and $E \cap v_0^{\mathbf{f}} = \emptyset$ due to the fact that the evaluation is unblocked, it cannot be the case that a_0 is decisively out w.r.t. w and thus it cannot be mapped to false by w. Let us continue with a_1 and its decisively in interpretation v_1. We know that $v_1^{\mathbf{t}} \subseteq \{a_0\}$. Since a_0 is not mapped to $\mathbf{f}$ by w and no element of $v_1^{\mathbf{f}}$ is mapped to $\mathbf{t}$ by w by the discussion above, it cannot be the case that a_1 is decisively out w.r.t. w. Thus, it is not mapped to $\mathbf{f}$ by w. We can continue in the same manner until we reach the conclusion that it cannot be the case that an argument is decisively out w.r.t. w and at the same time has an acyclic evaluation unblocked by v_E^p.

Let us focus now on the partially acyclic case. It is is easy to see that $F \subseteq w^{\mathbf{t}}$ and it cannot be the case that any of the arguments in F is decisively out w.r.t. w. Let us thus consider a_0 and its decisively in interpretation v_0 with which it entered the evaluation. Since $v_0^{\mathbf{t}} \subseteq F$ and $v_0^{\mathbf{f}} \cap w^{\mathbf{t}} = \emptyset$, then it is easy to see that a_0 cannot be decisively out w.r.t. w. Consequently, it cannot be mapped to $\mathbf{f}$. We can now proceed in a manner similar to the one above and again come to a conclusion and that a is decisively out w.r.t. w and at the same time has a partially acyclic evaluation unblocked by v_E^p and with a pd–set in E. Therefore, we reach a contradiction and can conclude that the v is a preferred labeling.

We will now analyze the labeling case. Let v be a preferred labeling and w its maximal two–valued subinterpretation. Let $v^{\mathbf{t}} = E$. Since every preferred labeling is also admissible, then by Theorem 7.6 E is at least ca$_2$ admissible. Let z be the $\mathbf{u}$–completion of v_E^p. By the definition of preferred labelings, it cannot be the case that $w^{\mathbf{f}} \subset z^{\mathbf{f}}$. We are thus left with cases in which $z^{\mathbf{f}} \subseteq w^{\mathbf{f}}$ or where the two sets are not comparable. Let us consider a two–valued interpretation x where $x^{\mathbf{t}} = E$, $x^{\mathbf{f}} = z^{\mathbf{f}} \cup w^{\mathbf{f}}$. Since x is a completion of both z and w, it follows that whatever is decisively in w.r.t. z or w is decisively in w.r.t. x and whatever is decisively out w.r.t. z or w is decisively out w.r.t. x. It is thus easy to see that the $\mathbf{u}$–completion of x is an admissible labeling. If the false mappings of z and w formed incomparable sets, then the $\mathbf{u}$–completion of x would contain more information than v and thus it could not have been the case that v was a preferred labeling. Therefore, it can only be the case that $z^{\mathbf{f}} \subseteq w^{\mathbf{f}}$ and w is a completion of v_E^p. If E is not ca$_2$–complete, then there exists an argument $a \in A \setminus E$ that is decisively in w.r.t. v_E^p and thus w.r.t. w as well. Thus, v could not have been a complete labeling and by Theorem 4.4, not a preferred one either. We reach a contradiction. Hence, if v is a preferred labeling, then $v^{\mathbf{t}}$ is a ca$_2$–complete extension.

As for counterexamples, please look at Example 26. The produced labelings correspond to sets $\emptyset, \{a\}, \{b\}$ and $\{a, b\}$. All of them are ca$_2$–complete, but only $\emptyset$ is aa/ac–complete. Furthermore, only $\{a, b\}$ and $\emptyset$ correspond to the preferred extensions we have obtained. In Example 27 we have obtained a preferred labeling corresponding to set $\{a, b\}$, which was neither cc–complete nor cc–preferred. Since not every ca$_1$–complete extension is ca$_2$–complete, we can finally conclude that the preferred labelings might not correspond to extension–based complete and preferred semantics with the exception of ca$_2$–complete. □

Theorem 7.9. *Let $D = (A, C)$ be an ADF. The following holds:*

1. *Let $E \subseteq A$ be an aa–preferred extension. There might be no preferred labeling of D corresponding to E.*

2. *Let $E \subseteq A$ be an ac–preferred extension. There might be no preferred labeling of D corresponding to E.*

3. *Let $E \subseteq A$ be a cc–preferred extension. There might be no preferred labeling of D corresponding to E.*

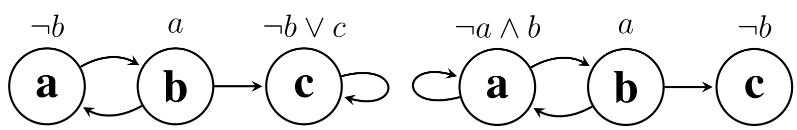
4. *Let $E \subseteq A$ be a ca$_1$–preferred extension. There might be no preferred labeling of D corresponding to E.*

Proof. See Example 19 for examples of extensions without corresponding labelings of a given type. □

Example 28. Recall the framework described in Example 21 and visible in Figure 14. The ca$_1$–preferred extension is $\{a\}$ and the only preferred labeling is $\{a : \mathbf{f}, b : \mathbf{f}\}$.

Let us now look at $ADF_1 = (\{a, b, c\}, \{a : \neg a, b : a, c : \neg b \vee c\})$, as depicted in Figure 19a. The only ac and aa–preferred extension is $\emptyset$. a and b cannot form a conflict–free extension to start with, so we are only left with c. However, the attack from b on c can be only overpowered by self–support, thus it cannot be part of an ac or aa–admissible extension in the first place. The single preferred labeling solution would be $v = \{a : \mathbf{u}, b : \mathbf{u}, c : \mathbf{t}\}$ and we obtain no correspondence. On the other hand, the result is in compliance with the cc, ca$_1$ and ca$_2$–preferred extension $\{c\}$.

Finally, we have $ADF_2 = (\{a, b, c\}, \{a : \neg a \wedge b, b : a, c : \neg b\})$ depicted in Figure 19b. The preferred labeling is $\{a : \mathbf{f}, b : \mathbf{f}, c : \mathbf{t}\}$. The single cc and ac–preferred extension is $\emptyset$ and again, we receive no correspondence. However, it is compliance with the aa, ca$_1$ and ca$_2$–preferred extension $\{c\}$.



(a) ADF_1 (b) ADF_2

Figure 19: Sample ADFs

Remark. We have already shown that stable extensions are aa–preferred. It also holds that they are labeling preferred. This means that although perfect correspondence will not be retrieved, in case a stable model exists we have at least one "meeting point" between the two preferred approaches.

7.4 Complete Semantics

Let us first explain complete labeling in terms of decisiveness:

Theorem 7.10. *Let $D = (A, C)$ be an ADF, v an admissible labeling and v' its (maximal) two–valued sub–interpretation. v is complete iff all arguments decisively out w.r.t. v' are mapped to* $\mathbf{f}$ *by v and all arguments decisively in w.r.t. v' are mapped to* $\mathbf{t}$ *by v.*

Proof. Assume that v is complete, but there exists an argument $s \in A$ that is decisively in/out w.r.t. v' and is not mapped to $\mathbf{t}/\mathbf{f}$ by v'. Since v is a three–valued model, then the outcome of the condition of a given argument mapped to $\mathbf{t}$ must be in and of one mapped to $\mathbf{f}$ has to be out. Thus, it cannot be the case that a decisively in argument is mapped to $\mathbf{f}$ and a decisively out one to $\mathbf{t}$. Therefore, s has to be mapped to $\mathbf{u}$ by v. However, since s is decisively in/out w.r.t. v' and all mappings in v' are already decisively in/out depending on their assignment, then the characteristic operator applied to v will produce a labeling v'' where the s assignment is replaced accordingly. Since $v \leq_i v''$, then v could not have been a fixpoint of the operator and thus could not have been a complete labeling.

Let us now focus on the other way around and assume that even though all decisive arguments are have according mappings in v, v is not complete. Let $v'' = \Gamma_D(v)$. Since v is admissible, but not complete, it means that $v \leq_i v''$ but not $v = v''$. Thus, there exists at least one argument $s \in A$ s.t. $v(s) = \mathbf{u}$ and $v''(s) \neq \mathbf{u}$. If $v''(s) = \mathbf{f}$, then by the way characteristic operator works it means that all completions of v' evaluated the condition of s to out. This means that s was decisively out w.r.t. v', but was not mapped to $\mathbf{f}$ by v; we reach a contradiction. If $v''(s) = \mathbf{t}$, then by a similar analysis s was decisively in w.r.t. v' but not mapped to $\mathbf{t}$ by v. Again, we reach a contradiction.

□

With the obvious exception of ca_1–complete semantics, we have that every cc, ac, aa and ca_2–complete extension has a corresponding complete labeling. However, also the labelings produce sets that might not be complete extensions of any type.

Theorem 7.11. *Let $D = (A, C)$ be an ADF. The following holds:*

1. *Let E be a cc–complete extension of D. The $\mathbf{u}$–completion of v_E is a complete labeling of D.*
2. *Let E be a ac–complete extension of D. The $\mathbf{u}$–completion of v_E is a complete labeling of D.*
3. *Let E be an aa–complete extension of D. The $\mathbf{u}$–completion of v^a_E is a complete labeling of D.*
4. *Let E be a ca_2–complete extension of D. The $\mathbf{u}$–completion of v^p_E is a complete labeling of D.*
5. *Let v be a complete labeling of D. There exists a ca_2–complete extension E of D s.t. $v^{\mathbf{t}} \subseteq E$.*

Proof.

1. By the definition of cc–completeness, all arguments that are decisively in w.r.t. v_E are already in E (and thus mapped to $\mathbf{t}$ by v_E). By the definition of the discarded set (and standard range), every argument decisively out w.r.t. v_E is mapped to $\mathbf{f}$ by v_E. Since the $\mathbf{u}$–completion of v_E is an admissible labeling by Theorem 7.5, then by Theorem 7.10 it is also a complete one.

2. By the definition of ac–completeness, all arguments that are decisively in w.r.t. v_E are already in E (and thus mapped to $\mathbf{t}$ by v_E). By the definition of the discarded set (and standard range), every argument decisively out w.r.t. v_E is mapped to $\mathbf{f}$ by v_E. Since the $\mathbf{u}$–completion of v_E is an admissible labeling by Theorem 7.5, then by Theorem 7.10 it is also a complete one.

3. By the definition of aa–completeness, all arguments that are decisively in w.r.t. v_E^a are already in E (and thus mapped to $\mathbf{t}$ by v_E^a). We also know by Proposition 6.1 that all arguments mapped to $\mathbf{f}$ by v_E^a are decisively out. We now need to show that all arguments decisively out w.r.t. v_E^a are mapped to $\mathbf{f}$ by v_E^a, i.e. all decisively out arguments are in E^{a+}. Assume there is an argument $a \in A$ that is decisively out w.r.t. v_E^a, but not falsified by it. However, if v_E^a has the power to decisively out a, then any decisively in interpretation for a has to "prevent" this interpretation from happening. Thus, v_E^a conflicts all (minimal) interpretations for which a is decisively in. Consequently, it has the means to block any acyclic pd–evaluation of a and by definition of the acyclic range interpretation, a must have already been mapped to $\mathbf{f}$ by v_E^a. Since the $\mathbf{u}$–completion of v_E^a is an admissible labeling by Theorem 7.5, then by Theorem 7.10 it is also complete.

4. By the definition of ca$_2$–completeness, all arguments that are decisively in w.r.t. v_E^p are already mapped to $\mathbf{t}$ by v_E^p. We also know by Proposition 6.1 that all arguments mapped to $\mathbf{f}$ by v_E^p are decisively out. By an analysis similar to the one as in aa–case, we can also show that there is no argument that is decisively out w.r.t. v_E^p, but is not mapped to $\mathbf{f}$. Since the $\mathbf{u}$–completion of v_E^p is an admissible labeling by Theorem 7.5, then by Theorem 7.10 it is also complete.

5. Every preferred labeling has a corresponding ca$_2$–complete extension by Theorem 7.8. Since every complete labeling is "contained" in some preferred labeling, it follows that for every complete labeling there is a ca$_2$–complete extension containing its $\mathbf{t}$ mappings.

□

Theorem 7.12. *Let $D = (A, C)$ be an ADF, E a ca$_1$–complete extension and v a complete labeling of D. There might be no complete labeling of D corresponding to E. The set $v^{\mathbf{t}}$ might not be a cc, ac, aa, ca$_1$ or ca$_2$–complete extension of D.*

Proof. Recall Example 21, where $\{a\}$ was a ca$_1$–complete extension. There existed no admissible labeling corresponding to it, thus no complete one will exist either.

Recall Example 27. The complete labelings corresponded to $\emptyset, \{a\}, \{b\}, \{c\}, \{d\}, \{a, b\}, \{a, d\}, \{b, c\}$ and $\{c, d\}$. Only the last four were ca$_1$/ca$_2$–complete, no set containing a or b was cc–complete, only $\{a, b\}$ was aa–complete and $\emptyset$ ac–complete.

□

7.5 Grounded Semantics

As the grounded semantics has a very clear meaning, it is no wonder that both available approaches coincide, as already noted in [11].

Theorem 7.13. *Let $D = (A, C)$ be an ADF. The two–valued grounded extension of D and the grounded labeling of D correspond.*

However, in the acyclic grounded case, the best we can get is that it has a complete labeling. It will of course not be the least one, since that corresponds to the standard grounded semantics.

Theorem 7.14. *Let $D = (A, C)$ be an ADF and E its acyclic grounded extension. The* $\mathbf{u}$*–completion of the acyclic range of E is a complete labeling of D.*

Proof. Follows straightforwardly from Theorems 6.12 and 7.11.

□

7.6 Comparison of Extensions and Labelings on AADF$^+$

In Section 6.4 we have defined a subclass of ADFs for which our xy–classification collapsed. In this section we will show that AADF$^+$ not only simplify the extension–based semantics, but are also sufficient for a more precise correspondence between extension and labeling–based ones.

Theorem 7.15. *Let $D = (A, C)$ be an AADF$^+$. The following holds:*

1. *Every admissible labeling of D has a corresponding aa, ac, cc, ca$_1$ and ca$_2$–admissible extension and vice versa.*

2. *Every complete labeling of D has a corresponding aa, ac, cc, ca$_1$ and ca$_2$–complete extension and vice versa.*

3. *Every preferred labeling of D has a corresponding aa, ac, cc, ca$_1$ and ca$_2$–preferred extension and vice versa.*

Proof.

1. Follows from Theorem 6.17 and 7.6.

2. The fact that for every aa/ac/cc/ca$_1$/ca$_2$–complete extension there exists a corresponding labeling follows from Theorems 6.17 and 7.11. Let us now focus on the other way around; let v be a complete labeling and $E = v^{\mathbf{t}}$ its $\mathbf{t}$ mappings. By the first point we know that E is at least cc–admissible. The fact that $v_E^{\mathbf{f}} \subseteq v^{\mathbf{f}}$ follows easily from the iterative definition of standard range, Theorem 7.10 and the fact that complete labelings are fix–points of the characteristic operator. Thus, v is a completion of v_E and whatever is decisively in w.r.t. v_E is decisively in w.r.t. v. Moreover, an argument decisively in w.r.t. v_E has a condition satisfied by E and thus cannot be mapped to $\mathbf{f}$ by v. Consequently, if E was not cc–complete, then v could not have been a complete labeling. Since E is cc–complete, then by Theorem 6.17 it is also aa/ac/ca$_1$/ca$_2$–complete.

3. By Theorem 7.8 we know that every ca$_2$–preferred extension has a corresponding preferred labeling. Since by Theorem 6.17 aa/ac/cc/ca$_1$/ca$_2$–preferred extensions coincide, it follows that every aa/ac/cc/ca$_1$–preferred extension also has a corresponding preferred labeling.

   Let v be a preferred labeling, w its maximal two–valued subinterpretation and $v^{\mathbf{t}} = E$. By Theorem 7.8 we know that E is at least ca$_2$–complete. Let us assume it is not ca$_2$ preferred. This means there exists a ca$_2$–complete extension $E' \subseteq A$ s.t. $E \subset E'$. Let z be the $\mathbf{u}$–completion of $v^p_{E'}$. By Theorem 7.11 we know that z is a complete labeling. In order to show that v could not have been a preferred labeling, we need to prove that z contains more information than v. We already know that $v^{\mathbf{t}} \subset z^{\mathbf{t}}$; what remains is to show that $v^{\mathbf{f}} \subseteq z^{\mathbf{f}}$.

   Let us assume it is not the case; this means that either $z^{\mathbf{f}} \subset v^{\mathbf{f}}$ or the two sets are incomparable, which in any case means that there is an argument $a \in v^{\mathbf{f}}$ that is decisively out w.r.t. w, but not w.r.t. $v^p_{E'}$. Thus, a has a pd–acyclic evaluation unblocked by v or a partially acyclic one that is not blocked by v and has a pd–set contained in E'. We can show it cannot be the first case by going through the evaluation of a, as we have done in number of different proofs (for example, see the proof of Theorem 7.8). What is more interesting, is the latter case. Since $E \subset E'$, for general ADFs it can in fact be the case that a has an unblocked partially acyclic evaluation with a pd–set in E', but not in E, thus a can be decisively out w.r.t. w. However, since we are dealing with AADF$^+$, every standard (and thus also partially acyclic) evaluation can be made acyclic. Consequently, the pd–set of the partial evaluation for a will be in fact empty. We thus come back to the first case and conclude that it cannot be the case hat an argument is decisively out w.r.t. w, but not w.r.t. $v^p_{E'}$. Therefore, $v^{\mathbf{f}} \subseteq z^{\mathbf{f}}$ and if $v^{\mathbf{t}}$ is not ca$_2$–preferred, then v cannot be a preferred labeling. Since the classification collapses by Theorem 6.17, the correspondence for other preferred extension–based semantics follows.

□

8 Concluding Remarks

In this report we have updated and followed up on our previous work in [35], in which we have introduced a family of extension–based semantics, showed a number of their properties and provided an initial comparison of extension and labeling–based approaches. In this update we introduced new types of argument evaluations, new subfamily of the ca–semantics and the acyclic grounded semantics. We also proved a number of new properties and completed the analysis of the relation between labeling–based and extension–based semantics. Finally, we corrected various minor issues and clarified some explanations.

References

[1] Leila Amgoud and Srdjan Vesic. A new approach for preference-based argumentation frameworks. *Ann. Math. Artif. Intell*, 63:149–183, 2011.

[2] Katie Atkinson, Trevor J. M. Bench-Capon, and Peter McBurney. PARMENIDES: Facilitating deliberation in democracies. *Artif. Intell. Law*, 14(4):261–275, 2006.

[3] Pietro Baroni, Martin Caminada, and Massimiliano Giacomin. An introduction to argumentation semantics. *Knowledge Eng. Review*, 26(4):365–410, 2011.

[4] Pietro Baroni, Federico Cerutti, Massimiliano Giacomin, and Giovanni Guida. Encompassing attacks to attacks in abstract argumentation frameworks. In Claudio Sossai and Gaetano Chemello, editors, *Symbolic and Quantitative Approaches to Reasoning with Uncertainty*, volume 5590 of *LNCS*, pages 83–94. Springer Berlin Heidelberg, 2009.

[5] Pietro Baroni and Massimiliano Giacomin. Semantics of abstract argument systems. In Guillermo Simari and Iyad Rahwan, editors, *Argumentation in Artificial Intelligence*, pages 25–44. Springer, 2009.

[6] Pietro Baroni, Massimiliano Giacomin, and Giovanni Guida. SCC-Recursiveness: A general schema for argumentation semantics. *Artif. Intell.*, 168(1-2):162–210, 2005.

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

[8] Trevor J. M. Bench-Capon, Henry Prakken, and Giovanni Sartor. Argumentation in legal reasoning. In Guillermo Simari and Iyad Rahwan, editors, *Argumentation in Artificial Intelligence*, pages 363–382. Springer, 2009.

[9] Gustavo A. Bodanza and Fernando A. Tohmé. Two approaches to the problems of self-attacking arguments and general odd-length cycles of attack. *Journal of Applied Logic*, 7(4):403 – 420, 2009. Special Issue: Formal Models of Belief Change in Rational Agents.

[10] Guido Boella, Dov Gabbay, Leendert van der Torre, and Serena Villata. Support in abstract argumentation. In *Proc. of COMMA 2010*, pages 111–122, Amsterdam, The Netherlands, The Netherlands, 2010. IOS Press.

[11] Gerhard Brewka, Stefan Ellmauthaler, Hannes Strass, Johannes Peter Wallner, and Stefan Woltran. Abstract dialectical frameworks revisited. In *Proc. IJCAI'13*, pages 803–809. AAAI Press, 2013.

[12] Gerhard Brewka, Sylwia Polberg, and Stefan Woltran. Generalizations of Dung frameworks and their role in formal argumentation. *Intelligent Systems, IEEE*, 29(1):30–38, Jan 2014.

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

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

[15] Martin W. A. Caminada, Walter A. Carnielli, and Paul E. Dunne. Semi-stable semantics. *J. Log. and Comput.*, 22(5):1207–1254, October 2012.

[16] Claudette Cayrol and Marie-Christine Lagasquie-Schiex. Bipolar abstract argumentation systems. In Guillermo Simari and Iyad Rahwan, editors, *Argumentation in Artificial Intelligence*, pages 65–84. 2009.

[17] Claudette Cayrol and Marie-Christine Lagasquie-Schiex. Coalitions of arguments: A tool for handling bipolar argumentation frameworks. *Int. J. Intell. Syst.*, 25(1):83–109, 2010.

[18] Claudette Cayrol and Marie-Christine Lagasquie-Schiex. Bipolarity in argumentation graphs: Towards a better understanding. *Int. J. Approx. Reasoning*, 54(7):876–899, 2013.

[19] Andrea Cohen, Sebastian Gottifredi, Alejandro J. García, and Guillermo R. Simari. A survey of different approaches to support in argumentation systems. *The Knowledge Engineering Review*, FirstView:1–38, 2013.

[20] Sylvie Coste-Marquis, Caroline Devred, and Pierre Marquis. Inference from controversial arguments. In Geoff Sutcliffe and Andrei Voronkov, editors, *Proc. LPAR '05*, volume 3835 of *LNCS*, pages 606–620. Springer Berlin Heidelberg, 2005.

[21] Sylvie Coste-Marquis, Caroline Devred, and Pierre Marquis. Prudent semantics for argumentation frameworks. In *Proc. of ICTAI'05*, pages 568–572, Washington, DC, USA, 2005. IEEE Computer Society.

[22] Sylvie Coste-Marquis, Sébastien Konieczny, Pierre Marquis, and Mohand Akli Ouali. Weighted attacks in argumentation frameworks. In *Proc. KR*, pages 593–597. AAAI Press, 2012.

[23] Phan Minh Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artif. Intell.*, 77:321–357, 1995.

[24] Phan Minh Dung and Phan Minh Thang. A unified framework for representation and development of dialectical proof procedures in argumentation. In *Proc. of IJCAI'09*, pages 746–751, San Francisco, CA, USA, 2009. Morgan Kaufmann Publishers Inc.

[25] Stefan Ellmauthaler. Abstract dialectical frameworks: properties, complexity, and implementation. Master's thesis, Faculty of Informatics, Institute of Information Systems, Vienna University of Technology, 2012.

[26] John Fox, David Glasspool, Vivek Patkar, Mark Austin, Liz Black, Matthew South, Dave Robertson, and Charles Vincent. Delivering clinical decision support services: There is nothing as practical as a good theory. *Journal of Biomedical Informatics*, 43(5):831–843, 2010.

[27] Anthony Hunter and Matthew Williams. Aggregating evidence about the positive and negative effects of treatments. *Artificial Intelligence in Medicine*, 56(3):173–190, 2012.

[28] Hadassa Jakobovits and Dirk Vermeir. Dialectic semantics for argumentation frameworks. In *Proc. of ICAIL '99*, pages 53–62, New York, NY, USA, 1999. ACM.

[29] Peter McBurney and Simon Parsons. Dialogue games for agent argumentation. In Guillermo Simari and Iyad Rahwan, editors, *Argumentation in Artificial Intelligence*, pages 261–280. Springer, 2009.

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

[31] Søren Nielsen and Simon Parsons. A generalization of Dung's abstract framework for argumentation: Arguing with sets of attacking arguments. In *Proc. ArgMAS*, volume 4766 of *LNCS*, pages 54–73. Springer, 2007.

[32] Farid Nouioua. AFs with necessities: Further semantics and labelling characterization. In Weiru Liu, V.S. Subrahmanian, and Jef Wijsen, editors, *Proc. SUM '13*, volume 8078 of *LNCS*, pages 120–133. Springer Berlin Heidelberg, 2013.

[33] Farid Nouioua and Vincent Risch. Argumentation frameworks with necessities. In *Proc. SUM*, volume 6929 of *LNCS*, pages 163–176. Springer, 2011.

[34] Nir Oren and Timothy J. Norman. Semantics for evidence-based argumentation. In *Proc. COMMA '08*, volume 172 of *Frontiers in Artificial Intelligence and Applications*, pages 276–284. IOS Press, 2008.

[35] Sylwia Polberg. Extension–based semantics of abstract dialectical frameworks. Technical Report DBAI-TR-2014-85, Institute for Information Systems, Technical University of Vienna, 2014.

[36] Sylwia Polberg. Extension-based semantics of abstract dialectical frameworks. *CoRR*, abs/1405.0406, 2014.

[37] Sylwia Polberg. Extension-based semantics of abstract dialectical frameworks. In *STAIRS 2014 - Proceedings of the 7th European Starting AI Researcher Symposium, Prague, Czech Republic*, pages 240–249, 2014.

[38] Sylwia Polberg and Nir Oren. Revisiting support in abstract argumentation systems. In *Proceedings of COMMA 2014*, 2014.

[39] Sylwia Polberg, Johannes Peter Wallner, and Stefan Woltran. Admissibility in the abstract dialectical framework. In *Proc. CLIMA'13*, volume 8143 of *LNCS*, pages 102–118. Springer, 2013.

[40] Henry Prakken. Models of persuasion dialogue. In Guillermo Simari and Iyad Rahwan, editors, *Argumentation in Artificial Intelligence*, pages 281–301. Springer, 2009.

[41] Iyad Rahwan and Guillermo R. Simari. *Argumentation in Artificial Intelligence*. Springer, 1st edition, 2009.

[42] Hannes Strass. Approximating operators and semantics for abstract dialectical frameworks. *Artificial Intelligence*, 205:39 – 70, 2013.

[43] Hannes Strass. Instantiating knowledge bases in abstract dialectical frameworks. In *Proc. CLIMA'13*, volume 8143 of *LNCS*, pages 86–101. Springer, 2013.

[44] Hannes Strass and Johannes Peter Wallner. Analyzing the Computational Complexity of Abstract Dialectical Frameworks via Approximation Fixpoint Theory. In *Proceedings of the Fourteenth International Conference on the Principles of Knowledge Representation and Reasoning (KR)*, pages 101–110, Vienna, Austria, July 2014. AAAI Press.