*Computational Models of Argument*
*H. Prakken et al. (Eds.)*


*doi:10.3233/FAIA200504*

# Expressiveness of SETAFs and Support-Free ADFs Under 3-Valued Semantics

Wolfgang DVOŘÁK [a], Atefeh KESHAVARZI ZAFARGHANDI [b] and Stefan WOLTRAN [a]

[a] *Institute of Logic and Computation, TU Wien, Austria*
[b] *Department of Artificial Intelligence, Bernoulli Institute, University of Groningen, The Netherlands*

**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. Another approach is provided via abstract dialectical frameworks (ADFs), where acceptance conditions specify the relation between arguments; restricting these conditions naturally allows for so-called support-free ADFs. 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 discriminate the two approaches.

**Keywords.** Abstract argumentation frameworks, Abstract dialectical frameworks, Collective attack.

## 1. Introduction

Abstract argumentation frameworks (AFs) as introduced by Dung [1] are a core formalism in formal argumentation. A popular line of research investigates extensions of Dung AFs that allow for a richer syntax (see, e.g. [2]). In this work we investigate two generalisations of Dung AFs that allow for a more flexible attack structure (but do not consider support between arguments).

The first formalism we consider are SETAFs as introduced by Nielsen and Parsons [3]. SETAFs extend Dung AFs by allowing for collective attacks such that a set of arguments $B$ attacks another argument $a$ but no proper subset of $B$ attacks $a$. Argumentation frameworks with collective attacks have received increasing interest in the last years. For instance, semi-stable, stage, ideal, and eager semantics have been adapted to SETAFs in [4,5]; translations between SETAFs and other abstract argumentation formalisms are studied in [6]; [7] observed that for particular instantiations, SETAFs provide a more convenient target formalism than Dung AFs. The expressiveness of SETAFs with two-valued semantics has been investigated in [4] in terms of signatures. Signatures have been introduced in [8] for AFs. In general terms, a signature for a formalism and a semantics captures all possible outcomes that can be obtained by the instances of the formalism under the considered semantics. Besides that, signatures are recognized as crucial for operators in dynamics of argumentation (cf. [9]).

The second formalism we consider are support-free abstract dialectical frameworks (SFADFs), a subclass of abstract dialectical frameworks (ADFs) [10] which are known as an advanced abstract formalism for argumentation, that is able to cover several generalizations of AFs [2,6]. 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 [11].

For comparison of the two formalisms, we need to focus on 3-valued (labelling) semantics [12,13], which are integral for ADF semantics [10]. In terms of SETAFs, we can rely on the recently introduced labelling semantics in [5]. We first define a new class of ADFs (SETADFs) where the acceptance conditions strictly follow the nature of collective attacks in SETAFs and show that SETAFs and SETADFs coincide for the main semantics, i.e. the $\sigma$-labellings of a SETAF are equal to the $\sigma$-interpretations of the corresponding SETADF. We then provide exact characterisations of the 3-valued signatures for SETAFs (and thus for SETADFs) for most of the semantics under consideration. While SETADFs are a syntactically defined subclass of ADFs, the second formalism 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 briefly discuss related work. The expressiveness of SETAFs has first been investigated in [14] 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 different semantics. However, no explicit characterisations of the signatures are given. Recently, Pührer [15] presented explicit characterisations of the signatures of general ADFs (but not for the sub-classes discussed above). In contrast, [4] provides explicit characterisations of the two-valued signatures of SETAFs and shows that SETAFs are more expressive than AFs. In both works all arguments are relevant for the signature, while in [5] 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 [16] already implicitly showed that SFADFs with satisfiable acceptance conditions can be equivalently represented as SETAFs. This provides a sufficient condition for rewriting an ADF as SETAF and raises the question whether it is also a necessary condition. In fact, we will show 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 [11], but no exact characterisations of signatures as we provide here are given in that work.
To summarize, the main contributions of our paper are as follows:

- We embed SETAFs under 3-valued labeling based semantics [5] 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 [14] and [5].

- 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 [4] 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.

Some technical details had to be omitted but are available in an online appendix: https://www.dbai.tuwien.ac.at/research/argumentation/comma2020-1.pdf

## 2. Background

In this section we briefly recall the necessary definitions for SETAFs and ADFs.

**Definition 1.** A set argumentation framework (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 semantics, cf. [5].

**Definition 2.** A (3-valued) labelling of a SETAF $F = (A,R)$ is a total function $\lambda : A \mapsto \{\mathtt{in}, \mathtt{out}, \mathtt{undec}\}$. For $x \in \{\mathtt{in}, \mathtt{out}, \mathtt{undec}\}$ we write $\lambda_x$ to denote the sets of arguments $a \in A$ with $\lambda(a) = x$. We sometimes denote labellings $\lambda$ as triples $(\lambda_{\mathtt{in}}, \lambda_{\mathtt{out}}, \lambda_{\mathtt{undec}})$.

**Definition 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 $\lambda(a) \neq \mathtt{in}$ or there is a $b \in S$ with $\lambda(b) \neq \mathtt{in}$, and (ii) for all $a \in A$, if $\lambda(a) = \mathtt{out}$ then there is an attack $(S,a) \in R$ such that $\lambda(b) = \mathtt{in}$ for all $b \in S$. A labelling $\lambda$ which is conflict-free in $F$ is

- *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 $\lambda'$ with $\lambda'_{\mathtt{in}} \subset \lambda_{\mathtt{in}}$ complete in $F$;
- *preferred* in $F$ iff it is complete and there is no $\lambda'$ with $\lambda'_{\mathtt{in}} \supset \lambda_{\mathtt{in}}$ complete in $F$;
- *stable* in $F$ iff $\lambda_{\mathtt{undec}} = \emptyset$.

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

**Example 1.** 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 $\mathtt{in}$ but neither $a$ nor $b$ is mapped to $\mathtt{out}$). The labelling that maps all argument to $\mathtt{undec}$ is not a complete labelling, how-

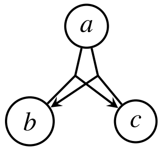


**Figure 1.** The SETAF of Example 1.

ever, 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, $prf_{\mathscr{L}}(F) = stb_{\mathscr{L}}(F) = \{\{a \mapsto \mathtt{in}, b \mapsto \mathtt{out}, c \mapsto \mathtt{in}\}, \{a \mapsto \mathtt{in}, b \mapsto \mathtt{in}, c \mapsto \mathtt{out}\}\}$.

We next turn to abstract dialectical frameworks [17].

**Definition 4.** 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, $\varphi_s$ 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 6 below). 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 $\mathscr{V}$ be the set of all interpretations for an ADF $D$. Then, we call a subset of all interpretations of the ADF, $\mathbb{V} \subseteq \mathscr{V}$, 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 $v|^b_x$, i.e. $v|^b_x(b) = x$ and $v|^b_x(a) = v(a)$ for $a \neq b$. Finally, the partial valuation of acceptance condition $\varphi_s$ by $v$, is given by $\varphi^v_s = v(\varphi_s) = \varphi_s[p/\top : v(p) = \mathbf{t}][p/\bot : v(p) = \mathbf{f}]$, for $p \in par(s)$.

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

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

**Definition 5.** Given an ADF $D = (S, L, C)$, an interpretation $v$ is

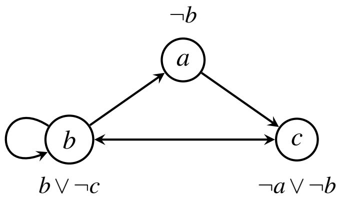


**Figure 2.** The ADF of Example 2.

- *conflict-free* in $D$ iff $v(s) = \mathbf{t}$ implies $\varphi_s^v$ is satisfiable and $v(s) = \mathbf{f}$ implies $\varphi_s^v$ 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 least fixed-point of $\Gamma_D$;
- *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 $\sigma$ 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.

**Example 2.** 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, $v'$ is not a stable model because the truth value of $b$ in $v'$ is since of self-support. 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 $com(D) = prf(D) \cup grd(D)$.

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 6 we consider two-valued interpretations that are only defined over the parents of $a$, that is, only give values to $par(a)$.

**Definition 6.** 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 $v|_{\mathbf{t}}^{b}(\varphi_a) = \mathbf{t}$;

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

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 links, is defined in [18], called *support free ADFs* (SFADFs) in the current work, defined formally as follows.

**Definition 7.** An ADF is called support-free if it has only attacking links.

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 [18].

**Proposition 1.** *For every SFADF D it holds that* $mod(D) = stb(D)$.

*Proof.* The result follows from the following observation: 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. Thus, $\varphi_s^v$ is irrefutable if and only if $\varphi_s[p/\bot \,:\, v(p) = \mathbf{f}]$ is irrefutable. □

## 3. Embedding SETAFs in ADFs

As observed by Polberg [19] and Linsbichler et.al [14], 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 8.** 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 $\mathscr{C}$ a set of non-empty clauses)

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

That is, in a SETADF each acceptance condition is either $\top$ (if $\mathscr{C}$ is empty) or a proper CNF formula over negative literals. SETADFs and SETAFs can be embedded in each other as follows.

**Definition 9.** 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

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

Let $D = (S, L, C)$ be a SETADF. We construct the SETAF $F_D = (A, R)$ in which, $A = S$, and $R$ is constructed as follows. For each argument $s \in S$ with acceptance formula $\bigwedge_{cl \in \mathscr{C}} \bigvee_{a \in cl} \neg a$ we add the attacks $\{(cl, s) \mid cl \in \mathscr{C}\}$ to $R$.

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 $u$.

**Theorem 2.** *For $\sigma \in \{cf, adm, com, prf, grd, stb\}$, a SETAF $F$ and its associated SETADF $D$, we have that $\sigma_{\mathscr{L}}(F)$ and $\sigma(D)$ are in one-to-one correspondence with each labelling $\mathbb{L} \in \sigma_{\mathscr{L}}(F)$ corresponding to an interpretation $v \in \sigma(D)$ such that $v(s) = \mathbf{t}$ iff $\lambda(s) = \mathtt{in}$, $v(s) = \mathbf{f}$ iff $\lambda(s) = \mathtt{out}$, and $v(s) = \mathbf{u}$ iff $\lambda(s) = \mathtt{undec}$.*

Notice that by the above theorem we have that the 3-valued SETAF semantics introduced in [14] coincide with the 3-valued labelling based SETAF semantics of [5] and the model semantics of [14] corresponds to the stable semantics of [5].

## 4. 3-valued Signatures of SETAFs

We adapt the concept of signatures [8] towards our needs first.

**Definition 10.** The signature of SETAFs under a labelling-based semantics $\sigma_{\mathscr{L}}$ is defined as $\Sigma^{\sigma_{\mathscr{L}}}_{SETAF} = \{\sigma_{\mathscr{L}}(F) \mid F \in SETAF\}$. The signature of an ADF-subclass $\mathscr{C}$ under a semantics $\sigma$ is defined as $\Sigma^{\sigma}_{\mathscr{C}} = \{\sigma(D) \mid D \in \mathscr{C}\}$.

By Theorem 2 we can use labellings of SETAFs and interpretations of the SETADF class of ADFs interchangeably, yielding that $\Sigma^{\sigma_{\mathscr{L}}}_{SETAF} \equiv \Sigma^{\sigma}_{SETADF}$, 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.

**Proposition 3.** *The signature $\Sigma^{stb_{\mathscr{L}}}_{SETAF}$ is given by all sets $\mathbb{L}$ of labellings such that*

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

*Proof.* We first show that for each SETAF $F$ the set $stb_{\mathscr{L}}(F)$ satisfies the conditions of the proposition. First clearly all $\lambda \in stb_{\mathscr{L}}(F)$ 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 $\lambda \in stb_{\mathscr{L}}(F)$ 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 in, a contradiction. Thus we obtain that there is at least one argument $a$ with $\lambda(a) = \mathtt{in}$. For Condition (3), towards a contradiction assume that for all arguments $a$ with $\lambda_1(a) = \mathtt{in}$ also $\lambda_2(a) = \mathtt{in}$ holds. As $\lambda_1 \neq \lambda_2$ there is an $a$ with $\lambda_2(a) = \mathtt{in}$ and $\lambda_1(a) = \mathtt{out}$. That is, there is an attack $(B, a)$ such that $\lambda_1(b) = \mathtt{in}$ for all $b \in B$. But then also $\lambda_2(b) = \mathtt{in}$ for all $b \in B$ and by $\lambda_2(a) = \mathtt{in}$ we obtain that $\lambda_2 \notin cf_{\mathscr{L}}(F)$, a contradiction.

Now assume that $\mathbb{L}$ satisfies all the conditions. We give a SETAF $F_{\mathbb{L}} = (A_{\mathbb{L}}, R_{\mathbb{L}})$ with $A_{\mathbb{L}} = \textsc{Args}_{\mathbb{L}}$ and $R_{\mathbb{L}} = \{(\lambda_{\mathtt{in}}, a) \mid \lambda \in \mathbb{L}, \lambda(a) = \mathtt{out}\}$. We show that $stb_{\mathscr{L}}(F_{\mathbb{L}}) = \mathbb{L}$.

To this end we first show $stb_{\mathscr{L}}(F_{\mathbb{L}}) \supseteq \mathbb{L}$. 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 $\lambda \in cf_{\mathscr{L}}(F_{\mathbb{L}})$. 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 $\texttt{out}$. Now towards a contradiction assume there is 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, $\lambda \in cf_{\mathscr{L}}(F_{\mathbb{L}})$ and therefore $\lambda \in stb_{\mathscr{L}}(F_{\mathbb{L}})$.

To show $stb_{\mathscr{L}}(F_{\mathbb{L}}) \subseteq \mathbb{L}$, consider $\lambda \in stb_{\mathscr{L}}(F_{\mathbb{L}})$. If $\lambda$ maps all arguments to $\texttt{in}$ then there is no attack in $R_{\mathbb{L}}$ which means that $\mathbb{L}$ contains only the labelling $\lambda$. 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 $\lambda'_{\texttt{in}} = B$. Then by construction 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 $\texttt{undec}$ to arguments. Note that stable is the only semantics allowing for an empty labelling set.

**Proposition 4.** *The signature $\Sigma^{prf_{\mathscr{L}}}_{SETAF}$ 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* $\texttt{out}$ *then it also assigns an argument to* $\texttt{in}$.
3. *For arbitrary $\lambda_1, \lambda_2 \in \mathbb{L}$ with $\lambda_1 \neq \lambda_2$ there is an argument $a$ such $\lambda_1(a) = \texttt{in}$ and $\lambda_2(a) = \texttt{out}$.*

*Proof sketch.* We first show that for each SETAF $F$ the set $prf_{\mathscr{L}}(F)$ satisfies the conditions of the proposition. The first condition is satisfied as all $\lambda \in prf_{\mathscr{L}}(F)$ have the same domain. The second condition is satisfied by the definition of conflict-free labellings. Condition (3) is by the $\subseteq$-maximality of $\lambda_{\texttt{in}}$ which implies that there is a conflict between each two preferred extensions.

Now assume that $\mathbb{L}$ satisfies all the conditions. We give a SETAF $F_{\mathbb{L}} = (A_{\mathbb{L}}, R_{\mathbb{L}})$ with $A_{\mathbb{L}} = \textsc{Args}_{\mathbb{L}}$ and $R_{\mathbb{L}} = \{(\lambda_{\texttt{in}}, a) \mid \lambda \in \mathbb{L}, \lambda(a) = \texttt{out}\} \cup \{(\lambda_{\texttt{in}} \cup \{a\}, a) \mid \lambda \in \mathbb{L}, \lambda(a) = \texttt{undec}\}$. It remains to show that $prf_{\mathscr{L}}(F_{\mathbb{L}}) = \mathbb{L}$. To show $prf_{\mathscr{L}}(F_{\mathbb{L}}) \supseteq \mathbb{L}$, consider an arbitrary $\lambda \in \mathbb{L}$. $\lambda \in cf_{\mathscr{L}}(F_{\mathbb{L}})$ can be seen by construction, and $\lambda \in adm_{\mathscr{L}}(F_{\mathbb{L}})$ since argument labelled out is attacked by $\lambda$; finally $\lambda \in prf_{\mathscr{L}}(F_{\mathbb{L}})$ is guaranteed since the arguments $a$ with $\lambda(a) = \texttt{undec}$ are involved in self-attacks. To show $prf_{\mathscr{L}}(F_{\mathbb{L}}) \subseteq \mathbb{L}$ consider $\lambda \in prf_{\mathscr{L}}(F_{\mathbb{L}})$. It can be checked that $\lambda$ satisfies all the conditions of the proposition. □

**Proposition 5.** *The signature $\Sigma^{cf_{\mathscr{L}}}_{SETAF}$ 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* $\texttt{out}$ *then it also assigns an argument to* $\texttt{in}$.
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 sketch.* Let $F$ be an arbitrary SETAF we show that $cf_{\mathscr{L}}(F)$ satisfies the conditions of the proposition. The first two conditions are clearly satisfied by the definition of conflict-free labelling. 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 $B \cup \{a\} \subseteq C \subseteq \lambda_{\mathtt{in}}$, and thus $\lambda \notin cf_{\mathscr{L}}(F)$, 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. For Condition (5), consider $\lambda, \lambda' \in cf_{\mathscr{L}}(F)$ with $\lambda_{\mathtt{in}} \subseteq \lambda'_{\mathtt{in}}$ and $\lambda^* = (\lambda'_{\mathtt{in}}, \lambda_{\mathtt{out}} \cup \lambda'_{\mathtt{out}}, \lambda_{\mathtt{undec}} \cap \lambda'_{\mathtt{undec}})$. Since $\lambda, \lambda' \in \mathbb{L}$, it is easy to check that $\lambda^*$ is a well-founded labelling and $\lambda^* \in cf_{\mathscr{L}}(F)$. For Condition (6), consider $\lambda, \lambda' \in cf_{\mathscr{L}}(F)$ and a set $C \subseteq \lambda_{\mathtt{out}}$ containing an argument $a$ such that $\lambda(a) = \mathtt{out}$. That is, there is an attack $(B,a)$ with $B \subseteq \lambda_{\mathtt{in}}$ and thus $\lambda_{\mathtt{in}} \cup C \not\subseteq \lambda'_{\mathtt{in}}$. 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 $A_{\mathbb{L}} = \textsc{Args}_{\mathbb{L}}$ and $R_{\mathbb{L}} = \{(\lambda_{\mathtt{in}}, a) \mid \lambda \in \mathbb{L}, \lambda(a) = \mathtt{out}\} \cup \{(B,b) \mid b \in B, \nexists \lambda \in \mathbb{L} : \lambda_{\mathtt{in}} = B\}$. To complete the proof it remains to show that $cf_{\mathscr{L}}(F_{\mathbb{L}}) = \mathbb{L}$. □

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

**Proposition 6.** *The signature $\Sigma^{grd_{\mathscr{L}}}_{SETAF}$ is given by sets $\mathbb{L}$ of labellings such that $|\mathbb{L}| = 1$, and if $\lambda \in \mathbb{L}$ assigns one argument to* out *then $\lambda_{\mathtt{in}} \neq \emptyset$.*

Notice that Proposition 6 basically exploits that grounded semantics is a unique status semantics based on admissibility. The result thus immediately extends to other semantics satisfying these two properties, e.g. to ideal or eager semantics [5].

So far, we have provided characterisations for the signatures $\Sigma^{stb_{\mathscr{L}}}_{\mathrm{SETAF}}$, $\Sigma^{prf_{\mathscr{L}}}_{\mathrm{SETAF}}$, $\Sigma^{cf_{\mathscr{L}}}_{\mathrm{SETAF}}$, $\Sigma^{grd_{\mathscr{L}}}_{\mathrm{SETAF}}$. By Theorem 2 we get analogous characterizations of $\Sigma^{\sigma}_{\mathrm{SETADF}}$ 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 7.** *For each $\mathbb{L} \in \Sigma^{adm_{\mathscr{L}}}_{SETAF}$ 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 $C \subseteq \lambda_{\mathtt{out}}$ (s.t. $C \neq \emptyset$) we have $\lambda_{\mathtt{in}} \cup C \not\subseteq \lambda'_{\mathtt{in}}$.*
4. *For arbitrary $\lambda, \lambda' \in \mathbb{L}$ either (a) $(\lambda_{\mathtt{in}} \cup \lambda'_{\mathtt{in}}, \lambda_{\mathtt{out}} \cup \lambda'_{\mathtt{out}}, \lambda_{\mathtt{undec}} \cap \lambda'_{\mathtt{undec}}) \in \mathbb{L}$ or (b) there is an argument $a$ such $\lambda(a) = \mathtt{in}$ and $\lambda'(a) = \mathtt{out}$.*
5. *For $\lambda, \lambda' \in \mathbb{L}$ with $\lambda_{\mathtt{out}} \subseteq \lambda'_{\mathtt{out}}$, and $C \subseteq \lambda_{\mathtt{in}} \setminus \bigcup_{\lambda^* \in \mathbb{L}:\, \lambda^*_{\mathtt{in}} = \lambda'_{\mathtt{in}}} \lambda^*_{\mathtt{out}}$ we have $(\lambda'_{\mathtt{in}} \cup C, \lambda'_{\mathtt{out}}, \lambda'_{\mathtt{undec}} \setminus C) \in \mathbb{L}$.*
6. *For $\lambda, \lambda' \in \mathbb{L}$ with $\lambda_{\mathtt{in}} \subseteq \lambda'_{\mathtt{in}}$, and $C \subseteq \lambda_{\mathtt{out}}$ we have $(\lambda'_{\mathtt{in}}, \lambda'_{\mathtt{out}} \cup C, \lambda'_{\mathtt{undec}} \setminus C) \in \mathbb{L}$.*
7. *For $\lambda, \lambda' \in \mathbb{L}$ with $\lambda_{\mathtt{in}} \subseteq \lambda'_{\mathtt{in}}$ and $\lambda_{\mathtt{out}} \supseteq \lambda'_{\mathtt{out}}$ we have $(\lambda_{\mathtt{in}}, \lambda'_{\mathtt{out}}, \textsc{Args}_{\mathbb{L}} \setminus (\lambda_{\mathtt{in}} \cup \lambda'_{\mathtt{out}})) \in \mathbb{L}$.*
8. *$(\emptyset, \emptyset, \textsc{Args}_{\mathbb{L}}) \in \mathbb{L}$.*

*Proof.* We show that for each SETAF $F$ the set $adm_{\mathscr{L}}(F)$ satisfies the conditions of the proposition. Conditions (1)–(3) are by the fact that $adm_{\mathscr{L}}(F) \subseteq cf_{\mathscr{L}}(F)$. For Condition (4), let $\lambda, \lambda' \in adm_{\mathscr{L}}(F)$ 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. Further, since $\lambda, \lambda' \in adm_{\mathscr{L}}(F)$ it is easy to check that $\lambda^* \in adm_{\mathscr{L}}(F)$. For Condition (5), let $\lambda^* = (\lambda'_{\mathtt{in}} \cup C, \lambda'_{\mathtt{out}}, \lambda'_{\mathtt{undec}} \setminus C)$. First, $\lambda^*$ is a well-defined labelling.

Notice that the set $C$ contains arguments defended by $\lambda$ and not attacked by $\lambda'_{\mathtt{in}}$. Now, it is easy to check that $\lambda^*$ meets the condition for being an admissible labelling. For Condition (6), let $\lambda^* = (\lambda'_{\mathtt{in}}, \lambda'_{\mathtt{out}} \cup C, \lambda'_{\mathtt{undec}} \setminus C)$. Notice that the set $C$ contains only arguments attacked by $\lambda_{\mathtt{in}}$ and thus are also attacked by $\lambda'_{\mathtt{in}}$. Thus, starting from the admissible labelling $\lambda'$ we can relabel arguments in $C$ to out and obtain that $\lambda^*$ is also an admissible labelling. For Condition (7), let $\lambda^* = (\lambda_{\mathtt{in}}, \lambda'_{\mathtt{out}}, \text{ARGS}_{\mathbb{L}} \setminus (\lambda_{\mathtt{in}} \cup \lambda'_{\mathtt{out}}))$. First, $\lambda^*$ is a well-defined labelling. We have that setting $\lambda'_{\mathtt{out}}$ to out is sufficient to make all the in labels for arguments in $\lambda'_{\mathtt{in}}$ valid and thus are also sufficient to make the in labels for arguments $\lambda_{\mathtt{in}} \subseteq \lambda'_{\mathtt{in}}$ valid. Moreover, as $\lambda_{\mathtt{out}} \supseteq \lambda'_{\mathtt{out}}$ also labelling arguments $\lambda_{\mathtt{in}}$ with in is sufficient to make the out labels for $\lambda'_{\mathtt{out}}$ valid. Hence, $\lambda^*$ is admissible. For Condition (8), the conditions of admissible labelling for arguments labelled in or out in $(\emptyset, \emptyset, \text{ARGS}_{\mathbb{L}})$ are clearly met, since there are no such arguments. □

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

In order to compare SETAFs with SFADFs, we can rely on SETADFs (recall Theorem 2). In particular, we will compare the signatures $\Sigma^{\sigma}_{SETADF}$ and $\Sigma^{\sigma}_{SFADF}$, cf. Definition 10. We start with the observation that each SETADF can be rewritten as an equivalent SETADF that is also a SFADF.[1]

**Lemma 8.** *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 8, 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. □

By the above 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 $\sigma$, $v$ is a $\sigma$-interpretation of the SFADF $D = (\{a\}, \{\varphi_a = \bot\})$ but there is no SETADF with $v$ being a $\sigma$-interpretation. We thus obtain $\Sigma^{\sigma}_{\text{SETADF}} \subsetneq \Sigma^{\sigma}_{SFADF}$.

**Theorem 9.** $\Sigma^{\sigma}_{SETADF} \subsetneq \Sigma^{\sigma}_{SFADF}$, *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 $\Sigma^{\sigma}_{SFADF}$. To this end we first recall a characterisation of the acceptance conditions of SFADF that can be rewritten as collective attacks.

**Lemma 10.** *[16] Let $D = (S, L, C)$ be a SFADF. If $s \in S$ has at least one incoming link then the acceptance condition $\varphi_s$ 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 expressive-

[1] As discussed in [6], 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.

ness of SETADFs and SFADFs is solely rooted in the capability of SFADFs to set an argument to **f** via a $\bot$ acceptance condition.

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

**Theorem 11.** *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 sketch.* First 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 $b$. On the other hand one can show that when $\mathbb{V} \in \Sigma^{\sigma}_{\text{SFADF}}$ is such that each $v \in \mathbb{V}$ assigns some argument to **t** one can construct a SETADF $D$ with $\sigma(D) = \mathbb{V}$. This is by the fact that we can rewrite acceptance conditions via Lemma 10 and replace $\bot$ acceptance conditions by collective attacks, i.e. for each interpretation we add collective attacks from the arguments set to **t** to all argument with $\bot$ acceptance condition. □

Next, we provide stronger characterisations of $\Delta_\sigma$ for preferred and stable semantics.

**Proposition 12.** *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 sketch.* If a SFADF has a $\sigma$-interpretation $v$ that assigns some arguments to **f** without assigning an argument to **t** then we have that the arguments assigned to **f** are exactly the arguments with acceptance condition $\bot$. For *stb* and *mod* semantics 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 $\sigma$-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 3.** Let $D = (\{a,b,c\}, \{\varphi_a = \bot, \varphi_b = \neg c, \varphi_c = \neg b\})$. 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 11, $com(D)$ 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.

## 6. Discussion

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 [4] 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. Exact characterisations for admissible and complete semantics are subject of future work. 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 currently embedded in the Center of Data Science & Systems Complexity (DSSC) Doctoral Programme, at the University of Groningen.

## References

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

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

[3] Søren Holbech Nielsen and Simon Parsons. A generalization of Dung's abstract framework for argumentation: Arguing with sets of attacking arguments. In *Proc. ArgMAS*, LNCS 4766, pages 54–73, 2006.

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

[5] Giorgos Flouris and Antonis Bikakis. A comprehensive study of argumentation frameworks with sets of attacking arguments. *Int. J. Approx. Reason.*, 109:55–86, 2019.

[6] Sylwia Polberg. *Developing the abstract dialectical framework*. PhD thesis, TU Wien, Institute of Information Systems, 2017.

[7] Bruno Yun, Srdjan Vesic, and Madalina Croitoru. Toward a more efficient generation of structured argumentation graphs. In *Proc. COMMA*, pages 205–212. IOS Press, 2018.

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

[9] Ringo Baumann and Gerhard Brewka. Extension removal in abstract argumentation - an axiomatic approach. In *Proc. AAAI*, pages 2670–2677. AAAI Press, 2019.

[10] Gerhard Brewka, Stefan Ellmauthaler, Hannes Strass, Johannes P. Wallner, and Stefan Woltran. Abstract Dialectical Frameworks: An Overview. In *Handbook of Formal Argumentation*, chapter 5. College Publications, February 2018.

[11] Martin Diller, Atefeh Keshavarzi Zafarghandi, Thomas Linsbichler, and Stefan Woltran. Investigating subclasses of abstract dialectical frameworks. *Argument & Computation*, 11(1), 2020.

[12] Bart Verheij. Two approaches to dialectical argumentation: admissible sets and argumentation stages. *Proc. NAIC*, 96:357–368, 1996.

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

[14] Thomas Linsbichler, Jörg Pührer, and Hannes Strass. A uniform account of realizability in abstract argumentation. In *Proc. ECAI*, pages 252–260. IOS Press, 2016.

[15] Jörg Pührer. Realizability of three-valued semantics for abstract dialectical frameworks. *Artif. Intell.*, 278, 2020.

[16] Johannes Peter Wallner. Structural constraints for dynamic operators in abstract argumentation. *Argument & Computation*, 11(1-2): 151-190, 2020.

[17] Gerhard Brewka, Stefan Ellmauthaler, Hannes Strass, Johannes P. Wallner, and Stefan Woltran. Abstract dialectical frameworks revisited. In *Proc. IJCAI*, pages 803–809, 2013.

[18] Atefeh Keshavarzi Zafarghandi. Investigating subclasses of abstract dialectical frameworks. Master's thesis, TU Wien, 2017.

[19] Sylwia Polberg. Understanding the abstract dialectical framework. In *Proc. JELIA*, LNCS 10021, pages 430–446, 2016.