Joint Attacks and Accrual in Argumentation Frameworks

Antonis Bikakis
University College London (UCL), UK
a.bikakis@ucl.ac.uk

Andrea Cohen
Institute for Computer Science and Engineering, CONICET-UNS
Dept. of Computer Science and Engineering, Universidad Nacional del Sur
Bahía Blanca, Argentina
ac@cs.uns.edu.ar

Wolfgang Dvořák
Institute of Logic and Computation, TU Wien, Austria
dvorak@dbai.tuwien.ac.at

Giorgos Flouris
Foundation for Research and Technology - Hellas (FORTH), Greece
fgeo@ics.forth.gr

Simon Parsons
University of Lincoln, UK
sparsons@lincoln.ac.uk

Abstract

While modelling arguments, it is often useful to represent "joint attacks", i.e., cases where multiple arguments jointly attack another (note that this is different from the case where multiple arguments attack another in isolation). Based on this remark, the notion of joint attacks has been proposed as a useful extension of classical Abstract Argumentation Frameworks, and has been shown to constitute a genuine extension in terms of expressive power. In this chapter, we review various works considering the notion of joint attacks from various perspectives, including abstract and structured frameworks. Moreover, we present results detailing the relation among frameworks with joint attacks and classical argumentation frameworks, computational aspects, and applications of joint attacks. Last but not least, we propose a roadmap for future research on the subject, identifying gaps in current research and important research directions.

Vol. 8 No. 6 2021
Journal of Applied Logics — IfCoLog Journal of Logics and their Applications

1 Introduction

As many have already pointed out, the work of Dung [40] is a cornerstone, arguably *the* cornerstone, of current work on computational argumentation. It was the work that introduced the notion of abstract argumentation and the idea that argumentation could be modelled just as a set of arguments and attacks between them, and it provided an initial set of semantics — complete, grounded, preferred and stable — for the evaluation of a set of arguments and attacks. As such, it is the work on which all subsequent work on abstract argumentation has been built. In addition, because many structured argumentation systems adopt the Dung semantics as a means of establishing which arguments are acceptable, these systems are also built upon [40].

Much of the appeal of [40] lies in its elegant simplicity. The approach relies on just two concepts — arguments and attacks — and yet these simple components can capture a complex range of types of reasoning, reflected in the large set of semantics that have been defined for abstract argumentation systems. However, this very simplicity means that abstract argumentation has limitations in terms of what it can represent. The limitations of representing arguments as atomic entities is widely recognised, and is addressed by work on structured argumentation[1]. However, there are also limitations in the way that [40] handles interactions between arguments. Attacks are binary, so that a given attack is from a single argument to a single argument. Attacks are also atomic in the sense that their impact is assessed independently of other attacks. To use the terminology of [6], an argument will be `out` as soon as it is attacked by a single `in` argument, regardless of any other attacks that may exist. The evaluation of an argument does not, even where arguments have different strengths, take account of whether there are multiple attacks on it. Where strengths are taken into account, it is, effectively, only the strongest attacker that matters.

These limitations, and in particular how they may be overcome, is the subject of this chapter. We are primarily interested in the extension of the [40] model of abstract argumentation to allow non-binary, or "joint" attacks. In particular, we consider the "sets of attacking arguments" (SETAF) approach first suggested in [84]. In this approach it is possible to model situations in which two or more arguments jointly attack a single argument, and we explore this approach in depth. This focus also leads us to consider bipolar argumentation frameworks, where joint attacks are a key element, and these frameworks, in turn, lead us to consider joint supports between arguments. We also briefly discuss how joint attacks might be modelled in

[1] In some systems of structured argumentation, ASPIC+ [81] for example, it is possible to cleanly "lift" a set of abstract arguments from a set of structured arguments in such a way that Dung-style semantics can be applied. In other systems, DeLP [65] for example, this is not possible.

structured argumentation, and touch on the rather neglected topic of *accrual*, which models situations in which the strength of sets of independent attacking arguments is an aggregate of the strengths of the arguments it contains.

The rest of this chapter is structured as follows. Section 2 motivates the study of joint attacks. Section 3 is perhaps the most central section of the chapter. It introduces the formal model of SETAFs, relates the model to classical abstract argumentation models, considers the computational aspects of SETAFs, and looks at alternative formulations for set-based attacks. Section 4 looks at the uses of joint attacks in bipolar argumentation frameworks, and considers the models of joint support that occur in those frameworks as well, while also discussing the use of joint attacks to model higher-order interactions. Section 5 then briefly covers the related topic of accrual, the combination of arguments for or against a given claim. Finally Section 6 looks at future lines of work on joint attacks, and Section 7 provides a brief summary and draws some conclusions.

2 Motivating the need for joint attacks

There are a number of possible motivations for work on joint attacks. One comes from a purely formal consideration of [40]. Dung [40] considers argumentation frameworks that take the form of a directed graph, with nodes being arguments and edges being attacks between arguments. It is natural to consider a generalisation of these frameworks to ones where the directed graph becomes a directed hypergraph. In its most general form, such a framework would have nodes that represent sets of arguments, and edges that represent attacks between sets of nodes[2]. What we study here is a less general representation in which nodes represent single arguments, and edges represents attacks where the attackers can be a set, but the attackee is constrained to be a singleton. Though less general than the representation just sketched, this is, as we discuss below, a genuine extension of the Dung argumentation framework.

This representation can also be motivated by considering knowledge that is most elegantly represented in a formalism that allows for joint attacks. For example, taken from [60], consider the following aspects of the UK laws governing marriage and civil

[2] [84] briefly considers explicitly representing the most general case of sets of arguments as both attacker and attackee, before settling on the SETAF formalism that we describe below. As [84] points out, SETAFs were originally devised during work that allowed arguments about Bayesian networks — work summarised in [83] — and not only is the SETAF approach able to capture attacks of sets of arguments on sets of arguments (by attacking each member of the attackee set separately), but it also mirrors the structure of a Bayesian network where conditional probability distributions capture multiple parents affecting a common child, but do not capture a single parent affecting multiple children.

partnerships[3] (as of early 2020). One is not allowed to enter into a marriage or civil partnership if

(a) you are under 16;

(b) you are closely related to your partner;

(c) you are not single; or

(d) you are under 18 and do not have permission to marry from your parents or guardians.

Much of this can be represented in a standard Dung argumentation framework, with an argument to represent the right to get married or enter a civil partnership (M), which is attacked by arguments that represent being under 16 ($A16$), being closely related (R), and not being single (NS). One might also represent case (d) with a single argument, but this single argument captures both being under 18 *and* not having permission — let's call this argument MWP (for minor without permission). That is fine on its own, but now consider adding additional information about the UK legal system into the framework, [60] again, this time on voting rights. In the UK you are allowed to vote[4], unless you are under 18, and the natural way to capture this is with an argument (V) representing the right to vote, which is attacked by an argument ($A18$) representing being under 18. How, then, do we capture the relationship between MWP, which incorporates the fact that the individual in question is under 18, and $A18$? We would argue that a natural and elegant way to do this is by replacing MWP by the argument NP, representing the fact that there is no parental permission, and having $A18$ and NP jointly attack M. The resulting SETAF is shown in Figure 1.

Just to make the point that this example of a joint attack is not contrived, Figure 1 contains some other arguments that are found in UK legislation and have a natural representation as a SETAF. (These all reflect the age of majority in the UK, which, as one might expect, crops up a lot in the law.) For example, consider the law around alcohol consumption[5]. In the UK, one is allowed to consume alcohol in public (Alc), unless one is under 16, or one is under 18 and not accompanied by an adult (NA), or one is under 18 and not having a meal (NM).

Of course, we are not claiming that using joint attacks is the *only* way to represent the above information. As we mentioned, it is possible to capture all of this in a standard abstract argumentation framework, using what are effectively compound

[3] `https://www.gov.uk/marriages-civil-partnerships`
[4] `https://www.gov.uk/elections-in-the-uk`
[5] `https://www.gov.uk/alcohol-young-people-law`

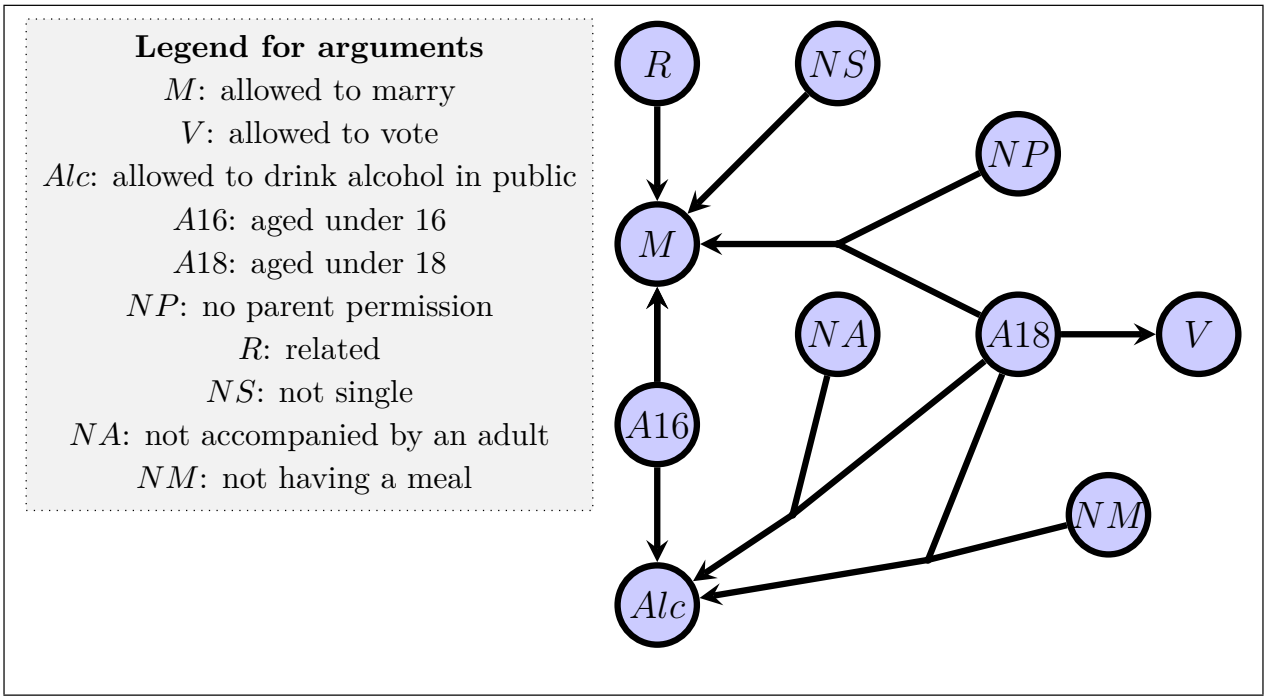


Figure 1: Example of a SETAF, encoding a part of UK legislation.

arguments such as "under 18 and not accompanied by an adult". Indeed, [60] shows that it is always possible to represent a SETAF as a standard abstract argumentation framework, albeit at the cost of a possibly substantial increase in the number of arguments. In addition to this potential cost, a cost both representational and computational, we echo the sentiment expressed in [84], that using standard frameworks rather than SETAFs in cases like that of Figure 1 tends to muddle the distinction between arguments and attacks which is the essence of the abstract argumentation approach.

3 Modelling joint attacks

In this section, we provide formal considerations associated with the use of collective attacks in argumentation frameworks. These are meant to provide the basic tools towards further formal results on the issue.

More specifically, in Section 3.1, we provide the basic formal definitions associated with SETAFs, as well as their semantics (provided both in terms of extensions, and in terms of labellings). Moreover, a series of formal results on extensions, labellings and their relations are presented, most of which are a direct adaptation of similar results from the standard AF setting.

Section 3.2 studies the relationship among AF and SETAF, and provides answers to the fundamental question of whether SETAFs constitute a more expressive tool than AFs for describing arguments and their relationships.

Section 3.3 provides various computational complexity results related to SETAFs, for different problems pertaining to different semantics. Moreover, algorithms and system implementations that address these problems are considered, including reduction-based approaches.

Further, in Section 3.4 we discuss various alternative models of abstract and structured argumentation accounting for collective attacks, i.e., attacks where a group (i.e., set) of arguments can act either as the attacker, or as the attackee.

3.1 Definitions and semantics

We start our description with the formal definition of SETAFs, including their semantics. In this subsection, we formally describe various types of semantics that have been proposed in the literature, as well as relevant results that should form the formal background and toolbox of anyone aiming to study SETAFs and their properties.

3.1.1 AFs and AF semantics: A brief reminder

An AF was defined in [40] as a pair $AF^{\mathcal{D}} = \langle Ar, att \rangle$ consisting of a (possibly infinite) set of arguments Ar and a binary attack relation att on this set. In principle, an AF is a directed graph, whose nodes correspond to arguments and whose edges correspond to attacks, which essentially represent the fact that a certain argument invalidates another. AFs are given semantics through *extensions*, which are sets of arguments (nodes) that are non-conflicting (i.e., they do not attack each other) and, as a group, "shield" themselves from attacks by other arguments (which are not in the extension). The exact formal meaning given to the term "shield" gives rise to a multitude of different semantics (complete, preferred, stable, etc.) which have been considered in the literature (e.g., see [6]).

Informally, a set of arguments $S \subseteq Ar$ is: (i) a *conflict-free* extension of $AF^{\mathcal{D}}$ iff it contains no arguments attacking each other; (ii) an *admissible* extension iff it is conflict-free and defends all its elements (i.e., for each argument $a \in Ar$ attacking an argument in S, there is an argument in S attacking a); (iii) a *complete* extension iff it is admissible and contains all the arguments it defends; (iv) a *grounded* extension iff it is minimal (w.r.t. set inclusion) among the complete extensions; (v) a *preferred* extension iff it is maximal among the complete extensions; (vi) a *stable* extension iff it is conflict-free and attacks all the arguments that it does not contain (i.e., all

arguments in $Ar \setminus S$); (*vii*) a *naive* extension iff it is maximal among the conflict-free extensions; (*viii*) a *semi-stable* extension iff its union with the set of arguments it attacks is maximal among the complete extensions; (*ix*) an *eager* extension iff it is maximal among the complete extensions that are subsets of every semi-stable extension; (*x*) an *ideal* extension iff it is maximal among the complete extensions that are subsets of every preferred extension; and (*xi*) a *stage* extension iff its union with the set of arguments it attacks is maximal among the conflict-free extensions.

3.1.2 A formalism for joint attacks (SETAFs)

To formally represent the notion of joint attacks, Dung's definition for argumentation frameworks was extended in [84] for the case where an argument can be attacked by a set of other arguments:

Definition 3.1. *A* Framework with Sets of Attacking Arguments *(*SETAF *for short) is a pair* $AF^{\mathcal{S}} = \langle Ar, \triangleright \rangle$ *such that* Ar *is a set of arguments and* $\triangleright \subseteq (2^{Ar} \setminus \{\emptyset\}) \times Ar$ *is the* attack *relation.*

It is interesting to note the asymmetry in Definition 3.1: a group of arguments can be the attacker, but not the recipient of an attack. The reason for this asymmetry is justified in [84], where it is shown that allowing a set of arguments to be jointly attacked by another does not add to the expressiveness of the proposed model. Indeed, there can be two ways in which a many-to-many attack (say $\{a_1, \ldots, a_n\} \triangleright \{b_1, \ldots, b_m\}$) can be interpreted:

1. The first, called "collective defeat" in [104], states that no b_i is accepted whenever all of $a_1, \ldots, a_n$ are accepted. This case can be easily modelled in the setting of Definition 3.1 by creating m attacks of the form $\{a_1, \ldots, a_n\} \triangleright b_i$.

2. The second, called "indeterministic defeat" in [104], states that at least one of b_i should not be accepted whenever all of $a_1, \ldots, a_n$ are accepted. This case can also be modelled in the setting of Definition 3.1, by creating m attacks of the form $\{a_1, \ldots, a_n, b_1, \ldots, b_{i-1}, b_{i+1}, \ldots, b_m\} \triangleright b_i$.

Nevertheless, for simplicity, the attack relationship $\triangleright$ of Definition 3.1 can be extended to apply among sets of arguments. Formally, we say that a set of arguments S attacks another set of arguments T (denoted by $S \blacktriangleright T$) iff there exist $U \subseteq S, a \in T$ such that $U \triangleright a$. Note that we used a different symbol for the extended relation, to avoid confusion. Importantly, $\blacktriangleright$ does not change the semantics of the attack and does not generalise it to attacks among sets of arguments; it is just a syntactic shorthand.

We will write $S \not\triangleright a$ when it is not the case that $S \triangleright a$, and $S \not\blacktriangleright T$ when it is not the case that $S \blacktriangleright T$. For singleton sets, we often write $S \blacktriangleright a$ to denote $S \blacktriangleright \{a\}$. We say that S *defends* an argument a from a set of arguments T that attacks a, iff $S \blacktriangleright T$.

An interesting note for SETAFs, is that certain attacks may be redundant. In particular, if we have that $S \triangleright a$ and $S' \triangleright a$, for $S \subseteq S'$, then the latter attack is implied by the former and is thus redundant (can be removed from the $AF^{\mathcal{S}}$ without change of semantics). This is also evident from the definition of $\blacktriangleright$, which is, in a sense, the "closure" of $\triangleright$.

3.1.3 Semantics (extensions) for SETAFs

With regards to semantics, it is easy to extend the definitions provided for the AF setting (e.g., in [40; 6]) so as to apply for the case of SETAFs (see [84; 60]). In all the following definitions, we consider a fixed SETAF $AF^{\mathcal{S}} = \langle Ar, \triangleright \rangle$ and a set of arguments $S \subseteq Ar$.

Definition 3.2. *S is said to be* conflict-free *iff it does not attack itself. Formally, S is conflict-free iff $S \not\blacktriangleright S$.*

Definition 3.3. *An argument $a \in Ar$ is said to be* acceptable *with respect to S, iff S defends a from all attacking sets of arguments in Ar. Formally, a is acceptable with respect to S iff $S \blacktriangleright T$ for all $T \subseteq Ar$ such that $T \blacktriangleright a$. S is said to be* admissible *iff it is conflict-free and each argument in S is acceptable with respect to S. Formally, S is admissible iff $S \not\blacktriangleright S$ and $S \blacktriangleright T$ for all $T \subseteq Ar$ such that $T \blacktriangleright S$.*

In [40], a characteristic function $\mathrm{F}_{AF^{\mathcal{D}}}$ was defined to return the arguments acceptable by a set of arguments in an argumentation framework $AF^{\mathcal{D}}$. This can be easily extended for SETAF (say $AF^{\mathcal{S}}$) as follows: $\mathrm{F}_{AF^{\mathcal{S}}} : 2^{Ar} \mapsto 2^{Ar}$, such that: $\mathrm{F}_{AF^{\mathcal{S}}}(S) = \{a \mid a \text{ is acceptable with respect to } S\}$.

Note that admissible extensions can (equivalently) be defined in terms of the characteristic function $\mathrm{F}_{AF^{\mathcal{S}}}$ as any conflict-free set such that $S \subseteq \mathrm{F}_{AF^{\mathcal{S}}}(S)$.

Definition 3.4. *An admissible set S is called a* complete extension *of $AF^{\mathcal{S}}$, iff all arguments that are acceptable with respect to S are in S. Formally, S is a complete extension of $AF^{\mathcal{S}}$ iff all the following conditions hold: (a) $S \not\blacktriangleright S$; (b) $S \blacktriangleright T$ for all $T \subseteq Ar$ such that $T \blacktriangleright S$; (c) If, for some $a \in Ar$, $S \blacktriangleright T$ for all $T \subseteq Ar$ such that $T \blacktriangleright a$, then $a \in S$.*

Obviously, complete extensions (of both AFs and SETAFs) can also be equivalently defined using the characteristic function: a conflict-free set S is a complete extension if and only if $\mathrm{F}_{AF^{\mathcal{S}}}(S) = S$.

Definition 3.5. *S is called a* preferred extension *of $AF^{\mathcal{S}}$, iff it is a complete extension and there is no other complete extension T such that $S \subset T$.*

In other words, a preferred extension is a maximal complete extension. In the standard AF setting, it has been shown that preferred extensions can be equivalently defined as maximal admissible extensions (see, e.g., [6]). It can be easily shown that the same holds true in the SETAF setting [55].

Definition 3.6. *S is called a* grounded extension *of $AF^{\mathcal{S}}$, iff it is a complete extension and there is no other complete extension T such that $T \subset S$.*

Essentially, grounded extensions are minimal complete extensions. Following similar results in the AF setting ([6]) we can easily show that the following are equivalent also in the SETAF setting:

- S is a grounded extension
- S is the complete extension such that $\{a \in Ar \mid S \triangleright a\}$ is minimal
- S is the complete extension such that $Ar \setminus (S \cup \{a \in Ar \mid S \triangleright a\})$ is maximal

Using the characteristic function, another equivalent characterisation can be formulated, namely that S is a grounded extension if and only if it is the least fixed point of $\mathrm{F}_{AF^{\mathcal{S}}}$ (see also [40]).

Definition 3.7. *S is called a* stable extension *of $AF^{\mathcal{S}}$, iff it is conflict-free and attacks all arguments in $Ar \setminus S$.*

Equivalently, S is stable if and only if $S = \{a \mid S \not\triangleright a\}$. Also, for a stable extension S it holds that $S \cup \{a \mid S \triangleright a\} = Ar$.

Moreover, we can easily show that stable extensions are also preferred, complete and admissible (see also [60] and Figure 4), thus S is a stable extension if and only if S is a preferred, complete or admissible extension that attacks all arguments in $Ar \setminus S$.

Example 3.8. *Consider the SETAF shown in Figure 2, whose extensions are shown in Table 1. Let us consider in more detail the complete extensions, which are: $\emptyset$, $\{a_1\}$, $\{a_2, a_3, a_5\}$. Note that, for example, $\{a_2, a_3\}$ is admissible and conflict-free but not complete, because it leaves out a_5, which is acceptable with respect to $\{a_2, a_3\}$. Similarly, $\{a_1, a_2\}$ is not a complete extension because it is not conflict-free, whereas $\{a_5\}$ and $\{a_1, a_5\}$ are not complete extensions because they are not admissible (a_5 is not acceptable with respect to the corresponding set in either case).*
Further, the minimal of the complete extensions (namely $\emptyset$) is also grounded, whereas

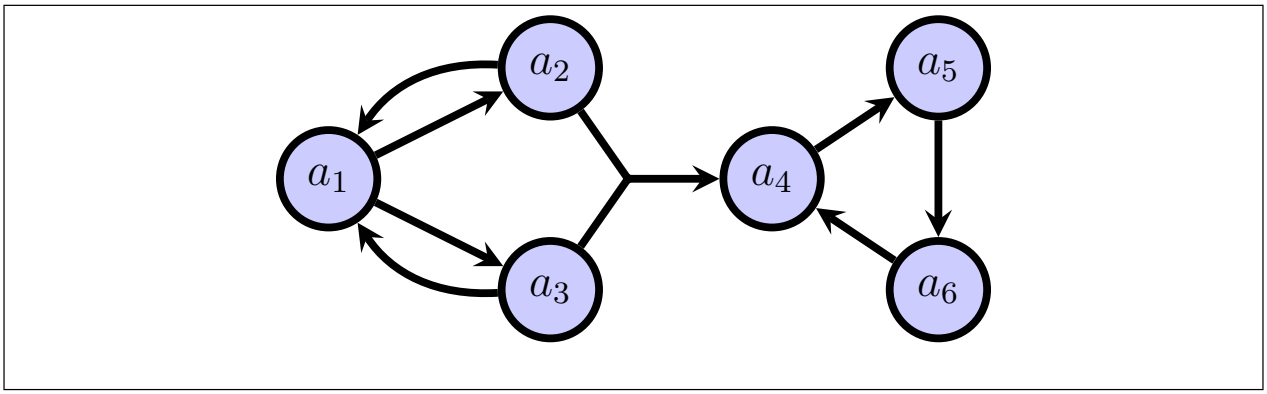


Figure 2: An example SETAF; set attacks are represented as arrows with multiple sources (e.g., $\{a_2, a_3\} \blacktriangleright a_4$); its extensions are shown in Table 1

Extension type	**Extensions**
Conflict-free	$\emptyset$, $\{a_1\}$, $\{a_2\}$, $\{a_3\}$, $\{a_4\}$, $\{a_5\}$, $\{a_6\}$, $\{a_1, a_4\}$, $\{a_1, a_5\}$, $\{a_1, a_6\}$, $\{a_2, a_3\}$, $\{a_2, a_3, a_5\}$, $\{a_2, a_3, a_6\}$, $\{a_2, a_4\}$, $\{a_2, a_5\}$, $\{a_2, a_6\}$, $\{a_3, a_4\}$, $\{a_3, a_5\}$, $\{a_3, a_6\}$
Admissible	$\emptyset$, $\{a_1\}$, $\{a_2\}$, $\{a_3\}$, $\{a_2, a_3\}$, $\{a_2, a_3, a_5\}$
Complete	$\emptyset$, $\{a_1\}$, $\{a_2, a_3, a_5\}$
Preferred	$\{a_1\}$, $\{a_2, a_3, a_5\}$
Grounded	$\emptyset$
Stable	$\{a_2, a_3, a_5\}$
Naive	$\{a_1, a_4\}$, $\{a_1, a_5\}$, $\{a_1, a_6\}$, $\{a_2, a_4\}$, $\{a_2, a_6\}$, $\{a_3, a_4\}$, $\{a_3, a_6\}$, $\{a_2, a_3, a_5\}$
Semi-stable	$\{a_2, a_3, a_5\}$
Eager	$\{a_2, a_3, a_5\}$
Ideal	$\emptyset$
Stage	$\{a_2, a_3, a_5\}$

Table 1: Extensions for the SETAF of Figure 2

the maximal ones ($\{a_1\}, \{a_2, a_3, a_5\}$) are also preferred. The latter ($\{a_2, a_3, a_5\}$) is also stable, because it attacks all other arguments.
Looking at the SETAF illustrated in Figure 3, we note that it also has three complete extensions (namely, $\{a_1\}$, $\{a_1, a_2, a_5\}$, $\{a_1, a_3, a_4\}$), the first of which is also the grounded one ($\{a_1\}$), whereas the other two are the preferred ones $\{a_1, a_2, a_5\}$, $\{a_1, a_3, a_4\}$). However, there is no stable extension, because none of the complete extensions attacks all other arguments in the SETAF.

Definition 3.9. *S is called a* naive extension *of $AF^{\mathcal{S}}$, iff it is conflict-free and is maximal w.r.t. set inclusion among the conflict-free subsets of Ar.*

Example 3.10. *Returning to the SETAF shown in Figure 2, we note that it has several naive extensions (see Table 1), which are essentially all the maximal subsets of Ar that do not attack themselves. On the other hand, the SETAF of Figure 3 has three naive extensions, namely $\{a_2, a_4\}$, $\{a_1, a_2, a_5\}$, $\{a_1, a_3, a_4\}$.*

Definition 3.11. *S is called a* semi-stable extension *of $AF^{\mathcal{S}}$, iff it is a complete extension and the set $S \cup \{a \in Ar \mid S \blacktriangleright a\}$ is maximal w.r.t. set inclusion among all complete extensions of $AF^{\mathcal{S}}$.*

Essentially, semi-stable semantics give up the strict requirement of stable semantics that $S \cup \{a \in Ar \mid S \blacktriangleright a\} = Ar$, and require just that $S \cup \{a \in Ar \mid S \blacktriangleright a\}$ is maximal.

Just like in stable extensions, semi-stable extensions are also preferred, complete and admissible (see also [60] and Figure 4), so the following are equivalent [55]:

- S is a semi-stable extension
- S is an admissible extension and the set $S \cup \{a \in Ar \mid S \blacktriangleright a\}$ is maximal w.r.t. set inclusion among all admissible extensions of $AF^{\mathcal{S}}$.
- S is a preferred extension and the set $S \cup \{a \in Ar \mid S \blacktriangleright a\}$ is maximal w.r.t. set inclusion among all preferred extensions of $AF^{\mathcal{S}}$.

Example 3.12. *For the SETAF illustrated in Figure 2, where a stable extension exists, this is also the (only) semi-stable extension of the SETAF (see Table 1). However, in the SETAF of Figure 3, where no stable extension exists, one can find two semi-stable extensions, namely: $\{a_1, a_2, a_5\}$, $\{a_1, a_3, a_4\}$. Each of these semi-stable extensions attack (or contain) all arguments except one (a_6 and a_7 respectively).*

Definition 3.13. *S is called an* eager extension *of $AF^{\mathcal{S}}$, iff it is a maximal (with respect to set inclusion) complete extension that is a subset of each semi-stable extension of $AF^{\mathcal{S}}$.*

The maximality requirement implies that we can replace the completeness requirement regarding S with admissibility, i.e., S is an eager extension of $AF^{\mathcal{S}}$, iff it is a maximal (with respect to set inclusion) admissible extension that is a subset of each semi-stable extension of $AF^{\mathcal{S}}$ (see [55]).

Example 3.14. *For the SETAF shown in Figure 2, there is only one semi-stable extension, so this is also the eager extension. In the SETAF of Figure 3, where there are two semi-stable extensions, the only eager extension is their intersection, i.e., $\{a_1\}$.*

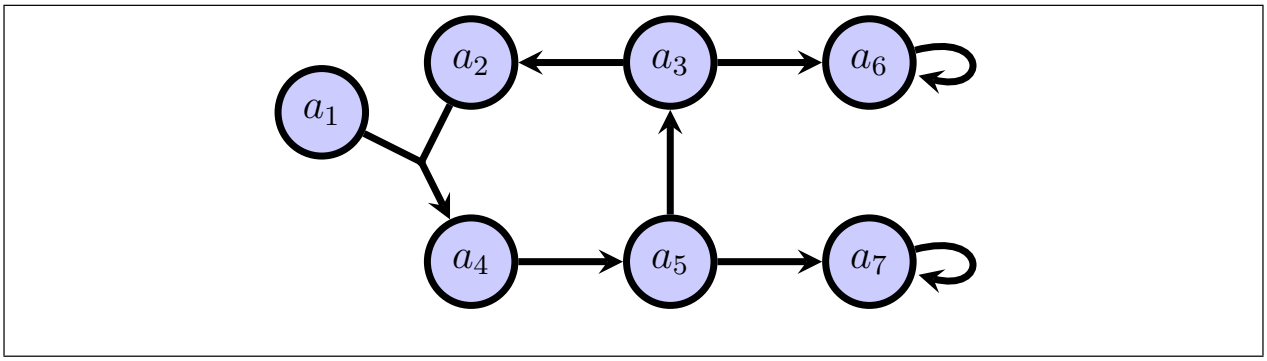


Figure 3: An example SETAF; set attacks are represented as arrows with multiple sources (e.g., $\{a_1, a_2\} \blacktriangleright a_4$); its extensions are shown in Table 2

Definition 3.15. *S is called an* ideal extension *of $AF^{\mathcal{S}}$, iff it is a maximal (with respect to set inclusion) complete extension that is a subset of each preferred extension of $AF^{\mathcal{S}}$.*

Extension type	**Extensions**
Complete	$\{a_1\}$, $\{a_1, a_2, a_5\}$, $\{a_1, a_3, a_4\}$
Preferred	$\{a_1, a_2, a_5\}$, $\{a_1, a_3, a_4\}$
Grounded	$\{a_1\}$
Stable	(none exists)
Naive	$\{a_2, a_4\}$, $\{a_1, a_2, a_5\}$, $\{a_1, a_3, a_4\}$
Semi-stable	$\{a_1, a_2, a_5\}$, $\{a_1, a_3, a_4\}$
Eager	$\{a_1\}$
Ideal	$\{a_1\}$
Stage	$\{a_1, a_2, a_5\}$, $\{a_1, a_3, a_4\}$

Table 2: Extensions for the SETAF of Figure 3

Again, we can replace the requirement of S being complete, with S being preferred, or admissible (see [55]). Moreover, since an ideal extension is a subset of all preferred ones, it is not attacked by any preferred extension, and is in fact the largest complete extension (and admissible set) with this property. Using similar arguments, we can show that an ideal extension is the largest admissible set not attacked by any admissible set, and the largest admissible set not attacked by any complete extension [55].

Example 3.16. *For the SETAF shown in Figure 2, the two preferred extensions have an empty intersection, and $\emptyset$ happens to be a complete extension, so the only*

ideal extension is $\emptyset$. Similarly, for the SETAF of Figure 3, there are two preferred extensions, whose intersection is equal to $\{a_1\}$, and this happens to be a complete extension, so it is also ideal.

Definition 3.17. *S is called a* stage extension *of $AF^{\mathcal{S}}$, iff it is conflict-free and $S \cup \{a \in Ar \mid S \blacktriangleright a\}$ is maximal among all conflict-free subsets of Ar.*

Apparently, a stage extension is also naive (see also Figure 4), and, in fact, a stage extension can be equivalently defined as a naive extension such that $S \cup \{a \in Ar \mid S \blacktriangleright a\}$ is maximal among all naive extensions of Ar.

Example 3.18. *For the SETAF shown in Figure 2, which has a stable extension, the (only) stage extension is the stable one, i.e., $\{a_2, a_3, a_5\}$. For the SETAF illustrated in Figure 3, which has no stable extension, there are two stage extensions, which happen to be the same as the semi-stable ones, namely $\{a_1, a_2, a_5\}$, $\{a_1, a_3, a_4\}$. As explained in Example 3.12, each of these semi-stable extensions attack (or contain) all arguments except one (a_6 and a_7 respectively).*

3.1.4 Relationships among extensions

The various extensions are related, in the sense that certain types of extensions are stronger than others (e.g., a preferred extension is also complete, but not vice-versa). Moreover, some types of extensions are guaranteed to exist, others are not, and some extensions are unique. These results have been shown in various works for standard AFs, but [60] recast them for the SETAF case.

Figure 4 summarises these results. Each arrow in the graph pointing from semantics σ to σ' indicates that every σ-extension of a SETAF is also a σ'-extension of the same SETAF (e.g., every stable extension is also a stage extension). The number (possibly followed by +) that appears next to each semantics indicates the multiplicity of extensions for the specific semantics (e.g., every SETAF has at least one preferred extension). Similarly to Dung-style AFs, for certain semantics, the multiplicity of extensions is different for finite and infinite SETAFs, i.e., SETAFs with finite (respectively infinite) number of arguments. All such arrows are strict, i.e., no semantics is equivalent to another. Note also that in [60] the relationship among stage and naive semantics is missing.

3.1.5 Labellings

The semantics of AFs can be alternatively defined through labellings, as proposed in [27]. A labelling is formally defined as a function from arguments to the set $\{\texttt{in}, \texttt{out}, \texttt{undec}\}$. Intuitively, an argument belongs to the extension iff it is labelled

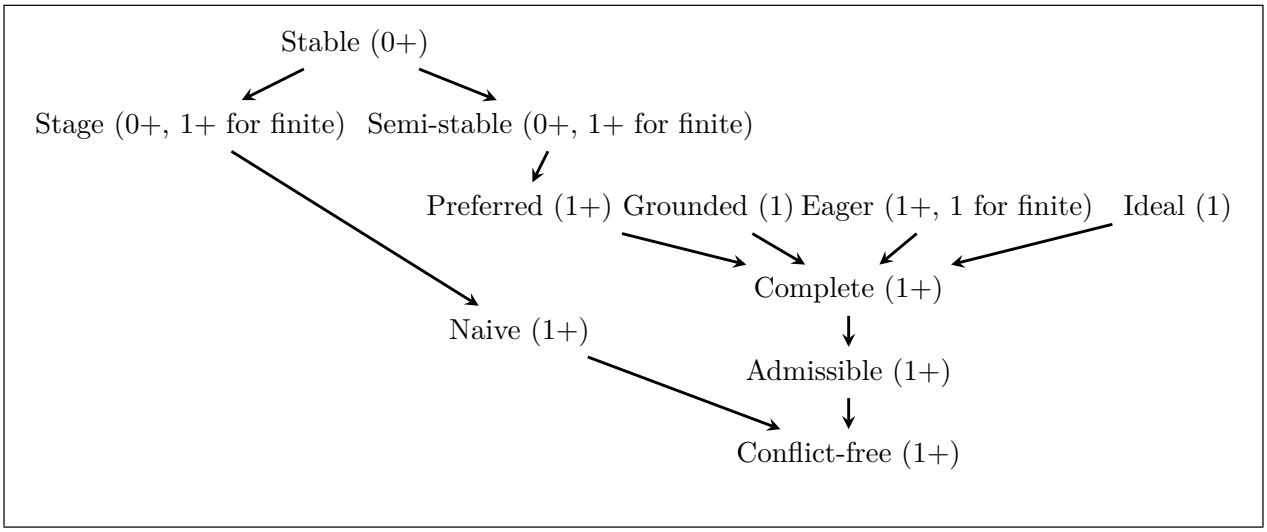


Figure 4: Inclusion relations and multiplicity of extensions for SETAF acceptability semantics

as `in`, whereas arguments labelled `out` are those attacked by the ones labelled `in`. Finally, the `undec` labelling is reserved for arguments that are not accepted, but are not attacked by an accepted argument either. Although labellings have been originally defined for AFs only [27], an adaptation for the SETAF case appears in [60]. Formally, a labelling is a function as follows:

Definition 3.19. *Consider a SETAF* $AF^{\mathcal{S}} = \langle Ar, \triangleright \rangle$. *A* labelling *for* $AF^{\mathcal{S}}$ *is a total function* $\mathcal{L}ab : Ar \mapsto \{\texttt{in}, \texttt{out}, \texttt{undec}\}$.

Note that the labellings of a SETAF are defined over arguments (just like in AFs [27]), not sets of arguments.

Special classes of labellings can be defined (e.g., conflict-free labellings, admissible labellings, complete labellings, etc) and formally shown to correspond to the respective extensions (conflict-free, admissible, complete, etc). The correspondence is realised through two functions ($Ext2Lab$, $Lab2Ext$), which determine how to generate an extension given a labelling, or vice-versa. It can be shown that, if $\mathcal{L}ab$ is a labelling of a certain type (e.g., complete), then $Lab2Ext(\mathcal{L}ab)$ is an extension of the same type, and, vice-versa, if S is an extension of a certain type (e.g., complete), then $Ext2Lab(S)$ is a labelling of the same type.

In this section, we illustrate these ideas, dealing with complete labellings only, and refer to [27] and [60] for further details. We start with the definition of the functions $Lab2Ext$, $Ext2Lab$:

Definition 3.20. *Consider a SETAF $AF^{\mathcal{S}} = \langle Ar, \triangleright \rangle$, and let $\mathcal{E}$ be the set of all possible extensions that can be created over $AF^{\mathcal{S}}$, $\mathcal{L}$ be the set of all possible labellings that can be created over $AF^{\mathcal{S}}$. Then:*

Ext2Lab: *We define the function $Ext2Lab : \mathcal{E} \mapsto \mathcal{L}$ such that, for $S \in \mathcal{E}$, $\mathcal{L}ab = Ext2Lab(S)$:*

- $\mathcal{L}ab(a) = \texttt{in}$ *for all* $a \in S$
- $\mathcal{L}ab(a) = \texttt{out}$ *for all* $a \notin S$, $S \blacktriangleright a$
- $\mathcal{L}ab(a) = \texttt{undec}$ *for all* $a \notin S$, $S \not\blacktriangleright a$

Lab2Ext: *We define the function $Lab2Ext : \mathcal{L} \mapsto \mathcal{E}$ such that, for $\mathcal{L}ab \in \mathcal{L}$, $Lab2Ext(\mathcal{L}ab) = \{a \in Ar \mid \mathcal{L}ab(a) = \texttt{in}\}$.*

Clearly, both *Ext2Lab* and *Lab2Ext* are well-defined. Moreover, note that $Ext2Lab(S)$ essentially labels `in` those arguments that are in S, `out` those arguments attacked by S, and `undec` the rest. On the other hand, $Lab2Ext(\mathcal{L}ab)$ contains only the arguments that are labelled `in` by $\mathcal{L}ab$.

Now, we can define complete labellings as follows:

Definition 3.21. *Let $AF^{\mathcal{S}} = \langle Ar, \triangleright \rangle$ be a SETAF. A labelling $\mathcal{L}ab : Ar \mapsto \{\texttt{in}, \texttt{out}, \texttt{undec}\}$ of $AF^{\mathcal{S}}$ is called* complete *iff for all $a \in Ar$:*

1. $\mathcal{L}ab(a) = \texttt{in}$ *if and only if* $\forall S \blacktriangleright a, \exists b \in S : \mathcal{L}ab(b) = \texttt{out}$
2. $\mathcal{L}ab(a) = \texttt{out}$ *if and only if* $\exists S \subseteq Ar$ *such that* $S \blacktriangleright a$ *and* $\mathcal{L}ab(b) = \texttt{in}$ *for all* $b \in S$

The next step is to prove that complete labellings correspond to complete extensions and vice-versa. The following two theorems prove these points:

Theorem 3.22. *Let $AF^{\mathcal{S}} = \langle Ar, \triangleright \rangle$ be a SETAF and $S \subseteq Ar$ a complete extension of $AF^{\mathcal{S}}$. Then, $Ext2Lab(S)$ is a complete labelling of $AF^{\mathcal{S}}$.*

Theorem 3.23. *Let $AF^{\mathcal{S}} = \langle Ar, \triangleright \rangle$ be a SETAF and $\mathcal{L}ab : Ar \mapsto \{\texttt{in}, \texttt{out}, \texttt{undec}\}$ a complete labelling of $AF^{\mathcal{S}}$. Then, $Lab2Ext(\mathcal{L}ab)$ is a complete extension of $AF^{\mathcal{S}}$.*

The above theorems show that complete labellings and complete extensions are essentially analogous ways to define the semantics of a SETAF. Similar theorems hold for the other types of extensions/labellings (see [60], Theorems 5.10, 5.11).

Complete extensions	Complete labellings
$S_1 = \emptyset$	$\mathcal{L}ab_1(a_1) = \texttt{undec}$, $\mathcal{L}ab_1(a_2) = \texttt{undec}$, $\mathcal{L}ab_1(a_3) = \texttt{undec}$, $\mathcal{L}ab_1(a_4) = \texttt{undec}$, $\mathcal{L}ab_1(a_5) = \texttt{undec}$, $\mathcal{L}ab_1(a_6) = \texttt{undec}$
$S_2 = \{a_1\}$	$\mathcal{L}ab_2(a_1) = \texttt{in}$, $\mathcal{L}ab_2(a_2) = \texttt{out}$, $\mathcal{L}ab_2(a_3) = \texttt{out}$, $\mathcal{L}ab_2(a_4) = \texttt{undec}$, $\mathcal{L}ab_2(a_5) = \texttt{undec}$, $\mathcal{L}ab_2(a_6) = \texttt{undec}$
$S_3 = \{a_2, a_3, a_5\}$	$\mathcal{L}ab_3(a_1) = \texttt{out}$, $\mathcal{L}ab_3(a_2) = \texttt{in}$, $\mathcal{L}ab_3(a_3) = \texttt{in}$, $\mathcal{L}ab_3(a_4) = \texttt{out}$, $\mathcal{L}ab_3(a_5) = \texttt{in}$, $\mathcal{L}ab_3(a_6) = \texttt{out}$

Table 3: Complete extensions and complete labellings for the SETAF of Figure 2.

Example 3.24. *Table 3 shows the complete labellings that correspond to the SETAF of Figure 2. Comparing complete extensions with complete labellings, we see that, e.g., the third labelling explicitly rejects a_6 (because it is attacked by a_5, which is accepted), but the second one makes no explicit decision on a_6, as the agent cannot make up its mind on how to resolve the cyclic attack among a_4, a_5, a_6. This distinction cannot be made with the corresponding complete extensions (first column of Table 3).*
Moreover, we can easily verify that:

- *the labellings can be generated through the corresponding extensions, using Definition 3.20;*
- *the labellings are all complete labellings (under Definition 3.21);*
- *the extensions could be generated from the labellings, using Definition 3.20.*

Another interesting point to note is that, for complete extensions and labellings, the relationship established by $Ext2Lab$, $Lab2Ext$, is bijective. In other words, for every labelling $\mathcal{L}ab$ and extension S of a SETAF, it holds $Ext2Lab(Lab2Ext(\mathcal{L}ab)) = \mathcal{L}ab$ and $Lab2Ext(Ext2Lab(S)) = S$. This is true for most, but not all, types of labellings; e.g., for admissible labellings, several different labellings may correspond to the same extension through $Lab2Ext$. A complete analysis of this phenomenon can be found in [60], where the concept of *proper labellings* is introduced to settle this question. Moreover, a rich set of results showing various properties of labellings can be found in [5; 6]. Although these results have been shown for AFs, recasting them for SETAFs is in most cases easy. Further details on the above are omitted, and the reader is referred to [5; 6; 60; 27] for more information.

3.2 Relating models for joint attacks with classical AFs

One of the obvious questions regarding SETAFs is whether they constitute a genuine extension of standard AFs (with more expressive power), or whether they are just a shorthand, i.e., syntactic sugar for knowledge that can be anyway represented in the standard Dung setting.

This is a very important question, because, if it turns out that AFs can be used to represent SETAFs, then we would be able to use the more intuitive SETAF formalism for modelling the attacks among arguments, while at the same time exploiting implementations and tools (and complexity results) developed for simple AFs to perform reasoning over the SETAF, by exploiting these translations. In the opposite case, SETAFs should be viewed as a separate, and more expressive branch of computational argumentation, and would require a different set of tools to support reasoning over them.

Interestingly, different works have addressed this problem, and answers have been given from different perspectives. In the rest of this section, we analyse four such works, namely:

- [48], who characterise the expressive power of AFs and SETAFs based on the notion of signatures [44], showing that SETAFs are strictly more expressive than AFs for the most popular semantics.
- [60], who circumvent the negative result of [48] by considering an exponential-sized translation of SETAFs to AFs and appropriate mappings among their semantics, for various semantics.
- [94], who applies an approach similar to [60], considering various alternative (and more condensed) translations with similar results (for the most popular semantics).
- [20], who consider the problem of translating Abstract Dialectical Frameworks (ADFs) [22] to AFs; given that SETAFs are a special case of ADFs, this result can be applied for the purposes of this chapter as well, albeit for a limited set of semantics.

3.2.1 Characterising the expressive power using signatures

The approach of [48] is based on *signatures* of different semantics (namely complete, grounded, preferred, stable, semi-stable, stage and naive [40; 24; 105; 19]) for AFs and SETAFs. Signatures have been originally defined in [44] as a way to characterise the expressive power of an AF, by way of conditions under which a candidate set of

subsets of arguments are "realistic", i.e., they correspond to the extensions of some argumentation framework AF for a semantics of interest.

The idea has been extended to other types of argumentation frameworks (e.g., in [77; 98; 99; 100] for the ADF case [22]), and employed heavily as a means to compare the expressiveness of different argumentation frameworks with, e.g., normal logic programs and propositional logic [99; 100].

Formally, given a set of extensions (i.e., a set of sets of arguments) $\mathcal{E}$, $\mathcal{E}$ belongs to the signature Σ_σ^{AF} iff there is an AF framework whose set of extensions, under σ-semantics, is $\mathcal{E}$. Similarly, one can define Σ_σ^k, where k corresponds to a SETAF that admits only attacks where the attacking set has arity at most k (note that Σ_σ^1 coincides with Σ_σ^{AF} and Σ_σ^∞ coincides with the generic SETAF framework Σ_σ^{SETAF}). By definition, the notion of a signature expresses exactly the sets of extensions that can be constructed given a certain framework type, and for a certain semantics.

The focus of [48] is to compute the signatures Σ_σ^k for the considered semantics and for different k. As an example, they define the notion of an incomparable set of sets, where a set of sets $\mathcal{E}$ is incomparable iff all elements of $\mathcal{E}$ are pairwise incomparable, i.e., for $T, U \in \mathcal{E}$, $T \subseteq U$ implies $T = U$. Then, they prove that the set comprising all stable extensions of a SETAF is incomparable, i.e., $\Sigma_{\mathcal{ST}}^\infty = \{\mathcal{E} \mid \mathcal{E}$ is incomparable$\}$.

Signatures are a powerful tool for determining expressive power. Larger signatures imply that the corresponding framework type is more flexible (and thus more expressive). In particular, if $\mathcal{E} \notin \Sigma_\sigma^1$, then this means that one cannot construct an AF whose σ-extensions are exactly the ones in $\mathcal{E}$. Thus, by comparing Σ_σ^k for various $k \in \{1, 2, ..., \infty\}$, we can determine the relative expressive power of the different framework types.

Using this reasoning, the main conclusion of the paper is that, for all the considered semantics, and for all $k > 0$, SETAFs that allow for collective attacks of $k + 1$ arguments are more expressive than SETAFs that only allow for collective attacks of at most k arguments, because $\Sigma_\sigma^k \subset \Sigma_\sigma^{k+1}$. As a corollary, SETAFs are strictly more expressive than AFs, even if restricted to attacks of at most 2 arguments.

It is important however to interpret the above results under the correct lens. In particular, the results of [48] tell us that certain sets of extensions that can be constructed using SETAFs, cannot be *directly* constructed through AFs. More specifically, for a given $\mathcal{E} \in \Sigma_\sigma^{SETAF} \setminus \Sigma_\sigma^{AF}$, we know that one can create a SETAF whose set of σ-extensions is exactly $\mathcal{E}$; moreover, there is no AF whose set of σ-extensions is exactly $\mathcal{E}$.

However, if we don't insist on the *direct* construction, one may be able to succeed in constructing $\mathcal{E}$ through some AF, but in another, indirect way. In particular, one could define an appropriate mapping (algorithm) among sets of extensions (say f),

and then construct an AF $AF^{\mathcal{D}}$ whose set of extensions is, say, $\mathcal{E}'$, where $f(\mathcal{E}') = \mathcal{E}$. For generality, one should also define a generic way to construct $AF^{\mathcal{D}}$ from the original SETAF $AF^{\mathcal{S}}$, via some other mapping (algorithm), say g. By the results of [48], this transformation cannot be a simple rearrangement of the attacks among the existing arguments of the SETAF, but should necessarily involve new, artificial arguments that would somehow encode the "collectivity of attacks".

3.2.2 An exponential translation to encode collectivity of attacks

This approach of "expanding" the SETAF with new arguments in order to get rid of collective attacks (and thus result in an AF) is followed in [60]. In that paper, a rather straightforward translation is followed, where, for any given SETAF $AF^{\mathcal{S}} = \langle Ar, \triangleright \rangle$, one constructs a so-called *generated AF* $AF^{\mathcal{D}} = \langle Ar', att \rangle$, whose "arguments" are all the non-empty sets of arguments of the original SETAF (i.e., $Ar' = 2^{Ar} \setminus \{\emptyset\}$). The corresponding attack relation att follows in the obvious manner from $\triangleright$. In the above terminology, this is the mapping g.

Then, the authors go on to identify the relationship among the σ-extensions of the $AF^{\mathcal{S}}$ and its corresponding generated $AF^{\mathcal{D}}$, as well as how one can identify the σ-extensions of $AF^{\mathcal{S}}$ through the σ-extensions of $AF^{\mathcal{D}}$, and vice versa (i.e., the mapping f and its inverse).

Various different semantics are considered, including the ones originally defined in [40] (conflict-free, admissible, complete, grounded, preferred, stable), but also naive [19], semi-stable [24], eager [25], ideal [41] and stage [105].

The conclusion of the above analysis is that many of the semantics (namely, complete, preferred, grounded, stable and ideal) admit a very simple one-to-one correspondence among the semantics of the SETAF and the generated AF. In particular, a set of arguments $S \subseteq Ar$ of the SETAF ($AF^{\mathcal{S}}$) is a σ-extension if and only if the set $2^S \setminus \{\emptyset\}$ is a σ-extension of the generated AF (recall that an argument in $AF^{\mathcal{D}}$ is a set of arguments from $AF^{\mathcal{S}}$).

For conflict-free and admissible extensions, the situation is similar, except that there are some additional σ-extensions of $AF^{\mathcal{D}}$ which do not follow this exact pattern. This has effects on the correspondence among naive extensions as well (recall that a naive extension is a maximal conflict-free set). Further, more convoluted correspondences exist for semi-stable, stage, and eager semantics, where the characterisations are complicated by the requirement of maximality (see [60] for details).

Complexity of characterisations put aside, the work of [60] shows that one can model a SETAF as an AF in a way that "preserves" the semantics, in the sense that one can determine the σ-extensions of the SETAF by just looking at the AF (and vice-versa). Alas, the proposed transformation for achieving this effect, results to

an AF with an exponentially larger number of arguments compared to the SETAF. Note that if we count the size of a SETAF in terms of the number of arguments plus the number of attacks, then we may not get an exponential increase (if a sufficiently large number of attacks exist), although the exponential increase is still true in the worst-case scenario.

3.2.3 Considering more compact translations

A similar, but less extreme "expansion" scheme is followed in [94], where the problem of translating SETAFs to AFs is considered, among other things. The considered semantics are the standard Dung semantics, i.e., conflict-free, admissible, complete, preferred, grounded and stable [40].

To perform the translation, two translation schemes (and variations thereof) are considered: one is inspired by the so-called coalition approach and the other by the so-called defender approach. Both have a polynomial size compared to the SETAF (assuming that the size of the SETAF is considered to be equal to the number of attacks plus the number of arguments).

The coalition approach is similar to the one proposed in [60], where an argument in the AF is a set of arguments from the SETAF. However, in [94] a "condensed" version of the translation is considered, where not all subsets of Ar are included in the generated AF, but only those that are actually the initiators of an attack. Different ways to translate the attack relation are then considered, with different results with respect to the correspondence among the semantics of the SETAF and the corresponding AF.

The second translation scheme is inspired by [80], and uses arguments in the translated AF that represent "statements" regarding an argument in the SETAF (*e.g.* whether it is accepted, justified, rejected etc). More precisely, for every argument a in the SETAF, two arguments are included in the AF: the argument itself (a), as well as a' which stands for "a is rejected". Moreover, every attack in the SETAF is represented as an argument in the AF (these are called auxiliary arguments).

Then, appropriate attacks are introduced in the new framework. Namely, each argument a attacks its corresponding a', and a' attacks the auxiliary arguments representing an attack involving a as an attacker. The auxiliary arguments representing attacks, attack the corresponding recipient of the attack. In this way, a defends the auxiliary arguments it is involved in, so if a is not accepted, the attack itself (i.e., the auxiliary argument representing it) will not be accepted, and thus the recipient of the attack will be unaffected by the attack. Using this trick, the semantics of the SETAF can be appropriately captured by the AF.

For both translations, the correspondences provided among the σ-extensions of

the SETAF and its generated AF are generally elegant, and quite similar to the correspondences of [60] (note however that the more complex cases of semi-stable, stage and eager semantics are not considered by [94]).

Despite that, a strong statement is made in [94] that no full exact SETAF-AF translation can be created. This statement is based on the idea of signatures, and follows similar lines of reasoning as in [48]. Therefore, it should be interpreted in the sense of a direct translation, as explained also in our analysis of the results of [48].

3.2.4 An indirect translation path, through ADFs

Another interesting translation results as a corollary of the work in [20]. In that paper, the authors do not study SETAFs, but ADFs [22]. An ADF is similar to an AF, except that the acceptance of an argument is determined by an acceptance condition (expressed as a propositional formula) over the acceptance of all its attackers. Thus, for example, one could say that an argument is accepted iff no more than two of its attackers are accepted, or that an argument is accepted iff all of its attackers are accepted.

Note that the expressive power of acceptance conditions allows ADFs to model various different types of relations among arguments, including attack, support, joint attacks or supports, as well as hybrid cases. In particular, it is easy to see that AFs and SETAFs are special cases of ADFs [93; 77].

Three different types of semantics have been defined for ADFs in [22], namely models, well-founded models and stable models. In the special case where an ADF is used to describe an AF (or a SETAF), models of the ADF correspond to the stable extensions of the AF (or SETAF) and well-founded models of the ADF correspond to the grounded extensions of the AF (or SETAF). Moreover, for this special case, stable models of the ADF and models of the ADF coincide (see [50], Proposition 1), so stable models of the ADF also correspond to stable extensions of the AF (or SETAF). It should be noted here that stable models have been retrospectively redefined in [23], but this redefinition does not break the above correspondences (see Theorem 4 in [23]).

In [20], the authors show that, given an ADF, one can generate an AF such that the stable extensions of the AF correspond (in a formal manner made clear in the paper) to the models of the ADF. A similar correspondence is also shown among the grounded extensions of an AF and the well-founded models of the ADF, as well as among the stable extensions of the AF and the stable models of the ADF. Although SETAFs were not in the scope of the work of [20], the fact that SETAFs are a special case of ADFs, allows us to apply their results to the case considered in this chapter. Moreover, [20] show that the proposed translations are polynomial

in size, and can also be computed in polynomial time, where the size of the original ADF (corresponding to a SETAF) is computed as the number of arguments plus the size of the acceptance conditions of the arguments.

3.3 Computational considerations

In this section we give an overview on complexity results of SETAFs and discuss implementation approaches for evaluating SETAFs. As discussed in [47] understanding the inherent complexity of the reasoning tasks is crucial towards efficient implementations of argumentation systems. In particular, problems on different levels of complexity have different limits concerning scalability and require different techniques to be implemented in a scalable manner. We first introduce the computational tasks we are interested in, then discuss their complexity, and finally discuss algorithms and reduction-based approaches for these tasks.

3.3.1 Computational Problems

The standard problems studied in computational (abstract) argumentation are the tasks of computing extensions of a given semantics and computing the credulous or skeptical consequences under a given semantics [36; 47; 35]. These tasks are investigated in the literature on algorithms, systems, and complexity of abstract argumentation, and are the basis for the different tracks of the International Competition on Computational Models of Argumentation (ICCMA)[6] [101; 64]. In the following we provide formal definitions of these computational problems in the context of SETAFs. To this end we will use $\sigma(AF^{\mathcal{S}})$ to denote the σ-extensions of a SETAF $AF^{\mathcal{S}}$. We start with the function problems of computing one or all of the extensions of a SETAF w.r.t. a semantics σ:

- *Some Extension* SE_σ: Given SETAF $AF^{\mathcal{S}}$, compute an extension $E \in \sigma(AF^{\mathcal{S}})$.
- *Enumerate Extensions* EE_σ: Given SETAF $AF^{\mathcal{S}}$, compute the extension-set $\sigma(AF^{\mathcal{S}})$.

Beside these function problems we consider decision problems whose output is either yes or no. These problems are of particular interest as they are well-suited for being analysed with the techniques of complexity theory. To this end we consider the skeptical acceptance of an argument, i.e., an argument is skeptically accepted if it is contained in each extension, and credulous acceptance of an argument, i.e., an argument is credulously accepted if it is contained in some extension (for a given semantics σ):

[6] `http://argumentationcompetition.org/`

- *Credulous Acceptance* $Cred_\sigma$: Given SETAF $AF^{\mathcal{S}} = \langle Ar, \triangleright \rangle$ and an argument $a \in Ar$, is a contained in some $E \in \sigma(AF^{\mathcal{S}})$?
- *Skeptical Acceptance* $Skept_\sigma$: Given SETAF $AF^{\mathcal{S}} = \langle Ar, \triangleright \rangle$ and an argument $a \in Ar$, is a contained in each $E \in \sigma(AF^{\mathcal{S}})$?

Moreover, we consider the frequently-studied problems of verifying a given extension, deciding whether a SETAF has at least one extension, and deciding whether a SETAF has a non-empty extension. These problems are of some interest on their own but are in particular relevant as frequent sub-tasks of reasoning procedures. We next provide the formal definitions of these problems:

- *Verification of an extension* Ver_σ: Given SETAF $AF^{\mathcal{S}} = \langle Ar, \triangleright \rangle$ and a set of arguments $S \subseteq Ar$, is $S \in \sigma(AF^{\mathcal{S}})$?
- *Existence of an extension* $Exists_\sigma$: Given SETAF $AF^{\mathcal{S}} = \langle Ar, \triangleright \rangle$, is $\sigma(AF^{\mathcal{S}}) \neq \emptyset$?
- *Existence of a non-empty extension* $Exists_\sigma^{\neg\emptyset}$: Given SETAF $AF^{\mathcal{S}} = \langle Ar, \triangleright \rangle$, does there exist a set $E \neq \emptyset$ such that $E \in \sigma(AF^{\mathcal{S}})$?

3.3.2 Complexity results for SETAFs

We next discuss the computational complexity of the decision problems introduced in the previous section. The rationale behind the focus on decision problems is that tools of complexity theory are better suited for decision problems than for function problems and that, when chosen carefully, the complexity of the decision problems is also a good indicator for the complexity of the corresponding function problem. In computational argumentation the credulous and skeptical acceptance decision problems together are considered to be a good indicator for the complexity of a semantics.

In this section we assume the reader to have basic knowledge in computational complexity theory.[7] We will consider the following complexity classes: L (logarithmic space), P (polynomial time), NP (non-deterministic polynomial time), coNP (complement of a NP problem), Θ_2^{P} (polynomial time with non-adaptive NP-oracle calls), Σ_2^{P} (non-deterministic polynomial time with NP-oracle calls), Π_2^{P} (complement of a Σ_2^{P} problem), and $\mathsf{D}_2^{\mathsf{P}}$ (intersection of a Σ_2^{P} and a Π_2^{P} language).

[7] For a gentle introduction into complexity theory in the context of argumentation the reader is referred to [47]).

We have the following relations between these complexity classes:

$$\mathsf{L} \subseteq \mathsf{P} \subseteq \begin{matrix}\mathsf{NP}\\ \mathsf{coNP}\end{matrix} \subseteq \Theta_2^{\mathsf{P}} \subseteq \begin{matrix}\Sigma_2^{\mathsf{P}}\\ \Pi_2^{\mathsf{P}}\end{matrix} \subseteq \mathsf{D}_2^{\mathsf{P}}$$

We follow [49] and start our complexity analysis with the observation that SETAFs generalize Dung AFs and thus all the decision problems are at least as hard as the corresponding problem for Dung AFs (cf. [47, Table 1]). Interestingly, one can also obtain the same upper bounds (see Table 4) as we discuss below. These results for SETAFs show the same complexity as the corresponding Table for Dung AFs (cf. [47, Table 1][8]). However, there is a subtle difference between the complexity results for Dung AFs and SETAFs. In both cases the complexity is stated w.r.t. the size of the input framework, which in case of Dung AFs is often interpreted as w.r.t. the number of arguments $|Ar|$ in the input framework. This interpretation is not valid for SETAFs where the number of attacks $|\triangleright|$ can be exponentially larger than the number of arguments $|Ar|$ (this even holds for normal forms where redundant attacks are removed). Thus, one has to consider the complexity w.r.t. the number of arguments plus the representation size of the attacks $\triangleright$. The latter is bounded bound by $|Ar| \cdot |\triangleright|$, i.e., is polynomially bounded in $|Ar| + |\triangleright|$. We can thus interpret the complexity results for SETAFs in Table 4 as w.r.t. $|Ar| + |\triangleright|$ [9].

The crucial observation towards the upper bounds is that checking basic properties of a set of arguments, although it is more evolved than in Dung AFs, can still be performed in L. First, to test whether a set S is conflict-free one can iterate over all attacks $(T, a) \in \triangleright$ and check that $T \cup \{a\} \not\subseteq S$. Second, to test $S \blacktriangleright T$ one can iterate over all attacks $(U, b) \in \triangleright$ and test whether $U \subseteq S$ and $b \in T$. Finally, a simple algorithm for testing that a set S defends an argument a iterates over all attacks $(T, a) \in \triangleright$ and for each of these attacks checks that $S \blacktriangleright T$. That is, for all three problems we just need to store a small number of pointers to the input which can be done in logarithmic space.

Proposition 3.25. *Given a SETAF $AF^{\mathcal{S}} = \langle Ar, \triangleright \rangle$, a set of arguments $S \subseteq Ar$, and an argument $a \in Ar$, deciding whether S is conflict-free, deciding whether $S \blacktriangleright a$, and deciding whether $a \in \mathrm{F}_{AF^{\mathcal{S}}}(S)$ are in L.*

Notice that most of the complexity upper bounds for Dung AFs are based on the fact that these three problems can be solved in polynomial-time, and thus these

[8] Notice that [47, Table 1] includes $\mathcal{CF}2$ semantics which has not yet been generalised to SETAFs and is thus not included in Table 4. On the other hand, we include eager semantics which has not been considered in [47, Table 1] (see [45, Table 2] for the complexity results of eager semantics in Dung AFs).

[9] For a more fine-grained analysis of algorithms for SETAFs one might take into account the actual representation size of the attacks, cf. [53].

upper bounds also apply to SETAFs (cf. Table 4). We next exemplify this for the credulous acceptance problem of stable semantics.

Proposition 3.26. *We have that* $Ver_{\mathcal{ST}} \in \mathsf{L}$ *and* $Cred_{\mathcal{ST}}$ *is* NP*-complete.*

Proof. First, consider the verification problem $Ver_{\mathcal{ST}}$ and an arbitrary SETAF $AF^{\mathcal{S}} = \langle Ar, \triangleright \rangle$. We can verify that a given set S is a stable extension of $AF^{\mathcal{S}}$ by (a) checking that S is conflict-free and (b) checking that for each $a \in Ar \setminus S$ we have $S \blacktriangleright a$. As both can be done in L, we obtain the L membership of $Ver_{\mathcal{ST}}$.

Now consider the credulous acceptance problem $Cred_{\mathcal{ST}}$. The NP-hardness is by the corresponding result for AFs. For the upper bound consider an arbitrary SETAF $AF^{\mathcal{S}} = \langle Ar, \triangleright \rangle$ and an argument $a \in Ar$. We can decide the credulous acceptance of a in $AF^{\mathcal{S}}$ by a standard guess & check algorithm. That is, one first uses the non-determinism to guess a set E and then use a deterministic part to verify that E is a stable extension and contains the argument a. This gives an NP procedure for $Cred_{\mathcal{ST}}$. □

Next, let us consider the complexity of ideal semantics, as it is the only case where the upper bound for Dung AFs [43] does not directly apply to SETAFs. Recall that the ideal extension can be characterised as the maximal admissible set that is not attacked by any other admissible set (Definition 3.15). In order to compute the ideal extension we thus use NP-oracle queries that for each argument ask (a) whether it is credulously accepted w.r.t. preferred semantics and (b) whether it is attacked by some admissible set. We then consider the set E^0 of all arguments that are credulously accepted but not attacked by an admissible set. Notice that E^0 is conflict-free by construction and it is an over-approximation of the ideal extension. We then compute the maximal admissible subset of E^0 by iteratively computing sets E^{i+1} by removing arguments that are not defended by E^i until we reach a fixed-point E. We then have that E is the ideal extension. We have that the NP-oracle queries of the above procedure are independent of each other and thus can be executed in parallel. Moreover, each iteration of the fixed-point computation is in polynomial-time and the fixed-point is reached after at most $n/2$ iterations, i.e., one can compute the fixed-point in polynomial time. Thus the above is a Θ_2^{P}-algorithm for computing the ideal extension. Hence, we obtain Θ_2^{P} upper bounds for all reasoning tasks of ideal semantics.

3.3.3 Algorithms for SETAFs

The field of algorithms for SETAFs is rather under-explored with the exception of [82]. The former studies algorithmic ideas for preferred semantics. We recapitulate

σ	$Cred_\sigma$	$Skept_\sigma$	Ver_σ	$Exists_\sigma$	$Exists_\sigma^{\neg\emptyset}$
Conflict-free	in L	trivial	in L	trivial	in L
Naive	in L	in L	in L	trivial	in L
Grounded	P-c	P-c	P-c	trivial	in L
Stable	NP-c	coNP-c	in L	NP-c	NP-c
Admissible	NP-c	trivial	in L	trivial	NP-c
Complete	NP-c	P-c	in L	trivial	NP-c
Ideal	Θ_2^P-c	Θ_2^P-c	Θ_2^P-c	trivial	Θ_2^P-c
Eager	Π_2^P-c	Π_2^P-c	D_2^P-c	trivial	Π_2^P-c
Preferred	NP-c	Π_2^P-c	coNP-c	trivial	NP-c
Semi-stable	Σ_2^P-c	Π_2^P-c	coNP-c	trivial	NP-c
Stage	Σ_2^P-c	Π_2^P-c	coNP-c	trivial	in L

Table 4: Complexity of SETAFs ($\mathcal{C}$-c denotes completeness for class $\mathcal{C}$).

their main observations in terms of a simple algorithm (see Algorithm 1) in the style of today's labelling-based algorithms ([36; 35]).

The rough idea of labelling-based algorithms is to start with all arguments unlabelled, in each step pick an argument and then consider two branches: one where we add the argument to the extension, i.e., labelled `in`; and one where we decide that the argument is excluded from the extension, i.e., labelled `out` or `undec` (cf. Section 3.1.5). When all arguments are labelled, one tests whether the labelling is valid w.r.t. the considered semantics and, if so, it is added to the output. By that procedure we would consider all possible candidates for valid labellings and thus also obtain all the extensions. In order to design an efficient algorithm one aims to cut off branches that do not lead to valid labellings as soon as possible. One approach are the so-called label propagation rules, i.e., one uses the already fixed labels of the arguments to conclude that other arguments have to obtain a certain label and by that avoids unnecessary branching in the algorithm. For instance, for preferred semantics, given the set of arguments $\mathcal{L}ab_{\texttt{in}}$ labelled `in` by a partial labelling $\mathcal{L}ab$ we can conclude that all arguments in the set $\mathcal{L}ab^+_{\texttt{in}}$, i.e., arguments a with $\mathcal{L}ab_{\texttt{in}} \blacktriangleright a$, must be labelled `out`. Moreover, for attacks that target $\mathcal{L}ab_{\texttt{in}}$ and have only one argument outside of $\mathcal{L}ab^+_{\texttt{in}}$ we have that this argument has to be labelled `out`. This is captured by the set $\mathcal{L}ab^{\leftarrow}_{\texttt{in}}$ defined as $\mathcal{L}ab^{\leftarrow}_{\texttt{in}} = \{a \in Ar \mid \mathcal{L}ab_{\texttt{in}} \cup \{a\} \blacktriangleright \mathcal{L}ab_{\texttt{in}}\}$. This propagation of `out` labels is implemented in Line 8 of Algorithm 1 and triggered

whenever a new argument is labelled `in`. Another observation is that we cannot label an argument `in` if this would cause a conflict in the set $\mathcal{L}ab_{\texttt{in}}$. Many cases where this could happen are already covered by the propagation rules for `out` labels, but these rules do not cover attacks $(S \cup \{a\}, a) \in \triangleright$ with $S \subseteq \mathcal{L}ab_{\texttt{in}}$. This propagation is implemented by the if condition on Line 4, which prevents the algorithm from starting the branch where the argument a is added to the extension. Finally, when an argument a is already defended by $\mathcal{L}ab_{\texttt{in}}$ then, due to the maximality of preferred extensions, we know that this argument is in each preferred extension containing $\mathcal{L}ab_{\texttt{in}}$ and thus we must label a by `in`. This propagation is implemented by the if condition on Line 12, which prevents the algorithm from starting the branch where the argument a is excluded from the extension.

We obtain that Algorithm 1 returns the preferred labellings of a given SETAF $AF^{\mathcal{S}}$. Notice that the algorithm can be easily adapted to compute complete labellings, by removing the maximality check on Line 16, or admissible sets, by removing the maximality check on Line 16 and the if condition on Line 12. We can roughly estimate the running time of these algorithms by $O(\exp(|Ar|) \cdot \mathsf{poly}(|Ar|, |\triangleright|))$. Notably only the polynomial part depends on the number of attacks while the exponential part solely depends on the number of arguments. Finally, recent work [53] suggests to not just label arguments but also label the attacks of a SETAF. It then studies possible label propagation-rules for stable and complete semantics and provides a linear time algorithm (linear w.r.t. the representation of the SETAF) for grounded semantics.

3.3.4 Systems and Reduction-based Approaches

Reduction-based approaches have been successfully applied in the design of argumentation systems, most prominently by systems that are based on modern SAT-solver technology or answer-set programming [35]. For SETAFs the only system discussed in the literature, i.e., the SETAF module of the ASPARTIX[10] system [49], is based on answer-set programming.

Reduction to Answer-set Programming. Answer-set programming (ASP) [79; 85] is a declarative problem solving paradigm with its roots in logic programming and non-monotonic reasoning. Today's answer-set systems [66; 75] support a rich language and are capable of solving hard problems efficiently. Thus, ASP is a convenient formalism to implement argumentation systems. The ASPARTIX approach [57] to argumentation problems relies on a query-based implementation

[10] `https://www.dbai.tuwien.ac.at/research/argumentation/aspartix`

Algorithm 1 pref-lab($AF^{\mathcal{S}}$)

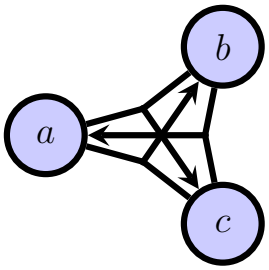
Require: SETAF $AF^{\mathcal{S}} = \langle Ar, \triangleright \rangle$, global variable $\mathcal{L}$
Ensure: $\mathcal{L}$ is the set of preferred labellings
1: $\mathcal{L} = \emptyset$, $\mathcal{L}ab = \langle \emptyset, \emptyset, \emptyset \rangle$
2: pref-lab($AF^{\mathcal{S}}, \mathcal{L}ab$)
3: **function** pref-lab($F, \mathcal{L}ab$)
Require: SETAF $F = \langle A, R \rangle$, partial labelling $\mathcal{L}ab$, global variable $\mathcal{L}$
4: **if** there is an argument $a \in A$ not labeled by $\mathcal{L}ab$ **then**
5: $a \leftarrow$ pick some unlabeled argument
6: **if** $\mathcal{L}ab_{\mathtt{in}} \cup \{a\} \in \mathcal{CF}(F)$ **then**
7: $\mathcal{L}ab'_{\mathtt{in}} = \mathcal{L}ab_{\mathtt{in}} \cup \{a\}$,
8: $\mathcal{L}ab'_{\mathtt{out}} = \mathcal{L}ab_{\mathtt{out}} \cup \mathcal{L}ab'^{+}_{\mathtt{in}} \cup \mathcal{L}ab'^{\leftarrow}_{\mathtt{in}}$
9: $\mathcal{L}ab'_{\mathtt{undec}} = \mathcal{L}ab_{\mathtt{undec}} \setminus \mathcal{L}ab'_{\mathtt{out}}$
10: pref-lab($AF^{\mathcal{S}}, \langle \mathcal{L}ab'_{\mathtt{in}}, \mathcal{L}ab'_{\mathtt{out}}, \mathcal{L}ab'_{\mathtt{undec}} \rangle$)
11: **end if**
12: **if** $\{a\} \notin \mathcal{F}_F(\mathcal{L}ab_{\mathtt{in}})$ **then**
13: pref-lab($AF^{\mathcal{S}}, \langle \mathcal{L}ab_{\mathtt{in}}, \mathcal{L}ab_{\mathtt{out}}, \mathcal{L}ab_{\mathtt{undec}} \cup \{a\} \rangle$)
14: **end if**
15: **else**
16: **if** $\mathcal{L}ab_{\mathtt{in}} \in \mathcal{AD}(F)$ **and** $\mathcal{L}ab_{\mathtt{in}}$ $\subseteq$-max among $\{\mathcal{L}ab_{\mathtt{in}} \mid \mathcal{L}ab \in \mathcal{L}\}$ **then**
17: $\mathcal{L} = \mathcal{L} \cup \{\mathcal{L}ab\}$
18: **end if**
19: **end if**
20: **endFunction**

where the argumentation framework is provided as an input database, and one provides fixed queries encoding the different argumentation semantics and reasoning tasks.

Here we briefly highlight the main differences between the ASP encodings of Dung AFs [57] and SETAFs [49]. To this end, we first briefly recall the basic terminology for logic programs (for rigorous definitions see [35] or [36]). A logic program (under the answer-set semantics) is a set of disjunctive rules r of the form

$$a_1 \vee \cdots \vee a_n \ \leftarrow b_1, \ldots, b_k, \ \mathit{not}\ b_{k+1}, \ldots, \ \mathit{not}\ b_m$$

where $a_1, \ldots, a_n$ and $b_1, \ldots, b_m$ are atoms, and *not* stands for *default negation*. We refer to a as a positive literal, while we refer to *not* a as a default negated literal. The *head* of r is the set $\{a_1, \ldots, a_n\}$ and the *body* of r is $\{b_1, \ldots, b_m\}$, and a rule



$\pi_{setaf}(AF^{\mathcal{S}})$

$\mathbf{arg}(a).\ \mathbf{arg}(b).\ \mathbf{arg}(c).$
$\mathbf{att}(r1, c).\ \mathbf{mem}(r1, a).\ \mathbf{mem}(r1, b).$
$\mathbf{att}(r2, b).\ \mathbf{mem}(r2, a).\ \mathbf{mem}(r2, c).$
$\mathbf{att}(r3, a).\ \mathbf{mem}(r3, b).\ \mathbf{mem}(r3, c).$

Figure 5: The SETAF $AF^{\mathcal{S}} = \langle \{a, b, c\}, \{(\{a, b\}, c), (\{a, c\}, b), (\{b, c\}, a)\} \rangle$ and its ASP-encoding $\pi_{setaf}(AF^{\mathcal{S}})$.

$\pi_{\mathcal{CF}}$

$\mathbf{in}(Y) \leftarrow \mathbf{arg}(Y), \mathit{not}\ \mathbf{out}(Y).$
$\mathbf{out}(Y) \leftarrow \mathbf{arg}(Y), \mathit{not}\ \mathbf{in}(Y).$
$\mathbf{blocked}(R) \leftarrow \mathbf{mem}(R, X), \mathbf{out}(X).$
$\leftarrow \mathbf{in}(X), \mathbf{att}(R, X), \mathit{not}\ \mathbf{blocked}(R).$

$\pi_{\mathcal{AD}}$

$\mathbf{in}(Y) \leftarrow \mathbf{arg}(Y), \mathit{not}\ \mathbf{out}(Y).$
$\mathbf{out}(Y) \leftarrow \mathbf{arg}(Y), \mathit{not}\ \mathbf{in}(Y).$
$\mathbf{blocked}(R) \leftarrow \mathbf{mem}(R, X), \mathbf{out}(X).$
$\leftarrow \mathbf{in}(X), \mathbf{att}(R, X), \mathit{not}\ \mathbf{blocked}(R).$
$\mathbf{defeated}(R) \leftarrow \mathbf{att}(R, X), \mathbf{mem}(R, Y),$
$\mathbf{att}(R2, Y), \mathit{not}\ \mathbf{blocked}(R2).$
$\leftarrow \mathbf{in}(X), \mathbf{att}(R, X), \mathit{not}\ \mathbf{defeated}(R).$

Figure 6: ASP Encodings $\pi_{\mathcal{CF}}$, $\pi_{\mathcal{AD}}$ for $\mathcal{CF}$ and $\mathcal{AD}$ semantics of SETAFs.

r is a *constraint* if $n = 0$. A *fact* is a ground rule without disjunction ($n = 1$) and with an empty body. An *input database* is a set of facts.

In order to evaluate SETAFs with ASP, in a first step, we have to encode SETAFs as an input database for the ASP-program. We introduce three predicates **arg**, **att**, and **mem** to encode a SETAF $AF^{\mathcal{S}} = \langle Ar, \triangleright \rangle$. The predicate **arg** is used to encode arguments, the latter two to encode the set attacks, i.e., **att** encodes which argument is attacked by an attack and **mem** encode which arguments are required to attack that argument. Notice that, this encoding uses a unique identifier for each attack in $\triangleright$. The encoding of a SETAF $AF^{\mathcal{S}} = \langle Ar, \triangleright \rangle$ is then given by $\pi_{setaf}(AF^{\mathcal{S}}) = \{\mathbf{arg}(a). \mid \text{for } a \in Ar\} \cup \{\mathbf{att}(r, x). \mid \text{for } r \in \triangleright \text{ and } r = (S, x)\} \cup \{\mathbf{mem}(r, y). \mid \text{for } r \in \triangleright, r = (S, x), \text{ and } y \in S\}$ (cf. Figure 5). While arguments are represented in the same way as in Dung AFs, Dung AFs allow for a simpler representation of attacks. That is, the encoding of AFs ([35]) only uses one binary predicate **att** to encode the attacks, containing the attacker and the attacked argument of each attack, and does not use identifiers for attacks.

When it comes to the encoding of semantics one uses predicates $\mathbf{in}(\cdot)$, $\mathbf{out}(\cdot)$

to guess whether an argument is in the extension or not (in the same way as for AFs). Notice that the predicate **out**$(\cdot)$ encodes that an argument is not in the extension and does not correspond to the label `out`. This guess builds up all possible subsets of arguments which are then filtered by adding constraints that reflect the specific semantics. Here the SETAF encodings differ from the AF encodings as they explicitly define statuses of attacks. First, we call an attack $(T, a) \in \triangleright$ blocked w.r.t. a set $E \subseteq Ar$ if $T \not\subseteq E$. Second, we consider an attack $(T, a) \in \triangleright$ to be defeated by a set E iff $E \blacktriangleright T$. We will exemplary discuss the encodings $\pi_{\mathcal{CF}}$, $\pi_{\mathcal{AD}}$ for conflict-free sets and admissible sets respectively (cf. Figure 6). In the encoding of the conflict-freeness, with the first two rules one guesses a subset of arguments, the third rule computes the blocked attacks, and the constraint in the fourth line rules out all sets that contain an argument X and have a non-blocked rule attacking X. That is, if we compute the answer-sets of the combined program $\pi_{setaf}(AF^{\mathcal{S}}) \cup \pi_{\mathcal{CF}}$ the answer-sets correspond to the conflict-free sets, i.e., the conflict-free sets are given by the **in**$(\cdot)$ predicate in the answer-sets. Next, we further extend $\pi_{\mathcal{CF}}$ to an encoding $\pi_{\mathcal{AD}}$ for admissible semantics. That is, we add a rule that computes the defeated attacks and a constraint that rules out sets where an argument of the set is attacked by an undefeated attack. Thus, if we compute the answer-sets of the combined program $\pi_{setaf}(AF^{\mathcal{S}}) \cup \pi_{\mathcal{AD}}$ the answer-sets correspond to the admissible sets.

Other Reduction-based Approaches. For Dung AFs and their generalisations, several reduction-based approaches have been studied in the literature and often resulted in argumentation systems [36]. In particular, systems based on modern SAT-solving systems have been successful [101; 64]. Beside ASP, none of these approaches have been considered in the literature on SETAFs so far. However, very recently a first version of the SAT-based SETAF system *joukko* appeared online[11]. Thus, one approach towards an efficient SETAF system would be to extend existing approaches that have been successful for AFs to SETAFs. Another approach is to translate SETAFs to AFs or ADFs and use one of the existing systems for these formalisms to evaluate SETAFs. Translations from SETAFs to AFs have been presented in [94] and [60](see also Section 3.2 in this chapter). However, when using these translations one is faced with an exponential blow-up in the arguments and thus these translations are not well-suited for computational matters. Recall, that algorithms for SETAFs scale polynomially w.r.t. the number of attacks and exponentially w.r.t. the number of arguments. Thus translating attacks to arguments and using AFs tools can results in a serious computational overhead. Con-

[11] `https://bitbucket.org/andreasniskanen/joukko`

cerning the latter, there are rather simple translations of SETAFs into ADFs [77; 93] (see Section 3.2.4) which do not increase the number of arguments. That is, one can efficiently encode a SETAF as an ADF and then use one of the existing systems for ADFs, e.g., k++ADF[12] [76], YADF[13] [21], or DIAMOND[14] [58], to evaluate the SETAF. The attentive reader may argue that the computational complexity of ADFs is higher than that of SETAFs and thus such a reduction might result in significant overheads. However, modern ADF systems are sensitive to the actual complexity of the acceptance conditions in the processed ADF and thus the overheads when processing ADFs with acceptance conditions generated from SETAFs probably will not be as high as one would expect from the worst-case complexity gap.

3.4 Alternative models for attacks involving sets of arguments

SETAFs have not been the only attempt to formalise *collective* attacks[15] in argumentation systems. There have been earlier or more recent related approaches, both in abstract and structured argumentation, each of which captures a slightly different notion of collective attack and with a different aim.

One of the earliest approaches to formalise collective attacks in abstract argumentation was the *collective argumentation theories* proposed by [18]. These are generalisations of Dung's abstract argumentation frameworks aimed at the representation of the semantics of disjunctive logic programs, but also, more generally, at the description of "reasoning situations in which the conflict between incompatible views or theories is global and cannot be reduced to particular claims made by these theories". [18] proposes a four-valued semantics, i.e., each argument is assigned a subset of $2^{\{t,f\}}$ and attacks occur among sets of arguments (e.g. $S \hookrightarrow T$) and are interpreted as "at least one of the arguments in the attacked set (T) should be rejected whenever all the arguments from the attacking set (S) are accepted".

[33] introduced the notion of coalitions of arguments to represent sets of non-conflicting arguments that are related via the support relation in a bipolar argumentation framework (BAF). Using this notion, a bipolar argumentation framework $AF^{\mathcal{B}}$ can be translated into a Dung-style meta-argumentation framework $C(AF^{\mathcal{B}})$, called "Coalition AF", in which the arguments represent coalitions of arguments of $AF^{\mathcal{B}}$ and the attacks among arguments (called *c-attacks*) correspond to attacks among elements of the corresponding coalitions: S c-attacks T in $C(AF^{\mathcal{B}})$, iff there

[12] `https://www.cs.helsinki.fi/group/coreo/k++adf/`

[13] `http://www.dbai.tuwien.ac.at/proj/adf/yadf/`

[14] `http://diamond-adf.sourceforge.net/`

[15] We use the term collective to refer to any kind of attack relation that involves sets of arguments, either as attackers or as targets of an attack, or both.

exist arguments $a, b \in AF^{\mathcal{B}}$ such that $a \in S, b \in T$ and a attacks b in $AF^{\mathcal{B}}$. All arguments belonging to a coalition are then treated in the same way when computing the acceptable arguments: an argument a is acceptable (under the preferred, stable or grounded semantics) in $AF^{\mathcal{B}}$ iff it is a member of a coalition S, which is acceptable (under the same semantics) in $C(AF^{\mathcal{B}})$.

The framework proposed in [63] also considers sets of arguments, but as recipients of disjunctive attacks from single arguments. In this framework, the result of an attack from an argument a that is labelled `in`, to a set of arguments S, is that at least one of the arguments in S must be labelled `out`. Definition 2.8 and Theorem 2.9 of the same paper show how a finite disjunctive framework can be converted to a Dung-style AF with the same set of extensions, which, combined with the results on the relationship between SETAF and AF that we present in Section 3.2, provide a way to associate SETAF with disjunctive argumentation frameworks. Note also that [84] also provides a way to model disjunctive attacks using SETAFs, using the notion of "indeterministic defeat" [104] (see Section 3.1 for details).

CumulA [103; 104] is an example of a structured argumentation model that supports collective attacks. In this model, arguments are tree-like structures that represent how a conclusion is supported. In order to support situations where a set of arguments should be collectively defeated (*collective defeat*) or at least one of the arguments in a set should be defeated (*indeterministic defeat*), it uses *compound defeaters*, i.e., attack relations where either the source or the target of the attack (or both) are sets of arguments. The meaning of a compound defeater is different than that of joint attacks in SETAFs: if all arguments in the attacking set are undefeated, the arguments in the attacked set are defeated as a group unless one of the arguments in the attacked set has already been defeated by another defeater. In the latter case the compound defeater becomes *inactive*.

Another structured argumentation formalism that incorporates the notion of collective attacks is the Abstract Argumentation Systems (AAS) from [106]. An AAS is defined as a triple $(\mathcal{L}, \mathcal{R}, \leq)$, where $\mathcal{L}$ is a language containing the symbol $\bot$, which represents a contradictory proposition, $\mathcal{R}$ is a set of (strict and defeasible) inference rules, and $\leq$ is a preorder on the set of arguments, called *order of conclusive force*, and determining "the relative difference in strength among arguments". Arguments are defined as chains of rules organised as trees. The notion of defeat in AAS is used to capture and resolve conflicts among groupwise incompatible arguments: a set of arguments X defeats an argument a if $X \cup \{a\}$ is incompatible (there is a strict argument b that is based on the conclusions of $X \cup \{a\}$ and has conclusion $\bot$) and X is not undermined by a (there is no $c \in X$ such that $a < c$).

Defeasible Logic [89], which, as shown in [69] has an argumentation-theoretic semantics, also supports a type of collective attacks, called *team defeat*. This logic

includes a rule priority relation, which is used to resolve conflicts between rules with contradictory conclusions. An attack on a rule r with conclusion p from a rule r' with conclusion $\neg p$ can be invalidated by another rule r'' also with conclusion p that is superior to r'. In this case, we say that r and r'' team defeat r'. Using this feature, we conclude that p is true if for every applicable rule that supports $\neg p$, there is a superior rule for p; in other words, if the rules for $\neg p$ are team defeated by the rules for p[16]. In order to support this feature, the argumentation-theoretic characterisation of Defeasible Logic defines arguments as sets of proof trees supporting the same conclusion and team defeat as a relation between two arguments with opposite conclusions, and requires that an argument team defeats all its attacking arguments to become acceptable. Team defeat is also supported by other rule-based non-monotonic logics, which use preferences on rules, such as Courteous Logic Programs [70] and Order Logic [73]. An interesting problem is to study the possibility of mapping Defeasible Logic, or any of the other rule-based non-monotonic logic that supports team defeat, to SETAF by defining arguments as proof trees and by representing team defeat, between a set of rules R supporting the same conclusion and a rule s supporting the opposite conclusion, as a joint attack from the set of arguments that have a top rule in R to each argument that has s as its top rule.

[10] recently introduced a semi-structured formalism for argumentation, called LAF-ensembles, capturing a set of essential features of structured arguments, such as their conclusion, their "attackable elements" and their subarguments. They also defined a family of abstract argumentation frameworks, called *set-based* (as their nodes correspond to sets of arguments instead of individual arguments), which are appropriate for representing LAF-ensembles at the abstract level. In set-based argumentation frameworks, the attacks occur at the set level. The main differences between set-based frameworks and SETAFs are that the former allow attacks on sets of arguments and attacks where the source is the empty set; the latter are useful to capture inconsistencies of the theory at the language level (e.g., incompatible subsets of the language in Vreeswijk's AAS [106]).

Finally, it should be noted that, as also explained in Section 3.2 and shown in [93], ADFs are generalisations of SETAFs and can therefore model the type of collective attacks used in SETAFs. This is done by setting the following acceptance condition for each argument a: at least one argument from each of the sets of arguments attacking a should be rejected.

[16] For a more detailed discussion on team defeat, see [17]

4 Applications of joint attacks and models for joint supports

The ideas behind the characterisation of abstract argumentation frameworks with joint attacks, as those described in Section 3, have also been applied in other contexts. In this section we will focus on applications of joint attacks in Bipolar Argumentation Frameworks (BAFs) and argumentation frameworks with higher-order interactions[17].

Briefly, BAFs extend Dung's AF by incorporating a support relation intended to model a positive interaction between the elements it relates. The first works accounting for bipolarity in abstract argumentation conceived the support relation as a binary relation over the set of arguments in the framework (see [34; 37] for an overview on BAFs). However, later approaches adopted a different view of the support relation, to also account for *joint supports* (i.e., support relations whose source is a set of arguments) or, more generally, *higher-order supports* (i.e., support relations that can target other interactions, either attacks or supports), in addition to arguments.

In this section we will consider approaches to bipolar abstract argumentation that make use of joint attacks, joint supports or both. Finally, we will discuss the possibility of using joint attacks for modelling higher-order attacks and supports (i.e., interactions whose target is another interaction) and the generalised necessary support relation proposed in [88] and also accounted for in [31].

4.1 Flat bipolar argumentation frameworks with joint attacks or joint supports

In [91] the authors used the SETAF as the underlying framework for representing evidence against an argument in order to allow for evidence-based reasoning. They introduced the Evidential Argumentation System (EAS) which further extended the definition of SETAF by incorporating a specialised support relation to capture the notion of *evidential support*. The support relation in the EAS enables to distinguish between *prima-facie* and *standard* arguments; the former arguments do not require support from other arguments to stand, whereas the latter must be linked to at least one *prima-facie* argument through a chain of supports. Moreover, the prima-facie arguments are supported by a special argument η denoting support from the environment or the existence of supporting evidence. Also, analogously to the attack relation, the support relation in an EAS allows for supports to be originated on sets

[17] The latter are the subject of study in Chapter 1 of this handbook [29].

of arguments. Formally:

Definition 4.1. *An EAS is a tuple $\langle Ar, att, sup\rangle$, where Ar is a set of arguments, $att \subseteq (2^{Ar}\backslash\emptyset) \times Ar$ is the attack relation, and $sup \subseteq (2^{Ar}\backslash\emptyset) \times Ar$ is the support relation. A special argument $\eta \in Ar$ is distinguished, such that $\nexists(X, y) \in att$ where $\eta \in X$; and $\nexists X$ where $(X, \eta) \in att$ or $(X, \eta) \in sup$.*

The attack relation in an EAS is interpreted in the same way as the attack relation in the SETAF. Given $X \subseteq Ar$ and $a \in Ar$, $(X, a) \in att$ reads as follows: if all the arguments in X are accepted, then a cannot be accepted. In contrast, the evidential support relation is interpreted as follows. Given $X \subseteq Ar$ and $a \in Ar$, $(X, a) \in sup$ reads as: "the acceptance of a requires the acceptance of every argument in X".

Since the core idea of the EAS is that valid arguments (in particular, those originating attacks) need to trace back to the environment, the authors define the notion of evidence supported attack (e-supported attack). Then, based on this notion, semantics for the EAS have been characterized in [91] and then reformulated in [95], following Dung's methodology.

The Generalised Argumentation Frameworks with Necessities (GAFNs) [88] (directly referred to as AFNs in [87]) are another kind of bipolar argumentation frameworks that account for interactions between single arguments and sets of arguments but in a different way: a necessity relation between a set of arguments S and an argument a means that the acceptance of a requires the acceptance of *at least one* argument in S.

To illustrate the support relation of GAFNs, let us consider the following example. Suppose that in order to be awarded with a scholarship (s) a student is required to obtain a Bachelor's degree with honours (bh) or justify modest income (mi). In addition, suppose that the student has a bad mark (bm), and that having a bad mark prevents the student from obtaining the honours (regardless of the average of marks). We can represent this scenario by a GAFN with arguments s, bh, mi and bm. On the other hand, there exists an attack from bm to bh, and there exists a necessary support from the set $\{bh, mi\}$ to argument s. It is important to note that, even though the attack from bm to bh will result in bh not being accepted, this does not prevent s from being accepted (in other words, the student will obtain the scholarship). This is because the support towards s is originated in the set $\{bh, mi\}$, where each argument within this set provides an alternative condition for obtaining the scholarship.

Generalised Argumentation Frameworks with Necessities are formally defined as follows:

Definition 4.2. *A Generalised Argumentation Framework with Necessities (GAFN) is defined by a tuple* $\langle Ar, att, sup\rangle$ *where* Ar *is a set of arguments,* $att \subseteq Ar \times Ar$ *is an attack relation and* $sup \subseteq ((2^{Ar}\backslash\emptyset) \times Ar)$ *is a necessity relation.*

In [87] the author proposed a characterisation of semantics for the GAFN, in addition to those given in [88]. Finally, it should be noted that in [95] the authors provided a translation allowing the transformation of a GAFN into an EAS. Briefly, this translation is such that unsupported arguments in the GAFN will be arguments supported by η in the EAS; on the other hand, all sets of supporting arguments in the GAFN are combined into different sets of supporting arguments in the EAS by accounting for their Cartesian product. Finally, for the attack relation it suffices to map the attacking arguments in the GAFN into singleton sets of attacking arguments in the EAS. Then, [95] formally established a correspondence between the EAS and the GAFN in terms of their semantics, and identified correspondences between the properties of both frameworks and properties of Dung's AF.

Example 4.3. *Consider the GAFN* $\langle Ar, att, sup\rangle$*, where:*

- $Ar = \{a, b, c, d, e, f\}$
- $att = \{(b, a), (e, a), (c, d)\}$
- $sup = \{(\{b\}, e), (\{d, f\}, e), (\{a\}, d)\}$

This AFN could be translated into the EAS $\langle Ar', att', sup'\rangle$*, where:*

- $Ar' = Ar \cup \{\eta\}$
- $att' = \{(\{b\}, a), (\{e\}, a), (\{c\}, d)\}$
- $sup' = \{(\{b, d\}, e), (\{b, f\}, e), (\{a\}, d), (\{\eta\}, a), (\{\eta\}, b), (\{\eta\}, c), (\{\eta\}, f)\}$

For details about the characterisation of semantics for EAS and GAFN we refer the reader to [91] and [88; 87], respectively.

4.2 Bipolar argumentation frameworks with joint attacks or joint supports and higher-order interactions

The ideas adopted by the EAS and the GAFN described in the previous section were further exploited in [30] and [31], where the authors introduced the Recursive Evidence-Based Argumentation Framework (REBAF) and the Recursive Argumentation Framework with Necessity (RAFN). Briefly, these frameworks extend Dung's

AF by accounting for attack and support relations that can target not only arguments, but also attacks or supports at any level[18]. As a result, the REBAF adopts the evidential interpretation for the support relation of [91], whereas the RAFN adopts the generalised necessity interpretation of support proposed in [88]. The formal definitions of these frameworks are included below:

Definition 4.4. *A Recursive Evidence-Based Argumentation Framework (REBAF) is a tuple* $\langle Ar, att, sup, \mathbf{s}, \mathbf{t}, PF\rangle$ *where* Ar*, att and sup are pairwise disjoint sets respectively representing the names of arguments, attacks and supports, and* $PF \subseteq Ar \cup att \cup sup$ *is a set representing the* prima-facie *elements of the framework that do not need to be supported. The functions* $\mathbf{s} : (att \cup sup) \mapsto 2^{Ar} \setminus \emptyset$ *and* $\mathbf{t} : (att \cup sup) \mapsto (Ar \cup att \cup sup)$ *respectively map each attack and support to its source and its target.*

Definition 4.5. *A Recursive Argumentation Framework with Necessity (RAFN) is a tuple* $\langle Ar, att, sup, \mathbf{s}, \mathbf{t}\rangle$*, where* Ar*, att and sup are pairwise disjoint sets respectively representing the names of arguments, attacks and supports. The function* $\mathbf{s} : (att \cup sup) \mapsto 2^{Ar} \setminus \emptyset$ *and* $\mathbf{t} : (att \cup sup) \mapsto (Ar \cup att \cup sup)$ *respectively map each attack and support to its source and its target. It is assumed that* $\forall \alpha \in att$, $\mathbf{s}(\alpha)$ *is a singleton.*

Note that, according to Definition 4.4, attacks and supports in a REBAF can have a set of arguments as their source. In contrast, by Definition 4.5, the attack relation of a RAFN is restricted to only allow for arguments as the source of attacks. Then, in both cases, an attack or a support can also be the target of an interaction. Consequently, since these frameworks allow to reason about interactions in addition to arguments, the attacks and supports are also accounted for in the acceptability calculus.[19]

Semantics of REBAF and RAFN are defined using a notion of *structure*, defined as a triple $U = (S, \Gamma, \Delta)$ such that $S \subseteq Ar$, $\Gamma \subseteq att$ and $\Delta \subseteq sup$. Then, the notions of conflict-freeness, acceptability and admissibility as well as the subsequent semantics are defined over these structures, with the idea that the set S represents the set of "acceptable" arguments w.r.t. the structure U, and the sets Γ and Δ respectively represent the sets of "valid attacks" and "valid supports" w.r.t. U. For details about the definition of semantics for REBAF and RAFN, we refer the reader to [30] and [31], respectively, or to Chapter 1 of this handbook [29].

[18]The REBAF and the RAFN are studied in more detail in Chapter 1 of this handbook [29].

[19]This feature is also shared by other frameworks such as the AFRA and the ASAF, discussed in Section 4.3.

4.3 Using joint attacks to model higher-order interactions and generalised necessary supports

Gabbay [62] proposed the Higher-Level Argumentation Frames (HLAFs), which extend Dung's framework by allowing for attacks from arguments targeting not only arguments, but also attacks at any level. A HLAF can be defined as follows:

Definition 4.6. *Let Ar be a set of arguments.* Level $(0, n)$ argumentation frames *are defined as follows:*

1. *A pair $(a, b) \in Ar \times Ar$ is called a level $(0, 0)$ attack.*
2. *If $c \in Ar$ and α is a level $(0, n)$ attack then (c, α) is a level $(0, n + 1)$ attack.*
3. *A level $(0, n)$ argumentation frame is the pair $\langle Ar, att \rangle$ where att contains only level $(0, m)$ attacks for $0 \leq m \leq n$.*

Note that, although the level of HLAFs is expressed in terms of pairs $(0, n)$ with possibly different values for n, the first component of the pair denoting the level is always 0 (the part of the level associated with the set of arguments). In particular, [62] proposed different kinds of approaches in order to define the semantics of HLAFs: the first option consists in translating a HLAF into a Dung's AF; the second alternative corresponds to the characterisation of labellings for HLAF, similarly to the labellings for AFs [6]; finally, in the third approach Gabbay proposed to translate a HLAF into a logic program. In the following, we will consider the first translation approach, which consists of obtaining a Dung's AF corresponding to a HLAF. Specifically, a HLAF $\langle Ar, att \rangle$ can be translated into an AF $\langle Ar^*, att^* \rangle$, where:

- $Ar^* = Ar \cup \{x_\beta, y_\beta \mid \beta = (a, \alpha) \in att\}$.
- $att^* = \{(a, x_\beta), (x_\beta, y_\beta), (y_\beta, \alpha) \mid \beta = (a, \alpha) \in att\}$.

The new arguments $x_{(a,\alpha)}$ and $y_{(a,\alpha)}$ associated with an attack from a to α respectively represent that the attack is 'live' or 'dead'; moreover, Gabbay argued that the translation of attacks as in the second bullet above is sufficient for attacks which are under attack. This translation is illustrated in Figure 7, where two attacks $\alpha = (a, b) \in att$ and $\beta = (c, \alpha) \in att$ are considered.

Then, given a set of extensions $E_1^+, E_2^+, \ldots, E_n^+$ of the associated AF, the corresponding extensions of the HLAF are $E_i^+ \cap Ar$, $(i = 1, \ldots, n)$.

In spite of proposing the translation described above, [62] argued that an attack $(a, b) \in att$ should be viewed as an independent unit, the attack of a on b, which can

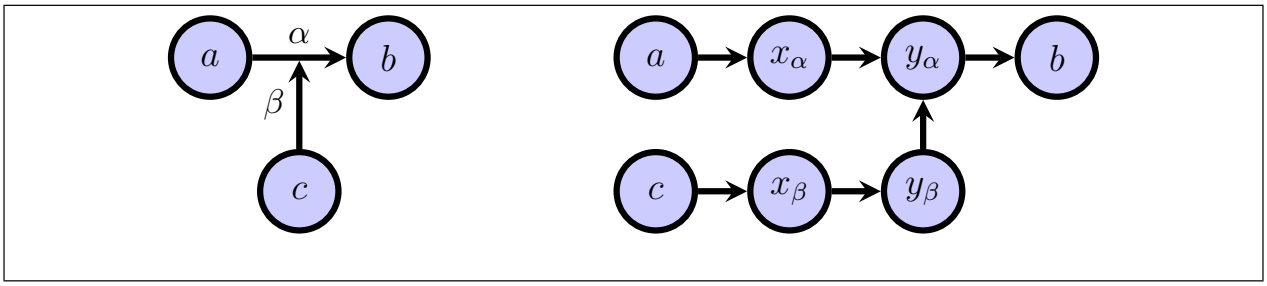


Figure 7: (LHS) HLAF with attacks $\alpha = (a, b)$ and $\beta = (c, \alpha)$; and (RHS) the translation into a series of AF attacks

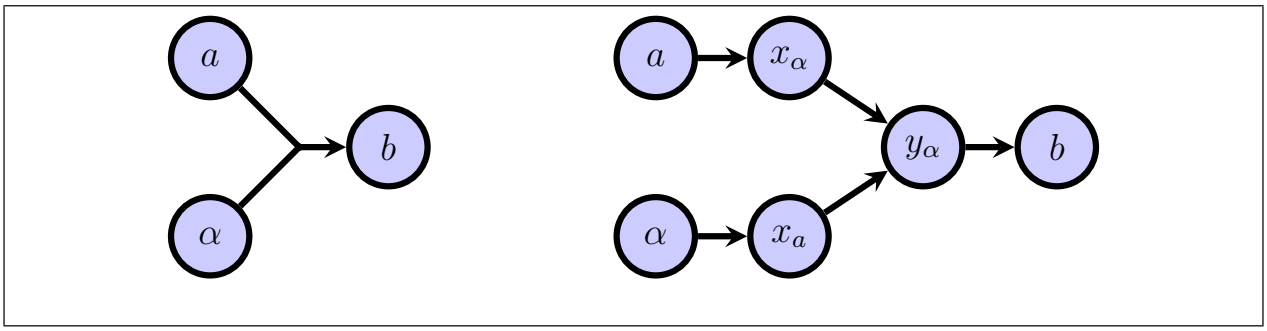


Figure 8: (LHS) Graphical representation of the joint attack by a and α on b, corresponding to an attack $\alpha = (a, b)$; and (RHS) its translation into a sequence of AF attacks.

be itself attacked. In particular, he stated that the preceding translation does not serve its purpose for modelling more general situations, such as attacks originated in other attacks (although the latter are not allowed in the frameworks of Definition 4.6). In that way, the author suggested that an attack $(a, b) \in att$ should be a unit kept 'live' unless attacked itself. Consequently, he proposed an alternative solution making use of *joint attacks*: an attack $\alpha = (a, b)$ is translated in a way such that the argument b is jointly attacked by two arguments a and α Then, both a and α must be 'live' in order for b to be 'dead'. The graphical representation of a joint attack by a and α on b, corresponding to an attack $\alpha = (a, b)$ is shown in Figure 8 on the left.

Given this notion of joint attack, [62] proposed a further translation of joint attacks into attacks in a Dung's AF. This translation has some similarities with the one introduced before for directly translating a HLAF into an AF, and is illustrated in Figure 8 on the right for the case of a joint attack by a and α on b.

Alternatively to the translation of joint attacks into attacks at the argument level in a Dung's AF, [62] introduced the *frames with joint attacks*:

Definition 4.7. *A* frame with joint attacks *has the form* $\langle Ar, att \rangle$*, where* Ar *is the set of arguments and* $att \subseteq Ar \times Ar \times Ar$ *is a ternary relation. We understand* $(x, y, z) \in att$ *as saying that the two arguments* x *and* y *are mounting a joint attack on* z*.*

The author remarked that single attacks can still appear in a frame with joint attacks; these would be attacks of the form $(x, x, y) \in att$. It is important to note that, following Definition 4.7, the frames with joint attacks are a particular case of the SETAFs, where the set of arguments originating an attack is restricted to a maximum of two elements. Consequently, the algorithms and reduction-based approaches for SETAFs discussed in Section 3.3 could also be applied to the frames with joint attacks.

Then, Gabbay introduced definitions analogous to those of [84], characterising the extensional semantics of these frameworks. Finally, he proposed a translation from HLAFs into frames with joint attacks, so that extensions of the former correspond to extensions of the latter.

Definition 4.8. *Let* $\langle Ar, att \rangle$ *be a HLAF. The corresponding frame with joint attacks* $\langle Ar', att' \rangle$ *is defined as follows:*

- $Ar' = Ar \cup att$
- $att' = \{(a, \alpha, \beta) \mid \alpha = (a, \beta) \in att\}$

Following this approach, for instance, the HLAF illustrated in Figure 9 on the top can be translated into a frame with joint attacks (or a SETAF) like the one depicted in Figure 9 at the bottom.

Next, we will discuss the possibility of using joint attacks for modelling attacks, including higher-order attacks, through Gabbay's Frames with Joint Attacks (a particular case of SETAFs) in frameworks such as the AFRA [7] or the ASAF[20] [68]. We will start by briefly recalling the definition of these frameworks, as proposed by their authors. As mentioned before, these frameworks are studied in another chapter of this book. Thus, for more details, we refer the interested reader to Chapter 1 of this handbook [29].

The Argumentation Framework with Recursive Attacks (AFRA) [7] generalises Dung's AF by incorporating a *recursive attack relation* where attacks are allowed to target other attacks as well as arguments, and the attacks can occur at any level.

Definition 4.9. *An* Argumentation Framework with Recursive Attacks (AFRA) *is a pair* $\langle Ar, att \rangle$ *where:*

[20] In the case of the ASAF, initially, without supports (where such an ASAF would be an AFRA). The means for modelling supports through joint attacks will be discussed later.

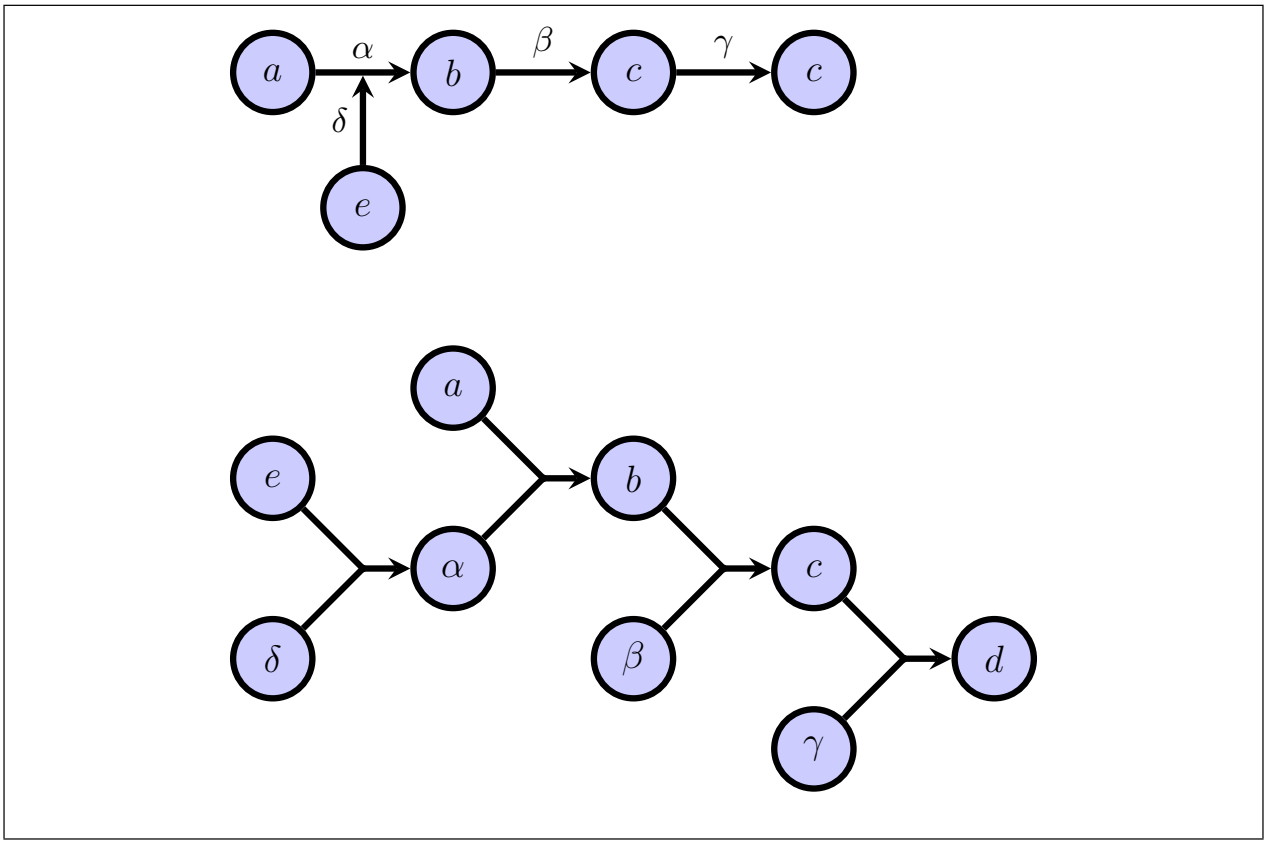


Figure 9: (Top) A HLAF with a higher-order attacks, where Greek letters denote the labels of the attacks; and (Bottom) its corresponding frame with joint attacks

- *Ar is a set of arguments;*
- *att is a set of attacks, namely pairs* (a, X) *such that* $a \in Ar$ *and* $(X \in Ar$ *or* $X \in att)$*.*

Given an attack $\alpha = (a, X) \in att$, a is said to be the source of α, denoted as $\mathbf{s}(\alpha) = a$, and X is the target of α, denoted as $\mathbf{t}(\alpha) = X$. Moreover, similarly to the notation used before for Gabbay's HLAF, [7] introduces an abbreviated notation for recursive attacks in the AFRA; for instance, an attack $(c, (a, b))$ can be expressed as (c, α), where $\alpha = (a, b)$.

Then, [7] establishes the different kinds of defeat that can occur between the elements of an AFRA. A key aspect of their formalisation is that they regard attacks (not their source arguments) as the subjects able to defeat arguments and other attacks. Then, an attack can be made ineffective (in other words, defeated) either by attacking the attack itself or by attacking its source. The notions of *direct defeat* and *indirect defeat* are introduced in [7] as follows:

Definition 4.10. *Let $\langle Ar, att\rangle$ be an AFRA, $\alpha \in att$ and $X \in Ar \cup att$. It is said that α defeats X, denoted $\alpha \rightarrow^R X$ if one of the following conditions holds:*

- $\mathbf{t}(\alpha) = X$ *(direct defeat); or*
- $X = \beta \in att$ *and* $\mathbf{t}(\alpha) = \mathbf{s}(\beta)$ *(indirect defeat).*

Then, based on this notion of defeat, the notions of conflict-freeness, acceptability, admissibility and extensions under different semantics are introduced following Dung's methodology. Consequently, the extensions of an AFRA will not only contain the accepted arguments under the corresponding semantics, but also the accepted attacks.

Example 4.11. *The arguments and attacks depicted at the top in Figure 9 correspond to the AFRA $\langle Ar, att\rangle$, where $Ar = \{a, b, c, d, e\}$ and $att = \{\alpha, \beta, \gamma, \delta\}$, with $\mathbf{s}(\alpha) = a$, $\mathbf{t}(\alpha) = b$, $\mathbf{s}(\beta) = b$, $\mathbf{t}(\beta) = c$, $\mathbf{s}(\gamma) = c$, $\mathbf{t}(\gamma) = d$, $\mathbf{s}(\delta) = e$, $\mathbf{t}(\delta) = \alpha$.*

Here, the direct defeats are: $\alpha \rightarrow^R b$, $\beta \rightarrow^R c$, $\gamma \rightarrow^R d$ and $\delta \rightarrow^R \alpha$. On the other hand, the indirect defeats are: $\alpha \rightarrow^R \beta$ and $\beta \rightarrow^R \gamma$. Consequently, β reinstates d, α reinstates c and γ, and δ reinstates α and b. As a result, for instance, the AFRA has only one complete extension (which is also its grounded and only preferred and stable extension), namely $\{a, e, \delta, b, \beta, d\}$. In contrast, if we apply the SETAF semantics on the framework depicted at the top on Figure 9[21]*, we have that the only complete extension is $\{a, e, \delta, b, \beta, \gamma, d\}$.*

The difference in the result obtained by applying the AFRA semantics, compared to the one obtained by translating the AFRA into a SETAF and then applying the SETAF semantics, has to do with the fact that the translation proposed in [62] does not take into account the indirect defeats. In particular, in the above example, the indirect defeat by β on γ is not captured, leaving γ as an accepted attack[22]. This suggests that we need to establish a different translation of AFRAs into SETAFs, in order to account for the effect of indirect defeats. An alternative translation of an AFRA into a SETAF could be:

Definition 4.12. *Let $\langle Ar, att\rangle$ be an AFRA. The corresponding SETAF $\langle Ar, \triangleright\rangle$ is defined as follows:*

$$\begin{aligned} Ar = \;& Ar \cup att \\ \triangleright = \;& \{(\{a, \alpha\}, X) \mid \alpha = (a, X) \in att\} \cup \\ & \{(\{a, \alpha\}, \alpha') \mid \alpha = (a, X) \in att,\ \alpha' \in att,\ \mathbf{s}(\alpha') = X\} \end{aligned}$$

[21] As stated before, the frames with joint attacks are a particular case of SETAFs.

[22] The indirect defeat by α on β is not captured either; however, since α is not accepted (because it is directly defeated by δ) it does not affect the outcome.

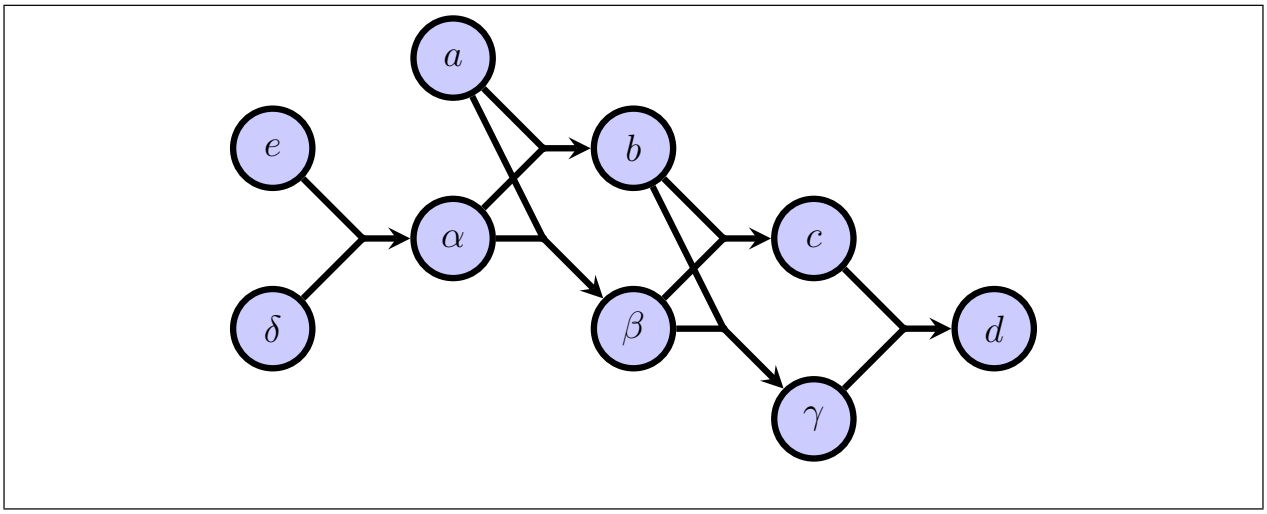


Figure 10: SETAF corresponding to the AFRA from Example 4.11

The AFRA from Example 4.11, corresponding to the framework depicted at the top of Figure 9, can be translated following Definition 4.12 to obtain the SETAF depicted in Figure 10. Then, applying the SETAF semantics on that framework, the only complete extension coincides with the one obtained with the AFRA semantics in Example 4.11.

Let us now consider the formalization of the Attack-Support Argumentation Framework (ASAF) [68]. Briefly, the ASAF extends Dung's AF by incorporating bipolar higher-order interactions. In that way, the ASAF allows for the representation and reasoning with attack and support relations not only between arguments, but also targeting the attack and support relations themselves. In particular, the support relation of the ASAF is interpreted as necessity [88]. That is, the necessary support relation in the ASAF imposes the following acceptability constraints on the elements it relates: if a supports b, then the acceptance of b implies the acceptance of a; equivalently, the non-acceptance of a implies the non-acceptance of b. Note that the support relation in the ASAF is set to be binary, differently from the necessary support relation of the GAFN introduced in Section 4.1. Some of the following definitions are taken from [2], where the background for the ASAF was succinctly introduced.

Definition 4.13. *An* Attack-Support Argumentation Framework (ASAF) *is a tuple* $\langle Ar, att, sup\rangle$ *where* Ar *is a set of arguments,* $att \subseteq W$ *is the attack relation, and* $sup \subseteq W$ *is the support relation, with* W *being the set iteratively defined as follows:* $W = Ar \times Ar$ *(basic step) and* $W = Ar \times W$ *(iterative step). It is assumed that sup is acyclic and* $att \cap sup = \emptyset$.

Similarly to the case of the AFRA, an attack $(a, b) \in att$ will be denoted as $\alpha_1 = (a, b)$; analogously, a support $(b, c) \in sup$ will be denoted as $\beta_1 = (b, c)$. Then, for instance, an attack from d to α_1 will be denoted as $\alpha_2 = (d, \alpha_1)$. In general, given an attack $\alpha = (a, X) \in att$, a is called the source of α, denoted $\mathbf{s}(\alpha) = a$, and X is called the target of α, denoted $\mathbf{t}(\alpha) = X$. Analogously, given a support $\beta = (b, Y) \in sup$, b is called the source of β, denoted $\mathbf{s}(\beta) = b$, and Y is called the target of β, denoted $\mathbf{t}(\beta) = Y$.

Like in the AFRA, different kinds of defeat that can occur between the elements of an ASAF. Specifically, they correspond to the two kinds of defeat identified for the AFRA, plus two additional kinds of defeat that arise from the coexistence of the attack and support relations.

Definition 4.14. *Let $\Delta = \langle Ar, att, sup \rangle$ be an ASAF, $\alpha \in att$, $X \in (Ar \cup att \cup sup)$ and $\mathbf{S} \subseteq sup$. We say that α defeats X (given $\mathbf{S}$), denoted α def X given $\mathbf{S}$ (or simply α def X whenever $\mathbf{S} = \emptyset$) iff one of the following conditions holds:*

- *there exists a (possibly empty) support path from $\mathbf{t}(\alpha)$ to X, whose corresponding set of supports is $\mathbf{S}$; or*
- *$X \in att$ and there exists a (possibly empty) support path from $\mathbf{t}(\alpha)$ to $\mathbf{s}(X)$, whose corresponding set of supports is $\mathbf{S}$.*

To illustrate these notions, let us consider the following example. Similarly to Dung's AF or the AFRA, an ASAF can be graphically represented using a graph-like notation where two kinds of edges are considered: $\rightarrow$ for the attack relation and $\Rightarrow$ for the support relation. In addition, attacks and supports are labelled with greek letters, following the convention that attacks are labelled with α (possibly with subscripts) and supports are labelled with β (again, possibly with subscripts).

Example 4.15. *Consider the ASAF $\langle Ar, att, sup \rangle$, where $Ar = \{a, b, c, d, e, f, g, h\}$, $att = \{\alpha_1, \alpha_2, \alpha_3, \alpha_4\}$ and $sup = \{\beta_1, \beta_2, \beta_3\}$, with $\alpha_1 = (a, b)$, $\alpha_2 = (c, e)$, $\alpha_3 = (g, f)$, $\alpha_4 = (h, \alpha_3)$, $\beta_1 = (b, c)$, $\beta_2 = (c, d)$ and $\beta_3 = (f, \beta_1)$. This framework is depicted in Figure 11, and the following defeats occur: α_1 def b, α_2 def e, α_3 def f, α_4 def α_3, α_1 def c given $\{\beta_1\}$, α_1 def α_2 given $\{\beta_1\}$, α_1 def d given $\{\beta_1, \beta_2\}$, α_3 def β_1 given $\{\beta_3\}$.*

The semantics of the ASAF are also defined following Dung's methodology, accounting for the notions of conflict-freeness, acceptability and admissibility, to later characterise the complete, preferred, stable and grounded semantics of the framework. It should be noted that, since the defeats in the ASAF may involve a set of supports, these notions cannot be directly defined by considering the definitions for AFs (see Section 3.1 and Definition 4.14).

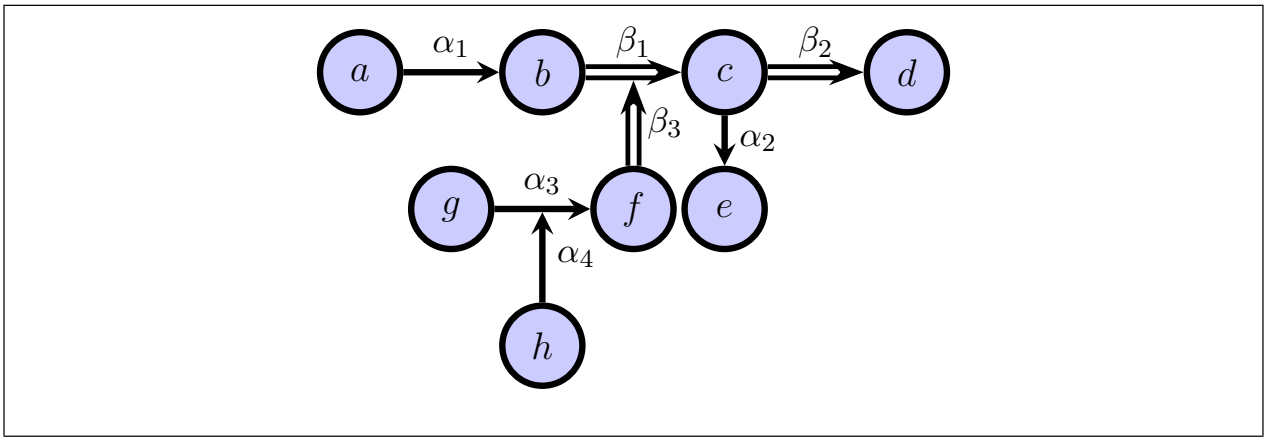


Figure 11: ASAF from Example 4.15

Definition 4.16. *Let $\Delta = \langle Ar, att, sup\rangle$ be an ASAF and $\mathbf{S} \subseteq (Ar \cup att \cup sup)$.*

- $\mathbf{S}$ *is* conflict-free *iff $\nexists \alpha, X \in \mathbf{S}$, $\nexists \mathbf{S}' \subseteq (\mathbf{S} \cap sup)$ such that α* def *X* given *$\mathbf{S}'$.*
- *$X \in (Ar \cup att \cup sup)$ is* acceptable w.r.t. *$\mathbf{S}$ iff $\forall \alpha \in att$, $\forall \mathbf{T} \subseteq sup$ such that α* def *X* given *$\mathbf{T}$: $\exists Y \in (\{\alpha\} \cup \mathbf{T})$, $\exists \alpha' \in \mathbf{S}$, $\exists \mathbf{S}' \subseteq (\mathbf{S} \cap sup)$ such that α'* def *Y* given *$\mathbf{S}'$.*
- $\mathbf{S}$ *is* admissible *iff it is conflict-free and for all $X \in \mathbf{S}$, X is acceptable w.r.t. $\mathbf{S}$.*

Definition 4.17. *Let $\Delta = \langle Ar, att, sup\rangle$ be an ASAF and $\mathbf{S} \subseteq (Ar \cup att \cup sup)$.*

- $\mathbf{S}$ *is a* complete extension *of Δ iff it is an admissible set and $\forall X \in (Ar \cup att \cup sup)$, if X is acceptable w.r.t. $\mathbf{S}$, then $X \in \mathbf{S}$.*
- $\mathbf{S}$ *is a* preferred extension *of Δ iff it is a maximal (w.r.t. $\subseteq$) complete extension of Δ.*
- $\mathbf{S}$ *is a* stable extension *of Δ iff it is a complete extension of Δ and $\forall X \in (Ar \cup att \cup sup) \backslash \mathbf{S}$, $\exists\, \alpha \in \mathbf{S}$, $\exists\, \mathbf{S}' \subseteq (\mathbf{S} \cap sup)$ such that α* def *X* given *$\mathbf{S}'$.*
- $\mathbf{S}$ *is the* grounded extension *of Δ iff it is the smallest (w.r.t. $\subseteq$) complete extension of Δ.*

The ASAF from Example 4.15 has only one complete extension, which is also the grounded extension and the only preferred and stable extension of the framework: $\{a, e, f, g, h, \alpha_1, \alpha_4, \beta_1, \beta_2, \beta_3\}$. In particular, it can be noted that whereas α_3 def β_1 given $\{\beta_3\}$, the support β_1 is reinstated by α_4, since α_4 def α_3. Then, the

defeats from α_1 on c and α_2 given $\{\beta_1\}$ are also reinstated, as well as the defeat from α_1 on d given $\{\beta_1, \beta_2\}$.

As shown in [68], an ASAF without support is an AFRA. So, when applying the AFRA semantics on ASAFs without supports we obtain the same outcome as the one obtained under the ASAF semantics. Consequently, the translation of an AFRA into a SETAF could also be applied to the ASAF; nevertheless, some adjustments need to be made in order to account for the defeats involving a set of supports. A possible translation of an ASAF into a SETAF is given below.

Definition 4.18. *Let $\langle Ar, att, sup\rangle$ be an ASAF. The corresponding SETAF $\langle Ar, \triangleright\rangle$ is defined as follows:*

$$\begin{aligned}
Ar = \;& Ar \cup att \cup sup \cup \{\beta^* \mid \beta \in sup\} \\
\triangleright = \;& \{(\{a, \alpha\}, X) \mid \alpha = (a, X) \in att\} \cup \\
& \{(\{a, \alpha\}, \alpha') \mid \alpha = (a, X) \in att,\ \alpha' \in att,\ \mathbf{s}(\alpha') = X\} \cup \\
& \{(\{a, \alpha\}, X^*) \mid \alpha = (a, X) \in att,\ X \in sup\} \cup \\
& \{(\{a, \beta\}, \beta^*), (\{\beta^*\}, X) \mid \beta = (a, X) \in sup\} \cup \\
& \{(\{\beta^*\}, X^*) \mid \beta \in sup,\ X \in sup, \mathbf{t}(\beta) = X\} \cup \\
& \{(\{\beta^*\}, \alpha) \mid \beta \in sup,\ \alpha \in att, \mathbf{t}(\beta) = \mathbf{s}(\alpha)\}
\end{aligned}$$

The ASAF from Example 4.15, corresponding to the framework depicted in Figure 11, can be translated following Definition 4.18 to obtain the SETAF depicted in Figure 12. Then, applying the SETAF semantics on that framework, the only complete extension coincides with the one obtained with the ASAF semantics, namely $\{a, e, f, g, h, \alpha_1, \alpha_4, \beta_1, \beta_2, \beta_3\}$.

Finally, we will briefly discuss the possibility of using joint attacks to model the generalised necessity relation proposed in [88] adopted in frameworks such as the RAFN (see Section 4.2).

Let us recall the example introduced in Section 2, represented using the SETAF from Figure 2. There, we can think of the argument NP as providing a context under which $A18$ attacks M; that is, a person aged under 18 is not allowed to marry whenever parent permission is not provided. So, we could think of this situation as corresponding to the existence of an attack $\alpha_1 = (A18, M)$ and a generalised necessary support $\beta_1 = (\{NP\}, \alpha_1)$. Similarly, given the restriction to drink alcohol, arguments NA and NM can be considered as providing alternative contexts under which $A18$ attacks Alc. Therefore, we could think of representing this situation through an attack $\alpha_2 = (A18, Alc)$ and a generalised necessary support $\beta_2 = (\{NA, NM\}, \alpha_2)$. This is because, in this situation, it suffices to have either NA or NM accepted in order to be able to accept α_2 (i.e., in order for the attack from $A18$ to Alc to hold).

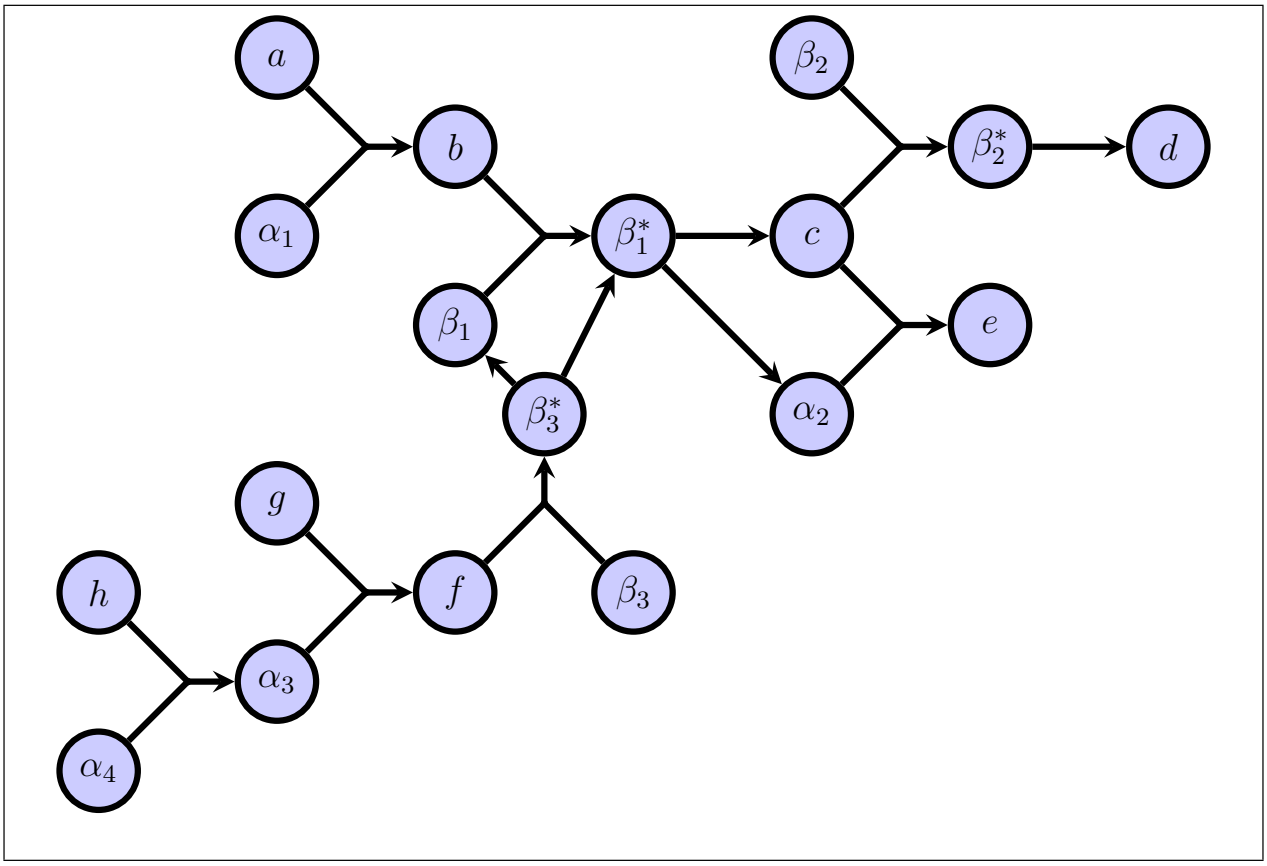


Figure 12: SETAF corresponding to the ASAF from Example 4.15

The preceding example suggests that joint attacks (as those in the SETAF from Figure 2) may be suitable for modelling the generalised necessary support relation in the case of higher-order supports targeting an attack. However, for instance, if there exists another interaction (say, an attack α_3) targeting β_1, we should be able to model on the SETAF the fact that if α_3 is accepted then β_1 no longer holds and, consequently, that NP does not provide a context under which the attack α_1 from $A18$ to M holds. Nevertheless, if the support β_1 from NP to α_1 is modeled by a joint attack from NP and $A18$ as in Figure 2, we cannot model the attack from α_3 towards β_1 in the SETAF, since β_1 is not made explicit in this representation.

5 Accrual

A parallel line of research in computational argumentation studies the accrual of arguments, i.e., how arguments supporting or refuting the same claim can be combined. The main differences between accrual and joint attacks (at least under the

type of joint attacks used in SETAFs) is that, in joint attacks, the strength of an argument or a combination of arguments is not considered when evaluating the effectiveness of attacks, and each argument participating in a joint attack is an essential element of it (in other words if an argument is missing then the attack is ineffective), while in accrual the strength of each argument is taken into account, and adding an argument to an accrual makes the accrual stronger, or more generally it changes its strength and the effectiveness of its attacks (or supports).

A seminal study on the accrual of arguments [96] set out three principles for accrual:

1. An accrual is sometimes weaker than its accruing elements. This is due to the possibility that the accruing reasons are not independent.

2. An accrual makes its elements inapplicable. More generally, any 'larger' accrual that applies makes all its 'lesser' versions inapplicable. This is because an accrual is meant to consider all available information, while the individual arguments it consists of take only part of the information into account.

3. Flawed reasons or arguments may not accrue. Any treatment of accrual should capture that when an individual reason or argument turns out to be flawed, it does not take part in the accrual.

Prakken also described two general ways to formalise accrual: (*a*) the *knowledge representation* (or else *KR*) approach, which requires formulating a separate rule for each possible combination of the accruing reasons; (*b*) the *inference* approach, where the accrual is part of the inference process, i.e., after all individual reasons have been constructed, those that attack or support the same claim are somehow aggregated and some mechanism is then used to resolve any conflicts between the conflicting sets of reasons. He also proposed a formalisation of accrual using the inference approach, according to which the conclusion of each individual defeasible inference step is labelled with the premises of the applied defeasible inference rule:

$$\phi, \phi \Rightarrow \psi \ \mid\sim \ \psi^{\{\phi, \phi \Rightarrow \psi\}}$$

and a new defeasible inference rule is introduced that takes any set of labelled versions of a certain formula and produces the unlabelled version:

$$\phi^{l_1}, \cdots, \phi^{l_n} \ \mid\sim \ \phi$$

The attack relationships among arguments are adjusted as follows: rebuttal requires that the two arguments support opposite conclusions that are labelled in the same

way, while undercut requires that the attacking arguments have unlabelled conclusions. Finally, the following rules ensure that when a set of reasons accrues, any subset of it is inapplicable:

$$\phi^{l_1}, \cdots, \phi^{l_n} \ \not\sim \ \neg\lceil\phi^{l_1}, \cdots, \phi^{l_{n-1}}\rceil$$

The proposed formalisation satisfies all principles of accrual, but has a computational drawback: it requires considering all possible accruals for every conclusion, which may lead to an exponential increase in the number of arguments.

The idea of combining arguments for and against a claim, albeit under the name "aggregation" rather than "accrual" was studied by argumentation researchers before [96]. One line of work, that of Fox and colleagues, goes back at least as far as [90], where the idea of symbolically weighing evidence is formalised in what recognisably is a structured argumentation framework, and arguably as far back as [61] where the idea was first applied. The formal development of that work came to a conclusion with [72] and [59]. The former paper describes a model that links the simple form of accrual from [90], which effectively just looks at the numbers or arguments for and against a claim[23], with forms of accrual which connect to probabilistic models. The latter uses the same model to develop a hierarchy of notions of acceptability, coming close to Dung's work at about the same time that work was first published [39].

CumulA [103; 104] was another structured argumentation model that dealt with accrual. Additionally to the notion of *compound defeaters*, which we discussed in Section 3.4, it also includes the notions of *coordination* and *narrowings* of arguments. Different arguments supporting the same conclusion can be combined in a *coordinated* argument, while the *narrowing* of a coordinated argument a is an argument b supporting the same conclusion as a but containing a subset of the arguments combined in a (or narrowings of them). CumulA deals with accrual using compound defeaters and the following acceptability condition for arguments: if the narrowing of an argument a is `in` (meaning that the argument is accepted), then a should be `in` too. As shown in [96], CumulA satisfies all three principles of accrual but the second one (i.e. that an accrual makes its elements inapplicable) is satisfied in a way that is too strong. Because of the acceptability condition described above, which implies that if an accrual is `out` then all its narrowings are `out`, it cannot capture a situation where an accrual is defeated because of subargument defeat so that some of its narrowings can be undefeated.

More recently, [16] developed another account of accrual, based on their logic-based approach to argumentation, though again they do not describe it as such. In

[23] Before dismissing such a simple model, consider how effective such simple models can be [38].

[16], lines of discussion about a particular claim — the argument for it, the arguments against it, the arguments against those arguments, and so on — are brought together into an argument tree. Then, all argument trees for or against a claim are assembled into an argument structure. An argument structure thus gathers everything that is relevant to whether or not a claim should be accepted. This, of course, is not much different to what one would get from assembling all of the arguments in a structured framework like ASPIC+ or DeLP that bear on a specific formula into some super-structure. However, whereas most structured frameworks summarise this higher level structure in a notion of acceptability, [16] defines a "categoriser" which maps a structure to a number, and this can be thought of as the accrued value of the set of arguments in the structure.

[78] proposed an approach for formalising the accrual of arguments in Defeasible Logic Programming using the notion of *a-structure*, a special kind of argument which subsumes different chains of reasoning that provide support for the same conclusion, and *partial attacks* among a-structures, where the attacking a-structure generally affects only the narrowing of the attacked a-structure containing exactly the arguments affected by the conflict. A binary preference relation on a-structures is used to determine the relevant strength of conflicting a-structures and whether an attack succeeds (in which case it constitutes a *defeat*). To deal with combined attacks (situations where two or more a-structures simultaneously attack the same a-structure), they define a process, called *bottom-up sequential degradation*, according to which the defeats are applied in sequence with the "deeper" ones applied first. The described framework satisfies all three principles of accrual. Its main difference with the formalisation proposed in [96] is that when analysing a theory to determine the accepted (undefeated) a-structures, it only considers maximal accruals (a-structures) and not all possible accruals for a conclusion.

[67] proposed the use of argument weighing functions as a way to model different types of argument schemes, including some types of argument accrual, in Carneades, a structured argumentation framework. In this framework, an argument is defined as a tuple (s, P, c, u) where s is the scheme that the argument instantiates; P, the premises of the argument, is a finite subset of the underlying logical language $\mathcal{L}$; and c, its conclusion, and u, its undercutter, are elements of $\mathcal{L}$. Its semantics is defined in terms of a labelling, which assigns a value from $\{\texttt{in}, \texttt{out}, \texttt{undec}\}$ to each element of $\mathcal{L}$ and a weighing function, which assigns a value from $[0, 1]$ to each argument and 0 to all arguments such that their undercutter is `in`. Gordon also provided several examples of weighing functions, some of which are appropriate for modelling different types of accrual. To simulate *convergent arguments*, i.e., arguments that at least one of its premises must be `in` to support their conclusion, he defined a weighing function that assigns 1 to an argument if at least one of its premises is `in`

and its undercutter is not `in`, and 0 otherwise. A weighing function that simulates *cumulative arguments*, i.e., arguments whose strength increases with the number of their acceptable premises, assigns the percentage of the premises of the argument that are `in` to every argument whose undercutter is not `in`. Cumulative arguments are a special type of accrual that does not satisfy Prakken's second principle, since cumulation can only increase the strength of an argument. Another weighing function that simulates accrual takes into account all *factors* (statements) that need to be considered when evaluating the arguments for a certain issue. It does so by assigning to each argument the proportion of its factors that are premises of the argument and are labelled `in`. One limitation of this framework with respect to accrual is that although it handles various forms of accrual at the level of statements, e.g., premises or factors of an argument, it does not provide a way to handle accrual of multiple arguments.

[97] proposed a formalisation of accrual for ASPIC+, a structured argumentation framework where arguments are tree-like structures constructed from a knowledge base, which is a subset of an underlying logical language $\mathcal{L}$, and a set of inference, strict or defeasible, rules. In this framework, there are two ways to attack an argument a: either at the top inference rule r of a (*undercut*) or at the conclusion of r (*rebuttal*)[24] - in both cases r must be defeasible, otherwise a cannot be attacked. [97] extended ASPIC+ with the notion of *accrual sets*, which are defined relative to a labelling of the set of arguments S. An accrual set for a literal $\phi \in \mathcal{L}$, denoted as $s_l(\phi)$, is the set of arguments with conclusion ϕ satisfying the following two conditions: (i) for any argument in $s_l(\phi)$ no immediate subargument of a is `out` and no undercutter of a is `in`; (ii) any argument with conclusion ϕ whose undercutters are `out` and its immediate subarguments are `in` must be in $s_l(\phi)$. The extended framework also includes a preference relation $\leq$ on the power set of S, such that any set of arguments containing a strict argument is at least as preferred as every other subset of S. It also includes a new defeat relation on arguments, called *l-defeat*, which takes into account accruals: an argument a l-defeats an argument b iff a undercuts b; or a rebuts b, and for some accrual sets for the conclusions of a, $s_l(Conc(a))$, and b, $s_l(Conc(b))$, it holds that $s_l(Conc(a)) \not\leq s_l(Conc(b))$. A *characteristic function* F is used to compute the labelling of a framework, which satisfies the following conditions: an argument a is `in` iff all arguments that l-defeat a are `out` and all immediate subarguments of a are `in`; a is `out` iff it is defeated by an argument that is labelled `in` or one of its immediate subarguments are `out`.

[24]Note that these definitions are different from the standard ASPIC+ where arguments can also be attacked on their subarguments. As explained in [97], in the version of ASPIC+ considered in this paper, arguments are constructed recursively and the recursion takes care of subargument attacks.

The proposed framework satisfies all principles of accrual and preserves some of the properties of Dung's AFs, such as the existence of complete and preferred labellings and the relations between grounded, complete, stable and preferred semantics.

In the field of abstract argumentation, the most relevant approaches are the frameworks with graded semantics (e.g., see [11] for an overview and a study of their properties) or ranking semantics and social argumentation frameworks (e.g., see [74; 12; 92]). Such frameworks provide methods for assessing the strength of an argument based on the aggregate strength of its attackers and the aggregate strength of its supporters (and in some cases the initial valuation of the argument), capturing the main idea of accrual. Some of their general properties are: (*i*) the larger the set of the attackers on an argument, the lower the strength of the argument under attack; (*ii*) the larger the set of supporters or defenders of an argument, the higher the strength of the argument they support or defend; and (*iii*) an argument with 0 strength does not have an effect on the strength of the arguments it attacks or supports. The last property satisfies the third principle of accrual (i.e., that flawed arguments do not accrue), while by considering the aggregate strength of the attackers or supporters of an argument, they essentially satisfy the second principle, i.e., that an accrual makes its elements inapplicable. Properties (*i*) and (*ii*), however, violate the second principle, since they imply that an accrual is always stronger than the individual accrued arguments.

6 Proposals for future work on joint attacks

In this section we highlight some emerging topics for future research on joint attacks and accrual.

There are several interesting directions for further research concerning semantics of SETAFs. Standard semantics of AFs have been generalised to SETAFs and their basic properties and relations are settled. However, several prominent semantics have not yet been generalised and analysed on SETAFs, e.g., cf2 [8], strong admissibility [9; 26] and weak admissibility [14]. Recently, a first approach to transfer also ranking-based semantics to SETAFs has been undertaken [108]. Properties of AF semantics have been studied in versatile aspects [102; 13] beyond the existing analysis for SETAFs. For instance, generalising the principle-based approach for analysing and comparing semantics to SETAFs would be valuable for the selection of the right argumentation semantics, and understanding the different notions of equivalence also on SETAFs is fundamental for using SETAFs in dynamic settings. Concerning the latter, a first investigation of strong equivalence notions for SETAFs has been done in [54].

As another research direction one could consider enhancing the expressiveness of SETAF by extending its basic model with features similar to the ones used in extensions of the AF model, such as the introduction of a joint support relation, weights on (joint) attacks, values promoted by (sets of) arguments, or a preference relation among (sets of) arguments. This would allow associating SETAFs with the corresponding AF extensions, i.e., frameworks for bipolar argumentation [3; 32], graded [71] or weighted argumentation [46], value-based [15], or preference-based argumentation [4] respectively. A related but somehow orthogonal research direction is the investigation of the relations of SETAFs and other extensions of AFs concerning their expressiveness. Existing investigations in that direction are the embedding of SETAFs in ADFs [77] and translations between SETAFs and claim-augmented AFs [56].

The translations from SETAF to AFs discussed in Section 3.2 either had the weakness that they might increase the size exponentially or only supported a selection of the semantics. For future research one could investigate alternative translation schemes in order to avoid this pitfall. Ideally, we would like to have a transformation that applies for all semantics and causes a polynomial increase in the size of the framework (in the sense of [20]), while at the same time resulting in elegant correspondences for all the semantics (unlike [60], where this is true only for some of the semantics). Recall, that [94] provide a translation that satisfies the latter two properties but only for a selection of the semantics, i.e., the semantics based on complete extensions, with the exception of semi-stable semantics. Also notice that a polynomial increase in the size of the framework can still result in an exponential increase in the number of arguments. Thus, another open question is whether such a potential exponential increase in the number of arguments can be avoided.

On the computational side there are several open challenges. From the theoretical perspective one would be interested in identifying classes of instances that provide milder complexity than general SETAFs. One approach that has been extensively studied for AFs are the so called tractable fragments [42], i.e., special graph classes like acyclic or bipartite, that allow for efficient reasoning procedures. A more general approach are graph parameters and techniques for parametrised complexity theory that allow for algorithms which are only exponential w.r.t. a graph parameter but polynomial in the size of the AF [51; 52]. From a more practical view one would be interested in efficient labelling-based algorithms [86; 36] for SETAFs as well as systems that extend methods that have been successfully applied for AFs [35]. An important step to boost the development of such systems would be to establish standard formats to share SETAF instances and standard benchmark sets.

Regarding the application of joint attacks, the ideas discussed in Section 4.3 could

be further explored. As shown in Section 4.3, the translations from an AFRA [7] or an ASAF [68] into a SETAF yield the same outcomes as those obtained directly by applying the AFRA or ASAF semantics, respectively. However, this was only shown for the examples illustrated in that section. A formal analysis of this correspondence for the general case of an arbitrary ASAF or AFRA is left for future work. In addition, the brief discussion at the end of Section 4.3 can also be the subject of future work, also considering the translations discussed in Section 3.2. Specifically, studying the possibility of using the SETAF for modelling the generalised support relation of frameworks like the RAFN [30; 31], accounting for all cases: first-order supports, higher-order supports targeting attacks and supports, and higher-order supports which can be themselves attacked.

Finally, the potential of collective attacks in structured models of argumentation is rather unexplored. Consider an instantiation scheme like ASPIC+ [97], or instantiations for logic programs [28] or assumption-based argumentation [1] that construct AFs from a knowledge base. Using SETAFs instead of AFs as target formalism can significantly reduce the number of arguments and, in certain cases one can even ensure that each statement has a unique argument supporting that statement [56]. A first investigation in that direction is [107; 109] where SETAFs are instantiated from Datalog knowledge bases. There is also scope for relating this kind of approach to work on accrual and the other models that collect related arguments such as the argument trees of [16] and the coalitions of [33].

The accrual of arguments is a less studied problem compared to joint attacks. As we discussed in Section 5 most existing approaches are focused on structured argumentation and only few of them satisfy all three principles proposed in [96]. There are a lot of interesting future research directions in this area such as the systematic comparison and evaluation of the frameworks that support accrual and the development of methods for handling accrual in abstract argumentation. The latter could rely on the recently proposed graded semantics for abstract argumentation or may require the development of a new abstract argumentation framework that explicitly models accrual. Another interesting direction, which could also lead to a solution for this problem, is to study the relation between current approaches for accrual and collective attacks and the mapping between the frameworks that deal with these two different problems.

7 Conclusions

In this chapter we have studied different formalisms that account for joint attacks (or more generally, collective attacks) in abstract argumentation. Also, we discussed the

consideration of joint attacks in the literature of structured argumentation as well as the application of joint attacks (and joint supports) in other frameworks such as bipolar argumentation frameworks or argumentation frameworks with higher-order interactions. We also touched upon works on argument accrual which, although not strictly related to the existing models of joint attacks, can be considered as a related topic. In particular, the SETAF [84] framework along with its computational complexity, algorithms and applications, was the main subject of study in this chapter.

In Section 3.1 the basic definitions of the framework were provided, followed by the presentation of extension-based semantics of the SETAF and the relationships between them, as well as the introduction of labelling-based semantics for the framework. Then, in Section 3.2 the expressive power of the SETAF was compared against that of Dung's AF [40]. For this purpose, the results and analyses reported in [48; 60; 94; 20] were accounted for. On the one hand, the characterisation of the expressive power using signatures was discussed, to then consider exponential and compact translations of SETAFs into AFs, and later discuss an indirect translation path consisting of a translation of a SETAF into an ADF [22] and a translation of the latter into an AF. The main conclusion here is that, although SETAFs could be represented by means of Dung's AFs, the SETAF indeed increases the expressive power of the AF. Moreover, as discussed in Section 3.3, the translations from a SETAF into an AF lead to an exponential blow-up in the arguments, making them not well-suited for computational matters.

In Section 3.3 different computational problems for SETAFs were characterised, following the definition of function problems and decision problems for Dung's AFs (cf. [36; 47; 35]). In particular, the decision problems include determining the credulous or skeptical acceptance of an argument under a given semantics, the verification of an extension, or determining the existence of a (non-empty) extension. Then, the computational complexity of these decision problems is addressed, and the results are linked to the existing results for decision problems in a Dung's AF. The conclusion here is that the complexity of decision problems for SETAFs is the same as the complexity of the corresponding problems for Dung's AFs. Notwithstanding this, we should note that although in both cases the complexity is stated w.r.t. the size of the input framework, the size is interpreted differently for AFs and SETAFs. On the one hand, the size of an AF is often interpreted in terms of the number of arguments of the input framework. On the other hand, since the number of attacks in a SETAF may be exponentially larger than the number of arguments in the framework (due to the existence of attacks by sets of arguments), the size of a SETAF should be interpreted in terms of the number of arguments plus the number of attacks.

Also, Section 3.3 briefly discussed the ideas behind labelling-based algorithms for

SETAFs, illustrating the algorithm for labelling enumeration under the preferred semantics. Moreover, as mentioned before, different reduction-based approaches for computing the extensions of a SETAF were presented. The former consists of encoding the SETAF and its semantics in Answer-set programming [79; 85], whereas the latter rely on translations of a SETAF into a Dung's AF or an ADF. While the drawbacks of the translations into AFs were pointed out above, we should note that the translation into an ADF offers the possibility of using existing systems for ADFs [76] without incurring significant overheads.

Section 3.4 recalled alternative models of abstract and structured argumentation which account for attacks involving sets of arguments. While in SETAF, a set of arguments can only be the source of an attack, in other models sets of arguments are only considered as a potential target of an attack (e.g. see [63]); or as both potential sources or targets of an attack (e.g. see [18; 33; 103]). The different models also differ in how the arguments within a set are treated. For example, while in the framework of [33], all arguments in a coalition are treated in the same way, i.e. they are all either accepted or rejected, in other frameworks, such as the ones proposed in [18; 63], a successful attack on a set of arguments has as a result that at least one of the arguments in the attacked set is rejected. Although the different approaches have different aims, an interesting problem is to study the extent to which they can be mapped to each other and whether there is a more general model that captures their different features.

Section 4 addresses the application of models for joint attacks in the context of Bipolar Argumentation Frameworks (BAFs) and argumentation frameworks with higher-order interactions such as those addressed in Chapter 1 [29]. First, BAFs that make use of joint attacks, joint supports, or both are recalled, highlighting the constraints they impose on the attack and support relations, as well as the adopted interpretations for the notion of support. Then, generalisations of these BAFs are presented, which incorporate higher-order interactions in order to allow for attacks and supports targeting other attacks or supports. Later, an analysis of the possibility of using joint attacks to model higher-order interactions is performed.

On the one hand, Section 4 considered the work by Gabbay on Higher-Level Argumentation Frames [62], as well as the proposed translations of HLAFs into Frames with Joint Attacks (a particular case of SETAF). Then, Gabbay's ideas are taken in the context of the AFRA [7] and the ASAF [68], two abstract argumentation frameworks allowing for binary higher-order interactions. Our findings are that, when applying the translation proposed by Gabbay to obtain the SETAF associated with an AFRA or an ASAF without supports, and then applying the SETAF semantics, the corresponding extensions might not be as expected (since the translation does not account for the existence of indirect defeats). Then, translations for obtaining

a SETAF corresponding to an AFRA or an ASAF are proposed, and illustrated through examples; their formalisations for the general case of an AFRA or an ASAF are left for future research. Moreover, the possibility of using the SETAF to model generalised necessary supports is briefly analysed in Section 4, leaving an in-depth discussion for future work.

In Section 5 different works addressing the topic of argument accrual were discussed, both at the abstract and structured levels of argumentation. As discussed there, the main difference between the approaches studying argument accrual and those accounting for joint attacks (e.g., as in SETAFs) is that the strength of the arguments combined to originate a joint attack is not accounted for when evaluating the effectiveness of attacks; notwithstanding this, each argument originating a joint attack is an essential element in the sense that the attack becomes ineffective whenever one of its source arguments is missing. In contrast, in accrual, the strength of each argument is taken into account, and adding an argument to an accrual causes changes in the strength and the effectiveness of its attacks (or supports).

Finally, as stated in Section 6, many open challenges remain for research on joint attacks, in addition to those mentioned above.

Acknowledgements

This work was partially supported by EPSRC grant EP/P010105/1 and by Universidad Nacional del Sur grant 24/N046. The opinions expressed in this paper are those of the authors and do not necessarily reflect the opinions of the funders. The authors are grateful to the reviewers of this chapter for their helpful comments.

References

[1] João F. L. Alcântara, Samy Sá, and Juan Carlos Acosta Guadarrama. On the equivalence between abstract dialectical frameworks and logic programs. *Theory and Practice of Logic Programming*, 19(5-6):941–956, 2019.

[2] Gianvincenzo Alfano, Andrea Cohen, Sebastian Gottifredi, Sergio Greco, Francesco Parisi, and Guillermo R. Simari. Dynamics in abstract argumentation frameworks with recursive attack and support relations. In *Proceedings of the 24th European Conference on Artificial Intelligence*, 2020.

[3] Leila Amgoud, Claudette Cayrol, Marie-Christine Lagasquie-Schiex, and Pierre Livet. On bipolarity in argumentation frameworks. *International Journal of Intelligent Systems, Bipolar Representations of Information and Preference (Part 2: reasoning and learning)*, 23(10):1062–1093, 2008.

[4] Leila Amgoud and Srdjan Vesic. Rich preference-based argumentation frameworks. *International Journal of Approximate Reasoning*, 55(2):585–606, 2014.

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

[6] Pietro Baroni, Martin Caminada, and Massimiliano Giacomin. Abstract argumentation frameworks and their semantics. In Pietro Baroni, Dov Gabbay, Massimiliano Giacomin, and Leendert van der Torre, editors, *Handbook of Formal Argumentation*, chapter 4, pages 159–236. College Publications, 2018.

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

[8] Pietro Baroni and Massimiliano Giacomin. Solving semantic problems with odd-length cycles in argumentation. In Thomas D. Nielsen and Nevin Lianwen Zhang, editors, *Proceedings of the 7th European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty*, pages 440–451. Springer, 2003.

[9] Pietro Baroni and Massimiliano Giacomin. On principle-based evaluation of extension-based argumentation semantics. *Artificial Intelligence*, 171(10-15):675–700, 2007.

[10] Pietro Baroni, Massimiliano Giacomin, and Beishui Liao. A general semi-structured formalism for computational argumentation: Definition, properties, and examples of application. *Artificial Intelligence*, 257:158 – 207, 2018.

[11] Pietro Baroni, Antonio Rago, and Francesca Toni. From fine-grained properties to broad principles for gradual argumentation: A principled spectrum. *International Journal of Approximate Reasoning*, 105:252–286, 2019.

[12] Pietro Baroni, Marco Romano, Francesca Toni, Marco Aurisicchio, and Giorgio Bertanza. Automatic evaluation of design alternatives with quantitative argumentation. *Argument & Computation*, 6(1):24–49, 2015.

[13] Ringo Baumann. On the nature of argumentation semantics: Existence and uniqueness, expressibility, and replaceability. In Pietro Baroni, Dov Gabbay, Massimiliano Giacomin, and Leendert van der Torre, editors, *Handbook of Formal Argumentation*, chapter 17, pages 839–936. College Publications, 2018. also appears in IfCoLog Journal of Logics and their Applications 4(8):2779–2886.

[14] Ringo Baumann, Gerhard Brewka, and Markus Ulbricht. Revisiting the foundations of abstract argumentation — semantics based on weak admissibility and weak defense. In *Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence*, pages 2742–2749. AAAI Press, 2020.

[15] Trevor J. M. Bench-Capon. Persuasion in practical argument using value-based argumentation frameworks. *Journal of Logic and Computation*, 13(3):429–448, 2003.

[16] Philippe Besnard and Anthony Hunter. A logic-based theory of deductive arguments. *Artificial Intelligence*, 128:203–235, 2001.

[17] David Billington, Grigoris Antoniou, Guido Governatori, and Michael Maher. An inclusion theorem for defeasible logics. *ACM Transactions on Computational Logic*,

12(1), November 2010.

[18] Alexander Bochman. Collective argumentation and disjunctive logic programming. *Journal of Logic and Computation*, 13(3):405–428, 2003.

[19] Andrei Bondarenko, Phan Minh Dung, Robert A. Kowalski, and Francesca Toni. An abstract, argumentation-theoretic approach to default reasoning. *Artificial Intelligence*, 93(1-2):63–101, 1997.

[20] Gerd Brewka, Paul E. Dunne, and Stefan Woltran. Relating the semantics of abstract dialectical frameworks and standard AFs. In *Proceedings of the 22nd International Joint Conference on Articial Intellignce*, 2011.

[21] Gerhard Brewka, Martin Diller, Georg Heissenberger, Thomas Linsbichler, and Stefan Woltran. Solving advanced argumentation problems with answer-set programming. In Satinder P. Singh and Shaul Markovitch, editors, *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, pages 1077–1083. AAAI Press, 2017.

[22] Gerhard Brewka, Stefan Ellmauthaler, Hannes Strass, Johannes P. Wallner, and Stefan Woltran. Abstract dialectical frameworks. In Pietro Baroni, Dov Gabbay, Massimiliano Giacomin, and Leendert van der Torre, editors, *Handbook of Formal Argumentation*, chapter 5, pages 237–285. College Publications, 2018. also appears in IfCoLog Journal of Logics and their Applications 4(8):2263–2318.

[23] Gerhard Brewka, Hannes Strass, Stefan Ellmauthaler, Johannes Peter Wallner, and Stefan Woltran. Abstract dialectical frameworks revisited. In *Proceedings of the 23rd International Joint Conference on Artificial Intelligence*, pages 803–809, 2013.

[24] Martin Caminada. Semi-stable semantics. In *Proceedings of the 1st Conference on Computational Models of Argument*, pages 121–130, 2006.

[25] Martin Caminada. Comparing two unique extension semantics for formal argumentation: Ideal and eager. In *Proceedings of the 19th Belgian-Dutch Conference on Artificial Intelligence*, pages 81–87, 2007.

[26] Martin Caminada and Paul E. Dunne. Strong admissibility revisited: Theory and applications. *Argument & Computation*, 10(3):277–300, 2019.

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

[28] Martin Caminada, Samy Sá, João F. L. Alcântara, and Wolfgang Dvořák. On the equivalence between logic programming semantics and argumentation semantics. *International Journal of Approximate Reasoning*, 58:87–111, 2015.

[29] Claudette Cayrol, Andrea Cohen, and Marie-Christine Lagasquie-Schiex. Higher-order interactions (bipolar or not) in abstract argumentation: A state of the art. In Dov Gabbay, Massimiliano Giacomin, Guillermo R. Simari, and Matthias Thimm, editors, *Handbook of Formal Argumentation*, volume 2, chapter 1. College Publications, 2021.

[30] Claudette Cayrol, Jorge Fandinno, Luis Fariñas del Cerro, and Marie-Christine Lagasquie-Schiex. Argumentation frameworks with recursive attacks and evidence-based support. In F. Ferrarotti and S. Woltran, editors, *Proceedings of the 10th International Symposium on Foundations of Information and Knowledge Systems*, pages

150–169. Springer-Verlag, 2018.

[31] Claudette Cayrol, Jorge Fandinno, Luis Fariñas del Cerro, and Marie-Christine Lagasquie-Schiex. Structure-based semantics of argumentation frameworks with higher-order attacks and supports. In Sanjay Modgil, Katarzyna Budzynska, and John Lawrence, editors, *International Conference on Computational Models of Argument*, pages 29–36. IOS Press, septembre 2018.

[32] Claudette Cayrol and Marie-Christine Lagasquie-Schiex. On the acceptability of arguments in bipolar argumentation frameworks. In *Proceedings of the 8th European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty*, pages 378–389, 2005.

[33] Claudette Cayrol and Marie-Christine Lagasquie-Schiex. Coalitions of arguments: A tool for handling bipolar argumentation frameworks. *International Journal of Intelligent Systems*, 25(1):83–109, 2010.

[34] Claudette Cayrol and Marie-Christine Lagasquie-Schiex. Bipolarity in argumentation graphs: Towards a better understanding. *International Journal of Approximate Reasoning*, 54(7):876–899, 2013.

[35] Federico Cerutti, Sarah A. Gaggl, Matthias Thimm, and Johannes P. Wallner. Foundations of implementations for formal argumentation. In Pietro Baroni, Dov Gabbay, Massimiliano Giacomin, and Leendert van der Torre, editors, *Handbook of Formal Argumentation*, chapter 15, pages 688–767. College Publications, 2018. also appears in IfCoLog Journal of Logics and their Applications 4(8):2623–2706.

[36] Günther Charwat, Wolfgang Dvořák, Sarah Alice Gaggl, Johannes Peter Wallner, and Stefan Woltran. Methods for solving reasoning problems in abstract argumentation - A survey. *Artificial Intelligence*, 220:28–63, 2015.

[37] Andrea Cohen, Sebastian Gottifredi, Alejandro Javier García, and Guillermo Ricardo Simari. A survey of different approaches to support in argumentation systems. *Knowledge Engineering Review*, 29(5):513–550, 2014.

[38] Robyn M. Dawes. The robust beauty of improper linear models in decision making. *American Psychologist*, 34(7):571, 1979.

[39] Phan Minh Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning and logic programming. In *Proceedings of the 13th International Joint Conference on Artificial Intelligence*, pages 852–857, Chambéry, France, 1993.

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

[41] Phan Minh Dung, Paolo Mancarella, and Francesca Toni. Computing ideal sceptical argumentation. *Artificial Intelligence*, 171(10-15):642–674, July 2007.

[42] Paul E. Dunne. Computational properties of argument systems satisfying graph-theoretic constraints. *Artificial Intelligence*, 171(10-15):701–729, 2007.

[43] Paul E. Dunne. The computational complexity of ideal semantics. *Artificial Intelligence*, 173(18):1559–1591, 2009.

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

[45] Paul E. Dunne, Wolfgang Dvořák, and Stefan Woltran. Parametric properties of ideal semantics. *Artificial Intelligence*, 202:1–28, 2013.

[46] Paul E. Dunne, Anthony Hunter, Peter McBurney, Simon Parsons, and Michael Wooldridge. Weighted argument systems: Basic definitions, algorithms, and complexity results. *Artificial Intelligence*, 175(2):457 – 486, 2011.

[47] Wolfgang Dvořák and Paul E. Dunne. Computational problems in formal argumentation and their complexity. In Pietro Baroni, Dov Gabbay, Massimiliano Giacomin, and Leendert van der Torre, editors, *Handbook of Formal Argumentation*, chapter 14, pages 631–687. College Publications, 2018. also appears in IfCoLog Journal of Logics and their Applications 4(8):2557–2622.

[48] Wolfgang Dvořák, Jorge Fandinno, and Stefan Woltran. On the expressive power of collective attacks. *Argument & Computation*, 10(2):191–230, 2019.

[49] Wolfgang Dvořák, Alexander Greßler, and Stefan Woltran. Evaluating SETAFs via Answer-Set Programming. In *Proceedings of the Second International Workshop on Systems and Algorithms for Formal Argumentation*, pages 10–21. CEUR-WS.org, 2018.

[50] Wolfgang Dvořák, Atefeh Keshavarzi Zafarghandi, and Stefan Woltran. Expressiveness of SETAFs and support-free ADFs under 3-valued semantics. In *Proceedings of the 8th Conference on Computational Models of Argument*, pages 191–202. IOS Press, 2020.

[51] Wolfgang Dvořák, Sebastian Ordyniak, and Stefan Szeider. Augmenting tractable fragments of abstract argumentation. *Artificial Intelligence*, 186:157–173, 2012.

[52] Wolfgang Dvořák, Reinhard Pichler, and Stefan Woltran. Towards fixed-parameter tractable algorithms for abstract argumentation. *Artificial Intelligence*, 186:1–37, 2012.

[53] Wolfgang Dvořák, Anna Rapberger, and Johannes Peter Wallner. Labelling-based algorithms for SETAFs. In *Proceedings of the Third International Workshop on Systems and Algorithms for Formal Argumentation*, pages 34–46. CEUR-WS.org, 2020.

[54] Wolfgang Dvořák, Anna Rapberger, and Stefan Woltran. Strong equivalence for argumentation frameworks with collective attacks. In *Proceedings of the 42nd German Conference on Artificial Intelligence*, pages 131–145. Springer, 2019.

[55] Wolfgang Dvořák, Anna Rapberger, and Stefan Woltran. On the different types of collective attacks in abstract argumentation: equivalence results for SETAFs. *Journal of Logic and Computation*, 30(5):1063–1107, 2020.

[56] Wolfgang Dvořák, Anna Rapberger, and Stefan Woltran. On the relation between claim-augmented argumentation frameworks and collective attacks. In *Proceedings of the 24th European Conference on Artificial Intelligence*, pages 721–728. IOS Press, 2020.

[57] Uwe Egly, Sarah Alice Gaggl, and Stefan Woltran. Answer-set programming encodings

for argumentation frameworks. *Argument & Computation*, 1(2):147–177, 2010.

[58] Stefan Ellmauthaler and Hannes Strass. The DIAMOND system for computing with abstract dialectical frameworks. In *Proceedings of the 5th Conference on Computational Models of Argumen*, volume 266, pages 233–240. IOS Press, 2014.

[59] Morten Elvang-Gøransson, Paul J. Krause, and John Fox. Acceptability of arguments as 'logical uncertainty'. In M. Clarke, R. Kruse, and S. Moral, editors, *Proceedings of the 2nd European Conference on Symbolic and Quantitative Approaches to Reasoning and Uncertainty*, pages 85–90. Springer, 1993.

[60] Giorgos Flouris and Antonis Bikakis. A comprehensive study of argumentation frameworks with sets of attacking arguments. *International Journal of Approximate Reasoning*, 109, 03 2019.

[61] J. Fox, D. Barber, and K. D. Bardhan. Alternatives to Bayes? a quantitative comparison with rule-based diagnostic inference. *Methods of Information in Medicine*, 19:210–215, 1980.

[62] Dov M. Gabbay. Semantics for higher level attacks in extended argumentation frames. *Studia Logica*, 93:357–381, 2009.

[63] Dov M. Gabbay and Michael Gabbay. Theory of disjunctive attacks, Part I. *Logic Journal of the IGPL*, 24(2):186–218, 2016.

[64] Sarah Alice Gaggl, Thomas Linsbichler, Marco Maratea, and Stefan Woltran. Design and results of the second international competition on computational models of argumentation. *Artificial Intelligence*, 279, 2020.

[65] Alejandro Javier García and Guillermo Ricardo Simari. Argumentation based on logic programming. In Pietro Baroni, Dov Gabbay, Massimiliano Giacomin, and Leendert van der Torre, editors, *Handbook of Formal Argumentation*, chapter 8, pages 409–435. College Publications, 2018.

[66] Martin Gebser, Benjamin Kaufmann, Roland Kaminski, Max Ostrowski, Torsten Schaub, and Marius T. Schneider. Potassco: The Potsdam answer set solving collection. *AI Communications*, 24(2):107–124, 2011.

[67] Thomas F. Gordon. Defining argument weighing functions. *Journal of Applied Logics*, 5(3):747–773, 2018.

[68] Sebastian Gottifredi, Andrea Cohen, Alejandro Javier García, and Guillermo Ricardo Simari. Characterizing acceptability semantics of argumentation frameworks with recursive attack and support relations. *Artificial Intelligence*, 262:336–368, 2018.

[69] Guido Governatori and Michael J. Maher. An Argumentation-Theoretic Characterization of Defeasible Logic. In *Proceedings of the 14th European Conference on Artificial Intelligence*, pages 469–473, 2000.

[70] Benjamin N. Grosof. Prioritized conflict handling for logic programs. In Jan Maluszynski, editor, *Proceedings of the 1997 International Symposium on LOgic Programming*, pages 197–211. MIT Press, 1997.

[71] Davide Grossi and Sanjay Modgil. On the graded acceptability of arguments. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence*, pages

868–874. AAAI Press, 2015.

[72] Paul J. Krause, Simon Ambler, Morten Elvang-Gøransson, and John Fox. A logic of argumentation for reasoning under uncertainty. *Computational Intelligence*, 11 (1):113–131, 1995.

[73] E. Laenens and D. Vermeir. A Fixpoint Semantics for Ordered Logic. *Journal of Logic and Computation*, 1(2):159–185, 12 1990.

[74] João Leite and João G. Martins. Social abstract argumentation. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence, Barcelona*, pages 2287–2292. IJCAI/AAAI, 2011.

[75] Nicola Leone, Gerald Pfeifer, Wolfgang Faber, Thomas Eiter, Georg Gottlob, Simona Perri, and Francesco Scarcello. The DLV system for knowledge representation and reasoning. *ACM Transactions on Computational Logic*, 7(3):499–562, 2006.

[76] Thomas Linsbichler, Marco Maratea, Andreas Niskanen, Johannes Peter Wallner, and Stefan Woltran. Novel algorithms for abstract dialectical frameworks based on complexity analysis of subclasses and SAT solving. In Jérôme Lang, editor, *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*, pages 1905–1911, 2018.

[77] Thomas Linsbichler, Jörg Pührer, and Hannes Strass. A uniform account of realizability in abstract argumentation. In *Proceedings of the 22nd European Conference on Artificial Intelligence*, pages 252–260. IOS Press, 2016.

[78] Mauro J. Gómez Lucero, Carlos I. Chesñevar, and Guillermo R. Simari. On the accrual of arguments in defeasible logic programming. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence*, pages 804–809. Morgan Kaufmann Publishers Inc., 2009.

[79] Victor W. Marek and Mirosław Truszczyński. Stable models and an alternative logic programming paradigm. In *The Logic Programming Paradigm – A 25-Year Perspective*, pages 375–398. Springer, 1999.

[80] Sanjay Modgil and Trevor J. M. Bench-Capon. Metalevel argumentation. *Journal of Logic and Computation*, 21(6):959–1003, 09 2010.

[81] Sanjay Modgil and Henry Prakken. Abstract rule-based argumentation. In Pietro Baroni, Dov Gabbay, Massimiliano Giacomin, and Leendert van der Torre, editors, *Handbook of Formal Argumentation*, chapter 6, pages 287–364. College Publications, 2018. also appears in IfCoLog Journal of Logics and their Applications 4(8):2319–2406.

[82] Søren Holbech Nielsen and Simon Parsons. Computing preferred extensions for argumentation systems with sets of attacking arguments. In *Proceedings of the 1st Conference on Computational Models of Argument*, pages 97–108. IOS Press, 2006.

[83] Søren Holbech Nielsen and Simon Parsons. An application of formal argumentation: Fusing Bayesian networks in multi-agent systems. *Artificial Intelligence*, 171(10–15):754–775, 2007.

[84] Søren Holbech Nielsen and Simon Parsons. A generalization of Dung's abstract framework for argumentation: Arguing with sets of attacking arguments. In *Proceedings*

of the 3rd International Workshop on Argumentation in Multi-Agent Systems, pages 54–73, 2007.

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

[86] Samer Nofal, Katie Atkinson, and Paul E. Dunne. Algorithms for decision problems in argument systems under preferred semantics. *Artificial Intelligence*, 207:23–51, 2014.

[87] Farid Nouioua. AFs with necessities: Further semantics and labelling characterization. In *7th International Conference on Scalable Uncertainty Management*, pages 120–133, 2013.

[88] Farid Nouioua and Vincent Risch. Argumentation frameworks with necessities. In *Proceedings of the 5th International Conference on Scalable Uncertainty Management*, pages 163–176, 2011.

[89] Donald Nute. *Defeasible Logic*, page 353?395. Oxford University Press, Inc., USA, 1994.

[90] Mike O'Neil, Andrzej J. Glowinski, and John Fox. A symbolic theory of decision-making applied to several medical tasks. In *AIME 89: Proceedings of the Second European Conference on Artificial Intelligence in Medicine*, pages 62–71. Springer, 1989.

[91] Nir Oren and Timothy J. Norman. Semantics for evidence-based argumentation. In *Proceedings of the 2nd Conference on Computational Models of Argument*, pages 276–284, 2008.

[92] Theodore Patkos, Giorgos Flouris, and Antonis Bikakis. Symmetric multi-aspect evaluation of comments - extended abstract. In *Proceedings of the 22nd European Conference on Artificial Intelligence*, pages 1672–1673. IOS Press, 2016.

[93] Sylwia Polberg. Understanding the abstract dialectical framework. In *Proceedings of the 15th European Conference on Logics in Artificial Intelligence*, pages 430–446, 2016.

[94] Sylwia Polberg. *Developing the abstract dialectical framework*. PhD thesis, TU Wien, Institute of Information Systems, 2017. available at `http://katalog.ub.tuwien.ac.at/AC13773888`.

[95] Sylwia Polberg and Nir Oren. Revisiting support in abstract argumentation systems. In *Proceedings of the 5th Conference on Computational Models of Argument*, pages 369–376, 2014.

[96] Henry Prakken. A study of accrual of arguments, with applications to evidential reasoning. In *Proceedings of the 10th International Conference on Artificial Intelligence and Law*, pages 85–94, New York, NY, USA, 2005. ACM.

[97] Henry Prakken. Modelling accrual of arguments in ASPIC+. In *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Law*, pages 103–112, New York, NY, USA, 2019. Association for Computing Machinery.

[98] Jörg Pührer. Realizability of three-valued semantics for abstract dialectical frame-

works. *Artificial Intelligence*, 278, 2020.

[99] Hannes Strass. Expressiveness of two-valued semantics for abstract dialectical frameworks. *Journal of Artificial Intelligence Research*, 54:193–231, 2015.

[100] Hannes Strass. The relative expressiveness of abstract argumentation and logic programming. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence*, pages 1625–1631, 2015.

[101] Matthias Thimm and Serena Villata. The first international competition on computational models of argumentation: Results and analysis. *Artificial Intelligence*, 252:267–294, 2017.

[102] Leendert van der Torre and Srdjan Vesic. The principle-based approach to abstract argumentation semantics. In Pietro Baroni, Dov Gabbay, Massimiliano Giacomin, and Leendert van der Torre, editors, *Handbook of Formal Argumentation*, chapter 16, pages 797–838. College Publications, 2018. also appears in IfCoLog Journal of Logics and their Applications 4(8):2735–2778.

[103] Bart Verheij. Accrual of arguments in defeasible argumentation. In *Proceedings of the Second Dutch/German Workshop on Nonmonotonic Reasoning*, pages 217–224, 1995.

[104] Bart Verheij. *Rules, Reasons, Arguments. Formal studies of argumentation and defeat.* PhD thesis, Universiteit Maastricht, 1996.

[105] Bart Verheij. Two approaches to dialectical argumentation: Admissible sets and argumentation stages. In *In Proceedings of the International Conference on Formal and Applied Practical Reasoning*, pages 357–368. Universiteit, 1996.

[106] Gerard A. W. Vreeswijk. Abstract argumentation systems. *Artificial Intelligence*, 90(1–2):225–279, February 1997.

[107] Bruno Yun, Madalina Croitoru, Srdjan Vesic, and Pierre Bisquert. NAKED: n-ary graphs from knowledge bases expressed in Datalog$^{\pm}$. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, pages 2390–2392, 2019.

[108] Bruno Yun, Srdjan Vesic, and Madalina Croitoru. Ranking-based semantics for sets of attacking arguments. In *Proceedings of the 34th Conference on Artificial Intelligence*, pages 3033–3040. AAAI Press, 2020.

[109] Bruno Yun, Srdjan Vesic, and Madalina Croitoru. Sets of attacking arguments for inconsistent Datalog knowledge bases. In *Proceedings of the 8th Conference on Computational Models of Argument*, pages 419–430. IOS Press, 2020.

Received May 2021