Expressiveness of SETAFs and Support-Free ADFs under 3-valued Semantics

W. Dvořák[a], A. Keshavarzi Zafarghandi[b,c] and S. Woltran[a]

[a]Institute of Logic and Computation, TU Wien, Austria;
[b]Human-Centered Data Analytics, Centrum Wiskunde & Informatica, The Netherlands;
[c]Vrije University Amsterdam, The Netherlands

ARTICLE HISTORY
Compiled February 2, 2024

ABSTRACT
Generalizing the attack structure in argumentation frameworks (AFs) has been studied in different ways. Most prominently, the binary attack relation of Dung frameworks has been extended to the notion of collective attacks. The resulting formalism is often termed SETAFs. Among the generalizations of AFs, abstract dialectical frameworks ADFs allow for a systematic and flexible generalization of AFs in which different kinds of logical relations, e.g. attack and support, among arguments can be represented. Restricting the logical relations among arguments leads to different subclasses of ADFs of interest. In this work we consider so-called support-free ADFs, that allow for all kinds of attacks but no support or other relations, and SETADFs, that embed SETAFs in the ADF setting.

The aim of the paper is to shed light on the relation between these two different approaches. To this end, we investigate and compare the expressiveness of SETAFs and support-free ADFs under the lens of 3-valued semantics. Our results show that it is only the presence of unsatisfiable acceptance conditions in support-free ADFs that discriminates the two approaches.

KEYWORDS
Abstract argumentation frameworks, Abstract dialectical frameworks, Collective attack.

1. Introduction

The last 25 years have seen an increasing interest in the area of formal argumentation. The ultimate goal of the field is to come up with computational models of how we make decisions, based on incomplete or inconsistent information. It is very natural to first construct arguments pro and con a particular position, and then to select amongst all possible arguments those which are most reliable or trustworthy and lead to a coherent view of the world. The main goal of formal argumentation is to make these ideas precise via formal, computational theories of argumentation. The concept of abstraction turned out be very useful in this context. Hereby, the tasks of constructing arguments and sorting out the relation between them is decoupled from the question which arguments are jointly acceptable. The latter is solely based on the relation between the arguments (and thus "abstracts away" from their actual contents), and

CONTACT A. Keshavarzi Zafarghandi. Email: akz@cwi.nl

it was Dung's seminal paper (Dung, 1995) which proposed fundamental mechanisms how to compute these acceptable sets (often referred to as extensions).

However, Dung's abstract frameworks (and semantics) only consider a very simple, binary, relation between the arguments (the "attack relation"). While this simple concept suffices in many settings, the community recognized that certain situations require a richer structure (relaxing attacks to be non-binary, introducing additional relations, e.g. support between arguments). Several such proposals can be found in the literature (we refer to (Brewka, Polberg, & Woltran, 2014) for an overview), and most of them have in common that the semantics as proposed by Dung are generalized in similar ways, making these different formalisms amenable to formal comparisons. It is this line of research, we want to tackle in this work. In fact, we shall investigate two generalisations of Dung's abstract frameworks (AFs) that allow for a more flexible attack structure (but do not consider support between arguments). Understanding the capacities of different formalisms is crucial to decide which approach to select for a certain application scenario, but it is also important for avoiding redundant research (for instance, in case it turns out that two formalisms are equally powerful).

The first formalism which we consider here are SETAFs as introduced by Nielsen and Parsons (Bikakis, Cohen, Dvořák, Flouris, & Parsons, 2021; Nielsen & Parsons, 2007). SETAFs extend Dung AFs by allowing for collective attacks with the intended meaning that a set of arguments B attacks another argument a but no proper subset of B attacks a (via this attack). Argumentation frameworks with collective attacks have received increasing interest in the last years, to some extend driven by the observation that for particular instantiations, SETAFs provide a more convenient target formalism than Dung AFs, see e.g., (Yun, Vesic, & Croitoru, 2018). We list a few of such new results on SETAFs: the adaption of semi-stable, stage, ideal, and eager semantics (Dvořák, Fandinno, & Woltran, 2019; Flouris & Bikakis, 2019); translations between SETAFs and other abstract argumentation formalisms (Polberg, 2017); principle-based investigations (Dvořák, König, Ulbricht, & Woltran, 2022); and tailored complexity results for SETAFs, e.g., Dvořák, König, and Woltran (2021).

The second formalism we consider are support-free abstract dialectical frameworks (SFADFs), a subclass of abstract dialectical frameworks (ADFs) (Brewka, Ellmauthaler, Strass, Wallner, & Woltran, 2018; Brewka & Woltran, 2010) which are known as an advanced abstract formalism for argumentation, that is able to cover several generalizations of AFs (Brewka et al., 2014; Polberg, 2017). This is accomplished by acceptance conditions which specify, for each argument, its relation to its neighbour arguments via propositional formulas. These conditions determine the links between the arguments which can be, in particular, attacking or supporting. SFADFs are ADFs where each link between arguments is attacking; they have been introduced in a recent study on different sub-classes of ADFs (Diller, Zafarghandi, Linsbichler, & Woltran, 2020).

For comparison of the two formalisms, we need to focus on 3-valued (labelling) semantics (Caminada & Gabbay, 2009; Verheij, 1996), which are integral for ADF semantics (Brewka et al., 2018). In terms of SETAFs, we can rely on the recently introduced labelling semantics in (Flouris & Bikakis, 2019). We first define a new class of ADFs (SETADFs) where the acceptance conditions strictly follow the nature of collective attacks in SETAFs. We then show that SETAFs and SETADFs coincide for the main semantics, i.e. the σ-labellings of a SETAF are equal to the σ-interpretations of the corresponding SETADF. While SETADFs are a syntactically defined subclass of ADFs, the second formalism (SFADFs) we study can be understood as semantical subclass of ADFs. In fact, for SFADFs it is not the syntactic structure of acceptance

conditions that is restricted but their semantic behavior, in the sense that all links need to be attacking. The second main contribution of the paper is to determine the exact difference in expressiveness between SETADFs and SFADFs.

We do so by employing the concept of *signatures*. As argued in Dunne, Dvořák, Linsbichler, and Woltran (2015), an approach to study and compare the capabilities of different semantics of AFs is via the notion of realizability. Realizability is the ability of a formalism under a semantics to express specific desired sets of extensions. Signatures then capture the exact expressiveness of a formalism under a semantics by collecting all sets of extensions that can be realized.[1] Previous work in this respect includes Dunne et al. (2015) which studies the expressiveness of AFs, and Pührer (2020a); Strass (2015a) for the expressiveness of ADFs from the perspective of realizability. Signature have been used to compare the capability of different formalisms Strass (2015b) and are recognized as crucial for operators in dynamics of argumentation (cf. (Baumann & Brewka, 2019)). For SETAFs the expressiveness of two-valued semantics has been investigated in (Dvořák et al., 2019), but the expressiveness of three-valued semantics has not been characterised so far.

We deepen this kind of comparison by also investigating symmetric versions of these classes (we recall that syntactic restrictions often lead to lower computational complexity, see e.g. (Linsbichler, Maratea, Niskanen, Wallner, & Woltran, 2022) for recent work in this direction in terms of ADFs).
To summarize, the main contributions of our paper are as follows:

- We embed SETAFs under 3-valued labeling based semantics (Flouris & Bikakis, 2019) in the more general framework of ADFs. That is, we show 3-valued labeling based SETAF semantics to be equivalent to the corresponding ADF semantics. As a side result, this also shows the equivalence of the 3-valued SETAF semantics in (Linsbichler, Pührer, & Strass, 2016) and (Flouris & Bikakis, 2019).
- We investigate the expressiveness of SETAFs under 3-valued semantics by providing exact characterizations of the signatures for preferred, stable, grounded and conflict-free semantics, thus complementing the investigations on expressiveness of SETAFs (Dvořák et al., 2019) in terms of extension-based semantics.
- We study the relations between SETAFs and support-free ADFs (SFADFs). In particular we give the exact difference in expressiveness between SETAFs and SFADFs under conflict-free, admissible, preferred, grounded, complete, stable and two-valued model semantics.
- Finally, we consider symmetric variants of SETAFs, SFADFs and SETADFs. In particular, we can show that the class of symmetric SFADFs is a strict superset of the class of symmetric SETADFs, while the expressiveness of the class of symmetric SETAFs and the class of symmetric SETADFs is the same.

This contribution is an extended version of the conference paper (Dvořák, Keshavarzi Zafarghandi, & Woltran, 2020). New material is concerned with symmetric subclasses of the SETAFs, SFADFs, and SETADFs.

Special issue on the occasion of Philippe Besnard's retirement. We have chosen this article as contribution to the special issue on the occasion of Philippe Besnard's retirement, since it touches on several areas Philippe has worked on. First of all, one has to mention the seminal book on argumentation (Besnard & Hunter, 2008) which

[1] We note this concept does not allow for auxiliary arguments that simulate certain behaviors, see e.g. Modgil and Bench-Capon (2011).

gave an important boost to the argumentation community and has stipulated many young academics to take up research in this field. Second, with the article (Besnard & Doutre, 2004) Philippe, together with Sylvie Doutre, has initiated research on reducing Dung AFs to other formalisms. The impact of their work is witnessed by the fact that the most performant systems for abstract argumentation still rely on reductions to SAT, see e.g. (Lagniez, Lonca, Mailly, & Rossit, 2021; Niskanen & Järvisalo, 2020). Finally, the usage of three-valued semantics (which is constitutional for ADF semantics) for inconsistency handling is something Philippe has already considered in the 90ies when introducing the idea of circumscribing inconsistency, see e.g. (Besnard & Schaub, 1997). In particular, the third author of the present contribution would like to express his gratefulness to Philippe for having had the opportunity to jointly work on these topics (Besnard, Hunter, & Woltran, 2009; Besnard, Schaub, Tompits, & Woltran, 2002, 2003).

Dear Philippe, the discussion we had were always enlightening and an important inspiration for my research agenda. We all wish you many laid-back years ahead. All the best and thank you so much!

2. Argumentation Frameworks

In this section we briefly recall the necessary definitions for the different types of argumentation frameworks we consider in this work, i.e., argumentation frameworks with collective attacks (SETAFs), Abstract Dialectical Frameworks (ADFs), and support-free ADFs (SFADFs).

2.1. Argumentation Frameworks with Collective Attacks

We recall the necessary background on SETAFs following Nielsen and Parsons (2007) and Bikakis et al. (2021).

Definition 2.1. An argumentation framework with collective attacks (SETAF) is an ordered pair $F = (A, R)$, where A is a finite set of arguments and $R \subseteq (2^A \setminus \{\emptyset\}) \times A$ is the attack relation.

The semantics of SETAFs are usually defined similarly to AFs, i.e., based on extensions. However, in this work we focus on 3-valued labelling based SETAF semantics introduced by Flouris and Bikakis (2019).

Definition 2.2. A (3-valued) labelling of a SETAF $F = (A, R)$ is a total function $\lambda : A \mapsto \{\texttt{in}, \texttt{out}, \texttt{undec}\}$. For $x \in \{\texttt{in}, \texttt{out}, \texttt{undec}\}$ we write λ_x to denote the sets of arguments $a \in A$ with $\lambda(a) = x$. We sometimes denote labellings λ as triples $(\lambda_{\texttt{in}}, \lambda_{\texttt{out}}, \lambda_{\texttt{undec}})$.

Next we give the generalization of conflict-freeness to labellings of SETAFs.

Definition 2.3. Let $F = (A, R)$ be a SETAF. A labelling is called conflict-free in F if

(i) for all $(S, a) \in R$ either $\lambda(a) \neq \texttt{in}$ or there is a $b \in S$ with $\lambda(b) \neq \texttt{in}$, and
(ii) for all $a \in A$, if $\lambda(a) = \texttt{out}$ then there is an attack $(S, a) \in R$ such that $\lambda(b) = \texttt{in}$ for all $b \in S$.

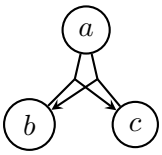

Figure 1. The SETAF of Example 2.5.

We are now ready to present the definitions of the SETAF semantics under our considerations.

Definition 2.4. A labelling λ which is conflict-free in F is

- *naive* in F iff $\lambda_{\mathtt{in}}$ is $\subseteq$-maximal among all conflict-free labellings, i.e. there is no conflict-free λ' with $\lambda_{\mathtt{in}} \subset \lambda'_{\mathtt{in}}$;
- *admissible* in F iff for all $a \in A$ if $\lambda(a) = \mathtt{in}$ then for all $(S, a) \in R$ there is a $b \in S$ such that $\lambda(b) = \mathtt{out}$;
- *complete* in F iff for all $a \in A$ (i) $\lambda(a) = \mathtt{in}$ iff for all $(S, a) \in R$ there is a $b \in S$ such that $\lambda(b) = \mathtt{out}$, and (ii) $\lambda(a) = \mathtt{out}$ iff there is an attack $(S, a) \in R$ such that $\lambda(b) = \mathtt{in}$ for all $b \in S$;
- *grounded* in F iff it is complete and there is no λ' with $\lambda'_{\mathtt{in}} \subset \lambda_{\mathtt{in}}$ complete in F;
- *preferred* in F iff it is complete and there is no λ' with $\lambda'_{\mathtt{in}} \supset \lambda_{\mathtt{in}}$ complete in F;
- *stable* in F iff $\lambda_{\mathtt{undec}} = \emptyset$.

The set of all σ labellings for a SETAF F is denoted by $\sigma_{\mathcal{L}}(F)$, where $\sigma \in \{nai, cf, adm, com, grd, prf, stb\}$ abbreviates the different semantics in the obvious manner.

Example 2.5. The SETAF $F = (\{a, b, c\}, \{(\{a, b\}, c), (\{a, c\}, b)\})$ is depicted in Figure 1. For instance, $(\{a, b\}, c) \in R$ says that there is a joint attack from a and b to c. This represents that neither a nor b is strong enough to attack c by themselves. Further, $\{a \mapsto \mathtt{in}, b \mapsto \mathtt{undec}, c \mapsto \mathtt{in}\}$ is an instance of a conflict-free labelling, that is not an admissible labelling (since c is mapped to `in` but neither a nor b is mapped to `out`). The labelling that maps all arguments to `undec` is not a complete labelling, however, it is an admissible labelling. Further, $\{a \mapsto \mathtt{in}, b \mapsto \mathtt{undec}, c \mapsto \mathtt{undec}\}$ is an admissible, the unique grounded and a complete labelling, which is not a preferred labelling because $\lambda_{\mathtt{in}} = \{a\}$ is not $\subseteq$-maximal among all complete labellings. Moreover, $prf_{\mathcal{L}}(F) = stb_{\mathcal{L}}(F) = \{\{a \mapsto \mathtt{in}, b \mapsto \mathtt{out}, c \mapsto \mathtt{in}\}, \{a \mapsto \mathtt{in}, b \mapsto \mathtt{in}, c \mapsto \mathtt{out}\}\}$.

Notice that *Dungs Abstract Argumentation Frameworks (AFs)* (Dung, 1995) and their semantics can be identified with SETAFs whose attacks are restricted a single argument attacking an argument (Nielsen & Parsons, 2007). That is all attacks are of the form $(\{b\}, a)$ for some arguments a, b (in the setting of Dung AFs such attacks are then denoted by pairs of arguments (b, a)).

2.2. Abstract Dialectical Frameworks

We now turn to abstract dialectical frameworks (Brewka, Ellmauthaler, Strass, Wallner, & Woltran, 2013; Brewka & Woltran, 2010). We start with the definition of ADFs.

Definition 2.6. An abstract dialectical framework (ADF) is a tuple $D = (S, L, C)$ where:

- S is a finite set of arguments (statements, positions);
- $L \subseteq S \times S$ is a set of links among arguments;
- $C = \{\varphi_s\}_{s \in S}$ is a collection of propositional formulas over arguments, called acceptance conditions.

An ADF can be represented by a graph in which nodes indicate arguments and links show the relation among arguments. Each argument s in an ADF is attached by a propositional formula, called acceptance condition, φ_s over $par(s)$ such that, $par(s) = \{b \mid (b, s) \in L\}$. Since in ADFs an argument appears in the acceptance condition of an argument s if and only if it belongs to the set $par(s)$, the set of links L of an ADF is given implicitly via the acceptance conditions. The acceptance condition of each argument clarifies under which condition the argument can be accepted and determines the type of links (see Definition 2.9 below). An argument s is considered an *initial argument* if it has no parent, i.e., $par(s) = \emptyset$, and it is called an *isolated argument* if it is an initial argument and has no children, meaning it does not have any outgoing links.

An *interpretation* v (for F) is a function $v : S \mapsto \{\mathbf{t}, \mathbf{f}, \mathbf{u}\}$, that maps arguments to one of the three truth values true ($\mathbf{t}$), false ($\mathbf{f}$), or undecided ($\mathbf{u}$). Truth values can be ordered via information ordering relation $<_i$ given by $\mathbf{u} <_i \mathbf{t}$ and $\mathbf{u} <_i \mathbf{f}$ and no other pair of truth values are related by $<_i$. Relation $\leq_i$ is the reflexive and transitive closure of $<_i$. An interpretation v is *two-valued* if it maps each argument to either $\mathbf{t}$ or $\mathbf{f}$. Let $\mathcal{V}$ be the set of all interpretations for an ADF D. Then, we call a subset of all interpretations of the ADF, $\mathbb{V} \subseteq \mathcal{V}$, an *interpretation-set*. Interpretations can be ordered via $\leq_i$ with respect to their information content, i.e. $w \leq_i v$ if $w(s) \leq_i v(s)$ for each $s \in S$. Further, we denote the update of an interpretation v with a truth value $x \in \{\mathbf{t}, \mathbf{f}, \mathbf{u}\}$ for an argument b by $v|_x^b$, i.e. $v|_x^b(b) = x$ and $v|_x^b(a) = v(a)$ for $a \neq b$. Finally, the partial valuation of acceptance condition φ_s by v, is given by $\varphi_s^v = v(\varphi_s) = \varphi_s[p/\top : v(p) = \mathbf{t}][p/\bot : v(p) = \mathbf{f}]$, for $p \in par(s)$.

Semantics for ADFs can be defined via a *characteristic operator* Γ_D for an ADF D. Given an interpretation v (for D), the characteristic operator Γ_D for D is defined as

$$\Gamma_D(v) = v' \text{ such that } v'(s) = \begin{cases} \mathbf{t} & \text{if } \varphi_s^v \text{ is irrefutable (i.e., a tautology),} \\ \mathbf{f} & \text{if } \varphi_s^v \text{ is unsatisfiable,} \\ \mathbf{u} & \text{otherwise.} \end{cases}$$

Definition 2.7. Given an ADF $D = (S, L, C)$, an interpretation v is

- *conflict-free* in D iff $v(s) = \mathbf{t}$ implies φ_s^v is satisfiable and $v(s) = \mathbf{f}$ implies φ_s^v is unsatisfiable;
- *admissible* in D iff $v \leq_i \Gamma_D(v)$;
- *complete* in D iff $v = \Gamma_D(v)$;
- *grounded* in D iff v is the (unique) least fixed-point of Γ_D;
- *preferred* in D iff v is $\leq_i$-maximal admissible in D;
- a *(two-valued) model* of D iff v is two-valued and for all $s \in S$, it holds that $v(s) = v(\varphi_s)$;
- a *stable model* of D if v is a model of D and $v^{\mathbf{t}} = w^{\mathbf{t}}$, where w is the grounded interpretation of the *stb*-reduct $D^v = (S^v, L^v, C^v)$, where $S^v = v^{\mathbf{t}}$, $L^v = L \cap (S^v \times S^v)$, and $\varphi_s[p/\bot : v(p) = \mathbf{f}]$ for each $s \in S^v$.

The set of all σ interpretations for an ADF D is denoted by $\sigma(D)$, where $\sigma \in \{cf, adm, com, grd, prf, mod, stb\}$ abbreviates the different semantics in the obvious manner.

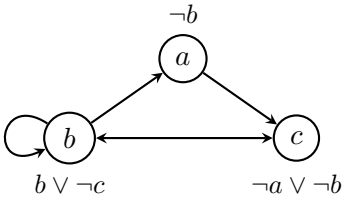

Figure 2. The ADF of Example 2.8.

Example 2.8. An example of an ADF $D = (S, L, C)$ is shown in Figure 2. To each argument a propositional formula is associated, the acceptance condition of the argument. For instance, the acceptance condition of c, namely $\varphi_c : \neg a \vee \neg b$, states that c can be accepted in an interpretation where either a or b (or both) are rejected.

In D the interpretation $v = \{a \mapsto \mathbf{u}, b \mapsto \mathbf{u}, c \mapsto \mathbf{t}\}$ is conflict-free. However, v is not an admissible interpretation, because $\Gamma_D(v) = \{a \mapsto \mathbf{u}, b \mapsto \mathbf{u}, c \mapsto \mathbf{u}\}$, that is, $v \not\leq_i \Gamma_D(v)$. The interpretation $v_1 = \{a \mapsto \mathbf{f}, b \mapsto \mathbf{t}, c \mapsto \mathbf{u}\}$ on the other hand is an admissible interpretation. Since $\Gamma_D(v_1) = \{a \mapsto \mathbf{f}, b \mapsto \mathbf{t}, c \mapsto \mathbf{t}\}$ and $v_1 \leq_i \Gamma_D(v_1)$. Further, $prf(D) = mod(D) = \{\{a \mapsto \mathbf{t}, b \mapsto \mathbf{f}, c \mapsto \mathbf{t}\}, \{a \mapsto \mathbf{f}, b \mapsto \mathbf{t}, c \mapsto \mathbf{t}\}\}$, but only the first interpretation in this set is a stable model. This is because for $v = \{a \mapsto \mathbf{t}, b \mapsto \mathbf{f}, c \mapsto \mathbf{t}\}$ the unique grounded interpretation w of D^v is $\{a \mapsto \mathbf{t}, c \mapsto \mathbf{t}\}$ and $v^{\mathbf{t}} = w^{\mathbf{t}}$. The interpretation $v' = \{a \mapsto \mathbf{f}, b \mapsto \mathbf{t}, c \mapsto \mathbf{t}\}$ is not a stable model, since the unique grounded interpretation w' of $D^{v'}$ is $\{b \mapsto \mathbf{u}, c \mapsto \mathbf{t}\}$ and $v'^{\mathbf{t}} \neq w'^{\mathbf{t}}$. Actually, the truth value of b in v' is assigned to $\mathbf{t}$ due to self-support, which makes v' an unstable model. Moreover, the unique grounded interpretation of D is $v = \{a \mapsto \mathbf{u}, b \mapsto \mathbf{u}, c \mapsto \mathbf{u}\}$. In addition, we have that for the given ADF D, $com(D) = prf(D) \cup grd(D)$.

2.3. Support-free ADFs

In ADFs links between arguments can be classified into four types, reflecting the relationship of attack and/or support that exists among the arguments. In Definition 2.9 we consider two-valued interpretations that are only defined over the parents of a, that is, only give values to $par(a)$.

Definition 2.9. Let $D = (S, L, C)$ be an ADF. A link $(b, a) \in L$ is called

- *supporting* (in D) if for every two-valued interpretation v of $par(a)$, $v(\varphi_a) = \mathbf{t}$ implies $v|^b_{\mathbf{t}}(\varphi_a) = \mathbf{t}$;
- *attacking* (in D) if for every two-valued interpretation v of $par(a)$, $v(\varphi_a) = \mathbf{f}$ implies $v|^b_{\mathbf{t}}(\varphi_a) = \mathbf{f}$;
- *redundant* (in D) if it is both attacking and supporting;
- *dependent* (in D) if it is neither attacking nor supporting.

We illustrate the different types of links in the following example.

Example 2.10. Consider the ADF illustrated in Figure 3. For the acceptance condition $\varphi_a : b \vee \neg c$ of argument a we have $\{b, c\}$ as the set of parents. With that we clarify the type of (b, a) and (c, a). There are three satisfying two-valued interpretations, i.e., $v_1 = \{b \mapsto \mathbf{t}, c \mapsto \mathbf{t}\}$, $v_2 = \{b \mapsto \mathbf{t}, c \mapsto \mathbf{f}\}$ and $v_3 = \{b \mapsto \mathbf{f}, c \mapsto \mathbf{f}\}$, and one that does not satisfy the formula, i.e., $v_4 = \{b \mapsto \mathbf{f}, c \mapsto \mathbf{t}\}$. By the definition of supporting links we have to check that whether $v_i(\varphi_a) = \mathbf{t}$ for i with $1 \leq i \leq 3$ implies $v_i|^b_{\mathbf{t}}(\varphi_a) = \mathbf{t}$.

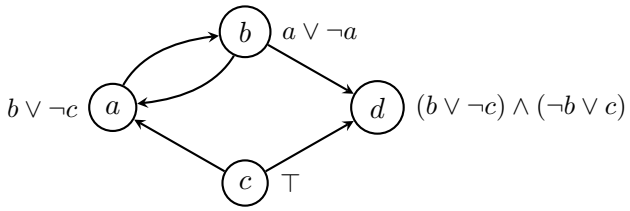

Figure 3. Different link types in ADFs: (b, a) is a supporting (but not attacking) link; (c, a) is an attacking (but not supporting) link; (a, b) is a redundant (both attacking and supporting) link; and (b, d), (c, d) are dependent (neither attacking nor supporting) links.

Since for i with $1 \leq i \leq 3$, $v_i(\varphi_a) = \mathbf{t}$ implies $v_i|^b_{\mathbf{t}}(\varphi_a) = \mathbf{t}$, it holds that (b, a) is a supporting link. Furthermore, since $v_4(\varphi_a) = \mathbf{f}$ but $v_4|^b_{\mathbf{t}}(\varphi_a) = \mathbf{t}$, link (b, a) is not an attack link. Moreover, since $v_3(a) = \mathbf{t}$ but $v_3|^c_{\mathbf{t}}(\varphi_a) = \mathbf{f}$, it holds that (c, a) is not a support link. However, it holds that $v_4(\varphi_a) = \mathbf{f}$ and $v_4|^c_{\mathbf{t}}(\varphi_a) = \mathbf{f}$. Thus, (c, a) is only an attacking link.

As an example for a link that is both attacking and supporting, consider $\varphi_b : a \vee \neg a$. There are two satisfying two-valued interpretations for the formula, i.e., $v_1 = \{a \mapsto \mathbf{t}\}$ and $v_2 = \{a \mapsto \mathbf{f}\}$. Since for i with $1 \leq i \leq 2$ it holds that $v_i(\varphi_b) = \mathbf{t}$ implies $v_i|^a_{\mathbf{t}}(\varphi_b) = \mathbf{t}$, it holds that (a, b) is a supporting link. Furthermore, since there is no two-valued interpretation that does not satisfy the formula, the link (a, b) is also an attacking link. Thus, (a, b) is a redundant link.

As an example for a link that is neither an attacking nor supporting, consider $\varphi_d : (\neg c \vee b) \wedge (c \vee \neg b)$. Let $v = \{b \mapsto \mathbf{f}, c \mapsto \mathbf{f}\}$ be a two-valued interpretation that satisfies the formula. However, $v|^b_{\mathbf{t}}(\varphi_d) = \mathbf{f}$. Thus, (b, d) is not a support link. Further, let $v = \{b \mapsto \mathbf{f}, c \mapsto \mathbf{t}\}$ be a two-valued interpretation that does not satisfy the formula. However, it holds that $v|^b_{\mathbf{t}}(\varphi_d) = \mathbf{t}$. Thus, (b, d) is not attacking. Hence (b, d) is a dependent link and the same reasoning applies to (c, d).

The classification of the types of the links of ADFs is also relevant for classifying ADFs themselves. One particularly important subclass of ADFs is that of *bipolar* ADFs or BADFs for short. In such an ADF each link is either attacking or supporting (or both; thus, the links can also be redundant). Another subclass of ADFs, having only attacking and no redundant links, is defined in Diller et al. (2020), called *support free ADFs* (SFADFs) in the current work, defined formally as follows.

Definition 2.11. An ADF is called support-free (SFADF for short) if it has no supporting links and all its links are attacking.

Notice that by the above definition SFADFs (a) allow for all kinds of attacks that can be formulated in ADFs and (b) do not contain neither redundant nor dependent links. This reflects the fact that dependent links become supporting as soon as the truth values of certain arguments have been fixed.
For SFADFs, it turns out that the intention of stable semantics, i.e. to avoid cyclic support among arguments, becomes immaterial, thus $mod(D) = stb(D)$ for any ADF D; the property is called weakly coherent in Keshavarzi Zafarghandi (2017), which however considers symmetric ADFs. In the following we generalize this observation to arbitrary SFADFs. In order to show that SFADFs are weakly coherent we have to introduce the following technical lemma.

Remark 1. To enhance the readability of this article, certain proofs have been relo-

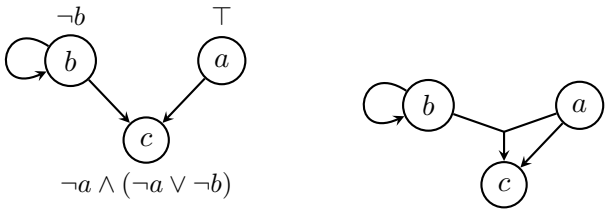

Figure 4. The SETADF of Example 3.2 and its associated SETAF

cated to Appendix A.

Lemma 2.12. *Let $D = (S, L, C)$ be an ADF, let v be a model of D and let $s \in S$ be an argument such that all parents of s are attackers. Then, φ_s^v is irrefutable if and only if $\varphi_s[p/\bot \ : \ v(p) = \mathbf{f}]$ is irrefutable.*

Proposition 2.13. *For every SFADF D it holds that $mod(D) = stb(D)$.*

3. Embedding SETAFs in ADFs

As observed by Polberg (2016) and Linsbichler et al. (2016), the notion of collective attacks can also be represented in ADFs by using the right acceptance conditions. We next introduce the class SETADFs of ADFs for this purpose.

Definition 3.1. An ADF $D = (S, L, C)$ is called SETAF-like (SETADF) if each of the acceptance conditions in C is given by a formula (with $\mathcal{C}$ a set of non-empty clauses)

$$\bigwedge_{cl \in \mathcal{C}} \bigvee_{a \in cl} \neg a.$$

That is, in a SETADF each acceptance condition is either $\top$ (if $\mathcal{C}$ is empty) or a proper CNF formula over negative literals.

Example 3.2. An instance of a SETADF is depicted in Figure 4 (on the left). Since argument a does not have any incoming link, i.e., $\mathcal{C} = \emptyset$, it holds that $\varphi_a : \top$. As we see, φ_b and φ_c are in CNF over negative literals.

SETADFs and SETAFs can be embedded in each other as follows. Definition 3.3 shows how a SETAF can be written as a SETADF.

Definition 3.3. Let $F = (A, R)$ be a SETAF. The ADF associated to F is a tuple $D_F = (S, L, C)$ in which $S = A$, $L = \{(a, b) \mid (B, b) \in R, a \in B\}$ and $C = \{\varphi_a\}_{a \in S}$ is the collection of acceptance conditions defined, for each $a \in S$, as

$$\varphi_a = \bigwedge_{(B,a) \in R} \bigvee_{a' \in B} \neg a'.$$

Definition 3.4 shows how a SETADF can directly be rewritten as a SETAF.

Definition 3.4. Let $D = (S, L, C)$ be a SETADF. We construct the associated SETAF $F_D = (A, R)$ in which, $A = S$, and R is constructed as follows. For

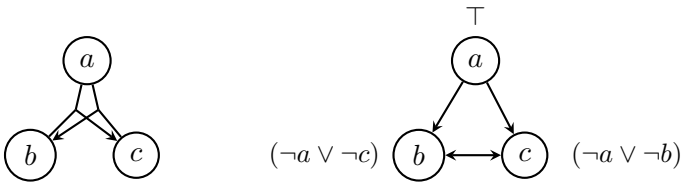

Figure 5. A SETAF (left) and its associated SETADF (right) and vice versa.

each argument $s \in S$ with acceptance formula $\bigwedge_{cl \in \mathcal{C}} \bigvee_{a \in cl} \neg a$ we add the attacks $\{(cl, s) \mid cl \in \mathcal{C}\}$ to R.

Example 3.5. First consider the SETADF of Example 3.2 illustrated in Figure 4 (on the left). When considering the associated SETAF (on the right of Figure 4) the acceptance condition $\neg a \wedge (\neg a \vee \neg b)$ of argument c corresponds to the two attacks $(\{a, b\}, c)$ and $(\{a\}, c)$. Next consider the SETAF of Example 2.5, i.e., $F = (\{a, b, c\}, \{(\{a, b\}, c), (\{a, c\}, b)\})$ as illustrated on the left of Figure 5. The ADF associated with F is $D_F = (A, L, C)$, where $\varphi_a : \top$ (since there is no B such that $(B, a) \in R$), $\varphi_b : \neg a \vee \neg c$ (since $(\{a, c\}, b) \in R$ in F), and $\varphi_c : \neg a \vee \neg b$ (since $(\{a, b\}, c) \in R$ in F) as illustrated at the right of Figure 5. Of course the SETAF associated with the constructed SETADF D_F is the original SETAF F.

Clearly the ADF D_F associated to a SETAF F is a SETADF and D is the ADF associated to the constructed SETAF F_D. We next deal with the fact that SETAF semantics are defined as three-valued labellings while semantics for ADFs are defined as three valued interpretations. In order to compare these semantics we associate the *in* label with t, the *out* label with f, and the *undec* label with u, presented formally in Definition 3.6.

Definition 3.6. The function $Lab2Int(\cdot)$ maps three-valued labellings to three-valued interpretations such that

- $Lab2Int(\lambda)(s) = \mathbf{t}$ iff $\lambda(s) = \texttt{in}$,
- $Lab2Int(\lambda)(s) = \mathbf{f}$ iff $\lambda(s) = \texttt{out}$, and
- $Lab2Int(\lambda)(s) = \mathbf{u}$ iff $\lambda(s) = \texttt{undec}$.

For a labelling λ and an interpretation I we write $\lambda \equiv I$ iff $Lab2Int(\lambda) = I$. For a set $\mathcal{L}$ of labellings and a set $\mathbb{V}$ of interpretations we write $\mathcal{L} \equiv \mathbb{V}$ iff $\{Lab2Int(\lambda) \mid \lambda \in \mathcal{L}\} = \mathbb{V}$.

Theorem 3.7. *For a SETAF F, its associated SETADF D and $\sigma \in \{cf, adm, com, prf, grd, stb\}$ we have $\sigma_{\mathcal{L}}(F) \equiv \sigma(D)$.*

Notice that by the above theorem we have that the 3-valued SETAF semantics introduced in Linsbichler et al. (2016) coincide with the 3-valued labelling based SETAF semantics of Flouris and Bikakis (2019) and the model semantics of Linsbichler et al. (2016) corresponds to the stable semantics of Flouris and Bikakis (2019).

4. 3-valued Signatures of SETAFs

In this section we investigate the expressiveness of 3-valued SETAF semantics. To provide exact characterisations of what can be expressed within a semantics, we investigate the expressiveness of SETAFs in terms of signatures. To this end, we adapt the concept of signatures Dunne et al. (2015) towards our needs. Intuitively, a signature for SETAFs is the sets of possible outcomes that can be achieved by SETAFs (of a particular class) under the different semantics, presented formally in Definition 4.1.

Definition 4.1. The signature of SETAFs under a labelling-based semantics $\sigma_{\mathcal{L}}$ is defined as $\Sigma^{\sigma_{\mathcal{L}}}_{\text{SETAF}} = \{\sigma_{\mathcal{L}}(F) \mid F \in \text{SETAF}\}$. The signature of an ADF-subclass $\mathcal{C}$ under a semantics σ is defined as $\Sigma^{\sigma}_{\mathcal{C}} = \{\sigma(D) \mid D \in \mathcal{C}\}$.

That is, a signature of a semantics is essentially a set of sets of labellings such that each of these sets of labellings corresponds to the evaluation of some SETAF under this semantics. We aim for compact and easy to test characterisations of the sets of labellings that are contained in the signature of a specific semantics. With such a characterisation we then can efficiently decide for a given set of labellings whether it can realized under a specific semantics and can also identify the sets that can be realized under one semantics but not under another semantics, i.e., identify the exact differences in the expressiveness of different semantics. By Theorem 3.7 we can use labellings of SETAFs and interpretations of the SETADF class of ADFs interchangeably, yielding that $\Sigma^{\sigma_{\mathcal{L}}}_{\text{SETAF}} \equiv \Sigma^{\sigma}_{SETADF}$, i.e. the 3-valued signatures of SETAFs and SETADFs only differ in the naming of the labels. For convenience, we will use the SETAF terminology in this section.

In Propositions 4.2– 4.10, we indicate the exact characterization of $\Sigma^{\sigma}_{\text{SETAF}}$, for $\sigma \in \{stb, prf, cf, nai, grd, adm\}$. We start with a characterisation of the signature for stable semantics.

Proposition 4.2. *The signature $\Sigma^{stb_{\mathcal{L}}}_{SETAF}$ is given by all sets $\mathbb{L}$ of labellings such that*

(1) all $\lambda \in \mathbb{L}$ have the same domain $\text{ARGS}_{\mathbb{L}}$; $\lambda(s) \neq$ `undec` for all $\lambda \in \mathbb{L}$, $s \in \text{ARGS}_{\mathbb{L}}$.
(2) If $\lambda \in \mathbb{L}$ assigns one argument to `out` *then it also assigns an argument to* `in`*.*
(3) For arbitrary $\lambda_1, \lambda_2 \in \mathbb{L}$ with $\lambda_1 \neq \lambda_2$ there is an argument a such that $\lambda_1(a) =$ `in` *and $\lambda_2(a) =$* `out`*.*

Proof. We first show that for each SETAF F the set $stb_{\mathcal{L}}(F)$ satisfies the conditions of the proposition. First clearly all $\lambda \in stb_{\mathcal{L}}(F)$ have the same domain and by the definition of stable semantics do not assign `undec` to any argument. That is the first condition is satisfied. For condition (2), towards a contradiction assume that the domain is non-empty and $\lambda \in stb_{\mathcal{L}}(F)$ assigns all arguments to `out`. Consider an arbitrary argument a. By definition of stable semantics a is only labeled `out` if there is an attack (B, a) such that all arguments in B are labeled `in`, a contradiction. Thus we obtain that there is at least one argument a with $\lambda(a) =$ `in`. For condition (3), towards a contradiction assume that for all arguments a with $\lambda_1(a) =$ `in` also $\lambda_2(a) =$ `in` holds. As $\lambda_1 \neq \lambda_2$ there is an a with $\lambda_2(a) =$ `in` and $\lambda_1(a) =$ `out`. That is, there is an attack (B, a) such that $\lambda_1(b) =$ `in` for all $b \in B$. But then also $\lambda_2(b) =$ `in` for all $b \in B$ and by $\lambda_2(a) =$ `in` we obtain that $\lambda_2 \notin cf_{\mathcal{L}}(F)$, a contradiction.

Now assume that $\mathbb{L}$ satisfies all the conditions of the proposition. We construct a

SETAF $F_{\mathbb{L}} = (A_{\mathbb{L}}, R_{\mathbb{L}})$ with

$$A_{\mathbb{L}} = \textsc{Args}_{\mathbb{L}}$$
$$R_{\mathbb{L}} = \{(\lambda_{\texttt{in}}, a) \mid \lambda \in \mathbb{L}, \lambda(a) = \texttt{out}\}$$

It remains to show that $stb_{\mathcal{L}}(F_{\mathbb{L}}) = \mathbb{L}$. To this end we first show $stb_{\mathcal{L}}(F_{\mathbb{L}}) \supseteq \mathbb{L}$. Consider an arbitrary $\lambda \in \mathbb{L}$: By condition (1) there is no $a \in \textsc{Args}_{\mathbb{L}}$ with $\lambda(a) = \texttt{undec}$ and it only remains to show $\lambda \in cf_{\mathcal{L}}(F_{\mathbb{L}})$. First, if $\lambda(a) = \texttt{out}$ for some argument a then by construction of $R_{\mathbb{L}}$ and condition (2) we have an attack $(\lambda_{\texttt{in}}, a)$ and thus a is legally labeled `out`.

Let us assume that $\lambda(a) = \texttt{in}$. We proceed with a proof by contradiction and suppose that $\lambda \notin cf_{\mathcal{L}}(F_{\mathbb{L}})$, meaning there exists a conflict (B, a) such that $B \cup \{a\} \subseteq \lambda_{\texttt{in}}$. Then, by construction of $R_{\mathbb{L}}$ there is a $\lambda' \in \mathbb{L}$ with $\lambda'_{\texttt{in}} = B$ and $\lambda_{\texttt{in}} \neq B$ (as $a \in \lambda_{\texttt{in}}$). That is, $\lambda'_{\texttt{in}} \subset \lambda_{\texttt{in}}$, a contradiction to condition (3). Thus, the assumption that there exists a conflict (B, a) such that $B \cup a \subseteq \lambda_{\texttt{in}}$ is incorrect. As a result, the assumption that $\lambda \notin cf_{\mathcal{L}}(F_{\mathbb{L}})$ is also incorrect. Hence, $\lambda \in cf_{\mathcal{L}}(F_{\mathbb{L}})$ and therefore $\lambda \in stb_{\mathcal{L}}(F_{\mathbb{L}})$.

To show $stb_{\mathcal{L}}(F_{\mathbb{L}}) \subseteq \mathbb{L}$, consider $\lambda \in stb_{\mathcal{L}}(F_{\mathbb{L}})$. If λ maps all arguments to `in` then there is no attack in $R_{\mathbb{L}}$ which means that $\mathbb{L}$ contains only the labelling λ. Thus, we assume that there is a with $\lambda(a) = \texttt{out}$ and there is $(B, a) \in R_{\mathbb{L}}$ with $B \subseteq \lambda_{\texttt{in}}$. By construction there is $\lambda' \in \mathbb{L}$ such that $\lambda'_{\texttt{in}} = B$. Moreover, as $\mathbb{L}$ labels all arguments either `in` or `out`, we have $(B, c) \in R_{\mathbb{L}}$ for all $c \notin B$ and thus $\lambda'_{\texttt{in}} = B = \lambda_{\texttt{in}}$ and moreover $\lambda'_{\texttt{out}} = \lambda_{\texttt{out}}$ and thus $\lambda = \lambda'$. □

We now turn to the signature for preferred semantics. Compared to the conditions for stable semantics, labelling may now assign `undec` to arguments. Note that stable is the only semantics allowing for an empty labelling set.

Proposition 4.3. *The signature $\Sigma^{prf_{\mathcal{L}}}_{SETAF}$ is given by all non-empty sets $\mathbb{L}$ of labellings s.t.*

(1) all labellings $\lambda \in \mathbb{L}$ have the same domain $\textsc{Args}_{\mathbb{L}}$.
(2) If $\lambda \in \mathbb{L}$ assigns one argument to `out` *then it also assigns an argument to* `in`*.*
(3) For arbitrary $\lambda_1, \lambda_2 \in \mathbb{L}$ with $\lambda_1 \neq \lambda_2$ there is an argument a such $\lambda_1(a) =$ `in` *and $\lambda_2(a) =$* `out`*.*

Proof. We first show that for each SETAF F the set $prf_{\mathcal{L}}(F)$ satisfies the conditions of the proposition.

(1) The first condition is satisfied as all $\lambda \in prf_{\mathcal{L}}(F)$ have the same domain.
(2) Assume that $\lambda \in prf_{\mathcal{L}}(F)$ assigns an argument a to `out`. By the definition of conflict-free labellings there is an attack (B, a) such that all arguments $b \in B$ are labeled `in`. Thus, the second condition is satisfied by the definition of conflict-free labellings.
(3) For condition (3), consider $\lambda, \lambda' \in prf_{\mathcal{L}}(F)$. Notice that there must be a conflict (S, a) with $S \cup \{a\} \subseteq \lambda_{\texttt{in}} \cup \lambda'_{\texttt{in}}$ as otherwise $(\lambda_{\texttt{in}} \cup \lambda'_{\texttt{in}}, \lambda_{\texttt{out}} \cup \lambda'_{\texttt{out}}, \lambda_{\texttt{undec}} \cap \lambda'_{\texttt{undec}})$ would be a larger admissible labelling. If $a \in \lambda'_{\texttt{in}}$ then, by the definition of admissible labellings, there is an attack (B, b) with $B \subseteq \lambda'_{\texttt{in}}$ and $b \in S \cap \lambda_{\texttt{in}}$. Thus b is an argument with $\lambda(b) = \texttt{in}$ and $\lambda'(b) = \texttt{out}$. Otherwise if $a \in \lambda_{\texttt{in}}$ then, by the definition of admissible labellings, there is an attack (B, b) with $B \subseteq \lambda_{\texttt{in}}$ and $b \in S \cap \lambda'_{\texttt{in}}$. Then, again by the definition of admissible labellings, there is an attack (C, c) with $C \subseteq \lambda'_{\texttt{in}}$ and $c \in B \subseteq \lambda_{\texttt{in}}$. Thus c is an argument

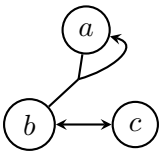

Figure 6. The SETAF of Example 4.4

with $\lambda(c) = \texttt{in}$ and $\lambda'(c) = \texttt{out}$.

Now assume that $\mathbb{L}$ satisfies all the conditions. We give a SETAF $F_{\mathbb{L}} = (A_{\mathbb{L}}, R_{\mathbb{L}})$ with $prf_{\mathcal{L}}(F_{\mathbb{L}}) = \mathbb{L}$. We use

$$A_{\mathbb{L}} = \textsc{Args}_{\mathbb{L}}$$
$$R_{\mathbb{L}} = \{(\lambda_{\texttt{in}}, a) \mid \lambda \in \mathbb{L}, \lambda(a) = \texttt{out}\} \cup \{(\lambda_{\texttt{in}} \cup \{a\}, a) \mid \lambda \in \mathbb{L}, \lambda(a) = \texttt{undec}\}$$

It remains to show that $prf_{\mathcal{L}}(F_{\mathbb{L}}) = \mathbb{L}$. We first show $prf_{\mathcal{L}}(F_{\mathbb{L}}) \supseteq \mathbb{L}$: Consider an arbitrary $\lambda \in \mathbb{L}$: We first show $\lambda \in cf_{\mathcal{L}}(F_{\mathbb{L}})$. We first consider out labeled arguments. First, if $\lambda(a) = \texttt{out}$ for some argument a then by construction and condition (2) we have an attack $(\lambda_{\texttt{in}}, a)$ and thus a is legally labeled out. Now towards a contradiction assume there is a conflict (B, a) such that $B \cup \{a\} \subseteq \lambda_{\texttt{in}}$.

If $|\mathbb{L}| = 1$, by the construction of $F_{\mathbb{L}}$ there is no $(B, a) \in R_{\mathbb{L}}$ such that $a \in \lambda_{\texttt{in}}$. That is, a is legally labeled in. If $|\mathbb{L}| > 1$, by construction there is a $\lambda' \in \mathbb{L}$ with $\lambda'_{\texttt{in}} = B \setminus \{a\}$, a contradiction to condition (3). Thus, $\lambda \in cf_{\mathcal{L}}(F_{\mathbb{L}})$. Next we show that $\lambda \in adm_{\mathcal{L}}(F_{\mathbb{L}})$. Consider an argument a with $\lambda(a) = \texttt{in}$ and an attack (B, a). Then, by construction there is a $\lambda' \in \mathbb{L}$ with $\lambda'_{\texttt{in}} = B \setminus \{a\}$ and, by condition (3), an argument $b \in B$ such that $\lambda(b) = \texttt{out}$. Thus, $\lambda \in adm_{\mathcal{L}}(F_{\mathbb{L}})$. Finally we show that $\lambda \in prf_{\mathcal{L}}(F_{\mathbb{L}})$. Towards a contradiction assume that there is a $\lambda' \in adm_{\mathcal{L}}(F_{\mathbb{L}})$ with $\lambda_{\texttt{in}} \subset \lambda'_{\texttt{in}}$. Let a be an argument such that $\lambda'(a) = \texttt{in}$ and $\lambda(a) \in \{\texttt{out}, \texttt{undec}\}$. By construction there is either an attack $(\lambda_{\texttt{in}}, a)$ or an attack $(\lambda_{\texttt{in}} \cup \{a\}, a)$. In both cases $\lambda' \notin adm_{\mathcal{L}}(F_{\mathbb{L}})$, a contradiction. Hence, $\lambda \in prf_{\mathcal{L}}(F_{\mathbb{L}})$.

We complete the proof by showing $prf_{\mathcal{L}}(F_{\mathbb{L}}) \subseteq \mathbb{L}$: Consider $\lambda \in prf_{\mathcal{L}}(F_{\mathbb{L}})$: If λ maps all arguments to in then there is no attack in $R_{\mathbb{L}}$ which means that $\mathbb{L}$ contains only the labelling λ. Thus we can assume that $\lambda(a) = \texttt{out}$ for some argument a and there is $(B, a) \in R_{\mathbb{L}}$ with $\lambda(b) = \texttt{in}$ for all $b \in B$. By construction there is $\lambda' \in \mathbb{L}$ such that $\lambda'_{\texttt{in}} = B$. Then by construction we have $(B, c) \in R_{\mathbb{L}}$ for all c with $\lambda'(c) = \texttt{out}$ and $(B \cup \{c\}, c) \in R_{\mathbb{L}}$ for all c with $\lambda'(c) = \texttt{undec}$. We obtain that $\lambda'_{\texttt{in}} = B = \lambda_{\texttt{in}}$ and thus $\lambda = \lambda'$. □

Example 4.4. Let $\mathbb{L} = \{\{a \mapsto \texttt{in}, b \mapsto \texttt{out}, c \mapsto \texttt{in}\}, \{a \mapsto \texttt{undec}, b \mapsto \texttt{in}, c \mapsto \texttt{out}\}\}$. It holds that $\mathbb{L} \in \Sigma^{prf_{\mathcal{L}}}_{\text{SETAF}}$ and $F = (\{a, b, c\}, \{(\{a, b\}, a), (\{c\}, b), (\{b\}, c)\})$, as depicted in Figure 6, is a witness of satisfying $\mathbb{L}$ as a preferred labelling in SETAFs, i.e., $prf(F) = \mathbb{L}$. However, $\mathbb{L} \notin \Sigma^{stb_{\mathcal{L}}}_{\text{SETAF}}$, since $\lambda(a) = \texttt{undec}$, where $\lambda = \{a \mapsto \texttt{undec}, b \mapsto \texttt{in}, c \mapsto \texttt{out}\}$. That is, λ does not satisfies the first item of Proposition 4.2.

Proposition 4.5. *The signature $\Sigma^{cf_{\mathcal{L}}}_{SETAF}$ is given by all non-empty sets $\mathbb{L}$ of labellings s.t.*

(1) all $\lambda \in \mathbb{L}$ have the same domain $\textsc{Args}_{\mathbb{L}}$.
(2) If $\lambda \in \mathbb{L}$ assigns one argument to out *then it also assigns an argument to* in*.*
(3) For $\lambda \in \mathbb{L}$ and $C \subseteq \lambda_{\texttt{in}}$ also $(C, \emptyset, \textsc{Args}_{\mathbb{L}} \setminus C) \in \mathbb{L}$.

(4) For $\lambda \in \mathbb{L}$ *and* $C \subseteq \lambda_{\texttt{out}}$ *also* $(\lambda_{\texttt{in}}, \lambda_{\texttt{out}} \setminus C, \lambda_{\texttt{undec}} \cup C) \in \mathbb{L}$.
(5) For $\lambda, \lambda' \in \mathbb{L}$ *with* $\lambda_{\texttt{in}} \subseteq \lambda'_{\texttt{in}}$ *also* $(\lambda'_{\texttt{in}}, \lambda_{\texttt{out}} \cup \lambda'_{\texttt{out}}, \lambda_{\texttt{undec}} \cap \lambda'_{\texttt{undec}}) \in \mathbb{L}$.
(6) For $\lambda, \lambda' \in \mathbb{L}$ *and* $C \subseteq \lambda_{\texttt{out}}$ *(s.t.* $C \neq \emptyset$*) we have* $\lambda_{\texttt{in}} \cup C \not\subseteq \lambda'_{\texttt{in}}$.

Proof. Let F be an arbitrary SETAF we show that $cf_{\mathcal{L}}(F)$ satisfies the conditions of the proposition.

The first condition is satisfied as clearly all $\lambda \in cf_{\mathcal{L}}(F)$ have the same domain. Now, assume that $\lambda \in cf_{\mathcal{L}}(F)$ assigns an argument a to `out`. By the definition of conflict-free labellings there is an attack (B, a) such that all arguments $b \in B$ are labeled `in`. Thus condition (2) is satisfied.

For condition (3), towards a contradiction assume that $(C, \emptyset, \textsc{Args}_{\mathbb{L}} \setminus C)$ is not conflict-free. Then there is an attack (B, a) such that $B \cup \{a\} \subseteq C$. But then also $B \cup \{a\} \subseteq \lambda_{\texttt{in}}$ and thus $\lambda \notin cf_{\mathcal{L}}(F)$, a contradiction.

Condition (4) is satisfied as in the definition of conflict-free labellings there are no conditions for labeling an argument `undec`. Further, the conditions that allow to label an argument `out` solely depend on the `in` labeled arguments. Since $\lambda_{\texttt{out}} \setminus C \subseteq \lambda_{\texttt{out}}$, the condition for arguments labeled `out` is satisfied.

For condition (5), consider $\lambda, \lambda' \in cf_{\mathcal{L}}(F)$ with $\lambda_{\texttt{in}} \subseteq \lambda'_{\texttt{in}}$ and $\lambda^* = (\lambda'_{\texttt{in}}, \lambda_{\texttt{out}} \cup \lambda'_{\texttt{out}}, \lambda_{\texttt{undec}} \cap \lambda'_{\texttt{undec}})$. First there cannot be an attack (B, a) such that $B \cup \{a\} \subseteq \lambda^*_{\texttt{in}}$ as $\lambda' \in cf_{\mathcal{L}}(F)$. Hence, $\lambda'_{\texttt{in}} \cap \lambda_{\texttt{out}} = \emptyset$ and thus λ^* is a well-defined labelling. Moreover, for each a with $\lambda^*(a) = \texttt{out}$ there is an attack (B, a) with $B \subseteq \lambda^*_{\texttt{in}}$ as either $\lambda(a) = \texttt{out}$ or $\lambda'(a) = \texttt{out}$. Thus, $\lambda^* \in cf_{\mathcal{L}}(F)$ and therefore the condition holds.

For condition (6), consider $\lambda, \lambda' \in cf_{\mathcal{L}}(F)$ and a set $C \subseteq \lambda_{\texttt{out}}$ containing an argument a such that $\lambda(a) = \texttt{out}$. That is, there is an attack (B, a) with $B \subseteq \lambda_{\texttt{in}}$ and thus $\lambda_{\texttt{in}} \cup C \not\subseteq \lambda'_{\texttt{in}}$. That is, condition (6) is satisfied.

Now assume that $\mathbb{L}$ satisfies all the conditions. We give a SETAF $F_{\mathbb{L}} = (A_{\mathbb{L}}, R_{\mathbb{L}})$ with

$$A_{\mathbb{L}} = \textsc{Args}_{\mathbb{L}}$$
$$R_{\mathbb{L}} = \{(\lambda_{\texttt{in}}, a) \mid \lambda \in \mathbb{L}, \lambda(a) = \texttt{out}\} \cup \{(B, b) \mid b \in B, \nexists \lambda \in \mathbb{L} : \lambda_{\texttt{in}} = B\}$$

We first show $cf_{\mathcal{L}}(F_{\mathbb{L}}) \supseteq \mathbb{L}$: Consider an arbitrary $\lambda \in \mathbb{L}$: First, if $\lambda(a) = \texttt{out}$ for some argument a then by construction and condition (2) we have an attack $(\lambda_{\texttt{in}}, a)$ and thus a is legally labeled `out`. Next, let us assume that $\lambda(a) = \texttt{in}$ and consider a proof by contradiction. Suppose there exists a conflict $(B, a) \in R_{\mathbb{L}}$ such that $B \cup \{a\} \subseteq \lambda_{\texttt{in}}$. By condition (3), it cannot be the case that $a \in B$. Therefore, there must be a $\lambda' \in \mathbb{L}$ such that $\lambda'_{\texttt{in}} = B$ due to the construction. This implies that $\lambda'_{\texttt{in}} \cup \{a\} \subseteq \lambda_{\texttt{in}}$, which contradicts condition (6). Thus, the assumption that $\lambda \notin cf_{\mathcal{L}}(F_{\mathbb{L}})$ is incorrect and we obtain $\lambda \in cf_{\mathcal{L}}(F_{\mathbb{L}})$.

We complete the proof by showing $cf_{\mathcal{L}}(F_{\mathbb{L}}) \subseteq \mathbb{L}$: Consider $\lambda \in cf_{\mathcal{L}}(F_{\mathbb{L}})$: If λ maps all arguments to `in` then there is no attack in $R_{\mathbb{L}}$ which means that $\mathbb{L}$ contains only the labelling λ. Thus we can assume that $\lambda(a) \in \{\texttt{out}, \texttt{undec}\}$ for some argument a. If $\lambda_{\texttt{in}} \neq \lambda'_{\texttt{in}}$ for all $\lambda' \in \mathbb{L}$ then by construction of the second part of $R_{\mathbb{L}}$ there would be attacks $(\lambda_{\texttt{in}}, b)$ for all $b \in \lambda_{\texttt{in}}$, which is in contradiction to $\lambda \in cf_{\mathcal{L}}(F_{\mathbb{L}})$. Thus, there is $\lambda' \in \mathbb{L}$ such that $\lambda'_{\texttt{in}} = \lambda_{\texttt{in}}$. For arguments a with $\lambda(a) = \texttt{out}$ there is an attack (B, a) with $B \subseteq \lambda_{\texttt{in}}$ and, by construction, a $\lambda^* \in \mathbb{L}$ such that $\lambda^*_{\texttt{in}} = B$ and $\lambda^*(a) = \texttt{out}$. By the existence of $\lambda' \in \mathbb{L}$ and condition (5) we have that there exists $\lambda'' \in \mathbb{L}$ such that $\lambda_{\texttt{in}} = \lambda''_{\texttt{in}}$, $\lambda'_{\texttt{out}} \subseteq \lambda''_{\texttt{out}}$ and $a \in \lambda''_{\texttt{out}}$. By iteratively applying this argument for each argument a with $\lambda(a) = \texttt{out}$ we obtain that there is a labelling $\hat{\lambda} \in \mathbb{L}$ such that $\lambda_{\texttt{in}} = \hat{\lambda}_{\texttt{in}}$ and $\lambda_{\texttt{out}} \subset \hat{\lambda}_{\texttt{out}}$. By condition (4) we obtain that $\lambda \in \mathbb{L}$. □

Example 4.6. Let $\mathbb{L} = \{\{a \mapsto \texttt{undec}, b \mapsto \texttt{undec}\}, \{a \mapsto \texttt{in}, b \mapsto \texttt{undec}\}, \{a \mapsto \texttt{undec}, b \mapsto \texttt{in}\}, \{a \mapsto \texttt{in}, b \mapsto \texttt{in}\}, \{a \mapsto \texttt{in}, b \mapsto \texttt{out}\}\}$. Labellings in $\mathbb{L}$ satisfy the four first items of Proposition 4.5. However, for $\lambda = \{a \mapsto \texttt{in}, b \mapsto \texttt{out}\}$ and $\lambda' = \{a \mapsto \texttt{in}, b \mapsto \texttt{in}\}$ the fifth and the sixth items does not hold. Thus, $\mathbb{L} \notin \Sigma^{cf_{\mathcal{L}}}_{\text{SETAF}}$. However, $\mathbb{L}$ can be realized by the conflict-free set of ADF $D = (\{a, b\}, \{\varphi_a : \top, \varphi_b : \neg a \vee b\})$, which is not a SETADF.

We next turn to naive semantics. For extension-based semantics we have that conflict-free sets fully determine naive extensions and vice versa. For labelling-based semantics only the former is true as the `out` labels of conflict-free subsets are not determined by the `out` labels of the naive labellings.

Proposition 4.7. *The signature $\Sigma^{nai_{\mathcal{L}}}_{SETAF}$ is given by all sets $\mathbb{L}$ of labellings such that*

(1) all $\lambda \in \mathbb{L}$ have the same domain $\textsc{Args}_{\mathbb{L}}$.
(2) if $\lambda \in \mathbb{L}$ assigns one argument to `out` *then it also assigns an argument to* `in`*.*
(3) for $\lambda \in \mathbb{L}$ and $C \subseteq \lambda_{\texttt{out}}$ also $(\lambda_{\texttt{in}}, \lambda_{\texttt{out}} \setminus C, \lambda_{\texttt{undec}} \cup C) \in \mathbb{L}$
(4) for $\lambda, \lambda' \in \mathbb{L}$ with $\lambda_{\texttt{in}} = \lambda'_{\texttt{in}}$ also $(\lambda_{\texttt{in}}, \lambda_{\texttt{out}} \cup \lambda'_{\texttt{out}}, \lambda_{\texttt{undec}} \cap \lambda'_{\texttt{undec}}) \in \mathbb{L}$.
(5) for arbitrary $\lambda, \lambda' \in \mathbb{L}$ we have $\lambda_{\texttt{in}} \not\subset \lambda'_{\texttt{in}}$.

Proof. Let F be an arbitrary SETAF. First we show that $nai_{\mathcal{L}}(F)$ satisfies the conditions of the proposition. Conditions (1)-(3) are by the fact that $nai_{\mathcal{L}}(F) \subseteq cf_{\mathcal{L}}(F)$. For condition (4), consider $\lambda, \lambda' \in nai_{\mathcal{L}}(F)$ with $\lambda_{\texttt{in}} = \lambda'_{\texttt{in}}$. We know that for each $a \in \lambda_{\texttt{out}} \cup \lambda'_{\texttt{out}}$ there is an attack (B, a) with $B \subseteq \lambda_{\texttt{in}}$. Thus also $(\lambda_{\texttt{in}}, \lambda_{\texttt{out}} \cup \lambda'_{\texttt{out}}, \lambda_{\texttt{undec}} \cap \lambda'_{\texttt{undec}}) \in nai_{\mathcal{L}}(F)$. Finally condition (5) is satisfied by the maximality of $\lambda_{\texttt{in}}$ in naive labellings.

Now assume that $\mathbb{L}$ satisfies all the conditions. We give a SETAF $F_{\mathbb{L}} = (A_{\mathbb{L}}, R_{\mathbb{L}})$ with

$$A_{\mathbb{L}} = \textsc{Args}_{\mathbb{L}}$$
$$R_{\mathbb{L}} = \{(\lambda_{\texttt{in}}, a) \mid \lambda \in \mathbb{L}, \lambda(a) = \texttt{out}\} \cup \{(\lambda_{\texttt{in}} \cup \{a\}, a) \mid \lambda \in \mathbb{L}, \lambda(a) = \texttt{undec}\}.$$

To complete the proof we show that $nai_{\mathcal{L}}(F_{\mathbb{L}}) = \mathbb{L}$. We first show $nai_{\mathcal{L}}(F_{\mathbb{L}}) \supseteq \mathbb{L}$: Consider an arbitrary $\lambda \in \mathbb{L}$: We first show $\lambda \in cf_{\mathcal{L}}(F_{\mathbb{L}})$. First, if $\lambda(a) = \texttt{out}$ for some argument a then by construction and condition (2) we have an attack $(\lambda_{\texttt{in}}, a)$ and thus a is legally labeled `out`. Now towards a contradiction assume there is a conflict (B, a) such that $B \cup \{a\} \subseteq \lambda_{\texttt{in}}$. If $|\mathbb{L}| > 1$, then, by construction there is a $\lambda' \in \mathbb{L}$ with $\lambda'_{\texttt{in}} = B \setminus \{a\}$, a contradiction to (5). Thus, $\lambda \in cf_{\mathcal{L}}(F_{\mathbb{L}})$. Finally we show that $\lambda \in nai_{\mathcal{L}}(F_{\mathbb{L}})$. Towards a contradiction assume that there is a $\lambda' \in cf_{\mathcal{L}}(F_{\mathbb{L}})$ with $\lambda_{\texttt{in}} \subset \lambda'_{\texttt{in}}$. Let a be an argument such that $\lambda'(a) = \texttt{in}$ and $\lambda(a) \in \{\texttt{out}, \texttt{undec}\}$. By construction there is either an attack $(\lambda_{\texttt{in}}, a)$ or an attack $(\lambda_{\texttt{in}} \cup \{a\}, a)$. In both cases $\lambda' \notin cf_{\mathcal{L}}(F_{\mathbb{L}})$ a contradiction. Hence, $\lambda \in nai_{\mathcal{L}}(F_{\mathbb{L}})$.

We complete the proof by showing $nai_{\mathcal{L}}(F_{\mathbb{L}}) \subseteq \mathbb{L}$: Consider $\lambda \in nai_{\mathcal{L}}(F_{\mathbb{L}})$: If λ maps all arguments to `in` then there is no attack in $R_{\mathbb{L}}$ which means that $\mathbb{L}$ contains only the labelling λ. Thus we can assume that $\lambda(a) \in \{\texttt{out}, \texttt{undec}\}$ for some argument a and there is $(B, a) \in R_{\mathbb{L}}$ with $B \subseteq \lambda_{\texttt{in}} \cup \{a\}$. By construction there is $\lambda' \in \mathbb{L}$ such that $\lambda'_{\texttt{in}} = B \setminus \{a\}$. By the above $\lambda' \in nai_{\mathcal{L}}(F_{\mathbb{L}})$ and thus $\lambda = \lambda'_{\texttt{in}}$ (cf. condition (5)). Moreover, for each argument b with $\lambda(b) = \texttt{out}$, by construction, we have a $\lambda^b \in \mathbb{L}$

with $\lambda^b_{\mathtt{in}} = \lambda_{\mathtt{in}}$ and $\lambda^b(b) = \mathtt{out}$. Let us next define the labelling

$$\lambda^* = (\lambda'_{\mathtt{in}}, \lambda'_{\mathtt{out}} \cup \bigcup_{b \in \lambda_{\mathtt{out}}} \lambda^b_{\mathtt{out}}, \lambda'_{\mathtt{undec}} \cap \bigcap_{b \in \lambda_{\mathtt{out}}} \lambda^b_{\mathtt{undec}}).$$

By condition (4) we have that $\lambda^* \in \mathbb{L}$. By the construction of λ^* we have $\lambda_{\mathtt{out}} \subseteq \lambda^*_{\mathtt{out}}$ and $\lambda_{\mathtt{in}} = \lambda^*_{\mathtt{in}}$. Thus, by condition (3), $\lambda \in \mathbb{L}$. □

Example 4.8. Let $\mathbb{L} = \{\{a \mapsto \mathtt{undec}, b \mapsto \mathtt{undec}\}, \{a \mapsto \mathtt{in}, b \mapsto \mathtt{undec}\}, \{a \mapsto \mathtt{undec}, b \mapsto \mathtt{in}\}, \{a \mapsto \mathtt{in}, b \mapsto \mathtt{out}\}\}$. $\mathbb{L}$ satisfied all conditions of Proposition 4.5, thus $\mathbb{L} \in \Sigma^{cf_{\mathcal{L}}}_{\text{SETAF}}$. However, the last item of Proposition 4.7 does not holds for $\mathbb{L}$, since for $\lambda = \{a \mapsto \mathtt{undec}, b \mapsto \mathtt{undec}\}$ and $\lambda' = \{a \mapsto \mathtt{in}, b \mapsto \mathtt{undec}\}$ it holds that $\lambda_{\mathtt{in}} \subset \lambda'_{\mathtt{in}}$. Now consider $\mathbb{L}' = \mathbb{L} \setminus \{\{a \mapsto \mathtt{undec}, b \mapsto \mathtt{undec}\}\}$. This set is realized by the SETAF $F_{\mathbb{L}'} = (\{a, b\}, \{(\{a\}, b), (\{a, b\}, b), (\{a, b\}, a)\})$, under naive semantics. Notice that the attacks $(\{a, b\}, b), (\{a, b\}, a)$ of $F_{\mathbb{L}'}$ are indeed redundant for naive semantics and can be removed without changing the naive extensions.

Finally, we give an exact characterisation of the signature of grounded semantics.

Proposition 4.9. *The signature $\Sigma^{grd_{\mathcal{L}}}_{SETAF}$ is given by sets $\mathbb{L}$ of labellings such that $|\mathbb{L}| = 1$, and for the unique labelling $\lambda \in \mathbb{L}$ we have that if $\lambda_{\mathtt{out}} \neq \emptyset$ then also $\lambda_{\mathtt{in}} \neq \emptyset$.*

Proof. We first show that for each SETAF F the set $grd_{\mathcal{L}}(F)$ satisfies the conditions of the proposition. Let $\lambda \in grd_{\mathcal{L}}(F)$ be the unique grounded labelling, which by definition is complete and thus conflict-free. Now, assume that λ assigns an argument a to $\mathtt{out}$. By the definition of conflict-free labellings there is an attack (B, a) such that $B \subseteq \lambda_{\mathtt{in}}$.

Now assume that $\mathbb{L}$ satisfies all the conditions. We give a SETAF $F_{\mathbb{L}} = (A_{\mathbb{L}}, R_{\mathbb{L}})$ with $grd_{\mathcal{L}}(F_{\mathbb{L}}) = \mathbb{L}$.

$$A_{\mathbb{L}} = \textsc{Args}_{\mathbb{L}}$$
$$R_{\mathbb{L}} = \{(\lambda_{\mathtt{in}}, a) \mid \lambda \in \mathbb{L}, \lambda(a) = \mathtt{out}\} \cup \{(\lambda_{\mathtt{in}} \cup \{a\}, a) \mid \lambda \in \mathbb{L}, \lambda(a) = \mathtt{undec}\}$$

Consider the unique $\lambda \in \mathbb{L}$ and the unique $\lambda^G \in grd_{\mathcal{L}}(F_{\mathbb{L}})$. For each argument $a \in \lambda_{\mathtt{in}}$ we have that a is not attacked in $F_{\mathbb{L}}$ and thus $a \in \lambda^G_{\mathtt{in}}$. For each argument $a \in \lambda_{\mathtt{out}}$ there is an attack $(\lambda_{\mathtt{in}}, a)$ in $F_{\mathbb{L}}$ and as $\lambda_{\mathtt{in}} \subseteq \lambda^G_{\mathtt{in}}$ by the definition of complete labellings we have $a \in \lambda^G_{\mathtt{out}}$. Finally for each argument $a \in \lambda_{\mathtt{undec}}$ the attack $(\lambda_{\mathtt{in}} \cup \{a\}, a)$ is the only attack towards a in $F_{\mathbb{L}}$. Thus, by the definition of complete labellings, we have that a is neither labelled $\mathtt{in}$ nor $\mathtt{out}$ in $F_{\mathbb{L}}$ and therefore $a \in \lambda^G_{\mathtt{undec}}$. We obtain that $\lambda^G = \lambda$ and thus $grd_{\mathcal{L}}(F_{\mathbb{L}}) = \mathbb{L}$. □

Notice that Proposition 4.9 basically exploits that grounded semantics is a unique status semantics based on admissibility. The result thus immediately extends to other SETAF semantics satisfying these two properties, e.g. to ideal or eager semantics (Flouris & Bikakis, 2019).

So far, we have provided characterisations for the signatures $\Sigma^{stb_{\mathcal{L}}}_{\text{SETAF}}$, $\Sigma^{prf_{\mathcal{L}}}_{\text{SETAF}}$, $\Sigma^{cf_{\mathcal{L}}}_{\text{SETAF}}$, $\Sigma^{nai_{\mathcal{L}}}_{\text{SETAF}}$, $\Sigma^{grd_{\mathcal{L}}}_{\text{SETAF}}$. By Theorem 3.7 we get analogous characterizations of $\Sigma^{\sigma}_{\text{SETADF}}$ for the corresponding ADF semantics.

We have not yet touched admissible and complete semantics. Here, the exact characterisations seem to be more cumbersome and are left for future work. However, for admissible semantics the following proposition provides necessary conditions for an

labelling-set to be *adm*-realizable, but it remains open whether they are also sufficient.

Proposition 4.10. *For each* $\mathbb{L} \in \Sigma^{adm_c}_{SETAF}$ *we have:*

(1) all $\lambda \in \mathbb{L}$ *have the same domain* $\textsc{Args}_{\mathbb{L}}$.
(2) If $\lambda \in \mathbb{L}$ *assigns one argument to* `out` *then it also assigns an argument to* `in`.
(3) For $\lambda, \lambda' \in \mathbb{L}$ *and* $C \subseteq \lambda_{\texttt{out}}$ *(s.t.* $C \neq \emptyset$*) we have* $\lambda_{\texttt{in}} \cup C \not\subseteq \lambda'_{\texttt{in}}$.
(4) For arbitrary $\lambda, \lambda' \in \mathbb{L}$ *either (a)* $(\lambda_{\texttt{in}} \cup \lambda'_{\texttt{in}}, \lambda_{\texttt{out}} \cup \lambda'_{\texttt{out}}, \lambda_{\texttt{undec}} \cap \lambda'_{\texttt{undec}}) \in \mathbb{L}$ *or (b) there is an argument* a *such* $\lambda(a) = \texttt{in}$ *and* $\lambda'(a) = \texttt{out}$.
(5) For $\lambda, \lambda' \in \mathbb{L}$ *with* $\lambda_{\texttt{out}} \subseteq \lambda'_{\texttt{out}}$*, and* $C \subseteq \lambda_{\texttt{in}} \setminus \bigcup_{\lambda^* \in \mathbb{L}:\ \lambda^*_{\texttt{in}} = \lambda'_{\texttt{in}}} \lambda^*_{\texttt{out}}$ *we have* $(\lambda'_{\texttt{in}} \cup C, \lambda'_{\texttt{out}}, \lambda'_{\texttt{undec}} \setminus C) \in \mathbb{L}$.
(6) For $\lambda, \lambda' \in \mathbb{L}$ *with* $\lambda_{\texttt{in}} \subseteq \lambda'_{\texttt{in}}$*, and* $C \subseteq \lambda_{\texttt{out}}$ *we have* $(\lambda'_{\texttt{in}}, \lambda'_{\texttt{out}} \cup C, \lambda'_{\texttt{undec}} \setminus C) \in \mathbb{L}$.
(7) For $\lambda, \lambda' \in \mathbb{L}$ *with* $\lambda_{\texttt{in}} \subseteq \lambda'_{\texttt{in}}$ *and* $\lambda_{\texttt{out}} \supseteq \lambda'_{\texttt{out}}$ *we have* $(\lambda_{\texttt{in}}, \lambda'_{\texttt{out}}, \textsc{Args}_{\mathbb{L}} \setminus (\lambda_{\texttt{in}} \cup \lambda'_{\texttt{out}})) \in \mathbb{L}$.
(8) $(\emptyset, \emptyset, \textsc{Args}_{\mathbb{L}}) \in \mathbb{L}$.

5. On the Relation between SETAFs and Support-Free ADFs

In order to compare SETAFs with SFADFs, we can rely on SETADFs (recall Theorem 3.7). In particular, we will compare the signatures $\Sigma^{\sigma}_{\text{SETADF}}$ and $\Sigma^{\sigma}_{\text{SFADF}}$, cf. Definition 4.1. We start with the observation that each SETADF can be rewritten as an equivalent SETADF that is also a SFADF.[2]

Lemma 5.1. *For each SETADF* $D = (S, L, C)$ *there is an equivalent SETADF* $D' = (S, L', C')$ *that is also a SFADF, i.e. for each* $s \in S$, $\varphi_s \in C$, $\varphi'_s \in C'$ *we have* $\varphi_s \equiv \varphi'_s$.

Proof. Given a SETADF D, by Definition 3.1, each acceptance condition is a CNF over negative literals and thus does not have any support link which is not redundant. We can thus obtain L' by removing the redundant links from L and C' by, in each acceptance condition, deleting the clauses that are super-sets of other clauses. □

Example 5.2 shows that there exists a SETADF which is not a SFADF, which shows the importance of Lemma 5.1.

Example 5.2. Let $D = (\{a, b, c\}, \{\varphi_a : \top, \varphi_b : \top, \varphi_c : \neg a \wedge (\neg a \vee \neg b)\})$. By Definition 3.1, D is a SETADF. We show that link (b, c) is a redundant link. Thus, by Definition 2.11, D is not a SFADF, since it contains a redundant link.

Here we aim to show that (b, c) is a redundant link. Let $v = \{a \mapsto \mathbf{f}, b \mapsto \mathbf{t}\}$ and $v' = \{a \mapsto \mathbf{f}, b \mapsto \mathbf{f}\}$. It holds that $v(\varphi_c) = v'(\varphi_c) = \mathbf{t}$, $v|^b_{\mathbf{t}}(\varphi_c) = v'|^b_{\mathbf{t}}(\varphi_c) = \mathbf{t}$, and there is no other two-valued interpretation over the parents of c that assigns c to $\mathbf{t}$. Therefore, by Definition 2.9, (b, c) is a supporting link.

Furthermore, let $v = \{a \mapsto \mathbf{t}, b \mapsto \mathbf{t}\}$ and $v' = \{a \mapsto \mathbf{t}, b \mapsto \mathbf{f}\}$. v and v' are two-valued interpretations over the parents of c that indicate (b, c) is an attacking link. Hence, by Definition 2.9, (b, c) is a redundant link.

In φ_c clause $(\neg a \vee \neg b)$ is a super-set of clause $\neg a$. Thus, we remove $(\neg a \vee \neg b)$ from the acceptance condition of c, as it is presented in the proof of Lemma 5.1. Thus, $D' = (\{a, b, c\}, \{\varphi_a : \top, \varphi_b : \top, \varphi_c : \neg a\})$ is a SETADF equivalent with D, which is also a SFADF.

[2] As discussed in Polberg (2017), in general, SETAFs translate to bipolar ADFs that contain attacking and redundant links. However, when we first remove redundant attacks from the SETAF we obtain a SFADF.

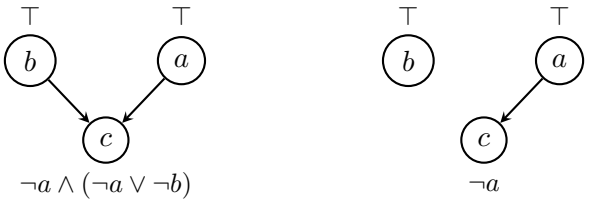

Figure 7. Left: D is a SETADF but not a SFADF, right: SFADF/SETADF D' equivalent with D, in Example 5.2

By Lemma 5.1 we have that $\Sigma^{\sigma}_{\text{SETADF}} \subseteq \Sigma^{\sigma}_{\text{SFADF}}$. Now consider the interpretation $v = \{a \mapsto \mathbf{f}\}$. We have that for all considered semantics σ, v is a σ-interpretation of the SFADF $D = (\{a\}, \{\varphi_a = \bot\})$ but there is no SETADF with v being a σ-interpretation. We thus obtain $\Sigma^{\sigma}_{\text{SETADF}} \subset \Sigma^{\sigma}_{SFADF}$.

Theorem 5.3. *$\Sigma^{\sigma}_{SETADF} \subset \Sigma^{\sigma}_{SFADF}$, for $\sigma \in \{cf, adm, stb, mod, com, prf, grd\}$.*

In the remainder of this section we aim to characterise the difference between $\Sigma^{\sigma}_{\text{SETADF}}$ and Σ^{σ}_{SFADF}. To this end, in Lemma 5.4 we first recall a characterisation of the acceptance conditions of SFADF that can be rewritten as collective attacks.

Lemma 5.4. *Wallner (2019) Let $D = (S, L, C)$ be a SFADF. If $s \in S$ has at least one incoming link then the acceptance condition φ_s can be written in CNF containing only negative literals.*

It remains to consider those arguments in an SFADF with no incoming links. Such arguments allow for only two acceptance conditions $\top$ and $\bot$. While condition $\top$ is unproblematic (it refers to an initial argument in a SETAF), an argument with unsatisfiable acceptance condition cannot be modeled in a SETADF. In fact, the different expressiveness of SETADFs and SFADFs is solely rooted in the capability of SFADFs to set an argument to $\mathbf{f}$ via a $\bot$ acceptance condition.

Lemma 5.5, by considering interpretation-sets, investigates the condition under which the acceptance condition of an argument that is equal to $\bot$ cannot be replaced by collective attacks. Later we use Lemma 5.5 in the proof of Theorem 5.6 to indicate which interpretation-set is not realizable in SETADF.

Lemma 5.5. *Given an interpretation-set $\mathbb{V} \in \Delta_\sigma$, for $\sigma \in \{adm, stb, mod, com, prf, grd\}$, such that there exists $v \in \mathbb{V}$, where $v \neq v_{\mathbf{u}}$ and $v(a) = \mathbf{f}/\mathbf{u}$, for each argument a. In all SFADFs that realize $\mathbb{V}$ under σ, the acceptance conditions of all arguments assigned to $\mathbf{f}$ by v are equal to $\bot$.*

Proof. Let D be a SFADF that realizes $\mathbb{V}$ under σ, for $\sigma \in \{adm, stb, mod, com, prf, grd\}$. Let $v \in \mathbb{V}$ be an non-trivial interpretation that assigns all arguments either to $\mathbf{f}$ or $\mathbf{u}$. Towards a contradiction, assume that there exists an argument a which is assigned to $\mathbf{f}$ by v, and $\varphi_a \neq \bot$ in D. First we show that $\mathbb{V}$ cannot be *adm*-realizable in SFADFs. Since a is assigned to $\mathbf{f}$ in v the acceptance condition of a cannot be equal to $\top$. If φ_a is neither $\top$, nor $\bot$, then it has an incoming link. Thus, by Lemma 5.4, the acceptance condition of a is in CNF and having only negative literals. Since all $b \in par(a)$ are either assigned to $\mathbf{f}$ or $\mathbf{u}$ by v, φ^v_a cannot be unsatisfiable. That is, $v(a) \not\leq_i \Gamma_D(v)(a)$. Therefore, v is not an admissible interpretation of D. Thus, any $\mathbb{V}$ that contains v is not *adm*-realizable in SFADF. To complete the proof it remains to see

that for each of the remaining semantics, each σ-interpretation is also admissible. $\square$

We next give a generic characterisations of the difference between $\Sigma^{\sigma}_{\text{SETADF}}$ and Σ^{σ}_{SFADF}.

Theorem 5.6. *For $\sigma \in \{cf, adm, stb, mod, com, prf, grd\}$, we have $\Delta_\sigma = \Sigma^{\sigma}_{SFADF} \setminus \Sigma^{\sigma}_{SETADF}$ with*

$$\Delta_\sigma = \{\mathbb{V} \in \Sigma^{\sigma}_{SFADF} \mid \exists v \in \mathbb{V} \text{ s.t. } \forall a : v(a) \in \{\mathbf{f}, \mathbf{u}\} \wedge \exists a : v(a) = \mathbf{f}\}.$$

Proof. First we show that $\Delta_\sigma \subseteq \Sigma^{\sigma}_{\text{SFADF}} \setminus \Sigma^{\sigma}_{\text{SETADF}}$. To this end, we use the facts shown in the previous sections. Let $\mathbb{L}$ be a σ-labelling for $\sigma \in \{cf, adm, stb, mod, com, prf, grd\}$. By Propositions 4.2-4.10, if $\lambda \in \mathbb{L}$ assigns one argument to `out` then, it also assigns an argument to `in`. Furthermore, by Theorem 3.7, labellings of SETAFs and interpretations of SETADFs can be used interchangeably. Thus, for $\mathbb{V} \in \Delta_\sigma$ the interpretation v cannot be realized in a SETADF as we cannot have $v(a) \in \mathbf{f}$ without $v(b) \in \mathbf{t}$ for some other argument b. Furthermore, by the definition of Δ_σ, it holds that $\mathbb{V} \in \Sigma^{\sigma}_{\text{SFADF}}$. That is, $\Delta_\sigma \subseteq \Sigma^{\sigma}_{\text{SFADF}} \setminus \Sigma^{\sigma}_{\text{SETADF}}$.

On the other hand, we show that $\Sigma^{\sigma}_{\text{SFADF}} \setminus \Sigma^{\sigma}_{\text{SETADF}} \subseteq \Delta_\sigma$ or equivalently $\Sigma^{\sigma}_{\text{SFADF}} \setminus \Delta_\sigma \subseteq \Sigma^{\sigma}_{\text{SETADF}}$. To this end, consider $\mathbb{V} \in \Sigma^{\sigma}_{\text{SFADF}}$ such that each $v \in \mathbb{V}$ assigns some argument to $\mathbf{t}$. We show that one can construct a SETADF D_F with $\sigma(D_F) = \mathbb{V}$.

- Let $\sigma = grd$. Assume that $F = (S, L, C)$ is a SFADF such that $\mathbb{V} = grd(F)$. Thus, $|\mathbb{V}| = 1$. Let $v \in \mathbb{V}$. We construct SETADF $D_F = (S, L', C')$, such that $\mathbb{V} = grd(D_F)$, where C' is a collection of φ'_a as follows.

$$\varphi'_a = \begin{cases} \top & \text{if } v(a) = \mathbf{t} \\ \bigwedge_{b \in v^{\mathbf{t}}} \neg b & \text{if } v(a) = \mathbf{f} \\ \neg a & \text{otherwise.} \end{cases}$$

  The second item of φ'_a in the above definition is well-defined, since we assume that $v^{\mathbf{t}} \neq \emptyset$. By Definition 3.1 it is clear that D_F is a SETADF and it is easy to check that $grd(D_F) = \mathbb{V}$.
- Let $\sigma \in \{cf, adm\}$. Assume that $F = (S, L, C)$ is a SFADF such that $\mathbb{V} = adm(F)$ ($\mathbb{V} = cf(F)$, respectively). F does not have any argument with $\bot$ acceptance condition. Otherwise, there exists a $v \in adm(F)$ ($\mathbb{V} = cf(F)$, respectively) that does not assign any argument to $\mathbf{t}$. This is a contradiction by the assumption that each $v \in \mathbb{V}$ assigns at least an argument to $\mathbf{t}$. All the other acceptance conditions can be rewritten in the associated SETADF D_F via Lemma 5.4.
- Let $\sigma = com$. Let F be a SFADF such that $com(F) = \mathbb{V}$. Let g be the grounded interpretation of F. If $g(a) \in \{\mathbf{t}, \mathbf{f}\}$, then let φ'_a as introduced in the first item of the proof. Note that if there exists a such that $\varphi_a : \bot$, then a assigns to $\mathbf{f}$ in the grounded interpretation F. All the other acceptance conditions in D_F can be rewritten via Lemma 5.4.
- For $\sigma \in \{stb, mod, prf\}$ assume there are arguments $s_1, \ldots, s_\ell$ with acceptance condition $\bot$ and thus s_i is denied by any $v_i \in \mathbb{V}$. For each $v_i \in \mathbb{V}$ let b_i be an argument such that $v_i(b_i) = \mathbf{t}$. We construct a SETADF $D_F = (S, L', C')$ such

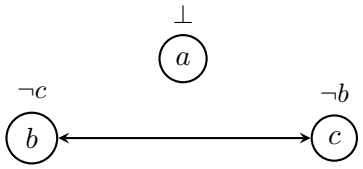

Figure 8. ADF of Example 5.8

that C' is a collection of φ'_a as follows.

$$\varphi'_a = \begin{cases} \varphi_a & \text{if } \varphi_a \neq \bot \\ \neg a \wedge \bigwedge_{v_i \in \mathbb{V}} \neg b_i & \text{otherwise.} \end{cases}$$

It is now easy to verify that $\mathbb{V} = \sigma(D_F)$ and as, by construction D_F, has no argument with acceptance condition $\bot$, by Lemma 5.4, $\mathbb{V}$ is σ-realizable in SETADFs.

That is for all semantics under our considerations we have $\Sigma^{\sigma}_{\text{SFADF}} \setminus \Sigma^{\sigma}_{\text{SETADF}} \subseteq \Delta_{\sigma}$ and together with the above we obtain $\Sigma^{\sigma}_{\text{SFADF}} \setminus \Sigma^{\sigma}_{\text{SETADF}} = \Delta_{\sigma}$. □

Next, we provide stronger characterisations of Δ_{σ} for preferred and stable semantics.

Proposition 5.7. *For $\mathbb{V} \in \Delta_{\sigma}$ and $\sigma \in \{stb, mod, prf\}$ we have $|\mathbb{V}| = 1$. For $\sigma \in \{stb, mod\}$ the unique $v \in \mathbb{V}$ assigns all arguments to* **f**.

Proof. If a SFADF has a σ-interpretation v that assigns some arguments to **f** without assigning an argument to **t** then, by Lemma 5.5, we have that the arguments assigned to **f** are exactly the arguments with acceptance condition $\bot$. Since stable and two-valued models are two-valued interpretations, this means all arguments have acceptance condition $\bot$ and the result follows. Each preferred interpretation assigns arguments with acceptance condition $\bot$ to **f** and thus the existence of another preferred interpretation would violate the $\leq_i$-maximality of v. □

In other words each interpretation-set which is σ-realizable in SFADFs and contains at least two interpretations can be realized in SETADFs, for $\sigma \in \{stb, prf, mod\}$. We close this section with an example illustrating that the above characterisation thus not hold for *cf*, *adm*, and *com*.

Example 5.8. Let $D = (\{a, b, c\}, \{\varphi_a = \bot, \varphi_b = \neg c, \varphi_c = \neg b\})$, depicted in Figure 8. We have $com(D) = \{\{a \mapsto \mathbf{f}, b \mapsto \mathbf{u}, c \mapsto \mathbf{u}\}, \{a \mapsto \mathbf{f}, b \mapsto \mathbf{t}, c \mapsto \mathbf{f}\}, \{a \mapsto \mathbf{f}, b \mapsto \mathbf{f}, c \mapsto \mathbf{t}\}\}$. By Theorem 5.6, $com(D)$ cannot be realized as SETADF. Moreover, as $com(D) \subseteq adm(D) \subseteq cf(D)$ for every ADF D, we have that, despite all three contain more than one interpretation, none of them can be realized via a SETADF.

In this section we obtained that each SETADF can be rewritten as SFADF if redundant links/attacks are removed and that there are SFADFs that cannot be rewritten as SETADFs. We then studied the exact difference an identified arguments with unsatisfiable acceptance condition as the source of this difference. The relation among the classes Σ_{SETAF}, Σ_{SETADF}, and Σ_{SFADF} is illustrated in Figure 9.

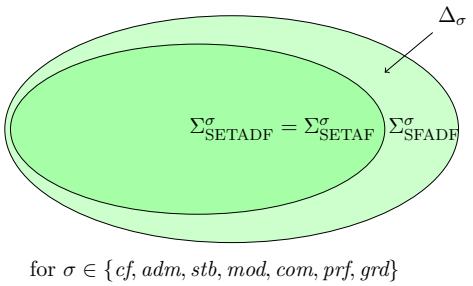

Figure 9. The relation among the signatures of SETAFs, SETADFs, and SFADFs. for $\sigma \in \{cf, adm, stb, mod, com, prf, grd\}$, where $\Delta_\sigma = \{\mathbb{V} \in \Sigma^\sigma_{\text{SFADF}} \mid \exists v \in \mathbb{V} \text{ s.t. } \forall a : v(a) \in \{\mathbf{f}, \mathbf{u}\} \wedge \exists a : v(a) = \mathbf{f}\}$.

6. Symmetric Argumentation Frameworks

Now we proceed our work by considering symmetric subclasses of these classes. These restrictions make decision problems often easier from a computational complexity perspective (Coste-Marquis, Devred, & Marquis, 2005; Dvořák & Dunne, 2018). In this section we investigate the effect of these restrictions to the relations between the different kind of argumentation frameworks under our considerations. That is, we consider symmetric subclasses of SETAFs and the different classes of ADFs. We start by recalling the definition of symmetric ADFs, presented in Diller et al. (2020).

Definition 6.1. An ADF $D = (S, L, C)$ is *symmetric* if L is irreflexive and symmetric and L does not contain any redundant links.

Notice that it is crucial to exclude redundant links as otherwise we are able to add arbitrary links without changing the semantics of the ADF at hand, i.e., we would be able to transform each ADF in equivalent symmetric one.

We continue by recalling the notion of support free symmetric ADF (for short SymSFADFs) Diller et al. (2020) and symmetric SETADFs, in Definitions 6.2 and 6.3, respectively.

Definition 6.2. A bipolar ADF $D = (S, L, C)$ is a *support free symmetric ADF* (SymSFADF for short) if it is symmetric and does not have any supporting links.

Definition 6.3. A SETADF $D = (S, L, C)$ is a *symmetric SETADF* if it is a symmetric ADF.

Note that, by Definition 6.1, SymSFADFs and symmetric SETADFs do not have redundant links. We can use this fact to strengthen Lemma 5.1 in the symmetric case. That is, Lemma 6.4 shows that the class of symmetric SETADFs is a subclass of SymSFADFs.

Lemma 6.4. *Each symmetric SETADF is a SymSFADF.*

Proof. Let $D = (S, L, C)$ be a symmetric SETADF. By Definition 6.1, L does not contain any redundant links. Since the acceptance condition of each argument of D can be written in CNF with only negative literals, it holds that $L = L^-$. Thus, D is a SymSFADF. □

Example 6.5 shows that the classes of symmetric SETADFs and SymSFADFs do not coincide. That is, it provides an example of an SymSFADFs that has no equivalent

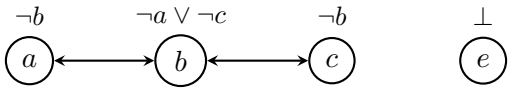

Figure 10. The ADF of Example 6.5

symmetric SETADFs.

Example 6.5. Let D be an ADF depicted in Figure 10. Since D does not contain any redundant links, and all links are symmetric and irreflexive and it does not contain any supporting links, it holds that D is a SymSFADF. However, since the acceptance condition of e cannot be written in CNF with negative literals, D is not equivalent to a symmetric SETADF.

Again, the problems are unattacked arguments, which are isolated arguments in the symmetric setting. with unsatisfiable acceptance condition. SymSFADF that do not have such arguments can be easily rewritten as equivalent symmetric SETADFs.

Lemma 6.6. *Each SymSFADF that does not have any isolated arguments with unsatisfiable acceptance condition is equal to a symmetric SETADF.*

Proof. Let $D = (S, L, C)$ be a SymSFADF with no isolated arguments. Since D is symmetric and does not have any isolated arguments, by Lemma 5.4 the acceptance condition of each argument D can be written in CNF containing only negative literals. Thus, By Definition 6.3, D is a symmetric SETADF. □

That is, with Lemmas 6.4 and 6.6 we obtain that the class of SETADFs correspond to the class of SymSFADFs with no isolated arguments with unsatisfiable acceptance condition.

Finally, we will relate these result to symmetric SETAFs. To this end we recall the notion of symmetric SETAFs presented in Diller et al. (2020). [3]

Definition 6.7. A SETAF $F = (A, R)$, in which $R \subseteq (2^A \setminus \{\emptyset\} \times A)$, is a *symmetric SETAF* if the following properties hold:

- for all $(S, t) \in R$ and for all $s \in S$, there exists $(T, s) \in R$ such that $t \in T$,
- for each argument s and for each $(S, s) \in R$, the set S does not include s.
- for each $(S, s) \in R$ there is no $(S', s) \in R$ with $S' \subset S$.

In Definition 6.7, the first item indicates that in the symmetric SETAFs all links are symmetric. The second item further means that there are also no reflexive links. Finally, the third item excludes redundant links. Note that the third item presents a crucial characterization of symmetric SETAFs. Intuitively, the third item is a guarantee that the associated ADF of a given symmetric SETAF is a SymSFADF which is proven in Theorem 9 in Diller et al. (2020).

We now show that in fact symmetric SETAFs correspond to symmetric SETADFs.

Lemma 6.8. *The ADF associated to a given symmetric SETAF is a symmetric SETADF.*

Proof. Let $F = (A, R)$ be a symmetric SETAF. From Definition 3.3 it follows that

[3] Recent work Dvořák, König, and Woltran (2021) studies different types of symmetry in SETAFs. Our notion of symmetry corresponds the notion of primal-symmetric SETAFs without self-attacks in Dvořák et al. (2021)

each SETAF can be represented as an ADF. We show that the ADF $D_F = (S, L, C)$ associated to F is a symmetric SETADF. Since the acceptance condition of each argument in D_F is in CNF formula over negative literals, by Definition 3.1, D_F is a SETADF. Thus, it remains to show that D_F is a symmetric ADF. It is clear that L does not have any redundant links, since F does not have any redundant links. By proof method presented in the proof of Theorem 9 in Diller et al. (2020) one can check that L is symmetric and irreflexive. Hence, D_F is a symmetric SETADFs. □

Lemma 6.9. *The SETAF associated to given symmetric SETADF is a symmetric SETAF.*

Proof. Let $D = (S, L, C)$ be a symmetric SETADF. We construct the SETAF $F_D = (A, R)$ associated to D as follows: We first define, for each $a \in A$ the sets R_a of all joint attacks to a, i.e., $R_a = \{(cl, a) \mid cl \text{ is a clause in } \varphi_a\}$. We then have:

- $A = S$;
- $R = \bigcup_{a \in S} R_a$.

It is easy to check that F_D is a symmetric SETAF. □

By Lemmas 6.4, 6.6, 6.8 and 6.9 we have that classes of (i) symmetric SETAFs, (ii) symmetric SETADFs and (iii) SymSFADFs with no isolated arguments with unsatisfiable acceptance condition coincide.

7. Related Work

The expressiveness of SETAFs has first been investigated in (Linsbichler et al., 2016) where different sub-classes of ADFs, i.e. AFs, SETAFs and Bipolar ADFs, are related w.r.t. their signatures of 3-valued semantics. Moreover, they provide an algorithm to decide realizability in one of the formalisms under admissible, preferred, complete, model and stable semantics. However, no explicit characterisations of the signatures are given. Pührer (2020b) presented explicit characterisations of the signatures of general ADFs (but not for the sub-classes under our considerations). In contrast, Dvořák et al. (2019) provide explicit characterisations of the two-valued signatures of SETAFs and show that SETAFs are more expressive than AFs. In both works all arguments are relevant for the signature, while in (Flouris & Bikakis, 2019) it is shown that when allowing to add extra arguments to an AF which are not relevant for the signature, i.e. the extensions/labellings are projected on common arguments, then SETAFs and AFs are of equivalent expressiveness. Other recent work (Wallner, 2019) already implicitly showed that SFADFs with satisfiable acceptance conditions can be equivalently represented as SETAFs. This provides a sufficient condition for when we can rewrite a SFADF as SETAF and raises the question whether it is also a necessary condition. In fact, we showed that a SFADF has an equivalent SETAF if and only if all acceptance conditions are satisfiable. Different sub-classes of ADFs (including SFADFs) have been compared in (Diller et al., 2020), but no exact characterisations of signatures as we provide here are given in that work.

Finally, an investigation quite similar to ours has independently been done by Alcântara and Sá (2021), where the main focus lies on the translation between support-free ADFs (there called attacking ADFs without redundant links) and SETAFs, while we have put more on emphasize on the actual expressiveness of these two formalisms. When comparing their technical results with ours they might seem to contradict each

other at first glance. In the following we briefly discuss the origins of these differences which are indeed due to slight differences in the basic definitions. First, by Definition 11 in (Alcântara & Sá, 2021), an attacking ADF (ADF^+) can have redundant links while our notion of support-free ADFs does not allow for redundant links. This is mirrored by the fact that some of the main results in (Alcântara & Sá, 2021) are stated for ADF^+ without redundant links. The second difference is in the definition of SETAFs. While the standard definition of SETAFs that we also use in our paper does not allow attacks from the empty set of arguments towards a single argument, Alcântara and Sá (2021) allow for such attacks. This comes into play when translating support-free ADFs with unsatisfiable acceptance conditions into SETAFs. As shown in Section 5, there are support-free ADFs with unsatisfiable acceptance conditions that cannot be translated in an SETAF (with the standard definition). However, with the extended definition of SETAFs it turns out that all support-free ADFs can be translated into SETAFs (Alcântara & Sá, 2021). Comparing our results with (Alcântara & Sá, 2021), one can thus see that allowing attacks from the empty-set is not just syntactic sugar but increases the expressiveness of SETAFs.

8. Conclusion

In this paper, we have characterised the expressiveness of SETAFs under 3-valued signatures. The more fine-grained notion of 3-valued signatures reveals subtle differences of the expressiveness of stable and preferred semantics which are not present in the 2-valued setting (Dvořák et al., 2019) and enabled us to compare the expressive power of SETAFs and SFADFs, a subclass of ADFs that allows only for attacking links. In particular, we have exactly characterized the difference for conflict-free, admissible, complete, stable, preferred, and grounded semantics; this difference is rooted in the capability of SFADFs to set an initial argument to false. Together with our exact characterisations on signatures of SETAFs for stable, preferred, grounded, and conflict-free semantics, this also yields the corresponding results for SFADFs. Moreover, we extended this results to the cases where we additionally require the attack relation of the frameworks to be symmetric. Our results indicates that notion collective attacks, despite its simplicity, is indeed rather expressive in the sense that we can simulate all kinds of attack relations that are expressible in ADFs.

As directions for future work, we identify exact characterisations for admissible and complete semantics. Another aspect to be investigated is to which extent our insights on labelling-based semantics for SETAFs and SFADFs can help to improve the performance of reasoning systems.

Acknowledgments

This research has been supported by FWF through projects I2854, P30168. The second researcher is supported by the Netherlands eScience Center project "The Eye of the Beholder" (project number 027.020.G15), and by the Netherlands Organisation for Scientific Research (NWO) through the Hybrid Intelligence Gravitation Programme with project number 024.004.022.

References

Alcântara, J., & Sá, S. (2021). Equivalence results between setaf and attacking abstract dialectical frameworks. In *Proceedings nmr* (Vol. 2021, pp. 139–48).

Baumann, R., & Brewka, G. (2019). Extension Removal in Abstract Argumentation - An Axiomatic Approach. In *Proc. AAAI* (pp. 2670–2677). AAAI Press.

Besnard, P., & Doutre, S. (2004). Checking the acceptability of a set of arguments. In *NMR* (pp. 59–64).

Besnard, P., & Hunter, A. (2008). *Elements of Argumentation*. MIT Press.

Besnard, P., Hunter, A., & Woltran, S. (2009). Encoding deductive argumentation in quantified boolean formulae. *Artif. Intell.*, *173*(15), 1406–1423.

Besnard, P., & Schaub, T. (1997). Circumscribing Inconsistency. In *Proc. IJCAI* (pp. 150–155). Morgan Kaufmann.

Besnard, P., Schaub, T., Tompits, H., & Woltran, S. (2002). Paraconsistent Reasoning via Quantified Boolean Formulas, I: Axiomatising Signed Systems. In *Proc. JELIA* (Vol. 2424, pp. 320–331). Springer.

Besnard, P., Schaub, T., Tompits, H., & Woltran, S. (2003). Paraconsistent Logics for Reasoning via Quantified Boolean Formulas, II: Circumscribing Inconsistent Theories. In *Proc. ECSQARU* (pp. 528–539). Springer.

Bikakis, A., Cohen, A., Dvořák, W., Flouris, G., & Parsons, S. (2021). Joint Attacks and Accrual in Argumentation Frameworks. In D. Gabbay, M. Giacomin, G. R. Simari, & M. Thimm (Eds.), *Handbook of formal argumentation* (chap. 2). College Publications.

Brewka, G., Ellmauthaler, S., Strass, H., Wallner, J. P., & Woltran, S. (2013). Abstract Dialectical Frameworks Revisited. In *Proc. IJCAI* (pp. 803–809). IJCAI/AAAI.

Brewka, G., Ellmauthaler, S., Strass, H., Wallner, J. P., & Woltran, S. (2018, February). Abstract Dialectical Frameworks: An Overview. In P. Baroni, D. Gabbay, M. Giacomin, & L. van der Torre (Eds.), *Handbook of formal argumentation* (chap. 5). College Publications.

Brewka, G., Polberg, S., & Woltran, S. (2014). Generalizations of Dung Frameworks and Their Role in Formal Argumentation. *IEEE Intelligent Systems*, *29*(1), 30–38.

Brewka, G., & Woltran, S. (2010). Abstract Dialectical Frameworks. In *Proc. KR* (pp. 102–111). AAAI Press.

Caminada, M. W. A., & Gabbay, D. M. (2009). A Logical Account of Formal Argumentation. *Studia Logica*, *93*(2-3), 109–145.

Coste-Marquis, S., Devred, C., & Marquis, P. (2005). Symmetric Argumentation Frameworks. In *Proc. ECSQARU* (pp. 317–328). Springer.

Diller, M., Zafarghandi, A. K., Linsbichler, T., & Woltran, S. (2020). Investigating subclasses of abstract dialectical frameworks. *Argument & Computation*, *11*(1-2), 191–219.

Dung, P. M. (1995). On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artif. Intell.*, *77*(2), 321–357.

Dunne, P. E., Dvořák, W., Linsbichler, T., & Woltran, S. (2015). Characteristics of multiple viewpoints in abstract argumentation. *Artif. Intell.*, *228*, 153–178.

Dvořák, W., König, M., & Woltran, S. (2021). Graph-Classes of Argumentation Frameworks with Collective Attacks. In *Proc. JELIA, virtual event, may 17-20, 2021, proceedings* (pp. 3–17). Springer.

Dvořák, W., & Dunne, P. E. (2018). Computational problems in formal argumentation and their complexity. In P. Baroni, D. Gabbay, M. Giacomin, & L. van der Torre (Eds.), *Handbook of formal argumentation* (pp. 631–687). College Publications. (also appears in IfCoLog Journal of Logics and their Applications 4(8):2557–2622)

Dvořák, W., Fandinno, J., & Woltran, S. (2019). On the expressive power of collective attacks. *Argument & Computation*, *10*(2), 191–230.

Dvořák, W., Keshavarzi Zafarghandi, A., & Woltran, S. (2020). Expressiveness of SETAFs and Support-Free ADFs Under 3-Valued Semantics. In *Proc. COMMA* (pp. 191–202). IOS Press.

Dvořák, W., König, M., Ulbricht, M., & Woltran, S. (2022). Rediscovering Argumentation

Principles Utilizing Collective Attacks. In *Proc. KR*.
Dvořák, W., König, M., & Woltran, S. (2021). On the Complexity of Preferred Semantics in Argumentation Frameworks with Bounded Cycle Length. In *Proc. KR* (pp. 671–675).
Flouris, G., & Bikakis, A. (2019). A comprehensive study of argumentation frameworks with sets of attacking arguments. *Int. J. Approx. Reason.*, *109*, 55–86.
Keshavarzi Zafarghandi, A. (2017). *Investigating Subclasses of Abstract Dialectical Frameworks* (Unpublished master's thesis). TU Wien.
Lagniez, J., Lonca, E., Mailly, J., & Rossit, J. (2021). Design and results of ICCMA 2021. *CoRR*, *abs/2109.08884*.
Linsbichler, T., Maratea, M., Niskanen, A., Wallner, J. P., & Woltran, S. (2022). Advanced algorithms for abstract dialectical frameworks based on complexity analysis of subclasses and SAT solving. *Artif. Intell.*, *307*, 103697.
Linsbichler, T., Pührer, J., & Strass, H. (2016). A Uniform Account of Realizability in Abstract Argumentation. In *Proc. ECAI* (pp. 252–260). IOS Press.
Modgil, S., & Bench-Capon, T. J. (2011). Metalevel argumentation. *Journal of Logic and Computation*, *21*(6), 959–1003.
Nielsen, S. H., & Parsons, S. (2007). A Generalization of Dung's Abstract Framework for Argumentation: Arguing with Sets of Attacking Arguments. In *Argumentation in multi-agent systems* (pp. 54–73). Springer Berlin Heidelberg.
Niskanen, A., & Järvisalo, M. (2020). μ-toksia: An efficient abstract argumentation reasoner. In *Proc. KR* (pp. 800–804).
Polberg, S. (2016). Understanding the Abstract Dialectical Framework. In *Proc. JELIA* (pp. 430–446).
Polberg, S. (2017). *Developing the abstract dialectical framework* (Unpublished doctoral dissertation). TU Wien, Institute of Information Systems.
Pührer, J. (2020a). Realizability of three-valued semantics for abstract dialectical frameworks. *Artif. Intell.*, *278*, 103-198.
Pührer, J. (2020b). Realizability of three-valued semantics for abstract dialectical frameworks. *Artif. Intell.*, *278*.
Strass, H. (2015a). Expressiveness of two-valued semantics for abstract dialectical frameworks. *J. Artif. Intell. Res. (JAIR)*, *54*, 193–231.
Strass, H. (2015b). The relative expressiveness of abstract argumentation and logic programming. In *AAAI* (pp. 1625–1631). AAAI Press.
Verheij, B. (1996). Two approaches to dialectical argumentation: admissible sets and argumentation stages. *Proc. NAIC*, *96*, 357–368.
Wallner, J. P. (2019). Structural constraints for dynamic operators in abstract argumentation. *Argument & Computation*, *Pre-press*(Pre-press), 1–40.
Yun, B., Vesic, S., & Croitoru, M. (2018). Toward a More Efficient Generation of Structured Argumentation Graphs. In *Proc. COMMA* (pp. 205–212). IOS Press.

Appendix A. Full Proofs

Proof of Lemma 2.12

Assume that $D = (S, L, C)$ is an ADF and v is a model of D. Further, assume $s \in S$ such that $\forall p \in par(s)$, (p, s) is an attacking link in D. Clearly if $\varphi_s[p/\bot : v(p) = \mathbf{f}]$ is irrefutable then also $\varphi_s^v = \varphi_s[p/\top : v(p) = \mathbf{t}][p/\bot : v(p) = \mathbf{f}]$ is irrefutable. It remains to show that if φ_s^v is irrefutable then also $\varphi_s[p/\bot : v(p) = \mathbf{f}]$ is irrefutable. Let $\varphi'_s = \varphi_s[p/\bot : v(p) = \mathbf{f}]$. Towards a contradiction, assume that φ_s^v is irrefutable and φ'_s is not irrefutable. That is, either φ'_s is unsatisfiable or it is undecided. In both cases, $\varphi'_s[p/\top : v(p) = \mathbf{t}]$ is unsatisfiable (as all the links are attacking). Since $\varphi_s^v = \varphi'_s[p/\top : v(p) = \mathbf{t}]$, it holds that φ_s^v is unsatisfiable as well. This is a contradiction with the assumption that φ_s^v is irrefutable.

Proof of Proposition 2.13

Let $D = (S, L, C)$ be a SFADF. Since $stb(D) \subseteq mod(D)$ for each ADF D, it remains to show that each model of D is also a stable model of D. Towards a contradiction assume that $mod(D) \not\subseteq stb(D)$. Thus, there exists a model v of D which is not a stable model. Let D^v be a *stb*-reduct of D and let w be the unique grounded interpretation of D^v. Since it is assumed that v is not a stable model, $v^{\mathbf{t}} \neq w^{\mathbf{t}}$. That is, there exists $s \in S$ such that $v(s) = \mathbf{t}$ and $w(s) \neq \mathbf{t}$. Since w is the grounded interpretation of D^v, it is the least fixed-point of Γ_{D^v}. Moreover, $w(s) \neq \mathbf{t}$ implies that $\Gamma_{D^v} \neq \mathbf{t}$. Therefore, according to the definition of the characteristic operator, $\varphi_s[p/\bot : v(p) = \mathbf{f}]$ cannot be irrefutable. Since, D is a SFADF, all parents of s are attackers. Hence, By Lemma 2.12, φ_s^v is not irrefutable, that is, $v(s) \neq \mathbf{t}$. This is a contradiction by the assumption that $v(s) = \mathbf{t}$. Thus, the assumption that D consists of a model which is not a stable model is incorrect.

Proof of Theorem 3.7

Let $F = (A, R)$ be a SETAF and $D = (S, L, C)$ be its corresponding SETADF. We show that $\{Lab2Int(\lambda) \mid \lambda \in \sigma_{\mathcal{L}}(F)\} = \sigma(D)$. Let λ be an arbitrary three-valued labelling and let $v = Lab2Int(\lambda)$. We investigate that $\lambda \in \sigma_{\mathcal{L}}(F)$ if and only if $v \in \sigma(D)$.

- Let $\sigma = adm$. We first assume that $\lambda \in adm_{\mathcal{L}}(F)$ and show that $v \in adm(D)$. Consider $s \in S$ and the acceptance condition $\varphi_s = \bigwedge_{(B,s)\in R} \bigvee_{a\in B} \neg a$. If $v(s) = \mathbf{t}$ we have that $\lambda(s) = \texttt{in}$ and thus that for all $(B, s) \in R$ there exists $b \in B$ s.t. $\lambda(b) = \texttt{out}$. The latter holds iff for all $(B, s) \in R$ there exists $b \in B$ s.t. $v(b) = \mathbf{f}$ iff partial evaluation of φ_s under v is irrefutable iff $\Gamma_D(v)(s) = \mathbf{t}$. If $v(s) = \mathbf{f}$ we have that $\lambda(s) = \texttt{out}$ and thus that there exists $(B, s) \in R$ s.t. for all $b \in B$: $\lambda(b) = \texttt{in}$. The latter holds iff there exists $(B, s) \in R$ s.t. for all $b \in B$: $v(b) = \mathbf{t}$ iff φ_s^v is unsatisfiable iff $\Gamma_D(v)(s) = \mathbf{f}$. We thus obtain that $v \leq_i \Gamma_D(v)$ and therefore $v \in adm(D)$.

  Now we assume $v \in adm(D)$ and show that $\lambda \in adm_{\mathcal{L}}(F)$. That is for each s with $\lambda(s) = \texttt{in}$ we have $\Gamma_D(v)(s) = \mathbf{t}$ and, as argued above, that for all $(B, s) \in R$ there exists $b \in B$ s.t. $\lambda(b) = out$. Moreover for each s with $\lambda(s) = \texttt{out}$ we have $\Gamma_D(v)(s) = \mathbf{f}$ and, as argued above, that there exists $(B, s) \in R$ s.t. for all $b \in B$: $\lambda(b) = \texttt{in}$. We obtain $\lambda \in adm_{\mathcal{L}}(F)$.

- Let $\sigma \in \{com, prf, grd\}$. Let $\lambda \in com_{\mathcal{L}}(F)$ and let $\varphi_s = \bigwedge_{(B,s)\in R} \bigvee_{a\in B} \neg a$ be the acceptance condition of $s \in S$ in D. For complete semantics it is enough to show that $\lambda(s) = \mathtt{in}$ iff $\Gamma_D(v)(s) = \mathbf{t}$ and $\lambda(s) = \mathtt{out}$ iff $\Gamma_D(v)(s) = \mathbf{f}$.
  - It holds that $\lambda(s) = \mathtt{in}$ (i.e. $v(s) = \mathbf{t}$) iff for all $(B, s) \in R$ there exists $b \in B$ s.t. $\lambda(b) = \mathtt{out}$ iff for all $(B, s) \in R$ there exists $b \in B$ s.t. $v(b) = \mathbf{f}$ iff partial evaluation of φ_s under v is irrefutable iff $\Gamma_D(v)(s) = \mathbf{t}$.
  - On the other hand, $\lambda(s) = \mathtt{out}$ (i.e. $v(s) = \mathbf{f}$) iff there exists $(B, s) \in R$ s.t. for all $b \in B$: $\lambda(b) = \mathtt{in}$ iff there exists $(B, s) \in R$ s.t. for all $b \in B$: $v(b) = \mathbf{t}$ iff φ_s^v is unsatisfiable iff $\Gamma_D(v)(s) = \mathbf{f}$.

  Now as complete semantics coincide it is easy to verify that also the maximal, i.e. the preferred, extensions and the minimal, i.e. the grounded, extension coincide.
- Let $\sigma = stb$. Recall that, by Proposition 2.13, on SETADFs we have that stable and models semantics coincide. We will show that $\lambda \in stb_{\mathcal{L}}(F)$ iff $v \in mod(D)$. That is we show that for each $s \in S$ we have (i) $\lambda(s) = \mathtt{in}$ iff $v(\varphi_s) = \mathbf{t}$ and (ii) $\lambda(s) = \mathtt{out}$ iff $v(\varphi_s) = \mathbf{f}$. To this end let $\varphi_s = \bigwedge_{(B,s)\in R} \bigvee_{a\in B} \neg a$ be the acceptance condition of s.
  - It holds that $\lambda(s) = \mathtt{in}$ (i.e. $v(s) = \mathbf{t}$) iff for all $(B, s) \in R$ there exists $b \in B$ s.t. $\lambda(b) = out$ iff for all $(B, s) \in R$ there exists $b \in B$ s.t. $v(b) = \mathbf{f}$ iff $v(\varphi_s) = \mathbf{t}$.
  - On the other hand, $\lambda(s) = out$ (i.e. $v(s) = \mathbf{f}$) iff there exists $(B, s) \in R$ s.t. for all $b \in B$: $\lambda(b) = \mathtt{in}$ iff there exists $(B, s) \in R$ s.t. for all $b \in B$: $v(b) = \mathbf{t}$ iff $v(\varphi_s) = \mathbf{f}$.
- Finally let $\sigma = cf$. We first assume that $\lambda \in cf_{\mathcal{L}}(F)$ and show that $v \in cf(D)$. Consider $s \in S$ and the acceptance condition $\varphi_s = \bigwedge_{(B,s)\in R} \bigvee_{a\in B} \neg a$. If $v(s) = \mathbf{t}$ we have that $\lambda(s) = \mathtt{in}$ and thus that for all $(B, s) \in R$ there exists $b \in B$ s.t. $\lambda(b) \neq \mathtt{in}$. The latter holds iff for all $(B, s) \in R$ there exists $b \in B$ s.t. $v(b) \neq \mathbf{t}$ iff φ_s^v is satisfiable. If $v(s) = \mathbf{f}$ we have that $\lambda(s) = \mathtt{out}$ and thus that there exists $(B, s) \in R$ s.t. for all $b \in B$: $\lambda(b) = \mathtt{in}$. The latter holds iff there exists $(B, s) \in R$ s.t. for all $b \in B$: $v(b) = \mathbf{t}$ iff φ_s^v is unsatisfiable. We thus obtain that $v \in cf(D)$.

  Now we assume $v \in cf(D)$ and show that $\lambda \in cf_{\mathcal{L}}(F)$. That is for each s with $\lambda(s) = \mathtt{in}$ we have φ_s^v is satisfiable and, as argued above, that for all $(B, s) \in R$ there exists $b \in B$ s.t. $\lambda(b) \neq \mathtt{in}$. Moreover for each s with $\lambda(s) = \mathtt{out}$ we have φ_s^v is unsatisfiable and, as argued above, that there exists $(B, s) \in R$ s.t. for all $b \in B$: $\lambda(b) = \mathtt{in}$. We obtain $\lambda \in cf_{\mathcal{L}}(F)$.

Proof of Proposition 4.10

We show that for each SETAF F the set $adm_{\mathcal{L}}(F)$ satisfies the conditions of the proposition. fonditions (1)–(3) are by the fact that $adm_{\mathcal{L}}(F) \subseteq cf_{\mathcal{L}}(F)$.

For condition (4), let $\lambda, \lambda' \in adm_{\mathcal{L}}(F)$ with $\lambda_{\mathtt{in}} \cap \lambda'_{out} = \{\}$ (since each admissible labelling defends itself, $\lambda'_{\mathtt{in}} \cap \lambda_{out} = \{\}$). Thus, $\lambda^* = (\lambda_{\mathtt{in}} \cup \lambda'_{\mathtt{in}}, \lambda_{\mathtt{out}} \cup \lambda'_{\mathtt{out}}, \lambda_{\mathtt{undec}} \cap \lambda'_{\mathtt{undec}})$ is a well-defined labelling. Consider that $\lambda^*(a) = \mathtt{in}$, that is, either $\lambda(a) = \mathtt{in}$ or $\lambda'(a) = \mathtt{in}$. Since λ, λ' are admissible labellings, for each conflict (B, a) there exists $b \in B$ s.t. $\lambda(b) = \mathtt{out}$ in the former case and $\lambda'(b) = \mathtt{out}$ in the latter case. Thus, for each conflict (B, a) there exists $b \in B$ s.t. $\lambda^*(b) = \mathtt{out}$. Moreover, if $\lambda^*(a) = \mathtt{out}$ there is an attack (B, a) with $B \subseteq \lambda_{\mathtt{in}}$ or $B \subseteq \lambda'_{\mathtt{in}}$, that is, there exists a conflict (B, a) such that $B \subseteq \lambda^*_{\mathtt{in}}$. On the other hand, assume that $\lambda_{\mathtt{in}} \cap \lambda'_{out} \neq \{\}$, for instance, $a \in \lambda_{\mathtt{in}} \cap \lambda'_{out}$. Therefore, $a \in \lambda^*_{\mathtt{in}}$ and $a \in \lambda^*_{\mathtt{out}}$. That is, λ^* is not a well-defined

labelling.

For condition (5), let $\lambda^* = (\lambda'_{\mathtt{in}} \cup C, \lambda'_{\mathtt{out}}, \lambda'_{\mathtt{undec}} \setminus C)$. By the definition of C, it is easy to check that $\lambda^*_{\mathtt{in}} \cap \lambda^*_{\mathtt{out}} = \{\}$, $\lambda^*_{\mathtt{in}} \cap \lambda^*_{\mathtt{undec}} = \{\}$, and $\lambda^*_{\mathtt{out}} \cap \lambda^*_{\mathtt{undec}} = \{\}$ hold. Thus, λ^* is a well-defined labelling. In the definition of admissible labelling there is no condition for label an argument `undec`. Further, $\lambda^*_{\mathtt{out}} = \lambda'_{\mathtt{out}}$, $\lambda'_{\mathtt{in}} \subseteq \lambda^*_{\mathtt{in}}$ and λ' is an admissible labelling, therefore, the condition for arguments which are labelled `out` in λ^* are also satisfied. For argument a with $\lambda^*(a) = \mathtt{in}$ either $a \in \lambda'_{\mathtt{in}}$ or $a \mapsto \mathtt{in} \in C \subseteq \lambda_{\mathtt{in}}$. Each of them implies that for each conflict (B, a) there exists $b \in B$ s.t. $\lambda^*(b) = \mathtt{out}$, since λ, λ' are admissible labelling and $\lambda_{\mathtt{out}} \subseteq \lambda'_{\mathtt{out}}$. Thus, λ^* is an admissible labelling.

For condition (6), first we show that $\lambda'_{\mathtt{in}} \cap (\lambda'_{\mathtt{out}} \cup C) = \{\}$. To this end, let $a \in C$ we show that $a \notin \lambda'_{\mathtt{in}}$. Since $C \subseteq \lambda_{\mathtt{out}}$, there exists $(B, a) \in R$ such that $\lambda(b) = \mathtt{in}$ for all $b \in B$. By the assumption of this condition, namely $\lambda_{\mathtt{in}} \subseteq \lambda'_{\mathtt{in}}$, the relation $B \subseteq \lambda'_{\mathtt{in}}$ holds. Thus, $\lambda'(a) \neq \mathtt{in}$. Since $\lambda' \in adm_{\mathcal{L}}(F)$, to show that $\lambda^* \in adm_{\mathcal{L}}(F)$ it is enough to show that each $a \in C$ is actually labelled `out` in λ^*. This condition is trivially satisfied, because $C \subseteq \lambda_{\mathtt{out}}$, $\lambda_{\mathtt{in}} \subseteq \lambda'_{\mathtt{in}}$ and $\lambda' \in adm_{\mathcal{L}}(F)$.

For condition (7), it is enough to show that $\lambda_{\mathtt{in}} \cap \lambda'_{out} = \{\}$, $\lambda_{\mathtt{in}} \cap (\textsc{Args}_{\mathbb{L}} \setminus (\lambda_{\mathtt{in}} \cup \lambda'_{\mathtt{out}})) = \{\}$, and $\lambda_{\mathtt{out}} \cap (\textsc{Args}_{\mathbb{L}} \setminus (\lambda_{\mathtt{in}} \cup \lambda'_{\mathtt{out}})) = \{\}$. Let $\lambda^* = (\lambda_{\mathtt{in}}, \lambda'_{\mathtt{out}}, \textsc{Args}_{\mathbb{L}} \setminus (\lambda_{\mathtt{in}} \cup \lambda'_{\mathtt{out}}))$. For a with $\lambda^*(a) = \mathtt{in}$ ($a \in \lambda_{\mathtt{in}}$) it holds that $a \notin \lambda_{out}$, because $\lambda \in adm_{\mathcal{L}}(F)$. Further, since $\lambda'_{\mathtt{out}} \subseteq \lambda_{\mathtt{out}}$, $a \notin \lambda'_{out}$, that is, $a \notin \lambda^*_{\mathtt{out}}$. If $a \in \lambda^*_{\mathtt{out}}$ ($a \in \lambda'_{\mathtt{out}}$), since $\lambda'_{\mathtt{out}} \subseteq \lambda_{\mathtt{out}}$, $a \in \lambda_{\mathtt{out}}$. Therefore, $a \notin \lambda_{\mathtt{in}}$ as $\lambda \in adm_{\mathcal{L}}(F)$. Thus, $a \notin \lambda^*_{\mathtt{in}}$. Moreover, a is included either in $\lambda^*_{\mathtt{in}}$ or $\lambda^*_{\mathtt{out}}$ if and only if $a \notin (\textsc{Args}_{\mathbb{L}} \setminus (\lambda_{\mathtt{in}} \cup \lambda'_{\mathtt{out}}))$. On the other hand, conditions of admissible labelling for arguments labelled `out` in λ^* are trivially satisfied as $\lambda^*_{\mathtt{in}} = \lambda_{\mathtt{in}}$ and $\lambda^*_{\mathtt{out}} \subseteq \lambda_{\mathtt{out}}$. Towards a contradiction, assume that $\lambda^*(a) = \mathtt{in}$ and there exists conflict (B, a) s.t. for each $b \in B$, $\lambda^*(b) \neq \mathtt{out}$, that is, $\lambda^*(b) = \mathtt{in}/\mathtt{undec}$. If $\lambda^*(b) = \mathtt{in}$, then $\lambda(b) = \mathtt{in}$ and if $\lambda^*(b) = \mathtt{undec}$, then $b \notin \lambda'_{\mathtt{out}} \subseteq \lambda_{\mathtt{out}}$. That is, $\lambda(b) \neq \mathtt{out}$ for each $b \in B$. This is a contradiction with the assumption that $\lambda \in adm_{\mathcal{L}}(F)$.

For condition (8) let $\lambda = (\emptyset, \emptyset, \textsc{Args}_{\mathbb{L}})$. The conditions of admissible labelling for arguments labelled with `in` or `out` in λ are satisfied, there is no such an argument, and there is no condition for arguments labelled with `undec` in the conditions of admissible labelling. Thus, $\lambda \in adm_{\mathcal{L}}(F)$.

Proof of Theorem 5.3

$\Sigma^{\sigma}_{\text{SETADF}} \subseteq \Sigma^{\sigma}_{SFADF}$ follows from Lemma 5.1. For showing $\Sigma^{adm}_{\text{SETADF}} \subset \Sigma^{adm}_{\text{SFADF}}$, let $\mathbb{V} = \{\{a \mapsto \mathbf{u}, b \mapsto \mathbf{u}\}, \{a \mapsto \mathbf{u}, b \mapsto \mathbf{f}\}, \{a \mapsto \mathbf{t}, b \mapsto \mathbf{f}\}\}$ be an interpretation-set. A witness of *adm*-realizability of $\mathbb{V}$ in SFADFs is $D = (\{a, b\}, \{\varphi_a = \neg a \vee \neg b, \varphi_b = \bot\})$. However, $\mathbb{V}$ is not realizable by any SETADF for admissible interpretations (cf. Proposition 4.10). To show $\Sigma^{\sigma}_{\text{SFADF}} \not\subseteq \Sigma^{\sigma}_{\text{SETADF}}$, for $\sigma \in \{stb, mod, com, prf, grd\}$, let $\mathbb{V} = \{\{a \mapsto \mathbf{f}\}\}$. The interpretation $\mathbb{V}$ is σ-realizable in SFADFs for $\sigma \in \{stb, mod, com, prf, grd\}$, and a witness of σ-realizability of $\mathbb{V}$ in SFADFs is $D = (\{a\}, \{\varphi_a = \bot\})$. However, $\mathbb{V}$ cannot be realized by any SETADF for semantics $\sigma \in \{adm, stb, prf, grd\}$ (cf. Propositions 4.2–4.9). The result for $\sigma = mod$ follows from Proposition 2.13 and for $\sigma = com$ by $|\mathbb{V}| = 1$ (i.e. complete and grounded semantics have to coincide). Further, *cf*(D) is not *cf*-realizable with any SETADF.