Argument & Computation 0 (0) 1
IOS Press

Attack Semantics and Collective Attacks Revisited

Martin Caminada [a,*], Matthias König [b], Anna Rapberger [c] and Markus Ulbricht [d]
[a] *Cardiff University, United Kingdom*
E-mail: CaminadaM@cardiff.ac.uk
[b] *Institute of Logic and Computation, TU Wien, Austria*
E-mail: matthias.koenig@tuwien.ac.at
[c] *Department of Computing, Imperial College London, United Kingdom*
E-mail: a.rapberger@imperial.ac.uk
[d] *ScaDS.AI, Department of Computer Science, Leipzig University, Germany*
E-mail: mulbricht@informatik.uni-leipzig.de

Abstract. In the current paper we re-examine the concepts of attack semantics and collective attacks in abstract argumentation, and examine how these concepts interact with each other. For this, we systematically map the space of possibilities. Starting with standard argumentation frameworks (which consist of a directed graph with nodes and arrows) we briefly state both node semantics and arrow semantics (the latter a.k.a. attack semantics) in both their extensions-based form and labellings-based form. We then proceed with SETAFs (which consist of a directed hypergraph of nodes and arrows, to take into account the notion of collective attacks) and state both node semantics and arrow semantics, in both their extensions-based and labellings-based form. We then show equivalence between the extensions-based and labellings-based form, for node semantics and arrow semantics of AFs, as well as for node semantics and arrow semantics of SETAFs. Moreover, we show equivalence between node semantics and arrow semantics for AFs, and equivalence between node semantics and arrow semantics for SETAFs (with the notable exception of semi-stable). We also provide a novel way of converting a SETAF to an AF such that semantics are preserved, without the use of any "meta arguments".

Although the main part of our work is on the level of abstract argumentation, we do provide an application of our theory on the instantiated level. More specifically, we show that the classical characterisation of Assumption-Based Argumentation (ABA) can be seen as an instantiation based on a SETAF, whereas the contemporary characterisation of ABA can be seen as an instantiation based on a standard AF. Our theory of how to convert a SETAF to an AF can then be used to account for both the similarities and the differences between the classical and contemporary characterisations of ABA. Most prominently, our theory is able to explain the semantic mismatch for semi-stable semantics that arises in the ABA instantiation process.

Keywords: Abstract Argumentation, Assumption-Based Argumentation, Attack Semantics, Collective Attacks

1. Introduction

The 1990s saw some of the foundational work in argumentation theory. This includes the work of Simari and Loui [1] that later evolved into Defeasible Logic Programming (DeLP) [2] as well as the ground-breaking work of Vreeswijk [3] whose way of constructing arguments has subsequently been applied in the various versions of the ASPIC formalism [4–7]. Two approaches, however, stand out for their ability to model a wide range of existing formalisms for non-monotonic inference. First of all,

[*]Corresponding author. E-mail: CaminadaM@cardiff.ac.uk.

there is the abstract argumentation approach of Dung [8], which is shown to be able to model formalisms such as Default Logic, logic programming under stable and well-founded model semantics [8], as well as Nute's Defeasible Logic [9] and logic programming under 3-valued stable model semantics [10] and regular semantics [11]. Secondly, there is the assumption-based argumentation (ABA) approach of Bondarenko, Dung, Kowalski and Toni [12], which is shown to model formalisms like Default Logic, logic programming under stable model semantics, auto epistemic logic and circumscription [12].

One of the essential differences between these two approaches is that abstract argumentation is argument-based. The idea is to use the information in the knowledge base to construct arguments and to examine how these arguments attack each other. Semantics are then defined on the resulting argumentation framework (AF), i.e., the directed graph in which the nodes represent arguments and the arrows represent the attack relation. In assumption-based argumentation, on the other hand, semantics are defined on sets of assumptions that attack each other based on their possible inferences.

To some extent, assumption-based argumentation can be regarded as applying a directed hypergraph of which the nodes contain assumptions and the arrows coincide with ABA-arguments that each attack an assumption. This is different from the ABA-AA instantiation [13], which applies a normal (binary) graph of which the nodes (not the arrows) contain ABA-arguments. In the current work, we investigate this hypergraph on the abstract level and refer to it as a SETAF. The hyperedges of such a SETAF can be interpreted as collective attacks, which have first been investigated by Nielsen and Parsons [14].

Even though SETAFs extend the limits of AFs by allowing for collective attacks, it has been shown that many semantics properties and principles still apply in this more general setting [15–18]. This holds even though SETAFs are more expressive than AFs [19]. Recent work examines the computational complexity of reasoning and the underlying hypergraph structure of SETAFs [20–23]. We will use the theory developed in this paper to show the semantic correspondence between ABA and its SETAF instantiation.

One particular claim in the literature is that assumption-based argumentation and abstract argumentation are equivalent to some extent [13, 24]. That is, the outcome (in terms of conclusions) of assumption-based argumentation would be the same as the outcome of its abstract argumentation interpretation (the ABA-AA instantiation of [13]). In the current paper, we re-examine this claim, by carefully analysing what happens on the abstract level. After all, if a SETAF is an abstraction of ABA, and an AF is an abstraction of the ABA-AA instantiation, then examining equivalence between ABA and the ABA-AA instantiation boils down to examining equivalence between SETAFs and AFs.

To carry out our inquiry, we need to borrow a number of concepts from the argumentation literature. The first concept is that of SETAF labellings [15]. Through our theory it becomes clear that these essentially coincide with the assumption labellings of [25]. The second concept is that of attack-semantics [26], which turns out to play an important role in the conversion process from SETAFs to AFs.

We want to highlight that our approach differs from existing conversions from SETAFs to AFs [15, 27]. These approaches can be seen as an instance of *flattening* [28], i.e., the additional meaning of collective attacks is modelled by the introduction of additional arguments to the "flat" AF. The semantics then coincide under projection. However, in our approach the original SETAF is turned "inside-out", i.e., we construct an AF where the arguments correspond to the attacks of the SETAFs. We then show the semantic correspondence via attack semantics.

In our current work, we systematically fill the gaps in the space of collective attacks and attack semantics via extensions and labellings. This orthogonal approach is summarised in Figure 1. Using the thus developed theory, connecting SETAFs and AFs, we then re-examine the often claimed equivalence between assumption-based argumentation and abstract argumentation. We find that some of the already observed equivalences (under complete [29], preferred [29], stable [24] and grounded semantics [30])

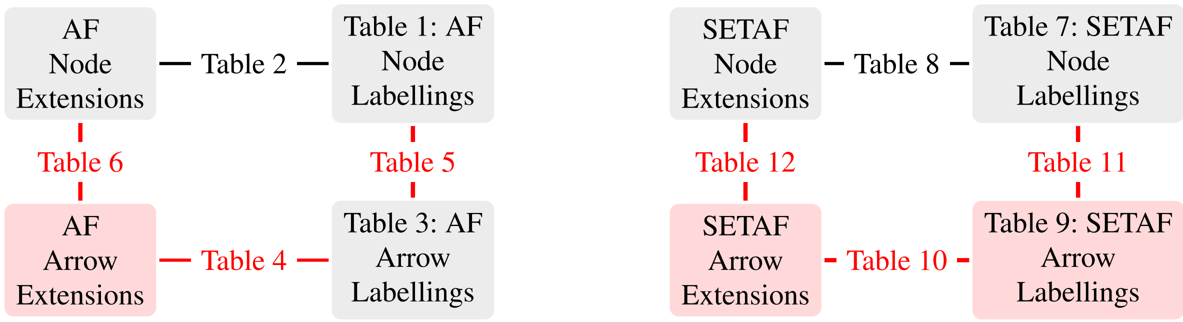


Fig. 1. Illustration of the contents of Section 2 and Section 3. The contributions of this paper are highlighted in red colour.

are special cases of our theory. In addition, for a particular non-equivalence (under semi-stable semantics [29]) our theory is able to explain *why* assumption-based argumentation is not equivalent to the ABA-AA instantiation. As we will see, this is due to the fact that on the abstract level the translation process from SETAFs to AFs does not preserve equivalence under semi-stable semantics.

The remaining part of this paper is structured as follows. First, in Section 2 (Argumentation Frameworks and their Semantics) we briefly summarise some key concepts from the formal argumentation literature, including that of an AF, extension-based semantics, labelling-based semantics, and the notion of attack-semantics. Then, in Section 3 (SETAFs and their Semantics) we recall the concept of SETAFs and how the semantics generalise to this setting. We then broaden the notion of attack-semantics to operate on SETAFs. The results of Section 2 and 3 form the theoretical underpinning of our theory; the obtained relationships between the semantics are summarised in Figure 1. In Section 4 (Relating SETAFs to AFs) we provide a translation procedure to convert SETAFs into AFs, based on the concept of attack-semantics. In Section 5 (Instantiating AFs and SETAFs using ABA) we examine the consequences of the thus obtained theory on AFs and SETAFs in the specific context of ABA, and what this means for the perceived equivalence between ABA and AA[1]. We round off with a discussion in Section 6.

In order to improve readability, we have moved the proofs to the appendices of the paper. A reader who is only interested in our main findings could decide only to read Section 1 to Section 6.

2. Argumentation Frameworks and their Semantics

In the current section, we provide a summary of some of the existing theory in formal argumentation that we will build on. We start with the concept of an argumentation framework, which is essentially a graph consisting of nodes (N) and arrows (arr). For current purposes, we restrict ourselves to finite argumentation frameworks.

Definition 1. *An* argumentation framework *is a pair* (N, arr) *where* N *is a finite set of nodes (called* arguments*) whose internal structure can be left implicit, and* $arr \subseteq N \times N$.

We will commonly refer to the elements of arr as the *arrows* of the argumentation framework. We say that $A \in N$ *attacks* $B \in N$ iff $(A, B) \in arr$. Semantics of argumentation frameworks can be defined in terms of the nodes [8] and in terms of the arrows [26]. We will now discuss each of these approaches.

[1]These results are based on preliminary considerations presented at COMMA 2022 [31].

2.1. Node Semantics for AFs

Semantics for argumentation frameworks were originally defined in terms of its nodes [8].

Definition 2. *Let (N, arr) be an argumentation framework, $M \subseteq N$ and $B \in N$. We say that M* attacks *B iff $\exists A \in M : (A, B) \in arr$. We say that $M_1 \subseteq N$ attacks $M_2 \subseteq N$ iff M_1 attacks some $B \in M_2$. We define M^+ as $\{A \in N \mid M$ attacks $A\}$. We say that M is* conflict-free *iff $M \cap M^+ = \emptyset$. We say that M* defends *A iff each $B \subseteq N$ that attacks A is attacked by M. We define the function $F : 2^N \to 2^N$ as follows: $F(M) = \{A \in N \mid A$ is defended by $M\}$.*

Definition 3. *Let (N, arr) be an argumentation framework. A set of nodes $M \subseteq N$ is called:*

(1) *an* admissible set *iff M is conflict-free and $M \subseteq F(M)$*
(2) *a* complete extension *iff M is conflict-free and $M = F(M)$*
(3) *a* grounded extension *iff M is minimal (w.r.t. $\subseteq$) among all complete extensions*
(4) *a* preferred extension *iff M is a maximal (w.r.t. $\subseteq$) complete extension*
(5) *a* semi-stable extension *iff M is a complete extension where $M \cup M^+$ is maximal (w.r.t. $\subseteq$) among all complete extensions*
(6) *a* stable extension *iff M is a complete extension with $M \cup M^+ = N$*

Although Definition 3 defines the common node semantics in a slightly different way than for instance in [8], equivalence can be observed [32].

An alternative way of defining node semantics is by applying labellings, as is done in the following definition based on [32, 33].

Definition 4. *Let (N, arr) be an argumentation framework. A node labelling is a function $NLab : N \to \{\texttt{in}, \texttt{out}, \texttt{undec}\}$. A node labelling NLab is called admissible iff for each $A \in N$:*

- *if $NLab(A) = \texttt{in}$ then for each $B \in N$ such that $(B, A) \in arr$ it holds that $NLab(B) = \texttt{out}$*
- *if $NLab(A) = \texttt{out}$ then there exists a $B \in N$ such that $(B, A) \in arr$ and $NLab(B) = \texttt{in}$*

A node labelling NLab is called complete iff it is admissible and also satisfies for each $A \in N$:

- *if $NLab(A) = \texttt{undec}$ then not for each $B \in N$ such that $(B, A) \in arr$ it holds that $NLab(B) = \texttt{out}$, and there does not exists a $B \in N$ such that $(B, A) \in arr$ and $NLab(B) = \texttt{in}$*

As a convention, we write $\texttt{in}(NLab)$ for $\{A \in N \mid NLab(A) = \texttt{in}\}$, $\texttt{out}(NLab)$ for $\{A \in N \mid NLab(A) = \texttt{out}\}$ and $\texttt{undec}(NLab)$ for $\{A \in N \mid NLab(A) = \texttt{undec}\}$. A complete node labelling NLab is called:

(1) *grounded iff $\texttt{in}(NLab)$ is minimal (w.r.t. $\subseteq$) among all complete node labellings*
(2) *preferred iff $\texttt{in}(NLab)$ is maximal (w.r.t. $\subseteq$) among all complete node labellings*
(3) *semi-stable iff $\texttt{undec}(NLab)$ is minimal (w.r.t. $\subseteq$) among all complete node labellings*
(4) *stable iff $\texttt{undec}(NLab) = \emptyset$*

As a node labelling essentially defines a partition, we sometimes write it as a triple $(\texttt{in}(NLab), \texttt{out}(NLab), \texttt{undec}(NLab))$.

It has been shown that the complete node labellings with maximal $\texttt{in}$ are equal to the complete node labellings with maximal $\texttt{out}$ [33], and that the complete node labelling where $\texttt{in}$ is minimal is unique and equal to both the complete node labelling where out is minimal and the complete node labelling where undec is maximal [33]. This means that the grounded, preferred and semi-stable node labellings cover all possibilities regarding the minimisation and maximisation of a particular label (among the complete node labellings). This is summarised in Table 1.

Table 1
Implementing different conditions on complete node labellings of argumentation frameworks

condition on node labelling	resulting semantics
maximal in	preferred
maximal out	preferred
maximal undec	grounded
minimal in	grounded
minimal out	grounded
minimal undec	semi-stable
no undec	stable

As was pointed out in [33], for the semantics we consider labellings and extensions are one-to-one related through the functions *NLab2Args* and *Args2NLab*, where

$$NLab2Args(NLab) = \mathtt{in}(NLab), \text{ and}$$
$$Args2NLab(M) = (M, M^+, N \setminus (M \cup M^+))^2.$$

That is, if *NLab* is a complete (resp. grounded, preferred, semi-stable or stable) node labelling, then $NLab2Args(NLab)$ is a complete (resp. grounded, preferred, semi-stable or stable) extension, and if M is a complete (resp. grounded, preferred, semi-stable or stable) extension, then $Args2NLab(M)$ is a complete (resp. grounded, preferred, semi-stable or stable) node labelling [33]. Moreover, under complete semantics (as well as under grounded, preferred, semi-stable and stable semantics) *NLab2Args* and *Args2NLab* are bijective functions and each other's inverses [33]. Table 2 provides an overview of the results.

Table 2
The relation between node extensions and node labellings of argumentation frameworks, through *Args2NLab* and *NLab2Args*

node extension	relation	node labelling
complete extension	$\Leftrightarrow$	complete labelling
grounded extension	$\Leftrightarrow$	grounded labelling
preferred extension	$\Leftrightarrow$	preferred labelling
semi-stable extension	$\Leftrightarrow$	semi-stable labelling
stable extension	$\Leftrightarrow$	stable labelling

[2] We note that *Args2NLab* is defined for *conflict-free* sets $M \subseteq N$.

2.2. Arrow Semantics for AFs

As an alternative to defining semantics based on the *nodes* of an argumentation framework, it is also possible to define semantics based on the *arrows* of the argumentation framework. This idea of "attack semantics" was first introduced in [26]—however, these semantics have not yet been defined in terms of extensions, as we introduce them in Definition 5.

Definition 5. *Let (N, arr) be an argumentation framework, $a \subseteq arr$ and $(A, B) \in arr$. We say that a* attacks *(A, B) iff $(C, A) \in a$ for some $C \in N$. We say that $a_1 \subseteq arr$* attacks *$a_2 \subseteq arr$ iff a_1 attacks some element of a_2. We define a^+ as $\{(C, D) \in arr \mid a \text{ attacks } (C, D)\}$. We say that a is* conflict-free *iff $a \cap a^+ = \emptyset$. We say that a* defends *(A, B) iff each $(C, A) \in arr$ is attacked by a. We define the function $F : 2^{arr} \to 2^{arr}$ as follows: $F(a) = \{(A, B) \in arr \mid (A, B) \text{ is defended by } a\}$.*

Definition 6. *Let (N, arr) be an argumentation framework. $a \subseteq arr$ is called:*

(1) *an* admissible set *iff a is conflict-free and $a \subseteq F(a)$*
(2) *a* complete extension *iff a is conflict-free and $a = F(a)$*
(3) *a* grounded extension *iff a is minimal (w.r.t. $\subseteq$) among all complete extensions*
(4) *a* preferred extension *iff a is a maximal (w.r.t. $\subseteq$) complete extension*
(5) *a* semi-stable extension *iff a is a complete extension where $a \cup a^+$ is maximal (w.r.t. $\subseteq$) among all complete extensions*
(6) *a* stable extension *iff a is a complete extension with $a \cup a^+ = arr$*

An alternative way of defining arrow semantics is by applying labellings, as is done in the following definition.[3]

Definition 7. *Let (N, arr) be an argumentation framework. An arrow labelling is a function $ALab : arr \to \{\mathtt{in}, \mathtt{out}, \mathtt{undec}\}$. An arrow labelling $ALab$ is called admissible iff for each $(A, B) \in arr$:*

- *if $ALab((A, B)) = \mathtt{in}$ then for each $(C, A) \in arr$ it holds that $ALab((C, A)) = \mathtt{out}$*
- *if $ALab((A, B)) = \mathtt{out}$ then there exists a $(C, A) \in arr$ such that $ALab((C, A)) = \mathtt{in}$*

An arrow labelling $ALab$ is called complete iff it is admissible and also satisfies for each $(A, B) \in arr$:

- *if $ALab((A, B)) = \mathtt{undec}$ then not for each $(C, A) \in arr$ it holds that $ALab((C, A)) = \mathtt{out}$, and there does not exist a $(C, A) \in arr$ such that $ALab((C, A)) = \mathtt{in}$*

As a convention, we write $\mathtt{in}(ALab)$ for $\{(A, B) \in arr \mid ALab((A, B)) = \mathtt{in}\}$, $\mathtt{out}(ALab)$ for $\{(A, B) \in arr \mid ALab((A, B)) = \mathtt{out}\}$ and $\mathtt{undec}(ALab)$ for $\{(A, B) \in arr \mid ALab((A, B)) = \mathtt{undec}\}$. A complete arrow labelling $ALab$ is called:

(1) *grounded iff $\mathtt{in}(ALab)$ is minimal (w.r.t. $\subseteq$) among all complete arrow labellings*
(2) *preferred iff $\mathtt{in}(ALab)$ is maximal (w.r.t. $\subseteq$) among all complete arrow labellings*
(3) *semi-stable iff $\mathtt{undec}(ALab)$ is minimal (w.r.t. $\subseteq$) among all complete arrow labellings*
(4) *stable iff $\mathtt{undec}(ALab) = \emptyset$*

[3] The labelling-based version of complete attack semantics is defined in a slightly different way in [26] than in Definition 7, but equivalence is proved in Appendix A.

As an arrow labelling essentially defines a partition, we sometimes write it as a triple $(\texttt{in}(ALab), \texttt{out}(ALab), \texttt{undec}(ALab))$.

It can be shown that the complete arrow labellings with maximal `in` are equal to the complete arrow labellings with maximal `out` (Appendix A) and that the complete arrow labelling where `in` is minimal is unique and equal to both the complete arrow labelling where `out` is minimal and the complete arrow labelling where `undec` is maximal (Appendix A). This means that the grounded, preferred and semi-stable arrow labellings cover all possibilities regarding the minimisation and maximisation of a particular label (among the complete arrow labellings). This is summarised in Table 3.

Table 3

Implementing different conditions on complete arrow labellings of argumentation frameworks

condition on arrow labelling	resulting semantics
maximal `in`	preferred
maximal `out`	preferred
maximal `undec`	grounded
minimal `in`	grounded
minimal `out`	grounded
minimal `undec`	semi-stable
no `undec`	stable

It can be shown that arrow extensions and arrow labellings are one-to-one related through the functions *ALab2a* and *a2ALab*, where

$$ALab2a(ALab) = \texttt{in}(ALab), \text{ and}$$

$$a2ALab(a) = (a, a^+, arr \setminus (a \cup a^+))^4.$$

If *ALab* is a complete (resp. grounded, preferred, semi-stable or stable) arrow labelling, then $ALab2a(ALab)$ is a complete (resp. grounded, preferred, semi-stable or stable) arrow extension, and if a is a complete (resp. grounded, preferred, semi-stable or stable) arrow extension then $a2ALab(a)$ is a complete (resp. grounded, preferred, semi-stable or stable) arrow labelling. Moreover, under complete semantics (as well as under grounded, preferred, semi-stable and stable semantics) *ALab2a* and *a2ALab* are bijective functions that are each other's inverses. Details and proofs can be found in Appendix H. Table 4 provides an overview of the results.

Table 4

The relation between arrow extensions and arrow labellings of argumentation frameworks, through *a2ALab* and *ALab2a*

arrow extension	relation	arrow labelling
complete extension	$\Leftrightarrow$	complete labelling
grounded extension	$\Leftrightarrow$	grounded labelling
preferred extension	$\Leftrightarrow$	preferred labelling
semi-stable extension	$\Leftrightarrow$	semi-stable labelling
stable extension	$\Leftrightarrow$	stable labelling

[4]We note that *a2ALab* is defined for *conflict-free* sets $a \subseteq arr$.

2.3. On the Equivalence between Node Semantics and Arrow Semantics for AFs

It turns out that arrow labellings and node labellings are one-to-one related through the functions *ALab2NLab* and *NLab2ALab*, where

$$\begin{aligned}
ALab2NLab(ALab) = (&\{A \in N \mid \forall (B,A) \in arr : ALab((B,A)) = \texttt{out}\},\\
&\{A \in N \mid \exists (B,A) \in arr : ALab((B,A)) = \texttt{in}\},\\
&\{A \in N \mid \neg\forall (B,A) \in arr : ALab((B,A)) = \texttt{out} \text{ and}\\
&\qquad \neg\exists (B,A) \in arr : ALab((B,A)) = \texttt{in}\}),\\
NLab2ALab(NLab) = (&\{(A,B) \in arr \mid NLab(A) = \texttt{in}\},\\
&\{(A,B) \in arr \mid NLab(A) = \texttt{out}\},\\
&\{(A,B) \in arr \mid NLab(A) = \texttt{undec}\}).
\end{aligned}$$

It can be shown that if *ALab* is a complete (resp. grounded, preferred or stable) arrow labelling, then *ALab2NLab* is a complete (resp. grounded, preferred or stable) node labelling, and if *NLab* is a complete (resp. grounded, preferred, semi-stable or stable) node labelling, then $NLab2ALab(NLab)$ is a complete (resp. grounded, preferred, semi-stable or stable) arrow labelling (see Appendix B for proofs). Moreover, under complete semantics (as well as under grounded, preferred and stable semantics) *ALab2NLab* and *NLab2ALab* are bijective functions and each other's inverses (see Appendix B for proofs). Table 5 and Theorem 8, respectively, provide an overview of these results.

Table 5
The relation between node labellings and arrow labellings of argumentation frameworks, through *NLab2ALab* and *ALab2NLab*

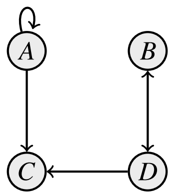
node labelling	relation	arrow labelling
complete node labelling	$\Leftrightarrow$	complete arrow labelling
grounded node labelling	$\Leftrightarrow$	grounded arrow labelling
preferred node labelling	$\Leftrightarrow$	preferred arrow labelling
semi-stable node labelling	$\not\Leftrightarrow$	semi-stable arrow labelling
stable node labelling	$\Leftrightarrow$	stable arrow labelling

Theorem 8. *Let $AF = (N, arr)$ be an argumentation framework and let NLab and ALab be a node labelling and an arrow labelling of AF, respectively. It holds that:*

(1) *If NLab is a complete node labelling, then $NLab2ALab(NLab)$ is a complete arrow labelling. If ALab is a complete arrow labelling, then $ALab2NLab(ALab)$ is a complete node labelling.*
(2) *When restricted to complete node labellings and complete arrow labellings, the functions ALab2NLab and NLab2ALab become bijections and each other's inverses.*
(3) *If NLab is a grounded node labelling, then $NLab2ALab(NLab)$ is a grounded arrow labelling. If ALab is a grounded arrow labelling, then $ALab2NLab(ALab)$ is a grounded node labelling.*
(4) *If NLab is a preferred node labelling, then $NLab2ALab(NLab)$ is a preferred arrow labelling. If ALab is a preferred arrow labelling, then $ALab2NLab(ALab)$ is a preferred node labelling.*
(5) *If NLab is a stable node labelling, then $NLab2ALab(NLab)$ is a stable arrow labelling. If ALab is a stable arrow labelling, then $ALab2NLab(ALab)$ is a stable node labelling.*

Note that this implication does not hold for semi-stable semantics (in neither direction), as the following counter-examples illustrate. Intuitively, the problem are nodes that have no outgoing arrows, which means in arrow semantics we cannot distinguish the cases `out` and `undec`.

Example 9. *Let $AF = (N, arr)$ be an argumentation framework with $N = \{A, B, C, D\}$ and $arr = \{(A, A), (A, C), (B, D), (D, B), (D, C)\}$.*



AF has three complete node labellings:

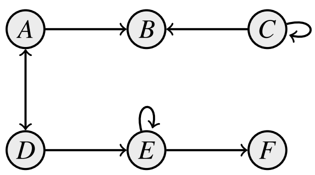
$$NLab_1 = (\emptyset, \emptyset, \{A, B, C, D\})$$
$$NLab_2 = (\{B\}, \{D\}, \{A, C\})$$
$$NLab_3 = (\{D\}, \{B, C\}, \{A\})$$

and three complete arrow labellings:

$$ALab_1 = (\emptyset, \emptyset, \{(A, A), (A, C), (B, D), (D, B), (D, C)\})$$
$$ALab_2 = (\{(B, D)\}, \{(D, B), (D, C)\}, \{(A, A), (A, C)\})$$
$$ALab_3 = (\{(D, B), (D, C)\}, \{(B, D)\}, \{(A, A), (A, C)\})$$

These node labellings and arrow labellings correspond to each other through the functions NLab2ALab and ALab2NLab. While $ALab_2$ is a semi-stable arrow labelling, $NLab_2 = ALab2NLab(ALab_2)$ is not *a semi-stable node labelling (the only semi-stable node labelling is $NLab_3$).*

Example 10. *Let $AF = (N, arr)$ be an argumentation framework with $N = \{A, B, C, D, E, F\}$ and $arr = \{(A, B), (C, B), (C, C), (A, D), (D, A), (D, E), (E, E), (E, F)\}$.*



AF has three complete node labellings:

$$NLab_1 = (\emptyset, \emptyset, \{A, B, C, D, E, F\})$$
$$NLab_2 = (\{A\}, \{B, D\}, \{C, E, F\})$$
$$NLab_3 = (\{D, F\}, \{A, E\}, \{B, C\})$$

and three complete arrow labellings:

$$ALab_1 = (\emptyset, \emptyset, \{(A,B),(C,B),(C,C),(A,D),(D,A),(D,E),(E,E),(E,F)\})$$
$$ALab_2 = (\{(A,B),(A,D)\}, \{(D,A),(D,E)\}, \{(C,B),(C,C),(E,E),(E,F)\})$$
$$ALab_3 = (\{(D,A),(D,E)\}, \{(A,B),(A,D),(E,E),(E,F)\}, \{(C,B),(C,C)\})$$

These node labellings and arrow labellings correspond to each other through the functions NLab2ALab and ALab2NLab. While $NLab_2$ is a semi-stable node labelling, $ALab_2 = NLab2ALab(NLab_2)$ is not *a semi-stable arrow labelling (the only semi-stable arrow labelling is $ALab_3$).*

Node extensions and arrow extensions are one-to-one related via the functions

$$Args2a = ALab2a \circ NLab2ALab \circ Args2NLab, \text{ and}$$
$$a2Args = Args2NLab \circ ALab2NLab \circ ALab2a.$$

The following table provides an overview (see Appendix I for proofs).

Table 6
The relation between node extensions and arrow extensions of argumentation frameworks.

node extension	relation	arrow extension
complete node extension	$\Leftrightarrow$	complete arrow extension
grounded node extension	$\Leftrightarrow$	grounded arrow extension
preferred node extension	$\Leftrightarrow$	preferred arrow extension
semi-stable node extension	$\not\Leftrightarrow$	semi-stable arrow extension
stable node extension	$\Leftrightarrow$	stable arrow extension

3. SETAFs and their Semantics

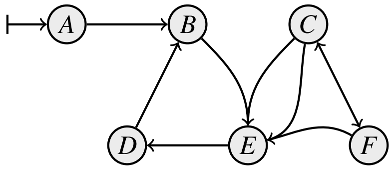
Apart from defining argumentation semantics based on a normal (binary) directed graph, one can also define argumentation semantics that take collective attacks into account. The idea is that instead of one node attacking another node, there is a *set* of nodes attacking another node. We note that these frameworks are a special case of directed hypergraphs where the target element is a singleton. For current purposes, we restrict ourselves to hypergraphs that are finite.

Definition 11. *An* argumentation framework with set attacks (SETAF) *is a tuple $\mathfrak{SF} = (\mathfrak{N}, \mathfrak{arr})$ where $\mathfrak{N}$ is a finite set of nodes, whose structure can be left implicit, and* $\mathfrak{arr} \subseteq 2^{\mathfrak{N}} \times \mathfrak{N}$.

As for AFs, we refer to the elements of $\mathfrak{N}$ as the *nodes* and to the elements of $\mathfrak{arr}$ as the *arrows* of the SETAF. Moreover, we say $\mathfrak{M}$ *attacks* A iff $(\mathfrak{M}, A) \in \mathfrak{arr}$. Consider the following example for an illustration.

Example 12. *Consider the following SETAF $\mathfrak{SF} = (\mathfrak{N}, \mathfrak{arr})$ with*

$$\mathfrak{N} = \{A, B, C, D, E, F\},$$
$$\mathfrak{arr} = \{(\emptyset, A), (\{A\}, B), (\{B, C\}, E), (\{C, F\}, E), (\{C\}, F), (\{D\}, B), (\{E\}, D), (\{F\}, C)\}\,.$$



Remark 13. *Originally, set attacks have been introduced without allowing for the empty set attacking an argument [14]: an* argumentation framework with collective attacks *is a pair $(\mathfrak{N}, \mathfrak{arr})$ where $\mathfrak{N}$ is a finite set of nodes and $\mathfrak{arr} \subseteq (2^{\mathfrak{N}} \setminus \emptyset) \times \mathfrak{N}$. Hence each AF with collective attacks is a SETAF but not vice versa. However, the difference can be neglected, as we show in Appendix G: given a SETAF $\mathfrak{SF} = (\mathfrak{N}, \mathfrak{arr})$, if we simply delete all nodes $A \in \mathfrak{N}$ with $(\emptyset, A) \in \mathfrak{arr}$ and all arrows $(\mathfrak{M}, B)$ where either $A \in \mathfrak{M}$ or $B = A$, we obtain an AF with collective attacks that is equivalent (w.r.t. all semantics under consideration) to our original SETAF $\mathfrak{SF}$.*

Remark 14. *Note that (with a slight abuse of notation) every AF can be seen as a SETAF (cf. [14]): let $AF = (N, arr)$ be an AF, the corresponding SETAF $\mathfrak{SF}_{AF}$ is defined as $\mathfrak{SF}_{AF} = (N, \{(\{A\}, B) \mid (A, B) \in arr\})$. Both the node semantics and arrow semantics (see next sections) coincide in this case.*

Just as is the case for AFs, semantics of SETAFs can be defined in terms of nodes [14, 15] and in terms of arrows (which is a novel contribution of this paper). We will now discuss each of these approaches.

3.1. Node Semantics for SETAFs

Semantics for SETAFs were originally defined in terms of its nodes [14].

Definition 15. *Let $\mathfrak{SF} = (\mathfrak{N}, \mathfrak{arr})$ be a SETAF, $\mathfrak{M} \subseteq \mathfrak{N}$ and $A \in \mathfrak{N}$. We say that $\mathfrak{M}$* attacks *A iff $\exists \mathfrak{M}' \subseteq \mathfrak{M} : (\mathfrak{M}', A) \in \mathfrak{arr}$. We say that $\mathfrak{M}_1 \subseteq \mathfrak{N}$ attacks $\mathfrak{M}_2 \subseteq \mathfrak{N}$ iff $\mathfrak{M}_1$ attacks some $B \in \mathfrak{M}_2$. We define $\mathfrak{M}^+_{\mathfrak{SF}}$ as $\{A \in \mathfrak{N} \mid \mathfrak{M}$ attacks $A\}$. We say that $\mathfrak{M}$ is* conflict-free *iff $\mathfrak{M} \cap \mathfrak{M}^+_{\mathfrak{SF}} = \emptyset$. We write $cf(\mathfrak{SF}) = \{Mh \subseteq \mathfrak{N} \mid \mathfrak{M} \cap \mathfrak{M}^+_{\mathfrak{SF}} = \emptyset\}$ to denote the set of conflict-free sets. We say that $\mathfrak{M}$* defends *A iff for each $\mathfrak{M}' \subseteq \mathfrak{N}$ that attacks A, $\mathfrak{M}$ attacks $\mathfrak{M}'$. We define the* characteristic function *$F_{\mathfrak{SF}} : 2^{\mathfrak{N}} \to 2^{\mathfrak{N}}$ with $F_{\mathfrak{SF}}(\mathfrak{M}) = \{A \in \mathfrak{N} \mid A$ is defended by $\mathfrak{M}\}$. We omit subscript $\mathfrak{SF}$ if it is clear from the context.*

Definition 16 ([14, 15])**.** *Let $(\mathfrak{N}, \mathfrak{arr})$ be a SETAF. $\mathfrak{M} \subseteq \mathfrak{N}$ is called:*

(1) *an* admissible set *iff $\mathfrak{M}$ is conflict-free and $\mathfrak{M} \subseteq F(\mathfrak{M})$*
(2) *a* complete extension *iff $\mathfrak{M}$ is conflict-free and $\mathfrak{M} = F(\mathfrak{M})$*
(3) *a* grounded extension *iff $\mathfrak{M}$ is minimal (w.r.t. $\subseteq$) among all complete extensions*
(4) *a* preferred extension *iff $\mathfrak{M}$ is a maximal (w.r.t. $\subseteq$) complete extension*
(5) *a* semi-stable extension *iff $\mathfrak{M}$ is a complete extension where $\mathfrak{M} \cup \mathfrak{M}^+$ is maximal (w.r.t. $\subseteq$) among all complete extensions*
(6) *a* stable extension *iff $\mathfrak{M}$ is a complete extension where $\mathfrak{M} \cup \mathfrak{M}^+ = \mathfrak{N}$*

If one interprets an arrow $(\mathfrak{M}, A)$ as a collective attack [14] from $\mathfrak{M}$ to A, then the semantics of Definition 16 coincide with those defined in [14] (minus semi-stable, which is missing in [14] but has been considered in later work, e.g., in [15]). In contrast to the original definitions, we have decided to follow the approach from [15] and use complete semantics as the basis for defining preferred, semi-stable, grounded, and stable semantics. It holds that the grounded extension is unique, and that there always exists at least one preferred and at least one semi-stable extension.

An alternative way of defining SETAF semantics is by applying node labellings as done in [15].

Definition 17 ([15])**.** *Let* $(\mathfrak{N}, \mathfrak{arr})$ *be a SETAF. A SETAF node labelling is a function* $\mathfrak{NLab} : \mathfrak{N} \to \{\texttt{in}, \texttt{out}, \texttt{undec}\}$*. A SETAF node labelling* $\mathfrak{NLab}$ *is called admissible iff for each* $A \in \mathfrak{N}$*:*

- *if* $\mathfrak{NLab}(A) = \texttt{in}$ *then for each* $\mathfrak{M} \subseteq \mathfrak{N}$ *such that* $(\mathfrak{M}, A) \in \mathfrak{arr}$ *it holds that* $\exists B \in \mathfrak{M} : \mathfrak{NLab}(B) = \texttt{out}$
- *if* $\mathfrak{NLab}(A) = \texttt{out}$ *then there exists an* $\mathfrak{M} \subseteq \mathfrak{N}$ *such that* $(\mathfrak{M}, A) \in \mathfrak{arr}$ *and* $\forall B \in \mathfrak{M} : \mathfrak{NLab}(B) = \texttt{in}$

A SETAF node labelling $\mathfrak{NLab}$ *is called complete iff it is admissible and also satisfies for each* $A \in \mathfrak{N}$*:*

- *if* $\mathfrak{NLab}(A) = \texttt{undec}$ *then not for each* $\mathfrak{M} \subseteq \mathfrak{N}$ *such that* $(\mathfrak{M}, A) \in \mathfrak{arr}$ *it holds that* $\exists B \in \mathfrak{M} : \mathfrak{NLab}(B) = \texttt{out}$ *and there does not exist an* $\mathfrak{M} \subseteq \mathfrak{N}$ *such that* $(\mathfrak{M}, A) \in \mathfrak{arr}$ *and* $\forall B \in \mathfrak{M} : \mathfrak{NLab}(B) = \texttt{in}$

As a convention, we write $\texttt{in}(\mathfrak{NLab})$ *for* $\{A \in \mathfrak{N} \mid \mathfrak{NLab}(A) = \texttt{in}\}$, $\texttt{out}(\mathfrak{NLab})$ *for* $\{A \in \mathfrak{N} \mid \mathfrak{NLab}(A) = \texttt{out}\}$ *and* $\texttt{undec}(\mathfrak{NLab})$ *for* $\{A \in \mathfrak{N} \mid \mathfrak{NLab}(A) = \texttt{undec}\}$*. A complete SETAF node labelling* $\mathfrak{NLab}$ *is called:*

(1) *grounded iff* $\texttt{in}(\mathfrak{NLab})$ *is minimal (w.r.t.* $\subseteq$*) among all complete SETAF node labellings*
(2) *preferred iff* $\texttt{in}(\mathfrak{NLab})$ *is maximal (w.r.t.* $\subseteq$*) among all complete SETAF node labellings*
(3) *semi-stable iff* $\texttt{undec}(\mathfrak{NLab})$ *is minimal (w.r.t.* $\subseteq$*) among all complete SETAF node labellings*
(4) *stable iff* $\texttt{undec}(\mathfrak{NLab}) = \emptyset$

As a SETAF node labelling essentially defines a partition, we sometimes write it as a triple $(\texttt{in}(\mathfrak{NLab}), \texttt{out}(\mathfrak{NLab}), \texttt{undec}(\mathfrak{NLab}))$.

It can been shown that the complete SETAF node labellings with maximal in are equal to the complete SETAF node labellings with maximal out (see Appendix D) and that the complete SETAF node labelling where in is minimal is unique and equal to both the complete SETAF node labelling where out is minimal and the complete SETAF node labelling where undec is maximal (See Appendix D). This means that the grounded, preferred and semi-stable SETAF node labellings cover all possibilities regarding the minimisation and maximisation of a particular label (among the complete SETAF node labellings). This is summarised in Table 7.

As shown in [15, Theorem 5.10, Theorem 5.11], it holds that SETAF node labellings and SETAF extensions are one-to-one related through the functions $\mathfrak{NLab2Args}$ and $\mathfrak{Args2NLab}$, where

$$\begin{aligned}\mathfrak{NLab2Args}(\mathfrak{NLab}) &= \texttt{in}(\mathfrak{NLab}), \text{ and}\\ \mathfrak{Args2NLab}(\mathfrak{M}) &= (\mathfrak{M}, \mathfrak{M}^+, \mathfrak{N} \setminus (\mathfrak{M} \cup \mathfrak{M}^+))^5.\end{aligned}$$

If $\mathfrak{NLab}$ is a complete (resp. grounded, preferred, semi-stable or stable) SETAF node labelling, then $\mathfrak{NLab2Args}(\mathfrak{NLab})$ is a complete (resp. grounded, preferred, semi-stable or stable) SETAF exten-

[5] We note that $\mathfrak{Args2NLab}$ is defined for *conflict-free* sets $\mathfrak{M} \subseteq \mathfrak{N}$.

Table 7
Implementing different conditions on complete node labellings of SETAFs

condition on node labelling	resulting semantics
maximal `in`	preferred
maximal `out`	preferred
maximal `undec`	grounded
minimal `in`	grounded
minimal `out`	grounded
minimal `undec`	semi-stable
no `undec`	stable

sion, and if $\mathfrak{M}$ is a complete (resp. grounded, preferred, semi-stable or stable) SETAF extension, then $\mathfrak{Args2NLab}(\mathfrak{M})$ is a complete (resp. grounded, preferred, semi-stable or stable) SETAF node labelling (see Appendix D for proofs). Moreover, under complete semantics (as well as under grounded, preferred, semi-stable and stable semantics) $\mathfrak{NLab2Args}$ and $\mathfrak{Args2NLab}$ are bijective functions and each other's inverses (see Appendix D for proofs). Table 8 provides an overview of these results.

Table 8
The relation between node extensions and node labellings of SETAFs, through $\mathfrak{Args2NLab}$ and $\mathfrak{NLab2Args}$

node extension	relation	node labelling
complete extension	$\Leftrightarrow$	complete labelling
grounded extension	$\Leftrightarrow$	grounded labelling
preferred extension	$\Leftrightarrow$	preferred labelling
semi-stable extension	$\Leftrightarrow$	semi-stable labelling
stable extension	$\Leftrightarrow$	stable labelling

3.2. Arrow Semantics for SETAFs

As an alternative to defining semantics based on the nodes of a SETAF, it is also possible to define semantics based on the arrows of the SETAF. This can be done using either arrow extensions or arrow labellings. We start with arrow extensions.

Definition 18. *Let* $(\mathfrak{N}, \mathfrak{arr})$ *be a SETAF,* $\mathfrak{a} \subseteq \mathfrak{arr}$ *and* $(\mathfrak{M}, A) \in \mathfrak{arr}$*. We say that* $\mathfrak{a}$ attacks $(\mathfrak{M}, A)$ *iff* $(\mathfrak{M}', B) \in \mathfrak{a}$ *with* $B \in \mathfrak{M}$ *for some* $\mathfrak{M}' \subseteq \mathfrak{N}$*. We say that* $\mathfrak{a}_1 \subseteq \mathfrak{arr}$ attacks $\mathfrak{a}_2 \subseteq \mathfrak{arr}$ *iff* $\mathfrak{a}_1$ *attacks some element of* $\mathfrak{a}_2$*. We define* $\mathfrak{a}^+$ *as* $\{(\mathfrak{M}, A) \in \mathfrak{arr} \mid \mathfrak{a}$ *attacks* $(\mathfrak{M}, A)\}$*. We say that* $\mathfrak{a}$ *is* conflict-free *iff* $\mathfrak{a} \cap \mathfrak{a}^+ = \emptyset$*. We say that* $\mathfrak{a}$ defends $(\mathfrak{M}, A)$ *iff each* $(\mathfrak{M}', B) \in \mathfrak{arr}$ *with* $B \in \mathfrak{M}$ *is attacked by* $\mathfrak{a}$*. We define the function* $F : 2^{\mathfrak{arr}} \to 2^{\mathfrak{arr}}$ *as follows:* $F(\mathfrak{a}) = \{(\mathfrak{M}, A) \in \mathfrak{arr} \mid (\mathfrak{M}, A)$ *is defended by* $\mathfrak{a}\}$*.*

Definition 19. *Let* $(\mathfrak{N}, \mathfrak{arr})$ *be a SETAF.* $\mathfrak{a} \subseteq \mathfrak{arr}$ *is called:*

(1) *an* admissible set *iff* $\mathfrak{a}$ *is conflict-free and* $\mathfrak{a} \subseteq F(\mathfrak{a})$
(2) *a* complete extension *iff* $\mathfrak{a}$ *is conflict-free and* $\mathfrak{a} = F(\mathfrak{a})$
(3) *a* grounded extension *iff* $\mathfrak{a}$ *is minimal (w.r.t.* $\subseteq$*) among all complete extensions*
(4) *a* preferred extension *iff* $\mathfrak{a}$ *is a maximal (w.r.t.* $\subseteq$*) complete extension*

(5) *a* semi-stable extension *iff* $\mathfrak{a}$ *is a complete extension where* $\mathfrak{a} \cup \mathfrak{a}^+$ *is maximal (w.r.t.* $\subseteq$*) among all complete extensions*
(6) *a* stable extension *iff* $\mathfrak{a}$ *is a complete extension with* $\mathfrak{a} \cup \mathfrak{a}^+ = \mathfrak{arr}$

An alternative way of defining arrow semantics for SETAF is by applying labellings, as is done in the following definition.

Definition 20. *Let* $(\mathfrak{N}, \mathfrak{arr})$ *be a SETAF. A SETAF arrow labelling is a function* $\mathfrak{ALab} : \mathfrak{arr} \to \{\mathtt{in}, \mathtt{out}, \mathtt{undec}\}$*. A SETAF arrow labelling* $\mathfrak{ALab}$ *is called admissible iff for each* $(\mathfrak{M}, A) \in \mathfrak{arr}$*:*

- *if* $\mathfrak{ALab}((\mathfrak{M}, A)) = \mathtt{in}$ *then for each* $(\mathfrak{M}', B) \in \mathfrak{arr}$ *such that* $B \in \mathfrak{M}$ *it holds that* $\mathfrak{ALab}((\mathfrak{M}', B)) = \mathtt{out}$
- *if* $\mathfrak{ALab}((\mathfrak{M}, A)) = \mathtt{out}$ *then there exists an* $(\mathfrak{M}', B) \in \mathfrak{arr}$ *such that* $B \in \mathfrak{M}$ *and* $\mathfrak{ALab}((\mathfrak{M}', B)) = \mathtt{in}$

A SETAF arrow labelling $\mathfrak{ALab}$ *is called complete iff it is admissible and satisfies for each* $(\mathfrak{M}, A) \in \mathfrak{arr}$*:*

- *if* $\mathfrak{ALab}((\mathfrak{M}, A)) = \mathtt{undec}$ *then not for each* $(\mathfrak{M}', B) \in \mathfrak{arr}$ *such that* $B \in \mathfrak{M}$ *it holds that* $\mathfrak{ALab}((\mathfrak{M}', B)) = \mathtt{out}$ *and there does not exist an* $(\mathfrak{M}', B) \in \mathfrak{arr}$ *such that* $B \in \mathfrak{M}$ *and* $\mathfrak{ALab}((\mathfrak{M}', B)) = \mathtt{in}$

As a convention, we write $\mathtt{in}(\mathfrak{ALab})$ *for* $\{(\mathfrak{M}, A) \in \mathfrak{arr} \mid \mathfrak{ALab}((\mathfrak{M}, A)) = \mathtt{in}\}$*,* $\mathtt{out}(\mathfrak{ALab})$ *for* $\{(\mathfrak{M}, A) \in \mathfrak{arr} \mid \mathfrak{ALab}((\mathfrak{M}, A)) = \mathtt{out}\}$ *and* $\mathtt{undec}(\mathfrak{ALab})$ *for* $\{(\mathfrak{M}, A) \in \mathfrak{arr} \mid \mathfrak{ALab}((\mathfrak{M}, A)) = \mathtt{undec}\}$*. A complete SETAF arrow labelling* $\mathfrak{ALab}$ *is called:*

(1) *grounded iff* $\mathtt{in}(\mathfrak{ALab})$ *is minimal (w.r.t.* $\subseteq$*) among all complete SETAF arrow labellings*
(2) *preferred iff* $\mathtt{in}(\mathfrak{ALab})$ *is maximal (w.r.t.* $\subseteq$*) among all complete SETAF arrow labellings*
(3) *semi-stable iff* $\mathtt{undec}(\mathfrak{ALab})$ *is minimal (w.r.t.* $\subseteq$*) among all complete SETAF arrow labellings*
(4) *stable iff* $\mathtt{undec}(\mathfrak{ALab}) = \emptyset$

As a SETAF arrow labelling essentially defines a partition, we sometimes write it as a triple $(\mathtt{in}(\mathfrak{ALab}), \mathtt{out}(\mathfrak{ALab}), \mathtt{undec}(\mathfrak{ALab}))$.

It can be shown that the complete arrow labellings with maximal `in` are equal to the complete arrow labellings with maximal `out` (Appendix E) and that the complete arrow labelling where `in` is minimal is unique and equal to both the complete arrow labelling where `out` is minimal and the complete arrow labelling where `undec` is maximal (Appendix E). This means that the grounded, preferred and semi-stable arrow labellings cover all possibilities regarding the minimisation and maximisation of a particular label (among the complete arrow labellings). This is summarised in Table 9.

It can be shown that arrow extensions and arrow labellings are one-to-one related through the functions $\mathfrak{ALab2a}$ and $\mathfrak{a2ALab}$, where

$$\mathfrak{ALab2a}(\mathfrak{ALab}) = \mathtt{in}(\mathfrak{ALab}),$$
$$\mathfrak{a2ALab}(\mathfrak{a}) = (\mathfrak{a}, \mathfrak{a}^+, \mathfrak{arr} \setminus (\mathfrak{a} \cup \mathfrak{a}^+))^6.$$

[6] We note that $\mathfrak{ALab2a}$ is defined for *conflict-free* sets $\mathfrak{a} \subseteq \mathfrak{arr}$.

Table 9

Implementing different conditions on complete arrow labellings of SETAF

condition on arrow labelling	resulting semantics
maximal `in`	preferred
maximal `out`	preferred
maximal `undec`	grounded
minimal `in`	grounded
minimal `out`	grounded
minimal `undec`	semi-stable
no `undec`	stable

If $\mathfrak{ALab}$ is a complete (resp. grounded, preferred, semi-stable or stable) arrow labelling, then $\mathfrak{ALab2a}(\mathfrak{ALab})$ is a complete (resp. grounded, preferred, semi-stable or stable) arrow extension, and if $\mathfrak{a}$ is a complete (resp. grounded, preferred, semi-stable or stable) arrow extension then $\mathfrak{a2ALab}(\mathfrak{a})$ is a complete (resp. grounded, preferred, semi-stable or stable) arrow labelling. Moreover, under complete semantics (as well as under grounded, preferred, semi-stable and stable semantics) $\mathfrak{ALab2a}$ and $\mathfrak{a2ALab}$ are bijective functions that are each other's inverses. Details and proofs can be found in Appendix J. Table 10 provides an overview of the results.

Table 10

The relation between arrow extensions and arrow labellings of SETAF, through $\mathfrak{a2ALab}$ and $\mathfrak{ALab2a}$

arrow extension	relation	arrow labelling
complete extension	$\Leftrightarrow$	complete labelling
grounded extension	$\Leftrightarrow$	grounded labelling
preferred extension	$\Leftrightarrow$	preferred labelling
semi-stable extension	$\Leftrightarrow$	semi-stable labelling
stable extension	$\Leftrightarrow$	stable labelling

3.3. On the Equivalence of Node Semantics and Arrow Semantics for SETAFs

It can be shown that SETAF arrow labellings and SETAF node labellings are one-to-one related through the functions $\mathfrak{ALab2NLab}$ and $\mathfrak{NLab2ALab}$, where

$$\begin{aligned}
\mathfrak{ALab2NLab}(\mathfrak{ALab}) = (&\{A \in \mathfrak{N} \mid \forall(\mathfrak{M}, A) \in \mathfrak{arr} : \mathfrak{ALab}((\mathfrak{M}, A)) = \texttt{out}\},\\
&\{A \in \mathfrak{N} \mid \exists(\mathfrak{M}, A) \in \mathfrak{arr} : \mathfrak{ALab}((\mathfrak{M}, A)) = \texttt{in}\},\\
&\{A \in \mathfrak{N} \mid \neg\forall(\mathfrak{M}, A) \in \mathfrak{arr} : \mathfrak{ALab}((\mathfrak{M}, A)) = \texttt{out} \text{ and}\\
&\qquad \neg\exists(\mathfrak{M}, A) \in \mathfrak{arr} : \mathfrak{ALab}((\mathfrak{M}, A)) = \texttt{in}\}),\\
\mathfrak{NLab2ALab}(\mathfrak{NLab}) = (&\{(\mathfrak{M}, A) \in \mathfrak{arr} \mid \forall B \in \mathfrak{M} : \mathfrak{NLab}(B) = \texttt{in}\},\\
&\{(\mathfrak{M}, A) \in \mathfrak{arr} \mid \exists B \in \mathfrak{M} : \mathfrak{NLab}(B) = \texttt{out}\},\\
&\{(\mathfrak{M}, A) \in \mathfrak{arr} \mid \neg\forall B \in \mathfrak{M} : \mathfrak{NLab}(B) = \texttt{in} \text{ and}\\
&\qquad \neg\exists B \in \mathfrak{M} : \mathfrak{NLab}(B) = \texttt{out}\}).
\end{aligned}$$

It can be shown that if $\mathfrak{ALab}$ is a complete (resp. grounded, preferred, semi-stable or stable) SETAF arrow labelling, then $\mathfrak{ALab2NLab}(\mathfrak{ALab})$ is a complete (resp. grounded, preferred or stable) SETAF node labelling, and if $\mathfrak{NLab}$ is a complete (resp. grounded, preferred or stable) node labelling, then $\mathfrak{NLab2ALab}(\mathfrak{NLab})$ is a complete (resp. grounded, preferred or stable) arrow labelling (see Appendix F for proofs).

However, the relation between SETAF arrow labellings and SETAF node labellings does *not* hold under semi-stable semantics (due to Remark 14, the same counter example as in AFs applies, i.e., Example 9 and Example 10). This is despite the fact that, under complete semantics (as well as under grounded, preferred, semi-stable and stable semantics) $\mathfrak{ALab2NLab}$ and $\mathfrak{NLab2ALab}$ are bijective functions and each other's inverses (see Appendix F for proofs). Theorem 21 and Table 11, respectively, provide an overview of these results.

Theorem 21. *Let $\mathfrak{SF} = (\mathfrak{N}, \mathfrak{arr})$ be a SETAF and let $\mathfrak{NLab}$ and $\mathfrak{ALab}$ be a node labelling and an arrow labelling of $\mathfrak{SF}$, respectively. It holds that:*

(1) *If $\mathfrak{NLab}$ is a complete node labelling, then $\mathfrak{NLab2ALab}(\mathfrak{NLab})$ is a complete arrow labelling.*
If $\mathfrak{ALab}$ is a complete arrow labelling, then $\mathfrak{ALab2NLab}(\mathfrak{ALab})$ is a complete node labelling.
(2) *When restricted to complete node labellings and complete arrow labellings, the functions $\mathfrak{ALab2NLab}$ and $\mathfrak{NLab2ALab}$ become bijections and each other's inverses.*
(3) *If $\mathfrak{NLab}$ is a grounded node labelling, then $\mathfrak{NLab2ALab}(\mathfrak{NLab})$ is a grounded arrow labelling.*
If $\mathfrak{ALab}$ is a grounded arrow labelling, then $\mathfrak{ALab2NLab}(\mathfrak{ALab})$ is a grounded node labelling.
(4) *If $\mathfrak{NLab}$ is a preferred node labelling, then $\mathfrak{NLab2ALab}(\mathfrak{NLab})$ is a preferred arrow labelling.*
If $\mathfrak{ALab}$ is a preferred arrow labelling, then $\mathfrak{ALab2NLab}(\mathfrak{ALab})$ is a preferred node labelling.
(5) *If $\mathfrak{NLab}$ is a stable node labelling, then $\mathfrak{NLab2ALab}(\mathfrak{NLab})$ is a stable arrow labelling.*
If $\mathfrak{ALab}$ is a stable arrow labelling, then $\mathfrak{ALab2NLab}(\mathfrak{ALab})$ is a stable node labelling.

Table 11
The relation between node labellings and arrow labellings of SETAFs, through $\mathfrak{NLab2ALab}$ and $\mathfrak{ALab2NLab}$

node labelling	relation	arrow labelling
complete node labelling	$\Leftrightarrow$	complete arrow labelling
grounded node labelling	$\Leftrightarrow$	grounded arrow labelling
preferred node labelling	$\Leftrightarrow$	preferred arrow labelling
semi-stable node labelling	$\not\Leftrightarrow$	semi-stable arrow labelling
stable node labelling	$\Leftrightarrow$	stable arrow labelling

Conflict-free node extensions and arrow extensions are one-to-one related via the functions

$$\mathfrak{Args2a} = \mathfrak{ALab2a} \circ \mathfrak{NLab2ALab} \circ \mathfrak{Args2NLab},$$

$$\mathfrak{a2Args} = \mathfrak{Args2NLab} \circ \mathfrak{ALab2NLab} \circ \mathfrak{ALab2a}.$$

Table 12 provides an overview (see Appendix K for proofs).

Table 12
The relation between node extensions and arrow extensions of SETAFs.

node extension	relation	arrow extension
complete node extension	$\Leftrightarrow$	complete arrow extension
grounded node extension	$\Leftrightarrow$	grounded arrow extension
preferred node extension	$\Leftrightarrow$	preferred arrow extension
semi-stable node extension	$\not\Leftrightarrow$	semi-stable arrow extension
stable node extension	$\Leftrightarrow$	stable arrow extension

4. Relating SETAFs to AFs

It is possible to translate a SETAF to an AF, and in the current section we will provide one particular way of doing so (other methods are presented, e.g., in [27]). The idea is that the arrows of the SETAF become the nodes of the AF.[7] As the arrows of the original framework become the nodes of the associated framework we say we turn the framework "inside-out". The proofs of this section can be found in Appendix L.

Definition 22. *Let $\mathfrak{SF} = (\mathfrak{N}, \mathfrak{arr})$ be a SETAF. We define the associated inside-out argumentation framework $AF_{\mathfrak{SF}}$ as (N, arr) with $N = \mathfrak{arr}$ and $arr = \{((\mathfrak{M}_1, A_1), (\mathfrak{M}_2, A_2)) \mid (\mathfrak{M}_1, A_1), (\mathfrak{M}_2, A_2) \in \mathfrak{arr}$ and $A_1 \in \mathfrak{M}_2\}$*

As a side effect of this translation, the arrow labellings of the SETAF become the node labellings of the associated argumentation framework.

Theorem 23. *Let $\mathfrak{SF} = (\mathfrak{N}, \mathfrak{arr})$ be a SETAF and $AF_{\mathfrak{SF}} = (N, arr)$ be the associated argumentation framework. It holds that $\mathfrak{ALab}$ is a complete (resp. grounded, preferred, semi-stable or stable) SETAF arrow labelling of $\mathfrak{SF}$ iff $\mathfrak{ALab}$ is a complete (resp. grounded, preferred, semi-stable or stable) node labelling of $AF_{\mathfrak{SF}}$.*

The relation described in Theorem 23 is summarised in Table 13. The fact that SETAF arrow labellings

Table 13
The relation between arrow labellings of a SETAF $\mathfrak{SF}$ and node labellings of its corresponding inside-out framework $AF_{\mathfrak{SF}}$

arrow labelling of $\mathfrak{SF}$	relation	node labelling of $AF_{\mathfrak{SF}}$
complete arrow labelling	$\Leftrightarrow$	complete node labelling
grounded arrow labelling	$\Leftrightarrow$	grounded node labelling
preferred arrow labelling	$\Leftrightarrow$	preferred node labelling
semi-stable arrow labelling	$\Leftrightarrow$	semi-stable node labelling
stable arrow labelling	$\Leftrightarrow$	stable node labelling

are equivalent to the associated argumentation framework node labellings (Theorem 23), together with the earlier observed equivalence between SETAF node labellings and SETAF arrow labellings (Theorem 21) allows for the connection between the node labellings of a SETAF and the node labellings of the associated argumentation framework. These are one-to-one related through the functions $\mathfrak{NLab2ALab}$ and

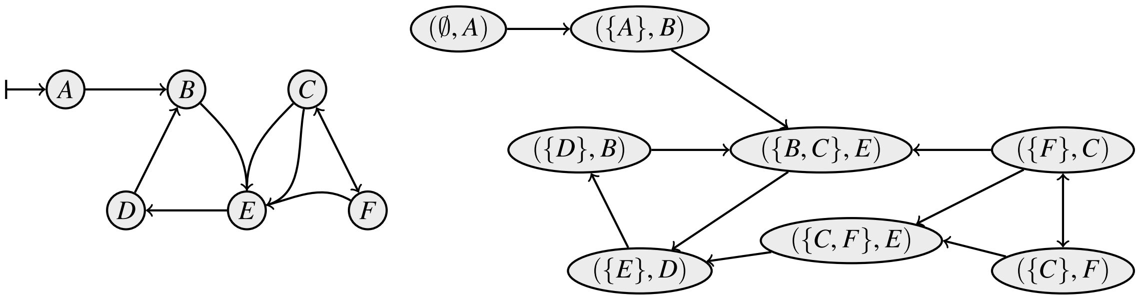
[7] This is similar to what was sketched at the end of Appendix A.

$\mathfrak{ALab}2\mathfrak{NLab}$, when applying complete, grounded, preferred, or stable semantics (but *not* when applying semi-stable semantics, as we have seen earlier). We will now provide an example for the "inside-out" framework of a SETAF.

Example 24. *Recall the SETAF $\mathfrak{SF} = (\mathfrak{N}, \mathfrak{arr})$ from Example 12 (see (a)). The associated inside-out argumentation framework is $AF_{\mathfrak{SF}} = (N, arr)$ with*

$$N = \big\{(\emptyset, A), (\{A\}, B), (\{B,C\}, E), (\{C,F\}, E), (\{C\}, F), (\{D\}, B), (\{E\}, D), (\{F\}, C)\big\}.$$

$$\begin{aligned} arr = \big\{ &\big((\emptyset, A),(\{A\}, B)\big), \big((\{A\}, B),(\{B,C\}, E)\big), \big((\{B,C\}, E),(\{E\}, D)\big), \big((\{D\}, B),(\{B,C\}, E)\big), \\ &\big((\{F\}, C),(\{B,C\}, E)\big), \big((\{E\}, D),(\{D\}, B)\big), \big((\{F\}, C),(\{C\}, F)\big), \big((\{F\}, C),(\{C,F\}, E)\big), \\ &\big((\{C\}, F),(\{F\}, C)\big), \big((\{C\}, F),(\{C,F\}, E)\big), \big((\{C,F\}, E),(\{E\}, D)\big)\big\} \end{aligned}$$



(a) *(b)*

$\mathfrak{SF}$ has three complete arrow labellings:

$$\mathfrak{ALab}_1 = (\{(\emptyset, A)\}, \{(\{A\}, B)\}, \{(\{B,C\}, E),(\{C,F\}, E),(\{C\}, F),(\{D\}, B),(\{E\}, D),(\{F\}, C)\})$$

$$\mathfrak{ALab}_2 = (\{(\emptyset, A), (\{F\}, C),(\{E\}, D)\}, \{(\{A\}, B),(\{C\}, F),(\{D\}, B),(\{B,C\}, E),(\{C,F\}, E)\}, \emptyset)$$

$$\mathfrak{ALab}_3 = (\{(\emptyset, A),(\{C\}, F)\}, \{(\{A\}, B),(\{F\}, C),(\{C,F\}, E)\}, \{(\{B,C\}, E),(\{D\}, B),(\{E\}, D)\})$$

$AF_{\mathfrak{SF}}$ has three complete node labellings:

$$NLab_1 = \mathfrak{ALab}_1,$$

$$NLab_2 = \mathfrak{ALab}_2,$$

$$NLab_3 = \mathfrak{ALab}_3$$

5. Instantiating AFs and SETAFs using ABA

In the current section, we show how assumption-based argumentation (ABA) can be used to instantiate both a standard AF and a SETAF. The common instantiation procedure translates a given ABA framework (ABAF) into a standard AF [34]. However, it turns out that the concepts underlying ABA are actually much closer in spirit to SETAFs. Here we discuss both instantiations and examine to what extent they can be considered equal to each other. Thereby, we will build upon our previously established results showing a close correspondence between the formalisms. We start with introducing ABAFs.

Definition 25 ([13])**.** *An ABAF is a tuple $D = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^{-})$ where:*

- *$(\mathcal{L}, \mathcal{R})$ is a deductive system, with $\mathcal{L}$ being a logical language, and $\mathcal{R}$ being a set of inference rules on this language,*
- *$\mathcal{A} \subseteq \mathcal{L}$ is a (non-empty) set of* assumptions*,*
- *$^{-}$ is the* contrary function*, i.e., a total mapping from $\mathcal{A}$ into $\mathcal{L}$; $\overline{\alpha}$ is called the contrary of α.*

For current purposes we only consider ABAFs that are *flat* in the sense of [12], which amounts to no assumption being the head of an inference rule.

Definition 26 ([13])**.** *Given an ABAF $D = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^{-})$, a* derivation tree *for $c \in \mathcal{L}$ (the* conclusion *or* claim*) supported by $Asms \subseteq \mathcal{A}$ is a finite tree t with nodes labelled by formulas in $\mathcal{L}$ or by the special symbol $\top$ such that:*

- *the root is labelled c*
- *for every node N*
  - $*$ *if N is a leaf then N is labelled either by an assumption or by $\top$*
  - $*$ *if N is not a leaf and b is the label of N, then there exists an inference rule $b \leftarrow b_1, \ldots, b_m$ ($m \geqslant 0$) and either $m = 0$ and the child of N is labelled by $\top$, or $m > 0$ and N has m children, labelled by $b_1, \ldots, b_m$ respectively*
- *$Asms(t)$ is the set of all assumptions labelling the leaves*

We denote by $cl(t) = c$ the conclusion of t.

Note that for each assumption $\gamma \in \mathcal{A}$, a trivial derivation tree consisting of a leaf labelled γ is induced.
Now that the notions of an ABAF and a derivation tree have been defined, we proceed with defining the ABA semantics [12, 34]. Historically, they have been defined in two different ways: using extensions of assumptions or using extensions of arguments [12, 13, 34]. We start with the more contemporary argument-based notion.

Definition 27. *Given an ABAF $D = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^{-})$, an* ABA-argument *is a pair $(Asms, c)$ where $Asms \subseteq \mathcal{A}$ and $c \in \mathcal{L}$ such that there is a derivation tree t with $cl(t) = c$ and $Asms(t) = Asms$. We say that an ABA-argument $(Asms_1, c_1)$* attacks *an ABA-argument $(Asms_2, c_2)$ iff $c_1 = \bar{\gamma}$ for some $\gamma \in Asms_2$.*

Our notion of an ABA-argument (Definition 27) is in line with the notation in the ABA literature [30], where a derivation tree is often denoted as $Asms \vdash c$. Notice, however, that as observed in [30] there can be several derivation trees (Definition 26) that yield the same assumptions-conclusion pair (Definition 27). However, for the purpose of the ABA semantics it does not matter whether one defines arguments as derivation trees or as pairs of assumptions and conclusions, since the semantics are characterised by considering this information only. Formally, ABA-arguments $(Asms, c)$ are equivalence classes of derivation trees for the purpose of argumentation semantics.
Using the notion of ABA-arguments and their attacks, it is straightforward to define the associated AF.

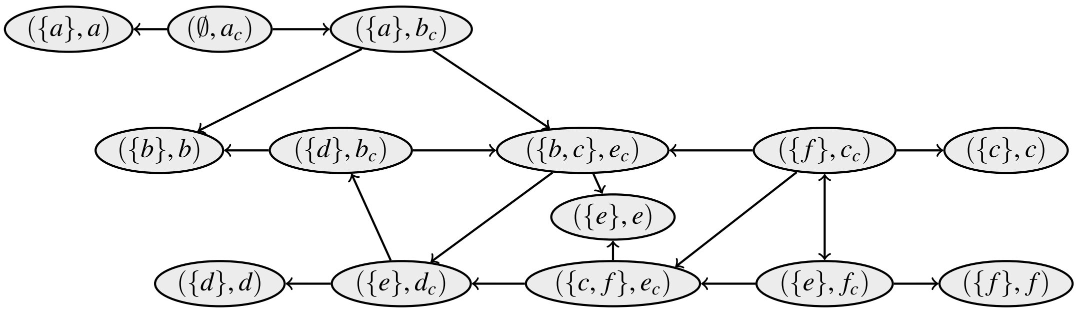
Definition 28. *Given an ABAF $D = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^{-})$, the associated AF AF_D is defined as (N, arr) with N being the set of ABA-arguments, and arr being the attack relation among ABA-arguments.*

The semantics of the ABAF D can now be determined by the semantics of the associated AF AF_D in the usual way. Due to the altered notion of ABA-arguments in Definition 27 as compared to [12, 30], the associated AF of an ABAF might contain fewer nodes (ABA-arguments) and arrows than the one constructed according to [30]. Nevertheless, the semantics correspond as proven in Appendix M.

Example 29. *We consider the ABAF* $D = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^-)$ *where* $\mathcal{A} = \{a, b, c, d, e, f\}$, $\mathcal{L} = \mathcal{A} \cup \{a_c \mid a \in \mathcal{A}\}$, $\overline{a} = a_c$ *for each* $a \in \mathcal{A}$, *and*

$$\begin{array}{llll} a_c \leftarrow & b_c \leftarrow d. & d_c \leftarrow e. & e_c \leftarrow b, c. \\ b_c \leftarrow a. & e_c \leftarrow c, f. & f_c \leftarrow c. & c_c \leftarrow f. \end{array}$$

In this example, each rule induces one ABA-argument, for instance $(\{c, f\}, e_c)$. *The attack relation is the natural one. Hence the AF* F_D *is given as follows.*



Note the structural similarity to the inside-out AF from Example 24 when ignoring the trivial ABA-arguments of the form $(\{a\}, a)$.

Now that we have provided the contemporary definition of ABA semantics which is based on extensions (resp. labellings) of *arguments*, we shift our attention to the more classical definition of ABA semantics which is based on extensions (resp. labellings) of *assumptions*. While the former can be captured by constructing an AF, the latter can be captured by a SETAF, which we are going to demonstrate subsequently. First, we define the semantics directly on the ABAF, without any kind of instantiation.

Definition 30. *Let* $D = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^-)$ *be an ABAF,* $Asms \subseteq \mathcal{A}$ *and* $\alpha \in \mathcal{A}$. *We say that Asms* attacks α *iff there is an ABA-argument* $(Asms', \overline{\alpha})$ *with* $Asms' \subseteq Asms$. *We say that* $Asms_1 \subseteq \mathcal{A}$ attacks $Asms_2 \subseteq \mathcal{A}$ *iff* $Asms_1$ *attacks some* $\beta \in Asms_2$. *We define* $Asms^+$ *as* $\{\alpha \in \mathcal{A} \mid Asms \text{ attacks } \alpha\}$. *We say that Asms is* conflict-free *iff* $Asms \cap Asms^+ = \emptyset$. *We say that Asms* defends α *iff for each* $Asms' \subseteq \mathcal{A}$ *that attacks* α, *Asms attacks* $Asms'$. *We define the function* $F : 2^{\mathcal{A}} \rightarrow 2^{\mathcal{A}}$ *as follows:* $F(Asms) = \{\alpha \in \mathcal{A} \mid \alpha \text{ is defended by } Asms\}$.

Definition 31. *Let* $D = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^-)$ *be an ABAF. The set* $Asms \subseteq \mathcal{A}$ *is called*

(1) *an* admissible assumption set *iff Asms is conflict-free and* $Asms \subseteq F(Asms)$
(2) *a* complete assumption extension *iff Asms is conflict-free and* $Asms = F(Asms)$
(3) *a* grounded assumption extension *iff Asms is the (unique) minimal (w.r.t.* $\subseteq$*) complete assumption extension*

(4) *a* preferred assumption extension *iff Asms is a maximal (w.r.t.* $\subseteq$*) complete assumption extension*
(5) *a* semi-stable assumption extension *iff Asms is a complete assumption extension where* $Asms \cup Asms^+$ *is maximal (w.r.t.* $\subseteq$*) among the complete assumption extensions*
(6) *a* stable assumption extension *iff Asms is a complete assumption extension with* $Asms \cup Asms^+ = \mathcal{A}$

Since the notion of ABA-arguments (Definition 27) influences the attack and defense relations between arguments (Definition 30), it also affects the definition of semantics of an ABAF as compared to [12, 30]. However, the semantics of an ABAF are the same no matter whether our definition of arguments or the one in [30] is used. It should also be noticed that description of preferred and stable semantics in Definition 31 is slightly different from [12, 30]. We prove equivalence in Appendix M.

Theorem 32. *Let* $D = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^{-})$ *be an ABAF.*

(1) *The set* $Asms \subseteq \mathcal{A}$ *is an admissible assumption set according to Definitions 26-31 iff Asms is an admissible assumption set according to the definitions in [12, 30].*
(2) *The set* $Asms \subseteq \mathcal{A}$ *is a complete (resp. grounded, preferred, semi-stable or stable) assumption extension according to Definitions 26-31 iff Asms is a complete (resp. grounded, preferred, semi-stable or stable) assumption extension according to the definitions in [12, 29, 30]*

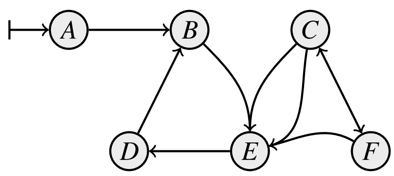
We next consider the SETAF instantiation of an ABAF. The SETAF is close to the original ABAF as for instance stated in [12, 30]. Instead of computing all derivation trees to construct an AF, the SETAF we instantiate only contains the set $\mathcal{A}$ of assumptions as arguments. Then, the induced attacks are defined in a very natural way: if $(Asms, \overline{\gamma})$ is an ABA-argument, then the set *Asms* attacks the assumption γ. Since SETAFs have the modeling capabilities of capturing this, we can simply use this as the definition of our attack relation. In summary, this yields the following definition of the associated SETAF.

Definition 33. *Given an ABAF* $D = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^{-})$*, the associated SETAF* $\mathfrak{S}\mathfrak{F}_D$ *is defined as* $(\mathfrak{N}, \mathfrak{arr})$ *with* $\mathfrak{N} = \mathcal{A}$ *and* $\mathfrak{arr} = \{(Asms, \gamma) \mid (Asms, \overline{\gamma}) \text{ is an ABA-argument with } \gamma \in \mathcal{A}\}$.

Example 34. *Recall the previous ABAF* $D = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^{-})$ *where* $\mathcal{A} = \{a, b, c, d, e, f\}$, $\mathcal{L} = \mathcal{A} \cup \{a_c \mid a \in \mathcal{A}\}$, $\overline{a} = a_c$ *for each* $a \in \mathcal{A}$*, and rules*

$a_c \leftarrow$	$b_c \leftarrow d.$	$d_c \leftarrow e.$	$e_c \leftarrow b, c.$
$b_c \leftarrow a.$	$e_c \leftarrow c, f.$	$f_c \leftarrow c.$	$c_c \leftarrow f.$

Each assumption in $\mathcal{A}$ *induces one argument in our SETAF* $\mathfrak{S}\mathfrak{F}_D$*. Moreover, the attacks can be read off of the rules: for instance, c and f collectively attack e due to the rule "*$e_c \leftarrow c, f.$*" Consequently,* $\mathfrak{S}\mathfrak{F}_D$ *is given as follows.*



Again, note the structural similarity to the SETAF from Example 24.

It is not difficult to see that the extensions of the thus constructed SETAF correspond to the traditional ABA extensions of assumptions.

Theorem 35. *Let $D = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^{-})$ be an ABAF, $\mathfrak{S}\mathfrak{F}_D$ be the associated SETAF, and Asms $\subseteq \mathcal{A}$. It holds that*

(1) *Asms is a complete extension of $\mathfrak{S}\mathfrak{F}_D$ iff Asms is a complete extension of D in the sense of [12, 30];*
(2) *Asms is a grounded extension of $\mathfrak{S}\mathfrak{F}_D$ iff Asms is a grounded extension of D in the sense of [12, 30];*
(3) *Asms is a preferred extension of $\mathfrak{S}\mathfrak{F}_D$ iff Asms is a preferred extension of D in the sense of [12, 30];*
(4) *Asms is a semi-stable extension of $\mathfrak{S}\mathfrak{F}_D$ iff Asms is a semi-stable extension of D in the sense of [35];*
(5) *Asms is a stable extension of $\mathfrak{S}\mathfrak{F}_D$ iff Asms is a stable extension of D in the sense of [12, 30].*

Hence, the essential difference between the classical ABA definitions (in which semantics are defined in terms of assumptions [12, 30]) and the more contemporary ABA-AF instantiation (in which semantics are defined in terms of arguments [13]) is that the classical ABA defintitions are based on a SETAF, in which the ABA-arguments form the arrows, whereas the ABA-AF instantiation is based on an AF, in which the ABA-arguments form the nodes.[8]

Now that the difference between the classical ABA definitions and the ABA-AF instantiation has been made clear, we can shed new light on the issue of whether these are actually equivalent. The idea is to apply the abstract theory on AF labellings and SETAF labellings in the particular context of ABA. This is done using the following steps:

(1) Start with the SETAF generated by the ABA SETAF instantiation. The complete (resp. grounded, preferred, semi-stable or stable) assumption extensions of the ABA-framework correspond one-to-one with the complete (resp. grounded, preferred, semi-stable and stable) node labellings of the SETAF.
(2) Convert the SETAF node labellings to the associated SETAF arrow labellings (that is, apply the concept of arrow-semantics to the SETAF). Since each arrow of the SETAF is associated with an ABA-argument, this will essentially define a labelling of ABA-arguments.
(3) Convert the SETAF into an AF (the associate inside-out AF, cf. Definition 22). The arrows of the SETAF become the nodes of the AF. The arrow labellings of the SETAF become the node labellings of the AF.
(4) The thus obtained AF is almost equivalent to the ABA-AF instantiation. However, two things still need to be taken care of. First, we should restore the contrary signs in the conclusions of the ABA-arguments, which were lost when generating the SETAF at step 1. Secondly, it can be observed that the resulting graph only contains the *attacking* ABA-arguments (arguments whose conclusion is the contrary of an assumption) whereas the ABA-AF instantiation also contains the *non-attacking* ABA-arguments (arguments whose conclusion is *not* the contrary of an assumption). These need to be added to the graph. The thus added nodes might have in-going arrows, but do not have any out-going arrows. Consequently, the node labellings can be extended in a straightforward way. Thereby, the overall set of accepted assumptions is preserved. The result will be the ABA-AF instantiation, with associated labellings.

---

[8]It should be observed, however, that in the SETAF not all ABA-arguments are represented, as only ABA-arguments that attack an assumption (that is, ABA-arguments whose conclusion is the contrary of an assumption) can be used as arrows of the SETAF.

At step 4, the effect of adding the non-attacking ABA-arguments basically means going from an AF $AF_1 = (N_1, arr_1)$ to an AF $AF_2 = (N_2, arr_2)$ such that

$$N_1 \subseteq N_2 \qquad \text{and} \qquad arr_2 \cap (N_1 \times N_1) = arr_1 \tag{1}$$

(no new arrows are added among the nodes that were already there) and

$$arr_2 \cap ((N_2 \setminus N_1) \times N_2) = \emptyset \tag{2}$$

(the new nodes do not have out-going arrows). In this situation, the complete (resp. grounded, preferred or stable) node labellings of AF_2 can be computed straightforwardly if we are given the labellings of AF_1. While this is intuitively clear, from a formal point of view we make use of the *directionality* principle here [36].

Proposition 36. *Let $AF_1 = (N_1, arr_1)$ and $AF_2 = (N_2, arr_2)$ be two AFs such that conditions (1) and (2) are met.*

(i) If $NLab_1$ is a complete (resp. grounded, preferred or stable) node labelling of AF_1, then

$$\begin{aligned} NLab_2 = NLab_1 &\cup \{(A, \mathtt{in}) \mid A \in N_2 \setminus N_1,\ \forall (B,A) \in arr_2 : NLab_1(B) = \mathtt{out}\} \\ &\cup \{(A, \mathtt{out}) \mid A \in N_2 \setminus N_1,\ \exists (B,A) \in arr_2 : NLab_1(B) = \mathtt{in}\} \\ &\cup \{(A, \mathtt{undec}) \mid A \in N_2 \setminus N_1,\ \neg\forall (B,A) \in arr_2 : NLab_1(B) = \mathtt{out}, \\ &\qquad\qquad \neg\exists (B,A) \in arr_2 : NLab_1(B) = \mathtt{in}\} \end{aligned}$$

is a complete (resp. grounded, preferred or stable) node labelling of AF_2.

(ii) If $NLab_2$ is a complete (resp. grounded, preferred or stable) node labelling of AF_2, then $NLab_1$ is a complete (resp. grounded, preferred or stable) node labelling of AF_1.

It can be observed that the thus defined conversion functions between node labellings of AF_1 and node labellings of AF_2 are bijective functions that are each other's inverses. Hence, the complete (resp. grounded, preferred and stable) node labellings of AF_1 and the complete (resp. grounded, preferred and stable) node labellings of AF_2 are one-to-one related.

We now formalise that the relation between the inside-out AF associated to $\mathfrak{SF}_D$ and the AF AF_D corresponding to D meets the conditions described in Proposition 36.

Proposition 37. *Let D be an ABAF. Let $\mathfrak{SF}_D$ be the associated SETAF and let $AF_{\mathfrak{SF}_D}$ be its inside-out AF. Let AF_D be the AF associated with D. Then the relation between $AF_1 := AF_{\mathfrak{SF}_D}$ and $AF_2 := AF_D$ is as described in (1) and (2) (up to argument names).*

The equivalences observed in the ABA-literature between extensions of assumptions (using the classical definition of ABA) and extensions of arguments (using the ABA-AF instantiation) follow from the above described theory on AFs and SETAFs. Let $D = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^-)$ be an ABAF and let $AF_D = (N, arr)$ be the associated AF (the ABA-AF instantiation). We define the functions $Asms2Args : 2^{\mathcal{A}} \to 2^N$ and $Args2Asms : 2^N \to 2^{\mathcal{A}}$ as follows:

$$Asms2Args(Asms) = \{(Asms', c) \in N \mid Asms' \subseteq Asms\}$$

and

$$Args2Asms(M) = \bigcup_{(Asms',c) \in M} Asms'.$$

We are now ready to give an alternative proof for the (known) result that an ABAF D can be evaluated by computing the semantics of the associated AF AF_D. Our proof will make use of several relations we showed throughout the present paper.[9]

Theorem 38. *Let $D = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^{-})$ be an ABA framework and $AF_D = (N, arr)$ be the associated argumentation framework.*

(1) *If Asms is a complete (resp. grounded, preferred or stable) assumption extension of D, then $Asms2Args(Asms)$ is a complete (resp. grounded, preferred or stable) extension of AF_D.*
(2) *If M is a complete (resp. grounded, preferred or stable) extension of AF_D then $Args2Asms(M)$ is a complete (resp. grounded, preferred or stable) assumption extension of D.*
(3) *When restricted to complete (resp. grounded, preferred and stable) assumption extensions of D and complete (resp. grounded, preferred and stable) extensions of AF_D the functions Asms2Args and Args2Asms are bijective and each other's inverses.*

Proof. We only do the case of complete semantics (the other semantics can be handled in a similar way).

(1) Let *Asms* be a complete assumption extension of D. We proceed in several steps, combining the results from our paper.
(i) By Theorem 35, the set *Asms* is a complete node extension of the associated SETAF $\mathfrak{S}\mathfrak{F}_D$.
(ii) Now let $\mathfrak{NLab}$ be the SETAF node labelling associated with *Asms*. From the correspondence between extensions and labellings (see Table 8), it follows that $\mathfrak{NLab}$ is a complete SETAF node labelling of $\mathfrak{S}\mathfrak{F}_D$.
(iii) In the next step, let $\mathfrak{ALab}$ be the associated SETAF arrow labelling associated with $\mathfrak{NLab}$. By Theorem 21, it holds that $\mathfrak{ALab}$ is a complete SETAF arrow labelling of $\mathfrak{S}\mathfrak{F}_D$.
(iv) Let $AF_{\mathfrak{S}\mathfrak{F}_D}$ be the inside-out AF that results from converting the SETAF $\mathfrak{S}\mathfrak{F}_D$ (Definition 22). By Theorem 23, it holds that $\mathfrak{ALab}$ is a complete node labelling of $AF_{\mathfrak{S}\mathfrak{F}_D}$.
(v) Now let AF_D be the AF associated to D. By Proposition 37, it holds that AF_D is the result of taking $AF_{\mathfrak{S}\mathfrak{F}_D}$ and adding some nodes without out-going arrows. Since $\mathfrak{ALab}$ is a complete node labelling of $AF_{\mathfrak{S}\mathfrak{F}}$ it follows that the labelling $\mathfrak{ALab}'$ resulting from applying Proposition 36 is a complete node labelling of AF_D.
(vi) Let M be the associated complete extension. It holds that M consists of precisely those ABA-arguments whose assumptions are in *Asms*. This can be seen as follows: Each attacking ABA-argument whose assumptions are in *Asms* will have been represented as an arrow in the SETAF that was labelled `in` by the SETAF arrow labelling (this is because the assumptions *Asms* were labelled `in` as nodes of the SETAF); consequently it was still labelled `in` by the node labelling of $AF_{\mathfrak{S}\mathfrak{F}}$, so still labelled `in` by the node labelling of AF_D, and thus an element of M. We have left to deal with the auxiliary nodes without out-going arrows. As we already mentioned, the node

[9]Our main contribution here is not the (known) correspondence between ABA and AFs, but the way we can proof this result by means of our theory. We thereby shed new light on this correspondence. Consequently, we include the proof here instead of moving it to the appendix.

labelling of AF_D can be obtained by applying Proposition 36 to $AF_{\mathfrak{S}\mathfrak{F}}$. So we analyse which further arguments are labelled `in`: By construction of AF_D, the additional nodes are ABA-arguments of the form $(Asms', \gamma)$ s.t. γ is not the contrary of any assumption in D. Since in-going arrows in ABA are determined by the assumptions required to construct an argument, $(Asms', \gamma)$ is defended by our complete labelling of AF_D iff $Asms' \subseteq Asms$. In this case, $(Asms', \gamma)$ is labelled `in` (Proposition 36); otherwise it is not. Consequently, the set *Asms* of accepted assumptions does not change when moving from $AF_{\mathfrak{S}\mathfrak{F}}$ to AF_D and applying Proposition 36 to the novel nodes without out-going arrows.

(2) Let M be a complete extension of AF_D and let *Asms* be the set

$$Asms = \bigcup_{(Asms',c)\in M} Asms'.$$

All of the aforementioned steps are applicable in both directions, so we can

(i) remove suitable nodes from AF_D to yield correspondence to the inside-out AF $AF_{\mathfrak{S}\mathfrak{F}_D}$ associated with $\mathfrak{S}\mathfrak{F}_D$;
(ii) move from the node labelling of $AF_{\mathfrak{S}\mathfrak{F}_D}$ to an arrow labelling of $\mathfrak{S}\mathfrak{F}_D$;
(iii) move from the arrow labelling of $\mathfrak{S}\mathfrak{F}_D$ to a node labelling of $\mathfrak{S}\mathfrak{F}_D$;
(iv) move from the node labelling of $\mathfrak{S}\mathfrak{F}_D$ to a node extension of $\mathfrak{S}\mathfrak{F}_D$.

By the same chain of reasoning, the thus obtained set $Asms'$ is a complete assumption set of D.

(3) In each step of the above proof, the applied mappings are bijective and each other's inverses. Consequently, this also applies to the whole process altogether. Note in particular that, as we pointed out, the auxiliary nodes added to move from the inside-out SETAF to the associated AF do not impact the accepted assumptions. □

An interesting observation is that this proof can be used to explain at which step the transformation fails for semi-stable semantics: While most of the steps would actually also work for semi-stable semantics (applying Theorem 35, Theorem 23, Table 8, and Proposition 37), the problem arises in step (iii) where we apply Theorem 21 moving from node labellings to arrow labellings in the SETAF under consideration.

6. Discussion

In this paper we thoroughly investigated abstract argumentation semantics in several dimensions: we studied *extension-based* and *labelling-based node* and *arrow semantics* for *standard argumentation frameworks* and for *argumentation frameworks with collective attacks (SETAFs)*, and investigated the relation between these semantics. We mapped out the entire space of possibilities, also regarding what happens when minimising or maximising a particular label. We systematically filled the gaps in the literature and introduced arrow extensions for AFs as well as arrow extensions and labellings for SETAFs (cf. red coloured elements in Figure 2). We studied the relation to the already established semantics and compared the obtained results for SETAFs with the results for AFs. We showed that each SETAF can be "turned inside-out" and efficiently transformed into a semantically corresponding AF (cf. red coloured arc in Figure 2). Note that arrow semantics for SETAFs can be defined in various ways (for example,

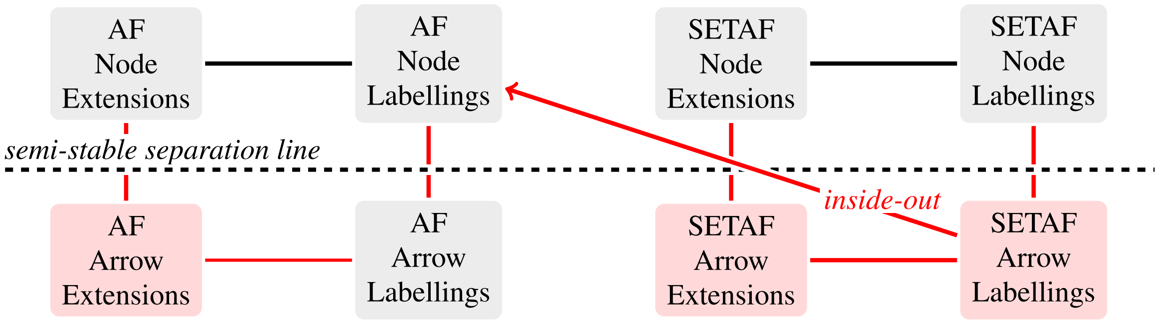


Fig. 2. The separation line for semi-stable semantics (dashed) indicates transformations for which semi-stable semantics is not preserved. The arc (red coloured, *inside-out* labelled) between SETAF Arrow Labellings and AF Node Labellings visualises the result from Theorem 23: each SETAF arrow labelling corresponds to the node labelling of the corresponding inside-out AF. Notably, semi-stable semantics are preserved in this translation. The contributions of this paper are highlighted in red colour.

preferred extensions can be defined in terms of $\subseteq$-maximal admissible or $\subseteq$-maximal complete extensions). We immediately obtain the same flexibility for these definitions that we are used to from AF semantics by applying Theorem 23.

Finally, we applied our findings in the realm of structured argumentation. We pointed out that for ABA frameworks our inside-out frameworks capture the instantiation process: while traditionally the semantics on the abstract level are evaluated via AFs, the original definitions are closer to the semantics of SETAFs. If this obtained SETAF is turned "inside-out", then we obtain the traditional instantiated AF.

The subtle difference of semi-stable semantics. We want to highlight that the semantics correspondence holds for some of the established semantics, with the notable exception of semi-stable semantics. As indicated in Figure 2, semi-stable semantics does not survive each transformation step. The parting line lies between node and arrow semantics for both AFs and SETAFs. Figure 2 shows that our transformations preserve all considered semantics on the horizontal axis, i.e., transforming node extensions to node labellings (arrow extensions and arrow labellings, respectively) and vice versa; but they fail to preserve semi-stable semantics vertically, i.e., for switching from node extensions to arrow extensions (node labellings to arrow labellings, respectively) and vice versa. Interestingly, when moving from SETAFs to AFs via the inside-out transformation, we switch levels: as shown in Section 4, semi-stable SETAF arrow semantics correspond to semi-stable AF node semantics of the resulting AF and vice versa.

The heterogeneous behaviour of semi-stable semantics has been already observed in several different settings, e.g., when comparing logic programs and AFs [11], in the context of ABA and AFs [29], and for different variants of claim semantics [37–39]. In our present work, we show that these differences can be found even within the same formalism, when comparing node and arrow semantics. Utilising our novel transformation from SETAFs to AFs, we furthermore reveal that AF node semantics and SETAF arrow semantics lie in the same category (with respect to semi-stable semantics).

An alternative view on attack semantics. Arrow semantics on AFs have been initially introduced as "attack semantics" [26]. However, our intuition differs slightly from the initial definition: in [26] the arrows (i.e., attacks) are partitioned into "successful" and "unsuccessful"; these correspond to the `in`/`undec` and `out` labelled arrows, respectively. In contrast, our arrow extensions capture precisely the `in` labelled arrows. In this way, we establish a closer correspondence to the traditional three-valued node labellings for AFs which are based on the same intuition [40]. We want to emphasise however that the labelling-based arrow semantics coincide with the notions of [26].

We furthermore note that labellings for SETAFs that assign both arrows and nodes `in`, `out`, or `undec` labels have been already considered [41]. In contrast, we consider arrow labelling semantics separately from node labellings. The generalisation of both arrow extension and labelling semantics to SETAFs is a novel contribution of this paper.

A radically new approach to flattening. Whether the simplicity of AFs is an advantage or a disadvantage has not yet been conclusively clarified; in any case, recent years have witnessed numerous generalisations of standard argumentation frameworks, e.g., generalisations that allow for preferences [42], values [43], collective attacks [14], as discussed in the present work, or a support relation [44], just to name a few. Regardless of other potential shortcomings, a unified model has its advantages; and in response to any generalisations, methods have been developed to reintegrate the generalisations into standard argumentation frameworks. The *flattening technique* [28] is the task to translate generalised argumentation frameworks into AFs. Usually, additional arguments are introduced to compensate the increased expressiveness of the generalised model [27, 45]. Moreover, while translations from SETAFs to AFs exist (cf. [27]) these methods usually capture the semantic correspondence in the arguments (under projection).

Our approach is radically different: each arrow in the SETAF becomes a node in the AF. Instead of handling the increased expressiveness with additional arguments, we exploit the close correspondence of arrow labellings of SETAFs to node labellings of AFs. That is, starting from a given SETAF,

(1) we exploit the connection between node extensions and node labellings (for SETAFs);
(2) we switch from node labellings to arrow labellings (for SETAFs);
(3) we turn the framework inside-out—now, each arrow in the SETAF becomes a node in the AF;
(4) we move from node labellings to node extensions (for AFs).

In this way, we obtain the desired AF *without any additional arguments*. Due to the correspondence of node labellings of AFs and arrow labellings on SETAFs we can establish a semantic correspondence between the nodes of a SETAF and its inside-out AF. This method is successful for all except semi-stable semantics when comparing the node semantics of the original SETAF instance with the node semantics of the obtained AF. As discussed above, semi-stable semantics is not preserved in point (2), i.e., when switching from node labellings to arrow labellings. We note that the proof of Theorem 38 in Section 5 makes use of the close connection of the formalisms we discussed.

Finally, we want to point out that a model that is similar to our inside-out AF has been studied in the context of dynamics in argumentation. For any given SETAF, the resulting inside-out AF resembles a cvAF [46]—an argumentation framework with explicit claims (conclusions) and vulnerabilities. For an arrow $(\mathfrak{M}, A)$ of the original SETAF $\mathfrak{SF}$ (i.e., a node of the associated inside-out argumentation framework $AF_{\mathfrak{SF}}$) the claim is A and the vulnerabilities are the elements of $\mathfrak{M}$.

Future Work. For future work, we want to investigate further semantics in all of the considered dimensions. In particular, we want to study ideal and eager semantics [47]. We anticipate that eager semantics admit a behaviour that is similar to semi-stable semantics. Ideal semantics, on the other hand, are expected to behave in line with the classical Dung semantics. Furthermore, the investigation of more involved semantics like $cf2$ [48], *stage*2 [49] or the more recent weak admissibility [50] would be a challenging, yet exciting task.

Abstract argumentation formalisms have been extensively investigated in terms of formal properties, principles [18, 36], and the expressive power of standard argumentation frameworks and their generalisations [38, 51–53]. However, the aforementioned results focus on node extensions in the respective formalism. It would be insightful to explore to which extent our inter-translations can help to study these properties also for e.g. arrow extensions.

Acknowledgements

This research has been supported by the Vienna Science and Technology Fund (WWTF) through project ICT19-065, the Austrian Science Fund (FWF) through project P32830, and by the Federal Ministry of Education and Research of Germany and by Sächsische Staatsministerium für Wissenschaft, Kultur und Tourismus in the programme Center of Excellence for AI-research "Center for Scalable Data Analytics and Artificial IntelligenceDresden/Leipzig", project identification number: ScaDS.AI.

References

[1] G.R. Simari and R.P. Loui, A mathematical treatment of defeasible reasoning and its implementation, *Artificial Intelligence* **53** (1992), 125–157.
[2] A.J. García, J. Dix and G.R. Simari, Argument-based Logic Programming, in: *Argumentation in Artificial Intelligence*, G. Simari and I. Rahwan, eds, Springer US, 2009, pp. 153–171. ISBN 978-0-387-98196-3.
[3] G.A.W. Vreeswijk, Abstract argumentation systems, *Artificial Intelligence* **90** (1997), 225–279.
[4] M.W.A. Caminada and L. Amgoud, On the evaluation of argumentation formalisms, *Artificial Intelligence* **171**(5–6) (2007), 286–310.
[5] H. Prakken, An abstract framework for argumentation with structured arguments, *Argument and Computation* **1**(2) (2010), 93–124.
[6] S. Modgil and H. Prakken, A general account of argumentation with preferences, *Artificial Intellligence* **195** (2013), 361–397.
[7] S. Modgil and H. Prakken, Abstract Rule-Based Argumentation, in: *Handbook of Formal Argumentation*, Vol. 1, College Publications, 2018, pp. 287–364.
[8] P.M. Dung, On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and *n*-person games, *Artificial Intelligence* **77**(2) (1995), 321–357.
[9] G. Governatori, M.J. Maher, G. Antoniou and D. Billington, Argumentation Semantics for Defeasible Logic, *Journal of Logic and Computation* **14**(5) (2004), 675–702.
[10] Y. Wu, M.W.A. Caminada and D.M. Gabbay, Complete Extensions in Argumentation Coincide with 3-Valued Stable Models in Logic Programming, *Studia Logica* **93**(1–2) (2009), 383–403, Special issue: new ideas in argumentation theory.
[11] M.W.A. Caminada, S. Sá, J. Alcântara and W. Dvořák, On the Equivalence between Logic Programming Semantics and Argumentation Semantics, *International Journal of Approximate Reasoning* **58** (2015), 87–111.
[12] A. Bondarenko, P.M. Dung, R.A. Kowalski and F. Toni, An abstract, argumentation-theoretic approach to default reasoning, *Artificial Intelligence* **93** (1997), 63–101.
[13] P.M. Dung, R.A. Kowalski and F. Toni, Assumption-Based Argumentation, in: *Argumentation in Artificial Intelligence*, G. Simari and I. Rahwan, eds, Springer US, 2009, pp. 199–218. ISBN 978-0-387-98196-3.
[14] S.H. Nielsen and S. Parsons, A Generalization of Dung's Abstract Framework for Argumentation: Arguing with Sets of Attacking Arguments, in: *Proceedings of ArgMAS 2006*, LNAI 4766, 2006, pp. 54–73.
[15] G. Flouris and A. Bikakis, A comprehensive study of argumentation frameworks with sets of attacking arguments, *International Journal of Approximate Reasoning* **109** (2019), 55–86. doi:10.1016/j.ijar.2019.03.006.
[16] W. Dvořák, A. Rapberger and S. Woltran, On the different types of collective attacks in abstract argumentation: equivalence results for SETAFs, *Journal of Logic and Computation* **30**(5) (2020), 1063–1107. doi:10.1093/logcom/exaa033.
[17] A. Bikakis, A. Cohen, W. Dvořák, G. Flouris and S. Parsons, Joint Attacks and Accrual in Argumentation Frameworks, *FLAP* **8**(6) (2021), 1437–1501. https://collegepublications.co.uk/ifcolog/?00048.
[18] W. Dvořák, M. König, M. Ulbricht and S. Woltran, Principles and their Computational Consequences for Argumentation Frameworks with Collective Attacks, *Journal of Artificial Intelligence Research* **79** (2024), 69–136.

[19] W. Dvořák, J. Fandinno and S. Woltran, On the expressive power of collective attacks, *Argument Comput.* **10**(2) (2019), 191–230. doi:10.3233/AAC-190457.

[20] W. Dvořák, A. Greßler and S. Woltran, Evaluating SETAFs via Answer-Set Programming, in: *Proceedings of SAFA 2018*, CEUR Workshop Proceedings, Vol. 2171, CEUR-WS.org, 2018, pp. 10–21. http://ceur-ws.org/Vol-2171/paper_2.pdf.

[21] W. Dvořák, M. König and S. Woltran, On the Complexity of Preferred Semantics in Argumentation Frameworks with Bounded Cycle Length, in: *Proceedings of KR 2021*, 2021, pp. 671–675. doi:10.24963/kr.2021/67.

[22] W. Dvořák, M. König and S. Woltran, Deletion-Backdoors for Argumentation Frameworks with Collective Attacks, in: *Proceedings of SAFA 2022*, Vol. 3236, 2022, pp. 98–110. http://ceur-ws.org/Vol-3236/paper8.pdf.

[23] W. Dvořák, M. König and S. Woltran, Treewidth for Argumentation Frameworks with Collective Attacks, in: *Proceedings of COMMA 2022*, FAIA, Vol. 353, IOS Press, 2022, pp. 140–151.

[24] F. Toni, Reasoning on the Web with Assumption-Based Argumentation, in: *Reasoning Web. Semantic Technologies for Advanced Query Answering*, T. Eiter and T. Krennwallner, eds, 2012, pp. 370–386.

[25] C. Schultz and F. Toni, Complete Assumption Labellings, in: *Proceedings of COMMA 2014*, 2014, pp. 405–412.

[26] S. Villata, G. Boella and L. van der Torre, Attack Semantics for Abstract Argumentation, in: *Proceedings of IJCAI 2011*, 2011, pp. 164–171.

[27] S. Polberg, Developing the Abstract Dialectical Framework, PhD thesis, Vienna University of Technology, Institute of Information Systems, 2017. https://permalink.obvsg.at/AC13773888.

[28] G. Boella, D.M. Gabbay, L.W.N. van der Torre and S. Villata, Meta-Argumentation Modelling I: Methodology and Techniques, *Stud Logica* **93**(2–3) (2009), 297–355. doi:10.1007/s11225-009-9213-2.

[29] M.W.A. Caminada, S. Sá, J. Alcântara and W. Dvořák, On the Difference between Assumption-Based Argumentation and Abstract Argumentation, *IfCoLog Journal of Logic and its Applications* **2** (2015), 15–34.

[30] P.M. Dung, P. Mancarella and F. Toni, Computing ideal sceptical argumentation, *Artificial Intelligence* **171**(10–15) (2007), 642–674.

[31] M. König, A. Rapberger and M. Ulbricht, Just a Matter of Perspective – Intertranslating Expressive Argumentation Formalisms, in: *Proceedings of COMMA 2022*, FAIA, Vol. 353, IOS Press, 2022, pp. 212–223.

[32] M.W.A. Caminada, On the Issue of reinstatement in Argumentation, in: *Proceedings of JELIA 2006*, Springer, 2006, pp. 111–123, LNAI 4160.

[33] M.W.A. Caminada and D.M. Gabbay, A logical account of formal argumentation, *Studia Logica* **93**(2–3) (2009), 109–145, Special issue: new ideas in argumentation theory.

[34] K. Čyras, X. Fan, C. Schulz and F. Toni, Assumption-Based Argumentation: Disputes, Explanations, Preferences, in: *Handbook of Formal Argumentation*, Vol. 1, College Publications, 2018, pp. 365–408.

[35] M.W.A. Caminada, S. Sá and J. Alcântara, On the Equivalence between Logic Programming Semantics and Argumentation Semantics, in: *Proceedings ECSQARU 2013*, 2013, pp. 97–108.

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

[37] A. Rapberger, Defining Argumentation Semantics under a Claim-centric View, in: *Proceedings of STAIRS 2020*, CEUR Workshop Proceedings, Vol. 2655, CEUR-WS.org, 2020. https://ceur-ws.org/Vol-2655/paper2.pdf.

[38] W. Dvořák, A. Rapberger and S. Woltran, Argumentation Semantics under a Claim-centric View: Properties, Expressiveness and Relation to SETAFs, in: *Proceedings of KR 2020*, 2020, pp. 341–350.

[39] W. Dvorák, A. Rapberger and S. Woltran, A claim-centric perspective on abstract argumentation semantics: Claim-defeat, principles, and expressiveness, *Artif. Intell.* **324** (2023), 104011. doi:10.1016/J.ARTINT.2023.104011. https://doi.org/10.1016/j.artint.2023.104011.

[40] B. Verheij, Two approaches to dialectical argumentation: admissible sets and argumentation stages, in: *Proceedings of NAIC 1996*, 1996, pp. 357–368.

[41] W. Dvořák, A. Rapberger and J.P. Wallner, Labelling-based Algorithms for SETAFs, in: *Proceedings of SAFA 2020*, CEUR Workshop Proceedings, Vol. 2672, CEUR-WS.org, 2020, pp. 34–46. https://ceur-ws.org/Vol-2672/paper_4.pdf.

[42] L. Amgoud and C. Cayrol, On the acceptability of arguments in preference-based argumentation framework, in: *Proceedings of UAI 1998*, 1998, pp. 1–7.

[43] T.J.M. Bench-Capon, Persuasion in Practical Argument Using Value-based Argumentation Frameworks, *Journal of Logic and Computation* **13**(3) (2003), 429–448.

[44] C. Cayrol and M. Lagasquie-Schiex, On the Acceptability of Arguments in Bipolar Argumentation Frameworks, in: *Proceedings of ECSQARU 2005*, LNCS, Vol. 3571, Springer, 2005, pp. 378–389.

[45] D.M. Gabbay, Semantics for Higher Level Attacks in Extended Argumentation Frames Part 1: Overview, *Stud Logica* **93**(2–3) (2009), 357–381. doi:10.1007/s11225-009-9211-4.

[46] A. Rapberger and M. Ulbricht, On Dynamics in Structured Argumentation Formalisms, *Journal of Artificial Intelligence Research* **77** (2023), 563–643. doi:10.1613/JAIR.1.14481. https://doi.org/10.1613/jair.1.14481.

[47] M.W.A. Caminada, Comparing two unique extension semantics for formal argumentation: ideal and eager, in: *Proceedings of BNAIC 2007*, 2007, pp. 81–87.

[48] P. Baroni, M. Giacomin and G. Guida, SCC-recursiveness: a general schema for argumentation semantics, *Artificial Intelligence* **168**(1–2) (2005), 165–210.
[49] W. Dvořák and S.A. Gaggl, Stage semantics and the SCC-recursive schema for argumentation semantics, *Journal of Logic and Computation* **26**(4) (2016), 1149–1202. doi:10.1093/logcom/exu006.
[50] R. Baumann, G. Brewka and M. Ulbricht, Shedding new light on the foundations of abstract argumentation: Modularization and weak admissibility, *Artificial Intelligence* **310** (2022), 103742. doi:10.1016/j.artint.2022.103742.
[51] R. Baumann and H. Strass, On the Maximal and Average Numbers of Stable Extensions, in: *Proceedings of TAFA 2013*, LNCS, Vol. 8306, Springer, 2013, pp. 111–126. doi:10.1007/978-3-642-54373-9_8.
[52] P.E. Dunne, W. Dvořák, T. Linsbichler and S. Woltran, Characteristics of multiple viewpoints in abstract argumentation, *Artificial Intelligence* **228** (2015), 153–178. doi:10.1016/j.artint.2015.07.006.
[53] M. Ulbricht, On the Maximal Number of Complete Extensions in Abstract Argumentation Frameworks, in: *Proceedings of KR 2021*, 2021, pp. 707–711.
[54] P. Baroni, M.W.A. Caminada and M. Giacomin, An Introduction to Argumentation Semantics, *Knowledge Engineering Review* **26**(4) (2011), 365–410.

Appendix A. Properties of Arrow Labellings for Argumentation Frameworks

The labelling-based version of complete attack-semantics is defined in a slightly different way in [26]. Instead of characterising a complete arrow labelling using three if-statements, as is done in Definition 7, it is characterised using two iff-statements [26, Theorem 7]. However, it can be proved that these two characterisations are equivalent.

Theorem 39. *Let $AF = (N, arr)$ be an argumentation framework and let ALab be an arrow labelling of AF. ALab is a complete arrow labelling of AF iff for every $(A, B) \in arr$ it holds that:*

(1) $ALab((A, B)) = \mathtt{in}$ *iff for each $(C, A) \in arr$ it holds that $ALab((C, A)) = \mathtt{out}$*
(2) $ALab((A, B)) = \mathtt{out}$ *iff there exists a $(C, A) \in arr$ such that $ALab((C, A)) = \mathtt{in}$*

Proof. "$\Rightarrow$ ": Let *ALab* be a complete arrow labelling of *AF*. Point 1 LTR follows directly from the first bullet of Definition 7. As for point 1 RTL, suppose that for each $(C, A) \in arr$ it holds that $ALab((C, A)) = \mathtt{out}$. Then $ALab((A, B))$ cannot be $\mathtt{out}$ (otherwise there would have to be a $(C, A) \in arr$ such that $ALab((C, A)) = \mathtt{in}$) and cannot be $\mathtt{undec}$ (otherwise not for each $(C, A) \in arr$ it holds that $ALab((C, A)) = \mathtt{out}$). Since $ALab((A, B))$ can only be $\mathtt{in}$, $\mathtt{out}$ or $\mathtt{undec}$, it then follows that $ALab((A, B)) = \mathtt{in}$.
Point 2 LTR follows directly from the second bullet of Definition 7. As for point 2 RTL, suppose that there exists a $(C, A) \in arr$ such that $ALab((C, A)) = \mathtt{in}$. Then $ALab((A, B))$ cannot be $\mathtt{in}$ (otherwise for each $(C, A) \in arr$ it holds that $ALab((C, A)) = \mathtt{out}$) and cannot be $\mathtt{undec}$ (otherwise there does not exist a $(C, A) \in arr$ such that $ALab((C, A)) = \mathtt{in}$). Since $ALab((A, B))$ can only be $\mathtt{in}$, $\mathtt{out}$ or $\mathtt{undec}$, it follows that $ALab((A, B)) = \mathtt{out}$.
"$\Leftarrow$": Let *ALab* be an arrow labelling satisfying points 1 and 2. We now need to show that *ALab* also satisfies the first three bullets of Definition 7. The first bullet follows directly from point 1. The second bullet follows directly from point 2. As for the third bullet, take an arbitrary $(A, B) \in arr$ such that $ALab((A, B)) = \mathtt{undec}$. The fact that $ALab((A, B)) \neq \mathtt{in}$, together with point 1, then implies that not for each $(C, A) \in arr$ it holds that $ALab((C, A)) = \mathtt{out}$. The fact that $ALab((A, B)) \neq \mathtt{out}$, together with point 2, then implies that there does not exist a $(C, A) \in arr$ such that $ALab((C, A)) = \mathtt{in}$. $\square$

We now proceed to prove a number of lemmas on how arrow labellings relate to each other, and how arrow labellings relate to node labellings.

Lemma 40. *Let $ALab_1$ and $ALab_2$ be complete arrow labellings of argumentation framework $AF = (N, arr)$. It holds that:*

(1) $\mathtt{in}(ALab_1) \subseteq \mathtt{in}(ALab_2)$ *iff* $\mathtt{out}(ALab_1) \subseteq \mathtt{out}(ALab_2)$
(2) $\mathtt{in}(ALab_1) = \mathtt{in}(ALab_2)$ *iff* $\mathtt{out}(ALab_1) = \mathtt{out}(ALab_2)$
(3) $\mathtt{in}(ALab_1) \subsetneq \mathtt{in}(ALab_2)$ *iff* $\mathtt{out}(ALab_1) \subsetneq \mathtt{out}(ALab_2)$

Proof. (1) "$\Rightarrow$": Suppose $\mathtt{in}(ALab_1) \subseteq \mathtt{in}(ALab_2)$. Let $(A, B) \in \mathtt{out}(ALab_1)$. This means that (Definition 7, second bullet point) there exists a $(C, A) \in arr$ such that $ALab_1((C, A)) = \mathtt{in}$. That is, $(C, A) \in \mathtt{in}(ALab_1)$. The fact that $\mathtt{in}(ALab_1) \subseteq \mathtt{in}(ALab_2)$ then implies that $(C, A) \in \mathtt{in}(ALab_2)$. This, together with the fact that $ALab_2$ is a complete arrow labelling implies (by point 2 of Theorem 39) that $(A, B) \in \mathtt{out}(ALab_2)$.
"$\Leftarrow$ ": Suppose $\mathtt{out}(ALab_1) \subseteq \mathtt{out}(ALab_2)$. Let $(A, B) \in \mathtt{in}(ALab_1)$. This means that (Definition 7, first bullet point) that for each $(C, A) \in arr$ it holds that $(C, A) \in \mathtt{out}(ALab_1)$. The fact that $\mathtt{out}(ALab_1) \subseteq \mathtt{out}(ALab_2)$ then implies that $(C, A) \in \mathtt{out}(ALab_2)$. This, together with the fact that $ALab_2$ is a complete arrow labelling implies (by point 1 of Theorem 39) that $(A, B) \in \mathtt{in}(ALab_2)$.

(2) This follows directly from point 1.
(3) This follows directly from point 1 and point 2.
□

Lemma 41. *Let $ALab_1$ and $ALab_2$ be complete arrow labellings of argumentation framework $AF = (N, arr)$. It holds that:*

(1) *if* $\mathtt{in}(ALab_1) \subseteq \mathtt{in}(ALab_2)$ *then* $\mathtt{undec}(ALab_1) \supseteq \mathtt{undec}(ALab_2)$
(2) *if* $\mathtt{in}(ALab_1) = \mathtt{in}(ALab_2)$ *then* $\mathtt{undec}(ALab_1) = \mathtt{undec}(ALab_2)$
(3) *if* $\mathtt{in}(ALab_1) \subsetneq \mathtt{in}(ALab_2)$ *then* $\mathtt{undec}(ALab_1) \supsetneq \mathtt{undec}(ALab_2)$
(4) *if* $\mathtt{out}(ALab_1) \subseteq \mathtt{out}(ALab_2)$ *then* $\mathtt{undec}(ALab_1) \supseteq \mathtt{undec}(ALab_2)$
(5) *if* $\mathtt{out}(ALab_1) = \mathtt{out}(ALab_2)$ *then* $\mathtt{undec}(ALab_1) = \mathtt{undec}(ALab_2)$
(6) *if* $\mathtt{out}(ALab_1) \subsetneq \mathtt{out}(ALab_2)$ *then* $\mathtt{undec}(ALab_1) \supsetneq \mathtt{undec}(ALab_2)$

Proof. (1) Suppose $\mathtt{in}(ALab_1) \subseteq \mathtt{in}(ALab_2)$. Then (Lemma 40, point 1) it follows that $\mathtt{out}(ALab_1) \subseteq \mathtt{out}(ALab_2)$. Let $(A, B) \in \mathtt{undec}(ALab_2)$. Then $(A, B) \notin \mathtt{in}(ALab_2)$ so $(A, B) \notin \mathtt{in}(ALab_1)$. Also, $(A, B) \notin \mathtt{out}(ALab_2)$ so $(A, B) \notin \mathtt{out}(ALab_1)$. From the fact that (A, B) is labelled either $\mathtt{in}$, $\mathtt{out}$ or $\mathtt{undec}$ by $ALab_1$, it follows that $(A, B) \in \mathtt{undec}(ALab_1)$.
(2) This follows directly from point 1.
(3) This follows directly from point 1 of this lemma, and Lemma 40 (point 3) and the fact that every arrow is labelled either $\mathtt{in}$, $\mathtt{out}$, or $\mathtt{undec}$.
(4) This follows directly from Lemma 40 (point 1) and point 1 of this lemma.
(5) This follows directly from point 4.
(6) This follows directly from point 4 of this lemma, and Lemma 40 (point 3) and the fact that every arrow is labelled either $\mathtt{in}$, $\mathtt{out}$, or $\mathtt{undec}$.
□

The following theorem states that minimising (resp. maximising) particular labels sometimes yields the same outcome.

Theorem 42. *Let $AF = (N, arr)$ be an argumentation framework, and let ALab be a complete arrow labelling of AF. The following two statements are equivalent:*

1. $\texttt{in}(ALab)$ *is maximal (w.r.t. set inclusion) among all complete arrow labellings of AF*
2. $\texttt{out}(ALab)$ *is maximal (w.r.t. set inclusion) among all complete arrow labellings of AF*

The following three statements are also equivalent:

3. $\texttt{in}(ALab)$ *is minimal (w.r.t. set inclusion) among all complete arrow labellings of AF*
4. $\texttt{out}(ALab)$ *is minimal (w.r.t. set inclusion) among all complete arrow labellings of AF*
5. $\texttt{undec}(ALab)$ *is maximal (w.r.t. set inclusion) among all complete arrow labellings of AF*

Furthermore, it holds that the complete arrow labelling with minimal in *is unique.*

Proof. **from 1 to 2** Suppose $\texttt{in}(ALab)$ is maximal among all complete arrow labellings of AF. That is, there is no complete arrow labelling $ALab'$ of AF such that $\texttt{in}(ALab) \subsetneq \texttt{in}(ALab')$. Suppose, towards a contradiction, that $\texttt{out}(ALab)$ is not maximal among all complete arrow labellings of AF. Then there exists a complete arrow labelling $ALab'$ such that $\texttt{out}(ALab) \subsetneq \texttt{out}(ALab')$. It then follows from Lemma 40 (point 3) that $\texttt{in}(ALab) \subsetneq \texttt{in}(ALab')$. Contradiction.

from 2 to 1 Similar to the previous point.

from 3 to 4 Suppose $\texttt{in}(ALab)$ is minimal among all complete arrow labellings of AF. That is, there is no complete arrow labelling $ALab'$ of AF such that $\texttt{in}(ALab') \subsetneq \texttt{in}(ALab)$. Suppose, towards a contradiction, that $\texttt{out}(ALab)$ is not minimal among all complete arrow labellings of AF. Then there exists a complete arrow labelling $ALab'$ such that $\texttt{out}(ALab') \subsetneq \texttt{out}(ALab)$. It then follows from Lemma 40 (point 3) that $\texttt{in}(ALab') \subsetneq \texttt{in}(ALab)$. Contradiction.

from 4 to 3 Similar to the previous point.

from 5 to 3 Suppose $\texttt{undec}(ALab)$ is maximal among all complete arrow labellings of AF. That is, there is no complete arrow labelling $ALab'$ of AF such that $\texttt{undec}(ALab) \subsetneq \texttt{undec}(ALab')$. Suppose, towards a contradiction, that $\texttt{in}(ALab)$ is not maximal among all complete arrow labellings of AF. Then there exists a complete arrow labelling $ALab'$ such that $\texttt{in}(ALab) \subsetneq \texttt{in}(ALab')$. It then follows from Lemma 41 (point 3) that $\texttt{undec}(ALab') \subsetneq \texttt{undec}(ALab)$. Contradiction.

As for the the last point to be proved (from 3 to 5), a particular difficulty is that we cannot just use the same proof strategy as the previous point (from 5 to 3). This is because point 3 of Lemma 41 only goes one-way (it's an "if" instead of an "iff"). To overcome this, we will need to make use of the uniqueness of the grounded arrow labelling.

uniqueness grounded arrow labelling Suppose $ALab_1$ and $ALab_2$ are complete arrow labellings of AF with minimal in. That is, they are grounded arrow labellings of AF. From Lemma 51 it follows[10] that $NLab_1 = ALab2NLab(ALab_1)$ and $NLab_2 = ALab2NLab(ALab_2)$ are grounded node labellings of AF. However, since the grounded node labelling is unique [33] it follows that $NLab_1 = NLab_2$, so also $NLab2ALab(NLab_1) = NLab2ALab(NLab_2)$. However, since $NLab2ALab(NLab_1) = ALab_1$ and $NLab2ALab(NLab_2) = ALab_2$ (by Lemma 48)[11] it follows that $ALab_1 = ALab_2$.

[10]Note that the proof of Lemma 51 does not depend on this result or its consequences.
[11]Note that the proof of Lemma 48 does not depend on this result or its consequences.

Using the uniqueness of the grounded arrow labelling, we can then proceed to show that point 3 implies point 5.

from 3 to 5 Suppose $ALab_1$ is a complete arrow labelling of AF with minimal in. That is, $ALab_1$ is a minimal element of the set of complete arrow labellings of AF (when applying an ordering based on set-inclusion on the in-labelled part of the labellings). As this minimal element is unique, it is also the smallest element, meaning that it is less or equal to each element of the set. That is, for each complete arrow labelling $ALab'$ of AF, it holds that $\texttt{in}(ALab) \subseteq \texttt{in}(ALab')$. From Lemma 41 (point 1) it then follows that $\texttt{undec}(ALab') \subseteq \texttt{undec}(ALab)$. It then follows that there is there is no complete arrow labelling $ALab'$ of AF such that $\texttt{undec}(ALab) \subsetneq \texttt{undec}(ALab')$. That is, $ALab$ is a complete arrow labelling with maximal undec.

□

From Theorem 42 it follows that the grounded, preferred and semi-stable arrow labellings cover all possibilities regarding the maximisation and minimisation of a particular label (among the complete arrow labellings).

As an aside, there exists an alternative way of proving the correctness of Theorem 39, Lemma 40, Lemma 41 and Theorem 42. The idea is that where a node labelling is based on nodes attacking each other, an arrow labelling is based on arrows attacking each other. Whereas a node A attacks a node B iff $(A, B) \in arr$, an arrow (C, D) attacks an arrow (E, F) iff $D = E$. Hence, the notion of attack becomes a binary relation between the arrows of the argumentation framework. This binary relation can in its turn be represented in the form of a graph (a meta-graph of the original argumentation framework). The idea is to take the original argumentation framework and "turn it inside out". That is, the arrows of the original graph become the nodes of the new meta-graph.

Definition 43. *Let $AF = (N, arr)$ be an argumentation framework. The inside out argumentation framework of AF is defined as $AF' = (N', arr')$ with $N' = arr$ and $arr' = \{((A, B), (B, C)) \mid (A, B), (B, C) \in arr\}$.*

Theorem 44. *Let $AF = (N, arr)$ be an argumentation framework and $AF' = (N', arr')$ be its inside out argumentation framework.*

(1) *If $ALab$ is a complete arrow labelling of AF, then $ALab$ is a complete node labelling of AF'.*
If $NLab$ is a complete node labelling of AF', then $NLab$ is a complete arrow labelling of AF.
(2) *If $ALab$ is a preferred (resp. grounded or stable) arrow labelling of AF, then $ALab$ is a preferred (resp. grounded or stable) node labelling of AF'.*
If $NLab$ is a preferred (resp. grounded, stable or semi-stable) node labelling of AF', then $NLab$ is a preferred (resp. grounded, stable or semi-stable) arrow labelling of AF.

Proof. (1) This follows directly from the definition of a complete node labelling (Definition 4, first three bullet points), the definition of a complete arrow labelling (Definition 7, first three bullet points) and the definition of an inside out argumentation framework (Definition 43).

(2) This follows directly from point 1, together with the definition of a preferred (resp. grounded, stable or semi-stable) node labelling (Definition 4) and the definition of a preferred (resp. grounded, stable or semi-stable) arrow labelling (Definition 7).

□

As arrow labellings are essentially node labellings (of the inside out argumentation framework) they satisfy the standard properties of node labellings described in the literature. Hence, Theorem 39 follows from [33, Definition 5, Definition 6 and Theorem 1], Lemma 40 follows from [33, Lemma 1], Lemma 41 follows from [11, Lemma 2] and Theorem 42 follows from [33, Theorem 6, Theorem 7].

Appendix B. Equivalence of Node Labellings and Arrow Labellings for Argumentation Frameworks

Lemma 45. *Let $AF = (N, arr)$ be an argumentation framework. If NLab is a complete node labelling of AF then $ALab = NLab2ALab(NLab)$ is a complete arrow labelling of AF.*

Proof. We need to prove that the three bullet points of Definition 7 are satisfied. Let $(A, B) \in arr$. We distinguish three cases:

(1) $ALab((A, B)) = \texttt{in}$. From the definition of *NLab2ALab* it then follows that $NLab(A) = \texttt{in}$. From the fact that *NLab* is a complete node labelling, it then follows that $NLab(C) = \texttt{out}$ for each $C \in N$ that attacks A. From the definition of *NLab2ALab* it then follows that $ALab((C, A)) = \texttt{out}$.
(2) $ALab((A, B)) = \texttt{out}$. From the definition of *NLab2ALab* it then follows that $NLab(A) = \texttt{out}$. From the fact that *NLab* is a complete node labelling, it then follows that $NLab(C) = \texttt{in}$ for some $C \in N$ that attacks A. From the definition of *NLab2ALab* it then follows that $ALab((C, A)) = \texttt{in}$.
(3) $ALab((A, B)) = \texttt{undec}$. From the definition of *NLab2ALab* it then follows that $NLab(A) = \texttt{undec}$. From the fact that *NLab* is a complete node labelling, it then follows that not for each $C \in N$ that attacks A it holds that $NLab(C) = \texttt{out}$ and there is no $C \in N$ that attacks A such that $NLab(C) = \texttt{in}$. From the definition of *NLab2ALab* it then follows that not for each $(C, A) \in arr$ it holds that $ALab((C, A)) = \texttt{out}$, and there is no $(C, A) \in arr$ such that $ALab((C, A)) = \texttt{in}$.

□

Lemma 46. *Let $AF = (N, arr)$ be an argumentation framework. If ALab is a complete arrow labelling of AF then $NLab = ALab2NLab(ALab)$ is a complete node labelling of AF.*

Proof. We need to prove that the three bullet points of Definition 4 are satisfied. Let $A \in N$. We distinguish three cases:

(1) $NLab(A) = \texttt{in}$. We need to show that for every $B \in N$ that attacks A, it holds that $NLab(B) = \texttt{out}$. Let $B \in N$ be a node that attacks A. This means that $(B, A) \in arr$. From the definition of *ALab2NLab* and the fact that $NLab(A) = \texttt{in}$ it follows that $ALab((B, A)) = \texttt{out}$. From the fact that *ALab* is a complete arrow labelling it then follows that there exists a $(C, B) \in arr$ such that $ALab((C, B)) = \texttt{in}$. From the definition of *ALab2NLab* it then follows that $NLab(B) = \texttt{out}$.
(2) $NLab(A) = \texttt{out}$. We need to show that there exists a $B \in N$ that attacks A such that $NLab(B) = \texttt{in}$. From the definition of *ALab2NLab* and the fact that $NLab(A) = \texttt{out}$ it follows that there is a $(B, A) \in arr$ such that $ALab((B, A)) = \texttt{in}$. From the fact that *ALab* is a complete arrow labelling, it then follows that for each $C \in N$ that attacks B, it holds that $ALab((C, B)) = \texttt{out}$. From the definition of *ALab2NLab*, it then follows that $NLab(B) = \texttt{in}$.
(3) $NLab(A) = \texttt{undec}$. We need to show that not for all $B \in N$ that attack A it holds that $NLab(B) = \texttt{out}$, and that there is no $B \in N$ that attacks A such that $NLab(B) = \texttt{in}$. From the definition

of *ALab2NLab* and the fact that $NLab(A) = \texttt{undec}$ it follows that (i) not for all $(B,A) \in arr$ it holds that $ALab((B,A)) = \texttt{out}$, and (ii) there is no $(B,A) \in arr$ such that $ALab((B,A)) = \texttt{in}$. From (i) it follows that there is a $(B,A) \in arr$ such that $ALab((B,A)) \neq \texttt{out}$. From the fact that *ALab* is a complete arrow labelling, it then follows that there is no $C \in N$ that attacks B such that $ALab((C,B)) = \texttt{in}$, which from the definition of *ALab2NLab* implies that $NLab(B) \neq \texttt{out}$. From (ii) it follows that for each $(B,A) \in arr$ it holds that $ALab((B,A)) \neq \texttt{in}$. Let B be an arbitrary attacker of A. It directly follows that $ALab((B,A)) \neq \texttt{in}$. From the fact that *ALab* is a complete arrow labelling, it follows that there is a $(C,B) \in arr$ such that $ALab((C,B)) \neq \texttt{out}$. From the definition of *ALab2NLab*, it then follows that $NLab(B) \neq \texttt{in}$.

□

Lemma 47. *Let NLab be a complete node labelling of argumentation framework $AF = (N, arr)$. It holds that $ALab2NLab(NLab2ALab(NLab)) = NLab$.*

Proof. Let $ALab = NLab2ALab(NLab)$. It suffices to prove the following three properties, for an arbitrary $A \in N$.

(1) If $NLab(A) = \texttt{in}$ then $ALab2NLab(ALab)(A) = \texttt{in}$.
Suppose $NLab(A) = \texttt{in}$. Then from *NLab* being a complete node labelling, it follows that for each B that attacks A it holds that $NLab(B) = \texttt{out}$, which from the definition of *NLab2ALab* implies that $ALab((B,A)) = \texttt{out}$. From the definition of *ALab2NLab* it then follows that $ALab2NLab(ALab)(A) = \texttt{in}$

(2) If $NLab(A) = \texttt{out}$ then $ALab2NLab(ALab)(A) = \texttt{out}$.
Suppose $NLab(A) = \texttt{out}$. Then from *NLab* being a complete node labelling, it follows that there is a $B \in N$ that attacks A such that $NLab(B) = \texttt{in}$, which from the definition of *NLab2ALab* implies that $ALab((B,A)) = \texttt{in}$. From the definition of *ALab2NLab* it then follows that $ALab2NLab(ALab)(A) = \texttt{out}$

(3) If $NLab(A) = \texttt{undec}$ then $ALab2NLab(ALab)(A) = \texttt{undec}$.
Suppose $NLab(A) = \texttt{undec}$. Then from *NLab* being a complete node labelling, it follows that (i) not for each $B \in N$ that attacks A it holds that $NLab(B) = \texttt{out}$, and (ii) there is no $B \in N$ that attacks A such that $NLab(B) = \texttt{in}$. From (i) it directly follows that there exists a $B \in N$ that attacks A such that $NLab(B) \neq \texttt{out}$, thus $ALab((B,A)) \neq \texttt{out}$. Then, from the definition of *ALab2NLab* it follows that $ALab2NLab(ALab)(A) \neq \texttt{in}$. From (ii) it directly follows that for each $B \in N$ that attacks A it holds that $NLab(B) \neq \texttt{in}$. Then, from the definition of *NLab2ALab* it follows that $ALab((B,A)) \neq \texttt{in}$. Then, from the definition of *ALab2NLab* it follows that $ALab2NLab(ALab)(A) \neq \texttt{out}$. This, together with the earlier obtained fact that $ALab2NLab(ALab)(A) \neq \texttt{in}$ implies that $ALab2NLab(NLab)(A) = \texttt{undec}$.

□

Lemma 48. *Let ALab be a complete arrow labelling of argumentation framework $AF = (N, arr)$. It holds that $NLab2ALab(ALab2NLab(ALab)) = ALab$.*

Proof. Let $NLab = ALab2NLab(ALab)$. It suffices to prove the following three properties, for an arbitrary $(A,B) \in arr$.

(1) If $ALab((A,B)) = \texttt{in}$ then $NLab2ALab(NLab)((A,B)) = \texttt{in}$.
Suppose $ALab((A,B)) = \texttt{in}$. Then from the fact that *ALab* is a complete arrow labelling, it follows that for each C that attacks A, $ALab((C,A)) = \texttt{out}$. From the definition of *ALab2NLab* it then follows that $NLab(A) = \texttt{in}$. From the definition of *NLab2ALab* it then follows that $NLab2ALab(NLab)((A,B)) = \texttt{in}$.

(2) If $ALab((A,B)) = \texttt{out}$ then $NLab2ALab(NLab)((A,B)) = \texttt{out}$.
Suppose $ALab((A,B)) = \texttt{out}$. Then, from the fact that *ALab* is a complete arrow labelling, it follows that there exists a C that attacks A such that $ALab((C,A)) = \texttt{in}$. From the definition of *ALab2NLab* it then follows that $NLab(A) = \texttt{out}$. From the definition of *NLab2ALab* it then follows that $NLab2ALab(NLab)((A,B)) = \texttt{out}$.

(3) If $ALab((A,B)) = \texttt{undec}$ then $NLab2ALab(NLab)((A,B)) = \texttt{undec}$.
Suppose $ALab((A,B)) = \texttt{undec}$. Then from the fact that *ALab* is a complete arrow labelling, it follows that (i) not for each C that attacks A it holds that $ALab((C,A)) = \texttt{out}$, and (ii) there is no C that attacks A such that $ALab((C,A)) = \texttt{in}$. From (i) it directly follows that there is a C that attacks A such that $ALab((C,A)) \neq \texttt{out}$. From the definition of *ALab2NLab* it then follows that $NLab(A) \neq \texttt{in}$. From the definition of *NLab2ALab* it then follows that $NLab2ALab(NLab)((A,B)) \neq \texttt{in}$. From (ii) it directly follows that for each C that attacks A it holds that $ALab((C,A)) \neq \texttt{in}$. From the definition of *ALab2NLab* it then follows that $NLab(A) \neq \texttt{out}$. From the definition of *NLab2ALab* it then follows that $NLab2ALab(NLab)((A,B)) \neq \texttt{out}$. From this, together with the earlier observed fact that $NLab2ALab(NLab)((A,B)) \neq \texttt{in}$, it follows that $NLab2ALab(NLab)((A,B)) = \texttt{undec}$.

□

Lemma 49. *Let $NLab_1$ and $NLab_2$ be complete node labellings of an argumentation framework $AF = (N, arr)$. Let $ALab_1 = NLab2ALab(NLab_1)$ and $ALab_2 = NLab2ALab(NLab_2)$. It holds that:*

(1) $\texttt{in}(NLab_1) \subseteq \texttt{in}(NLab_2)$ *iff* $\texttt{in}(ALab_1) \subseteq \texttt{in}(ALab_2)$.
(2) $\texttt{in}(NLab_1) = \texttt{in}(NLab_2)$ *iff* $\texttt{in}(ALab_1) = \texttt{in}(ALab_2)$.
(3) $\texttt{in}(NLab_1) \subsetneq \texttt{in}(NLab_2)$ *iff* $\texttt{in}(ALab_1) \subsetneq \texttt{in}(ALab_2)$.
(4) $\texttt{out}(NLab_1) \subseteq \texttt{out}(NLab_2)$ *iff* $\texttt{out}(ALab_1) \subseteq \texttt{out}(ALab_2)$.
(5) $\texttt{out}(NLab_1) = \texttt{out}(NLab_2)$ *iff* $\texttt{out}(ALab_1) = \texttt{out}(ALab_2)$.
(6) $\texttt{out}(NLab_1) \subsetneq \texttt{out}(NLab_2)$ *iff* $\texttt{out}(ALab_1) \subsetneq \texttt{out}(ALab_2)$.
(7) *If* $\texttt{undec}(NLab_1) \subseteq \texttt{undec}(NLab_2)$ *then* $\texttt{undec}(ALab_1) \subseteq \texttt{undec}(ALab_2)$.
(8) *If* $\texttt{undec}(NLab_1) = \texttt{undec}(NLab_2)$ *then* $\texttt{undec}(ALab_1) = \texttt{undec}(ALab_2)$.

Proof. (1) "$\Rightarrow$ ": Suppose $\texttt{in}(NLab_1) \subseteq \texttt{in}(NLab_2)$. Let $(A,B) \in \texttt{in}(ALab_1)$. Then, from the definition of *NLab2ALab* it follows that $A \in \texttt{in}(NLab_1)$. The fact that $\texttt{in}(NLab_1) \subseteq \texttt{in}(NLab_2)$ then implies that $A \in \texttt{in}(NLab_2)$. From the definition of *NLab2ALab* it then follows that $(A,B) \in \texttt{in}(ALab_2)$.
"$\Leftarrow$ ": Suppose $\texttt{in}(ALab_1) \subseteq \texttt{in}(ALab_2)$. Let $A \in \texttt{in}(NLab_1)$. First assume that there is a node $B \in N$ such that $(A,B) \in arr$, i.e., A has an outgoing arrow. Then, from the definition of *NLab2ALab* it follows that for each $B \in N$ such that $(A,B) \in arr$ it holds $(A,B) \in \texttt{in}(ALab_1)$. The fact that $\texttt{in}(ALab_1) \subseteq \texttt{in}(ALab_2)$ then implies that $(A,B) \in \texttt{in}(ALab_2)$. From the definition of *NLab2ALab* it then follows that $A \in \texttt{in}(NLab_2)$ and we are done. Now on the other hand assume that there is no node $B \in N$ such that $(A,B) \in arr$, i.e., A has no outgoing arrows. Towards

contradiction assume that $A \notin \mathtt{in}(NLab_2)$. This means either A is not defended by $\mathtt{in}(NLab_2)$ or there is some $B \in \mathtt{in}(NLab_2)$ such that $(B, A) \in arr$. Since we assume $A \in \mathtt{in}(NLab_1)$ and $NLab_1$ is complete we know that for each arrow $(C, A) \in arr$ towards A there is a counter-attack $(D, C) \in arr$ with $D \in \mathtt{in}(NLab_1)$. From the definition of *NLab2ALab* it then follows that $(D, C) \in \mathtt{in}(ALab_1)$. From $\mathtt{in}(ALab_1) \subseteq \mathtt{in}(ALab_2)$ we then get $(D, C) \in \mathtt{in}(ALab_2)$ and, hence, from the definition of *NLab2ALab* we get $D \in \mathtt{in}(NLab_2)$. Since we chose an arbitrary attacker C of A, we know that A is defended by $\mathtt{in}(NLab_2)$. It remains to handle the case where there is some $B \in \mathtt{in}(NLab_2)$ such that $(B, A) \in arr$, i.e., A is not in $\mathtt{in}(NLab_2)$ due to a conflict. However, since $NLab_1$ is complete this means that $B \in \mathtt{out}(NLab_1)$, and therefore $(B, A) \in \mathtt{out}(ALab_1)$. From this and the fact that $\mathtt{in}(ALab_1) \subseteq \mathtt{in}(ALab_2)$ and from Lemma 40 (point 1) we get $(B, A) \in \mathtt{out}(ALab_2)$, which gives us from the definition of *NLab2ALab* that $B \in \mathtt{out}(NLab_2)$, a contradiction to our assumption $B \in \mathtt{in}(NLab_2)$.

(2) This follows directly from point 1.

(3) This follows directly from point 1 and point 2.

(4) "$\Rightarrow$": Suppose $\mathtt{out}(NLab_1) \subseteq \mathtt{out}(NLab_2)$. Let $(A, B) \in \mathtt{out}(ALab_1)$. Then, from the definition of *NLab2ALab* it follows that $A \in \mathtt{out}(NLab_1)$. The fact that $\mathtt{out}(NLab_1) \subseteq \mathtt{out}(NLab_2)$ then implies that $A \in \mathtt{out}(NLab_2)$. From the definition of *NLab2ALab* it then follows that $(A, B) \in \mathtt{out}(ALab_2)$.

"$\Leftarrow$": Suppose $\mathtt{out}(ALab_1) \subseteq \mathtt{out}(ALab_2)$. Let $A \in \mathtt{out}(NLab_1)$. Then, since $NLab_1$ is complete, there is a $B \in \mathtt{in}(NLab_1)$ such that $(B, A) \in arr$. From the definition of *NLab2ALab* it follows that $(B, A) \in \mathtt{in}(ALab_1)$. From our assumption $\mathtt{out}(ALab_1) \subseteq \mathtt{out}(ALab_2)$ and Lemma 40 (point 1) we get $\mathtt{in}(ALab_1) \subseteq \mathtt{in}(ALab_2)$, and therefore $(B, A) \in \mathtt{in}(ALab_2)$. From the definition of *NLab2ALab* it follows that $B \in \mathtt{in}(NLab_2)$, and since $NLab_2$ is complete we get $A \in \mathtt{out}(NLab_2)$.

(5) This follows directly from point 4.

(6) This follows directly from point 4 and point 5.

(7) Suppose $\mathtt{undec}(NLab_1) \subseteq \mathtt{undec}(NLab_2)$. Let $(A, B) \in \mathtt{undec}(ALab_1)$. Then, from the definition of *NLab2ALab* it follows that $A \in \mathtt{undec}(NLab_1)$. The fact that $\mathtt{undec}(NLab_1) \subseteq \mathtt{undec}(NLab_2)$ then implies that $A \in \mathtt{undec}(NLab_2)$. From the definition of *NLab2ALab* it then follows that $(A, B) \in \mathtt{undec}(ALab_2)$.

(8) This follows directly from point 7.

□

Note that the following similar statements **do not hold**:

7' If $\mathtt{undec}(ALab_1) \subseteq \mathtt{undec}(ALab_2)$ then $\mathtt{undec}(NLab_1) \subseteq \mathtt{undec}(NLab_2)$. A counter example is *AF* in Example 9: We have $\mathtt{undec}(ALab_2) \subseteq \mathtt{undec}(ALab_3)$ but $\mathtt{undec}(NLab_2) \nsubseteq \mathtt{undec}(NLab_3)$.

8' If $\mathtt{undec}(ALab_1) = \mathtt{undec}(ALab_2)$ then $\mathtt{undec}(NLab_1) = \mathtt{undec}(NLab_2)$. A counter example is *AF* in Example 9: We have $\mathtt{undec}(ALab_2) = \mathtt{undec}(ALab_3)$ but $\mathtt{undec}(NLab_2) \neq \mathtt{undec}(NLab_3)$.

9 If $\mathtt{undec}(NLab_1) \subsetneq \mathtt{undec}(NLab_2)$ then $\mathtt{undec}(ALab_1) \subsetneq \mathtt{undec}(ALab_2)$. A counter example is *AF* in Example 9: We have $\mathtt{undec}(NLab_3) \subsetneq \mathtt{undec}(NLab_2)$ but $\mathtt{undec}(NLab_3) \nsubseteq \mathtt{undec}(NLab_2)$.

9' If $\texttt{undec}(ALab_1) \subsetneq \texttt{undec}(ALab_2)$ then $\texttt{undec}(NLab_1) \subsetneq \texttt{undec}(NLab_2)$. A counter example is *AF* in Example 10: We have $\texttt{undec}(ALab_3) \subsetneq \texttt{undec}(ALab_2)$ but $\texttt{undec}(NLab_3) \not\subsetneq \texttt{undec}(NLab_2)$.

Lemma 50. *Let $ALab_1$ and $ALab_2$ be complete arrow labellings of argumentation framework $AF = (N, arr)$. Let $NLab_1 = ALab2NLab(ALab_1)$ and $NLab_2 = ALab2NLab(ALab_2)$. It holds that:*

(1) $\texttt{in}(ALab_1) \subseteq \texttt{in}(ALab_2)$ *iff* $\texttt{in}(NLab_1) \subseteq \texttt{in}(NLab_2)$.
(2) $\texttt{in}(ALab_1) = \texttt{in}(ALab_2)$ *iff* $\texttt{in}(NLab_1) = \texttt{in}(NLab_2)$.
(3) $\texttt{in}(ALab_1) \subsetneq \texttt{in}(ALab_2)$ *iff* $\texttt{in}(NLab_1) \subsetneq \texttt{in}(NLab_2)$.
(4) $\texttt{out}(ALab_1) \subseteq \texttt{out}(ALab_2)$ *iff* $\texttt{out}(NLab_1) \subseteq \texttt{out}(NLab_2)$.
(5) $\texttt{out}(ALab_1) = \texttt{out}(ALab_2)$ *iff* $\texttt{out}(NLab_1) = \texttt{out}(NLab_2)$.
(6) $\texttt{out}(ALab_1) \subsetneq \texttt{out}(ALab_2)$ *iff* $\texttt{out}(NLab_1) \subsetneq \texttt{out}(NLab_2)$.
(7) *If* $\texttt{undec}(NLab_1) \subseteq \texttt{undec}(NLab_2)$ *then* $\texttt{undec}(ALab_1) \subseteq \texttt{undec}(ALab_2)$.
(8) *If* $\texttt{undec}(NLab_1) = \texttt{undec}(NLab_2)$ *then* $\texttt{undec}(ALab_1) = \texttt{undec}(ALab_2)$.

Proof. (1) "⇒": Suppose $\texttt{in}(ALab_1) \subseteq \texttt{in}(ALab_2)$. Let $A \in \texttt{in}(NLab_1)$. Then, from the definition of *ALab2NLab* it follows that for every C that attacks A, $(C, A) \in \texttt{out}(ALab_1)$. From the fact that $\texttt{in}(ALab_1) \subseteq \texttt{in}(ALab_2)$ it follows that (Lemma 40) $\texttt{out}(ALab_1) \subseteq \texttt{out}(ALab_2)$, so $(C, A) \in \texttt{out}(ALab_2)$. From the definition of *ALab2NLab* it then follows that $NLab_2(A) = \texttt{in}$. That is, $A \in \texttt{in}(NLab_2)$.

"⇐": Suppose $\texttt{in}(NLab_1) \subseteq \texttt{in}(NLab_2)$. Let $(A, B) \in \texttt{in}(ALab_1)$. Since we assume that $ALab_1$ is complete, it follows that for every (C, A) towards A, $(C, A) \in \texttt{out}(ALab_1)$. This means for some $(D, C) \in arr$ we have $(D, C) \in \texttt{in}(ALab_1)$. Then, from the definition of *ALab2NLab* it follows that $C \in \texttt{out}(NLab_1)$. From $\texttt{in}(NLab_1) \subseteq \texttt{in}(NLab_2)$ and [54, Lemma 1] we then get $\texttt{out}(NLab_1) \subseteq \texttt{out}(NLab_2)$. From this and $C \in \texttt{out}(NLab_1)$ we get $C \in \texttt{out}(NLab_2)$. Then from the definition of *ALab2NLab* it follows that there is some $(D, C) \in \texttt{in}(ALab_2)$ which means $(C, A) \in \texttt{out}(ALab_2)$, and then since $ALab_2$ is complete we get $(A, B) \in \texttt{in}(ALab_2)$.

(2) This follows directly from point 1.
(3) This follows directly from point 1 and point 2.
(4) "⇒": Suppose $\texttt{out}(ALab_1) \subseteq \texttt{out}(ALab_2)$. Let $A \in \texttt{out}(NLab_1)$. Then, from the definition of *ALab2NLab* it follows that there exists a C that attacks A such that $(C, A) \in \texttt{in}(ALab_1)$. From the fact that $\texttt{out}(ALab_1) \subseteq \texttt{out}(ALab_2)$ it follows that (Lemma 40) $\texttt{in}(ALab_1) \subseteq \texttt{in}(ALab_2)$, so $(C, A) \in \texttt{in}(ALab_2)$. From the definition of *ALab2NLab* it then follows that $NLab_2(A) = \texttt{out}$. That is, $A \in \texttt{out}(NLab_2)$.

"⇐": Suppose $\texttt{out}(NLab_1) \subseteq \texttt{out}(NLab_2)$. Let $(A, B) \in \texttt{out}(ALab_1)$. Since we assume that $ALab_1$ is complete, it follows that there exists a (C, A) towards A such that $(C, A) \in \texttt{in}(ALab_1)$. By completeness this means for every $(D, C) \in arr$ towards C we have $(D, C) \in \texttt{out}(ALab_1)$. Then, from the definition of *ALab2NLab* it then follows that $C \in \texttt{in}(NLab_1)$. From $\texttt{out}(NLab_1) \subseteq \texttt{out}(NLab_2)$ and [54, Lemma 1] we then get $\texttt{in}(NLab_1) \subseteq \texttt{in}(NLab_2)$. From this and $C \in \texttt{in}(NLab_1)$ we get $C \in \texttt{in}(NLab_2)$. Then from the definition of *ALab2NLab* it follows that for every $(D, C) \in arr$ towards C we have $(D, C) \in \texttt{out}(ALab_2)$, which means $(C, A) \in \texttt{in}(ALab_2)$, and then since $ALab_2$ is complete we get $(A, B) \in \texttt{out}(ALab_2)$.

(5) This follows directly from point 4.
(6) This follows directly from point 4 and point 5.

(7) Suppose $\texttt{undec}(NLab_1) \subseteq \texttt{undec}(NLab_2)$. Let $(A,B) \in \texttt{undec}(ALab_1)$. Then, by completeness of $ALab_1$ it follows that for no $(C,A) \in arr$ it holds $(C,A) \in \texttt{in}(ALab_1)$ and there is some $(C,A) \in arr$ such that $(C,A) \notin \texttt{out}(ALab_1)$, i.e., $(C,A) \in \texttt{undec}(ALab_1)$. From the definition of *ALab2NLab* it follows that $A \in \texttt{undec}(NLab_1)$. The fact that $\texttt{undec}(NLab_1) \subseteq \texttt{undec}(NLab_2)$ then implies that $A \in \texttt{undec}(NLab_2)$. From the definition of *ALab2NLab* it follows then that for no $(C,A) \in arr$ towards A it holds $(C,A) \in \texttt{in}(ALab_2)$ and not for all $(C,A) \in arr$ we have $(C,A) \in \texttt{out}(ALab_2)$. From this we get $(A,B) \in \texttt{undec}(ALab_2)$.

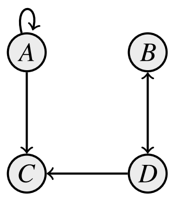
(8) This follows directly from point 7.

$\square$

Notice that the respective missing cases do not hold—analogous to Lemma 49 (the same counter examples apply in this case). In fact, the similarities between Lemma 49 and Lemma 50 are no coincidence, they are a direct consequence of the fact that the functions *NLab2ALab* and *ALab2NLab* are each others inverses (see Theorem 8, point 2).

To illustrate why these cases do not hold, we recall Example 9 from Section 2 (see below). It holds that $\texttt{undec}(ALab_3) \subseteq \texttt{undec}(ALab_2)$ but $\texttt{undec}(ALab2NLab(ALab_3)) \not\subseteq \texttt{undec}(ALab2NLab(ALab_2))$. Furthermore, it also holds that $\texttt{undec}(ALab_3) = \texttt{undec}(ALab_2)$ but $\texttt{undec}(ALab2NLab(ALab_3)) \neq \texttt{undec}(ALab2NLab(ALab_2))$.

Example 9. *Let $AF = (N, arr)$ be an argumentation framework with $N = \{A,B,C,D\}$ and $arr = \{(A,A),(A,C),(B,D),(D,B),(D,C)\}$.*



AF has three complete node labellings:

$$NLab_1 = (\emptyset, \emptyset, \{A,B,C,D\})$$
$$NLab_2 = (\{B\}, \{D\}, \{A,C\})$$
$$NLab_3 = (\{D\}, \{B,C\}, \{A\})$$

and three complete arrow labellings:

$$ALab_1 = (\emptyset, \emptyset, \{(A,A),(A,C),(B,D),(D,B),(D,C)\})$$
$$ALab_2 = (\{(B,D)\}, \{(D,B),(D,C)\}, \{(A,A),(A,C)\})$$
$$ALab_3 = (\{(D,B),(D,C)\}, \{(B,D)\}, \{(A,A),(A,C)\})$$

These node labellings and arrow labellings correspond to each other through the functions NLab2ALab and ALab2NLab. While $ALab_2$ is a semi-stable arrow labelling, $NLab_2 = ALab2NLab(ALab_2)$ is not *a semi-stable node labelling (the only semi-stable node labelling is $NLab_3$).*

Lemma 51. *Let $AF = (N, arr)$ be an argumentation framework.*

(1) *If NLab is a grounded node labelling of AF then $NLab2ALab(NLab)$ is a grounded arrow labelling of AF.*
(2) *If ALab is a grounded arrow labelling of AF then $ALab2NLab(ALab)$ is a grounded node labelling of AF.*

Proof. (1) Let *NLab* be a grounded node labelling of *AF*. Since a grounded node labelling is also a complete node labelling, it follows (Lemma 45) that $ALab = NLab2ALab(NLab)$ is a complete arrow labelling. Suppose, towards a contradiction, that *ALab* does not have minimal `in`. That is, there exists a complete arrow labelling $ALab'$ such that $\texttt{in}(ALab') \subsetneq \texttt{in}(ALab)$. From point 3 of Lemma 50 it then follows that $\texttt{in}(ALab2NLab(ALab')) \subsetneq \texttt{in}(ALab2NLab(ALab))$. Let $NLab' = ALab2NLab(ALab')$. It follows (Lemma 46) that $NLab'$ is a complete node labelling. Furthermore, it follows from Lemma 47 that $ALab2NLab(ALab) = NLab$. Hence, we obtain $\texttt{in}(NLab') \subsetneq \texttt{in}(NLab)$. But this is impossible since *NLab* is a grounded node labelling and therefore has minimal `in` among all complete node labellings.

(2) Let *ALab* be a grounded arrow labelling of *AF*. Since a grounded arrow labelling is also a complete arrow labelling, it follows (Lemma 46) that $NLab = ALab2NLab(ALab)$ is a complete node labelling. Suppose, towards a contradiction, that *NLab* does not have minimal `in`. That is, there exists a complete node labelling $NLab'$ such that $\texttt{in}(NLab') \subsetneq \texttt{in}(NLab)$. From point 3 of Lemma 49 it then follows that $\texttt{in}(NLab2ALab(NLab')) \subsetneq \texttt{in}(NLab2ALab(NLab))$. Let $ALab' = NLab2ALab(NLab')$. It follows (Lemma 45) that $ALab'$ is a complete arrow labelling. Furthermore, it follows from lemma 48 that $NLab2ALab(NLab) = ALab$. Hence, we obtain $\texttt{in}(ALab') \subsetneq \texttt{in}(ALab)$. But this is impossible since *ALab* is a grounded arrow labelling and therefore has minimal `in` among all complete arrow labellings.

□

Lemma 52. *Let $AF = (N, arr)$ be an argumentation framework.*

(1) *If NLab is a preferred node labelling of AF then $NLab2ALab(NLab)$ is a preferred arrow labelling of AF.*
(2) *If ALab is a preferred arrow labelling of AF then $ALab2NLab(ALab)$ is a preferred node labelling of AF.*

Proof. Similar to the proof of Lemma 51 □

Lemma 53. *Let $AF = (N, arr)$ be an argumentation framework.*

(1) *If NLab is a stable node labelling of AF then $NLab2ALab(NLab)$ is a stable arrow labelling of AF.*
(2) *If ALab is a stable arrow labelling of AF then $ALab2NLab(ALab)$ is a stable node labelling of AF.*

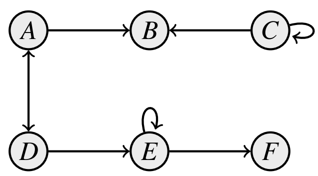
Proof. (1) Let *NLab* be a stable node labelling of *AF*. Since a stable node labelling is also a complete node labelling, it follows (Lemma 45) that $ALab = NLab2ALab(NLab)$ is a complete arrow labelling. In order to prove that *ALab* is also a stable arrow labelling, we need to show that no arrow is labelled `undec`. Let $(A, B) \in arr$. The fact that *NLab* is a stable node labelling implies that *A* is labelled either `in` or `out`. In case $NLab(A) = \texttt{in}$, it follows from the definition of *NLab2ALab* that $ALab((A, B)) = \texttt{in}$. In case $NLab(A) = \texttt{out}$, it follows from the definition of *NLab2ALab* that $ALab((A, B)) = \texttt{out}$. In both cases, $ALab((A, B)) \neq \texttt{undec}$.

(2) Let *ALab* be a stable arrow labelling of *AF*. Since a stable arrow labelling is also a complete arrow labelling, it follows (Lemma 46) that $NLab = ALab2NLab(ALab)$ is a complete node labelling. In order to prove that *NLab* is also a stable node labelling, we need to show that no node is labelled `undec`. Let $A \in N$ and let $\mathcal{B}$ be the set of attackers of A. The fact that *ALab* is a stable arrow labelling means that for every attacker $B \in \mathcal{B}$ it holds that $ALab((B,A))$ is either `in` or `out`. This implies that either there exists a $B \in \mathcal{B}$ such that $ALab((B,A)) = \mathtt{in}$, or for each $B \in \mathcal{B}$ it holds that $ALab((B,A)) = \mathtt{out}$. In the former case, it follows from the definition of *ALab2NLab* that $NLab(A) = \mathtt{out}$. In the latter case, it follows from the definition of *ALab2NLab* that $NLab(A) = \mathtt{in}$. In either case, it holds that $NLab(A) \neq \mathtt{undec}$.
□

Remark 54. *Notice that it does* not *hold that if NLab is a semi-stable node labelling of AF then* $NLab2ALab(NLab)$ *is a semi-stable arrow labelling of AF. Example 10 provides a counter example: while* $NLab_2$ *is a semi-stable node labelling,* $ALab_2$ *is not a semi-stable arrow labelling. Neither does the other direction hold (i.e., if ALab is a semi-stable arrow labelling of AF then* $ALab2NLab(ALab)$ *is a semi-stable node labelling of AF), Example 9 provides a counter example: while* $ALab_2$ *is a semi-stable arrow labelling,* $NLab_2$ *is not a semi-stable node labelling.*

We recall Example 10 from Section 2 (see below). It holds that $\mathtt{undec}(ALab_3) \subsetneq \mathtt{undec}(ALab_2)$ but $\mathtt{undec}(ALab2NLab(ALab_3)) \not\subseteq \mathtt{undec}(ALab2NLab(ALab_2))$.

Example 10. *Let* $AF = (N, arr)$ *be an argumentation framework with* $N = \{A, B, C, D, E, F\}$ *and* $arr = \{(A,B), (C,B), (C,C), (A,D), (D,A), (D,E), (E,E), (E,F)\}$.



AF has three complete node labellings:

$$NLab_1 = (\emptyset, \emptyset, \{A, B, C, D, E, F\})$$
$$NLab_2 = (\{A\}, \{B, D\}, \{C, E, F\})$$
$$NLab_3 = (\{D, F\}, \{A, E\}, \{B, C\})$$

and three complete arrow labellings:

$$ALab_1 = (\emptyset, \emptyset, \{(A,B), (C,B), (C,C), (A,D), (D,A), (D,E), (E,E), (E,F)\})$$
$$ALab_2 = (\{(A,B), (A,D)\}, \{(D,A), (D,E)\}, \{(C,B), (C,C), (E,E), (E,F)\})$$
$$ALab_3 = (\{(D,A), (D,E)\}, \{(A,B), (A,D), (E,E), (E,F)\}, \{(C,B), (C,C)\})$$

These node labellings and arrow labellings correspond to each other through the functions NLab2ALab and ALab2NLab. While $NLab_2$ *is a semi-stable node labelling,* $ALab_2 = NLab2ALab(NLab_2)$ *is* not *a semi-stable arrow labelling (the only semi-stable arrow labelling is* $ALab_3$*).*

We recall the following Theorem 8 from Section 2 that sums up our findings regarding the connections between arrow labellings and node labellings on AFs.

Theorem 8. *Let $AF = (N, arr)$ be an argumentation framework and let NLab and ALab be a node labelling and an arrow labelling of AF, respectively. It holds that:*

(1) *If NLab is a complete node labelling, then NLab2ALab(NLab) is a complete arrow labelling. If ALab is a complete arrow labelling, then ALab2NLab(ALab) is a complete node labelling.*
(2) *When restricted to complete node labellings and complete arrow labellings, the functions ALab2NLab and NLab2ALab become bijections and each other's inverses.*
(3) *If NLab is a grounded node labelling, then NLab2ALab(NLab) is a grounded arrow labelling. If ALab is a grounded arrow labelling, then ALab2NLab(ALab) is a grounded node labelling.*
(4) *If NLab is a preferred node labelling, then NLab2ALab(NLab) is a preferred arrow labelling. If ALab is a preferred arrow labelling, then ALab2NLab(ALab) is a preferred node labelling.*
(5) *If NLab is a stable node labelling, then NLab2ALab(NLab) is a stable arrow labelling. If ALab is a stable arrow labelling, then ALab2NLab(ALab) is a stable node labelling.*

Proof. (1) This follows from Lemma 45 and Lemma 46.
(2) This follows from Lemma 47 and Lemma 48.
(3) This follows from Lemma 51.
(4) This follows from Lemma 52.
(5) This follows from Lemma 53. $\square$

Appendix C. Node extensions for SETAFs: Equivalent definitions

In this section, we show that the semantics for AFs with collective attacks defined in [14] coincide with the formulations we present in Section 3.

Recall that AFs with collective attacks and SETAFs differ in their treatment of the empty attack: in AFs with collective attacks, the attack relation is a subset of $(2^{\mathfrak{N}} \setminus \emptyset) \times \mathfrak{N}$ while SETAFs allow for attacks of the form $(\emptyset, A)$, $A \in \mathfrak{N}$. However, this difference can be neglected, as we discuss in Appendix G. Another difference between the work of Nielsen and Parsons and the theory on SETAFs in Section 3 is how preferred and stable semantics are defined. Throughout the current paper, we have decided to use complete semantics as the basis for defining the other semantics (including preferred and stable). This is to provide a level of uniformity, and to allow for easy conversion between extensions and labellings. However, the work of Nielsen and Parsons stays closer to [8] in the sense that preferred semantics is defined in terms of admissibility and stable semantics in terms of conflict-freeness. The resulting notions, however, are equivalent, as is stated by the following results.

Lemma 55 (Fundamental Lemma)**.** *Let $\mathfrak{SF} = (\mathfrak{N}, \mathfrak{arr})$ be a SETAF, let $\mathfrak{M} \subseteq \mathfrak{N}$ be admissible, and let $A, A' \in F_{\mathfrak{SF}}(\mathfrak{M})$ (i.e., A and A' are defended by $\mathfrak{M}$). Then*

(1) $\mathfrak{M}' = \mathfrak{M} \cup \{A\}$ *is admissible, and*
(2) $A' \in F_{\mathfrak{SF}}(\mathfrak{M}')$.

Proof. The proof is analogous to the proof of the fundamental lemma for AFs [8] and AFs with collective attacks [14].

(1) By definition of admissibility, it suffices to show that $\mathfrak{M}'$ is conflict-free. Towards a contradiction, assume that there exists an argument $B \in \mathfrak{M}'$ and a set $\mathfrak{M}'' \subseteq \mathfrak{M}'$ such that $(\mathfrak{M}'', B) \in \mathfrak{arr}$. We consider three cases: (i) $A = B$ and $A \notin \mathfrak{M}''$; (ii) $A = B$ and $A \in \mathfrak{M}''$; (iii) $A \neq B$ and $A \in \mathfrak{M}''$, (iv) $A \neq B$ and $A \notin \mathfrak{M}''$.
(i) First assume $A = B$ and $A \notin \mathfrak{M}''$. That is, $\mathfrak{M}'' \subseteq \mathfrak{M}$ attacks A. Since A is defended by $\mathfrak{M}$, there is some $\mathfrak{M}''' \subseteq \mathfrak{M}$ that attacks $\mathfrak{M}''$. Hence we have $\mathfrak{M}$ is not conflict-free, contradiction.
(ii) Now assume $A = B$ and $A \in \mathfrak{M}''$. Since A is defended by $\mathfrak{M}$, there is some $\mathfrak{M}''' \subseteq \mathfrak{M}$ that attacks some element $C \in \mathfrak{M}''$. In case $C \in \mathfrak{M}'' \cap \mathfrak{M}$, it follows that $\mathfrak{M}$ is not conflict-free, contradiction. In case $C = A$, consider case (i).
(iii) Next assume $A \neq B$ and $A \in \mathfrak{M}''$. That is, the set $\mathfrak{M}$ is attacked by $\mathfrak{M}''$ containing A. Since $\mathfrak{M}$ is admissible, there is some set $\mathfrak{M}''' \subseteq \mathfrak{M}$ that attacks some element in $D \in b$. Since $\mathfrak{M}$ is conflict-free, we have $D \notin \mathfrak{M}$. It follows that $D = A$, i.e, there is an attack $(\mathfrak{M}''', A) \in \mathfrak{arr}$ with $\mathfrak{M}''' \subseteq \mathfrak{M}$. Hence, we can consider case (i) again to derive a contradiction.
(iv) Finally, suppose $A \neq B$ and $A \notin \mathfrak{M}''$. Hence, $\mathfrak{M}'' \cup \{B\} \nsubseteq \mathfrak{M}$, contradiction to admissibility of $\mathfrak{M}$.
(2) By of the characteristic function.

□

Theorem 56. *Let $\mathfrak{S}\mathfrak{F} = (\mathfrak{N}, \mathfrak{arr})$ be a SETAF and let $\mathfrak{M} \subseteq \mathfrak{N}$. The following two statements are equivalent.*

(1) *$\mathfrak{M}$ is a maximal (w.r.t. $\subseteq$) admissible set of $\mathfrak{S}\mathfrak{F}$*
(2) *$\mathfrak{M}$ is a maximal (w.r.t. $\subseteq$) complete extension if $\mathfrak{S}\mathfrak{F}$*

Proof. **from 1. to 2.** Consider a maximal (w.r.t. $\subseteq$) admissible set $\mathfrak{M}$ of $\mathfrak{S}\mathfrak{F}$. First, we show that $\mathfrak{M}$ is complete. We show that $\mathfrak{M}$ contains all nodes it defends. Let $A \in F(\mathfrak{M})$. By the fundamental lemma, it holds that $\mathfrak{M} \cup \{A\}$ is admissible. Since $\mathfrak{M}$ is a $\subseteq$-maximal admissible set, it follows that $A \in \mathfrak{M}$.
Next, we show that $\mathfrak{M}$ is $\subseteq$-maximal among all complete extensions. Towards a contradiction, assume that there is a complete extension $\mathfrak{M}'$ such that $\mathfrak{M}' \supset \mathfrak{M}$. By definition of complete semantics, $\mathfrak{M}'$ is admissible. Hence, we have a contradiction to $\subseteq$-maximality of $\mathfrak{M}$ among the admissible extensions of $\mathfrak{S}\mathfrak{F}$.

from 2. to 1. Consider a maximal (w.r.t. $\subseteq$) complete set $\mathfrak{M}$ of $\mathfrak{S}\mathfrak{F}$. By definition, each complete set is admissible. It remains to show that $\mathfrak{M}$ is $\subseteq$-maximal among admissible sets. Towards a contradiction, suppose there is an admissible set $\mathfrak{M}'$ such that $\mathfrak{M}' \supset \mathfrak{M}$. By monotonicity of the characteristic function, there is a complete extension $\mathfrak{M}''$ such that $\mathfrak{M}' \subseteq \mathfrak{M}''$. Hence, we obtain a contradiction to the $\subseteq$-maximality of $\mathfrak{M}$. □

Theorem 57. *Let $\mathfrak{S}\mathfrak{F} = (\mathfrak{N}, \mathfrak{arr})$ be a SETAF and let $\mathfrak{M} \subseteq \mathfrak{N}$. The following two statements are equivalent.*

(1) *$\mathfrak{M}$ is a conflict-free set that attacks all nodes in $\mathfrak{N} \setminus \mathfrak{M}$*
(2) *$\mathfrak{M}$ is a complete extension with $\mathfrak{M} \cup \mathfrak{M}^+ = \mathfrak{N}$*

Proof. **from 1. to 2.** Consider a conflict-free set $\mathfrak{M}$ of $\mathfrak{S}\mathfrak{F}$ that attacks all nodes in $\mathfrak{N} \setminus \mathfrak{M}$. Hence, it holds that $\mathfrak{M} \cup \mathfrak{M}^+ = \mathfrak{N}$.

We show that $\mathfrak{M}$ is complete:

By assumption, $\mathfrak{M}$ is conflict-free.

Moreover, it defends itself: towards a contradiction, assume there is some node $A \in \mathfrak{M}$ that is not defended by $\mathfrak{M}$. I.e., there is some attack $(\mathfrak{M}', A) \in \mathfrak{arr}$ such that $\mathfrak{M}' \cap \mathfrak{M}^+ = \emptyset$. By assumption $\mathfrak{M}$ attacks all nodes in $\mathfrak{N} \setminus \mathfrak{M}$, it follows that $\mathfrak{M}' \subseteq \mathfrak{M}$, contradiction to $\mathfrak{M}$ being conflict-free. Hence we conclude that $\mathfrak{M}$ is admissible.

We show that $\mathfrak{M}$ contains all nodes it defends: let $A \in F(\mathfrak{M})$. By the fundamental lemma, it holds that $\mathfrak{M} \cup \{A\}$ is admissible. Since $\mathfrak{M}$ attacks all nodes not contained in $\mathfrak{M}$, it follows that $A \in \mathfrak{M}$.

from 2. to 1. By definition, each complete set is conflict-free. Moreover, $\mathfrak{M} \cup \mathfrak{M}^+ = \mathfrak{N}$ iff $\mathfrak{M}$ attacks all nodes in $\mathfrak{N} \setminus \mathfrak{M}$ for all conflict-free sets by definition. This concludes the proof. □

Appendix D. Properties of Node Labellings for SETAFs

Theorem 58. *Let $\mathfrak{SF} = (\mathfrak{N}, \mathfrak{arr})$ be a SETAF, let $\mathfrak{M} \subseteq \mathfrak{N}$ and let $\mathfrak{NLab}$ be a SETAF node labelling of $\mathfrak{SF}$. It holds that:*

(1) *if $\mathfrak{M}$ is a complete extension of $\mathfrak{SF}$ then*
- $\mathfrak{Args2NLab}(\mathfrak{M})$ *is a complete SETAF node labelling of $\mathfrak{SF}$, and*
- $\mathfrak{NLab2Args}(\mathfrak{Args2NLab}(\mathfrak{M})) = \mathfrak{M}$

(2) *if $\mathfrak{NLab}$ is a complete SETAF node labelling of $\mathfrak{SF}$ then*
- $\mathfrak{NLab2Args}(\mathfrak{NLab})$ *is a complete extension of $\mathfrak{SF}$, and*
- $\mathfrak{Args2NLab}(\mathfrak{NLab2Args}(\mathfrak{NLab})) = \mathfrak{NLab}$

(3) *if $\mathfrak{M}$ is a grounded extension of $\mathfrak{SF}$ then $\mathfrak{Args2NLab}(\mathfrak{M})$ is a grounded SETAF node labelling of $\mathfrak{SF}$*

(4) *if $\mathfrak{NLab}$ is a grounded SETAF node labelling of $\mathfrak{SF}$ then $\mathfrak{NLab2Args}(\mathfrak{NLab})$ is a grounded extension of $\mathfrak{SF}$*

(5) *if $\mathfrak{M}$ is a preferred extension of $\mathfrak{SF}$ then $\mathfrak{Args2NLab}(\mathfrak{M})$ is a preferred SETAF node labelling of $\mathfrak{SF}$*

(6) *if $\mathfrak{NLab}$ is a preferred SETAF node labelling of $\mathfrak{SF}$ then $\mathfrak{NLab2Args}(\mathfrak{NLab})$ is a preferred extension of $\mathfrak{SF}$*

(7) *if $\mathfrak{M}$ is a semi-stable extension of $\mathfrak{SF}$ then $\mathfrak{Args2NLab}(\mathfrak{M})$ is a semi-stable SETAF node labelling of $\mathfrak{SF}$*

(8) *if $\mathfrak{NLab}$ is a semi-stable SETAF node labelling of $\mathfrak{SF}$ then $\mathfrak{NLab2Args}(\mathfrak{NLab})$ is a semi-stable extension of $\mathfrak{SF}$*

(9) *if $\mathfrak{M}$ is a stable extension of $\mathfrak{SF}$ then $\mathfrak{Args2NLab}(\mathfrak{M})$ is a stable SETAF node labelling of $\mathfrak{SF}$*

(10) *if $\mathfrak{NLab}$ is a stable SETAF node labelling of $\mathfrak{SF}$ then $\mathfrak{NLab2Args}(\mathfrak{NLab})$ is a stable extension of $\mathfrak{SF}$*

Proof. Shown in [15]. □

Note that the following Lemma 59 and Theorem 60 are a straightforward generalisation of the respective results of AFs [33].

Lemma 59. *Let* $\mathfrak{NLab}_1$ *and* $\mathfrak{NLab}_2$ *be complete node labellings of SETAF* $\mathfrak{SF} = (\mathfrak{N}, \mathfrak{arr})$*. It holds that:*

(1) $\mathtt{in}(\mathfrak{NLab}_1) \subseteq \mathtt{in}(\mathfrak{NLab}_2)$ *iff* $\mathtt{out}(\mathfrak{NLab}_1) \subseteq \mathtt{out}(\mathfrak{NLab}_2)$
(2) $\mathtt{in}(\mathfrak{NLab}_1) = \mathtt{in}(\mathfrak{NLab}_2)$ *iff* $\mathtt{out}(\mathfrak{NLab}_1) = \mathtt{out}(\mathfrak{NLab}_2)$
(3) $\mathtt{in}(\mathfrak{NLab}_1) \subsetneq \mathtt{in}(\mathfrak{NLab}_2)$ *iff* $\mathtt{out}(\mathfrak{NLab}_1) \subsetneq \mathtt{out}(\mathfrak{NLab}_2)$

Proof. (1) "$\Rightarrow$": Suppose $\mathtt{in}(\mathfrak{NLab}_1) \subseteq \mathtt{in}(\mathfrak{NLab}_2)$. Let $A \in \mathtt{out}(\mathfrak{NLab}_1)$. This means that (Definition 17, second bullet point) there exists a $(\mathfrak{M}, A) \in \mathfrak{arr}$ such that $\forall B \in \mathfrak{M} : \mathfrak{NLab}_1(B) = \mathtt{in}$. The fact that $\mathtt{in}(\mathfrak{NLab}_1) \subseteq \mathtt{in}(\mathfrak{NLab}_2)$ then implies that $\forall B \in \mathfrak{M} : \mathfrak{NLab}_2(B) = \mathtt{in}$. Then $\mathfrak{NLab}_2(A)$ cannot be $\mathtt{in}$ (otherwise there would have to be a $B \in \mathfrak{M}$ such that $\mathfrak{NLab}_2(B) = \mathtt{out}$) and cannot be $\mathtt{undec}$ (by Definition 17, third bullet point). Since $\mathfrak{NLab}_2(A)$ can only be $\mathtt{in}$, $\mathtt{out}$ or $\mathtt{undec}$, it then follows that $\mathfrak{NLab}_2(A) = \mathtt{out}$.
"$\Leftarrow$ ": Suppose $\mathtt{out}(\mathfrak{NLab}_1) \subseteq \mathtt{out}(\mathfrak{NLab}_2)$. Let $A \in \mathtt{in}(\mathfrak{NLab}_1)$. This means that (Definition 17, first bullet point) for each $\mathfrak{M} \subseteq \mathfrak{N}$ such that $(\mathfrak{M}, A) \in \mathfrak{arr}$ it holds that $\exists B \in \mathfrak{M} : \mathfrak{NLab}_1(B) = \mathtt{out}$. The fact that $\mathtt{out}(\mathfrak{NLab}_1) \subseteq \mathtt{out}(\mathfrak{NLab}_2)$ then implies that for each $\mathfrak{M} \subseteq \mathfrak{N}$ such that $(\mathfrak{M}, A) \in \mathfrak{arr}$ it holds that $\exists B \in \mathfrak{M} : \mathfrak{NLab}_2(B) = \mathtt{out}$. Then $\mathfrak{NLab}_2(A)$ cannot be $\mathtt{out}$ (otherwise there would have to be a $(\mathfrak{M}, A) \in \mathfrak{arr}$ such that $\forall B \in \mathfrak{M} : \mathfrak{NLab}_2(B) = \mathtt{in}$) and cannot be $\mathtt{undec}$ (by Definition 17, third bullet point). Since $\mathfrak{NLab}_2(A)$ can only be $\mathtt{in}$, $\mathtt{out}$ or $\mathtt{undec}$, it then follows that $\mathfrak{NLab}_2(A) = \mathtt{in}$.

(2) This follows directly from point 1.
(3) This follows directly from point 1 and point 2.
□

Theorem 60. *Let* $\mathfrak{SF} = (\mathfrak{N}, \mathfrak{arr})$ *be a SETAF, and let* $\mathfrak{NLab}$ *be a SETAF node labelling of* $\mathfrak{SF}$*. The following two statements are equivalent:*

(1) $\mathtt{in}(\mathfrak{NLab})$ *is maximal (w.r.t. set inclusion) among all complete SETAF node labellings of* $\mathfrak{SF}$
(2) $\mathtt{out}(\mathfrak{NLab})$ *is maximal (w.r.t. set inclusion) among all complete SETAF node labellings of* $\mathfrak{SF}$

The following three statements are also equivalent:

3. $\mathtt{in}(\mathfrak{NLab})$ *is minimal (w.r.t. set inclusion) among all complete SETAF node labellings of* $\mathfrak{SF}$
4. $\mathtt{out}(\mathfrak{NLab})$ *is minimal (w.r.t. set inclusion) among all complete SETAF node labellings of* $\mathfrak{SF}$
5. $\mathtt{undec}(\mathfrak{NLab})$ *is maximal (w.r.t. set inclusion) among all complete SETAF node labellings of* $\mathfrak{SF}$

Furthermore, it holds that the complete SETAF node labelling with minimal $\mathtt{in}$ *is unique.*

Proof. from 1 to 2 Let $\mathfrak{NLab}$ be a complete labelling where $\mathtt{out}(\mathfrak{NLab})$ is not maximal. Then there exists a complete labelling $\mathfrak{NLab}'$ with $\mathtt{out}(\mathfrak{NLab}) \subsetneq \mathtt{out}(\mathfrak{NLab}')$. From Lemma 59 it then follows that $\mathtt{in}(\mathfrak{NLab}) \subsetneq \mathtt{in}(\mathfrak{NLab}')$, so $\mathfrak{NLab}$ is a labelling where $\mathtt{in}(\mathfrak{NLab})$ is not maximal.
from 2 to 1 Let $\mathfrak{NLab}$ be a complete labelling where $\mathtt{in}(\mathfrak{NLab})$ is not maximal. Then there exists a complete labelling $\mathfrak{NLab}'$ with $\mathtt{in}(\mathfrak{NLab}) \subsetneq \mathtt{in}(\mathfrak{NLab}')$. From Lemma 59 it then follows that $\mathtt{out}(\mathfrak{NLab}) \subsetneq \mathtt{out}(\mathfrak{NLab}')$, so $\mathfrak{NLab}$ is a labelling where $\mathtt{out}(\mathfrak{NLab})$ is not maximal.
from 3 to 4 Note that this result is mentioned in [17] in the context of extensions. Let $\mathfrak{NLab}$ be a complete labelling where $\mathtt{out}(\mathfrak{NLab})$ is not minimal. Then there exists a complete labelling $\mathfrak{NLab}'$ with $\mathtt{out}(\mathfrak{NLab}') \subsetneq \mathtt{out}(\mathfrak{NLab})$. From Lemma 59 it then follows that $\mathtt{in}(\mathfrak{NLab}') \subsetneq \mathtt{in}(\mathfrak{NLab})$, so $\mathfrak{NLab}$ is a labelling where $\mathtt{in}(\mathfrak{NLab})$ is not minimal.

from 4 to 3 Note that this result is mentioned in [17] in the context of extensions. Let $\mathfrak{NLab}$ be a complete labelling where $\texttt{in}(\mathfrak{NLab})$ is not minimal. Then there exists a complete labelling $\mathfrak{NLab}'$ with $\texttt{in}(\mathfrak{NLab}') \subsetneq \texttt{in}(\mathfrak{NLab})$. From Lemma 59 it then follows that $\texttt{out}(\mathfrak{NLab}') \subsetneq \texttt{out}(\mathfrak{NLab})$, so $\mathfrak{NLab}$ is a labelling where $\texttt{out}(\mathfrak{NLab})$ is not minimal.

from 3 to 5 Let $\mathfrak{NLab}$ be a complete labelling where $\texttt{in}(\mathfrak{NLab})$ is minimal. Then by Theorem 58 $\mathfrak{NLab2Args}(\mathfrak{NLab})$ is the grounded extension. Now suppose that $\texttt{undec}(\mathfrak{NLab})$ is not maximal. Then there exists a complete labelling $\mathfrak{NLab}'$ with $\texttt{undec}(\mathfrak{NLab}) \subsetneq \texttt{undec}(\mathfrak{NLab}')$. It holds that $\mathfrak{NLab2Args}(\mathfrak{NLab}')$ is a complete extension, and from the fact that the grounded extension is a subset of each complete extension, it follows that $\mathfrak{NLab2Args}(\mathfrak{NLab}) \subseteq \mathfrak{NLab2Args}(\mathfrak{NLab}')$, so $\texttt{in}(\mathfrak{NLab}) \subseteq \texttt{in}(\mathfrak{NLab}')$. From Lemma 59 it then follows that $\texttt{out}(\mathfrak{NLab}) \subseteq \texttt{out}(\mathfrak{NLab}')$. From the fact that $\texttt{in}(\mathfrak{NLab}) \subseteq \texttt{in}(\mathfrak{NLab}')$ and $\texttt{out}(\mathfrak{NLab}) \subseteq \texttt{out}(\mathfrak{NLab}')$ it follows that $\texttt{undec}(\mathfrak{NLab}') \subseteq \texttt{undec}(\mathfrak{NLab})$. Contradiction.

from 5 to 3 Let $\mathfrak{NLab}$ be a complete labelling where $\texttt{in}(\mathfrak{NLab})$ is not minimal. Then there exists a complete labelling $\mathfrak{NLab}'$ with $\texttt{in}(\mathfrak{NLab}') \subsetneq \texttt{in}(\mathfrak{NLab})$. It then follows from Lemma 59 that $\texttt{out}(\mathfrak{NLab}') \subsetneq \texttt{out}(\mathfrak{NLab})$, so $\texttt{undec}(\mathfrak{NLab}) \subsetneq \texttt{undec}(\mathfrak{NLab}')$.

uniqueness grounded node labelling Shown in [15].

□

Appendix E. Properties of Arrow Labellings for SETAF

We show that for SETAF arrow labellings the same properties hold as for AF arrow labellings, as shown in Appendix A. In particular, we show that the complete arrow labellings with maximal in are equal to the complete arrow labellings with maximal out, and that the complete arrow labelling where in is minimal is unique and equal to both the complete arrow labelling where out is minimal and the complete arrow labelling where undec is maximal (Theorem 64).

We start with an alternative definition for the conditions of complete arrow labellings for SETAF.

Theorem 61. *Let $\mathfrak{SF} = (\mathfrak{N}, \mathfrak{arr})$ be a SETAF and let $\mathfrak{ALab}$ be an arrow labelling of $\mathfrak{SF}$. $\mathfrak{ALab}$ is a complete arrow labelling of $\mathfrak{SF}$ iff for every $(\mathfrak{M}, A) \in \mathfrak{arr}$ it holds that:*

(1) $\mathfrak{ALab}((\mathfrak{M}, A)) = \texttt{in}$ *iff for each* $(\mathfrak{M}', B) \in \mathfrak{arr}$ *with* $B \in \mathfrak{M}$ *it holds that* $\mathfrak{ALab}((\mathfrak{M}', B)) = \texttt{out}$

(2) $\mathfrak{ALab}((\mathfrak{M}, A)) = \texttt{out}$ *iff there exists a* $(\mathfrak{M}', B) \in \mathfrak{arr}$ *with* $B \in \mathfrak{M}$ *such that* $\mathfrak{ALab}((\mathfrak{M}', B)) = \texttt{in}$

Proof. "$\Rightarrow$ ": Let $\mathfrak{ALab}$ be a complete arrow labelling of $\mathfrak{SF}$. Point 1 LTR follows directly from the first bullet of Definition 20. As for point 1 RTL, suppose that for each $(\mathfrak{M}', B) \in arr$ with $B \in \mathfrak{M}$ it holds that $\mathfrak{ALab}((\mathfrak{M}', B)) = \texttt{out}$. Then $\mathfrak{ALab}((\mathfrak{M}, A))$ cannot be out (otherwise there would have to be a $(\mathfrak{M}', B) \in \mathfrak{arr}$ with $B \in \mathfrak{M}$ such that $\mathfrak{ALab}((\mathfrak{M}', B)) = \texttt{in}$) and cannot be undec (otherwise not for each $(\mathfrak{M}', B) \in \mathfrak{arr}$ with $B \in \mathfrak{M}$ it holds that $\mathfrak{ALab}((\mathfrak{M}', B)) = \texttt{out}$). Since $\mathfrak{ALab}((\mathfrak{M}, A))$ can only be in, out or undec, it then follows that $\mathfrak{ALab}((\mathfrak{M}, A)) = \texttt{in}$.
Point 2 LTR follows directly from the second bullet of Definition 20. As for point 2 RTL, suppose that there exists a $(\mathfrak{M}', B) \in \mathfrak{arr}$ with $B \in \mathfrak{M}$ such that $\mathfrak{ALab}((\mathfrak{M}', B)) = \texttt{in}$. Then $\mathfrak{ALab}((\mathfrak{M}, A))$ cannot be in (otherwise for each $(\mathfrak{M}', B) \in arr$ with $B \in \mathfrak{M}$ it holds that $\mathfrak{ALab}((\mathfrak{M}', B)) = \texttt{out}$) and cannot be undec (otherwise there does not exist a $(\mathfrak{M}', B) \in \mathfrak{arr}$ with $B \in \mathfrak{M}$ such that $\mathfrak{ALab}((\mathfrak{M}', B)) = \texttt{in}$). Since $\mathfrak{ALab}((\mathfrak{M}', B))$ can only be in, out or undec, it follows that $\mathfrak{ALab}((\mathfrak{M}, A)) = \texttt{out}$.
"$\Leftarrow$": Let $\mathfrak{ALab}$ be an arrow labelling satisfying points 1 and 2. We now need to show that $\mathfrak{ALab}$

also satisfies the first three bullets of Definition 20. The first bullet follows directly from point 1. The second bullet follows directly from point 2. As for the third bullet, take an arbitrary $(\mathfrak{M}, A) \in \mathfrak{arr}$ such that $\mathfrak{ALab}((\mathfrak{M}, A)) = \texttt{undec}$. The fact that $\mathfrak{ALab}((\mathfrak{M}, A)) \neq \texttt{in}$, together with point 1, then implies that not for each $(\mathfrak{M}', B) \in arr$ with $B \in \mathfrak{M}$ it holds that $\mathfrak{ALab}((\mathfrak{M}', B)) = \texttt{out}$. The fact that $\mathfrak{ALab}((\mathfrak{M}, A)) \neq \texttt{out}$, together with point 2, then implies that there does not exist a $(\mathfrak{M}', B) \in \mathfrak{arr}$ with $B \in \mathfrak{M}$ such that $\mathfrak{ALab}((\mathfrak{M}', B)) = \texttt{in}$. □

We now proceed to prove a number of lemmas on how arrow labellings relate to each other, and how arrow labellings relate to node labellings.

Lemma 62. *Let $\mathfrak{ALab}_1$ and $\mathfrak{ALab}_2$ be complete arrow labellings of SETAF $\mathfrak{SF} = (\mathfrak{N}, \mathfrak{arr})$. It holds that:*

(1) $\texttt{in}(\mathfrak{ALab}_1) \subseteq \texttt{in}(\mathfrak{ALab}_2)$ *iff* $\texttt{out}(\mathfrak{ALab}_1) \subseteq \texttt{out}(\mathfrak{ALab}_2)$
(2) $\texttt{in}(\mathfrak{ALab}_1) = \texttt{in}(\mathfrak{ALab}_2)$ *iff* $\texttt{out}(\mathfrak{ALab}_1) = \texttt{out}(\mathfrak{ALab}_2)$
(3) $\texttt{in}(\mathfrak{ALab}_1) \subsetneq \texttt{in}(\mathfrak{ALab}_2)$ *iff* $\texttt{out}(\mathfrak{ALab}_1) \subsetneq \texttt{out}(\mathfrak{ALab}_2)$

Proof. (1) "⇒": Suppose $\texttt{in}(\mathfrak{ALab}_1) \subseteq \texttt{in}(\mathfrak{ALab}_2)$. Let $(\mathfrak{M}, A) \in \texttt{out}(\mathfrak{ALab}_1)$. This means that (Definition 20, second bullet point) there exists a $(\mathfrak{M}', B) \in \mathfrak{arr}$ such that $B \in \mathfrak{M}$ and $\mathfrak{ALab}_1((\mathfrak{M}', B)) = \texttt{in}$. That is, $(\mathfrak{M}', B) \in \texttt{in}(\mathfrak{ALab}_1)$. The fact that $\texttt{in}(\mathfrak{ALab}_1) \subseteq \texttt{in}(\mathfrak{ALab}_2)$ then implies that $(\mathfrak{M}', B) \in \texttt{in}(\mathfrak{ALab}_2)$. This, together with the fact that $\mathfrak{ALab}_2$ is a complete arrow labelling implies (by point 2 of Theorem 61) that $(\mathfrak{M}, A) \in \texttt{out}(\mathfrak{ALab}_2)$.
"⇐ ": Suppose $\texttt{out}(\mathfrak{ALab}_1) \subseteq \texttt{out}(\mathfrak{ALab}_2)$. Let $(\mathfrak{M}, A) \in \texttt{in}(\mathfrak{ALab}_1)$. This means that (Definition 20, first bullet point) that for each $(\mathfrak{M}', B) \in \mathfrak{arr}$ with $B \in \mathfrak{M}$ it holds that $(\mathfrak{M}', B) \in \texttt{out}(\mathfrak{ALab}_1)$. The fact that $\texttt{out}(\mathfrak{ALab}_1) \subseteq \texttt{out}(\mathfrak{ALab}_2)$ then implies that $(\mathfrak{M}', B) \in \texttt{out}(\mathfrak{ALab}_2)$. This, together with the fact that $\mathfrak{ALab}_2$ is a complete arrow labelling implies (by point 1 of Theorem 61) that $(\mathfrak{M}, A) \in \texttt{in}(\mathfrak{ALab}_2)$.

(2) This follows directly from point 1.
(3) This follows directly from point 1 and point 2.
□

Lemma 63. *Let $\mathfrak{ALab}_1$ and $\mathfrak{ALab}_2$ be complete arrow labellings of SETAF $\mathfrak{SF} = (\mathfrak{N}, \mathfrak{arr})$. It holds that:*

(1) *if* $\texttt{in}(\mathfrak{ALab}_1) \subseteq \texttt{in}(\mathfrak{ALab}_2)$ *then* $\texttt{undec}(\mathfrak{ALab}_1) \supseteq \texttt{undec}(\mathfrak{ALab}_2)$
(2) *if* $\texttt{in}(\mathfrak{ALab}_1) = \texttt{in}(\mathfrak{ALab}_2)$ *then* $\texttt{undec}(\mathfrak{ALab}_1) = \texttt{undec}(\mathfrak{ALab}_2)$
(3) *if* $\texttt{in}(\mathfrak{ALab}_1) \subsetneq \texttt{in}(\mathfrak{ALab}_2)$ *then* $\texttt{undec}(\mathfrak{ALab}_1) \supsetneq \texttt{undec}(\mathfrak{ALab}_2)$
(4) *if* $\texttt{out}(\mathfrak{ALab}_1) \subseteq \texttt{out}(\mathfrak{ALab}_2)$ *then* $\texttt{undec}(\mathfrak{ALab}_1) \supseteq \texttt{undec}(\mathfrak{ALab}_2)$
(5) *if* $\texttt{out}(\mathfrak{ALab}_1) = \texttt{out}(\mathfrak{ALab}_2)$ *then* $\texttt{undec}(\mathfrak{ALab}_1) = \texttt{undec}(\mathfrak{ALab}_2)$
(6) *if* $\texttt{out}(\mathfrak{ALab}_1) \subsetneq \texttt{out}(\mathfrak{ALab}_2)$ *then* $\texttt{undec}(\mathfrak{ALab}_1) \supsetneq \texttt{undec}(\mathfrak{ALab}_2)$

Proof. (1) Suppose $\texttt{in}(\mathfrak{ALab}_1) \subseteq \texttt{in}(\mathfrak{ALab}_2)$. Then (Lemma 62, point 1) it follows that $\texttt{out}(\mathfrak{ALab}_1) \subseteq \texttt{out}(\mathfrak{ALab}_2)$. Let $(\mathfrak{M}, A) \in \texttt{undec}(\mathfrak{ALab}_2)$. Then $(\mathfrak{M}, A) \notin \texttt{in}(\mathfrak{ALab}_2)$ so $(\mathfrak{M}, A) \notin \texttt{in}(\mathfrak{ALab}_1)$. Also, $(\mathfrak{M}, A) \notin \texttt{out}(\mathfrak{ALab}_2)$ so $(\mathfrak{M}, A) \notin \texttt{out}(\mathfrak{ALab}_1)$. From the fact that $(\mathfrak{M}, A)$ is labelled either in, out or undec by $\mathfrak{ALab}_1$, it follows that $(\mathfrak{M}, A) \in \texttt{undec}(\mathfrak{ALab}_1)$.

(2) This follows directly from point 1.
(3) Assume $\mathtt{in}(\mathfrak{ALab}_1) \subsetneq \mathtt{in}(\mathfrak{ALab}_2)$. From point 1 of this lemma it follows $\mathtt{undec}(\mathfrak{ALab}_1) \supseteq \mathtt{undec}(\mathfrak{ALab}_2)$. From Lemma 62 (point 3) it follows $\mathtt{out}(\mathfrak{ALab}_1) \subsetneq \mathtt{out}(\mathfrak{ALab}_2)$. But since every arrow is labelled either in, out, or undec, this means that $\mathtt{in}(\mathfrak{ALab}_2) \setminus \mathtt{in}(\mathfrak{ALab}_1) \subseteq \mathtt{undec}(\mathfrak{ALab}_1)$ and $\mathtt{out}(\mathfrak{ALab}_2) \setminus \mathtt{out}(\mathfrak{ALab}_1) \subseteq \mathtt{undec}(\mathfrak{ALab}_1)$, which gives us $\mathtt{undec}(\mathfrak{ALab}_1) \supsetneq \mathtt{undec}(\mathfrak{ALab}_2)$.
(4) This follows directly from Lemma 62 (point 1) and point 1 of this lemma.
(5) This follows directly from point 4.
(6) Assume $\mathtt{out}(\mathfrak{ALab}_1) \subsetneq \mathtt{out}(\mathfrak{ALab}_2)$. From Lemma 62 (point 3) it follows $\mathtt{in}(\mathfrak{ALab}_1) \subsetneq \mathtt{in}(\mathfrak{ALab}_2)$. Then $\mathtt{undec}(\mathfrak{ALab}_1) \supsetneq \mathtt{undec}(\mathfrak{ALab}_2)$ follows from point 3 of this lemma.

□

The following theorem states that minimising (resp. maximising) particular labels sometimes yields the same outcome.

Theorem 64. *Let $\mathfrak{SF} = (\mathfrak{N}, \mathfrak{arr})$ be a SETAF, and let $\mathfrak{ALab}$ be a complete arrow labelling of $\mathfrak{SF}$. The following two statements are equivalent:*

1. $\mathtt{in}(\mathfrak{ALab})$ *is maximal (w.r.t. set inclusion) among all complete arrow labellings of* $\mathfrak{SF}$
2. $\mathtt{out}(\mathfrak{ALab})$ *is maximal (w.r.t. set inclusion) among all complete arrow labellings of* $\mathfrak{SF}$

The following three statements are also equivalent:

3. $\mathtt{in}(\mathfrak{ALab})$ *is minimal (w.r.t. set inclusion) among all complete arrow labellings of* $\mathfrak{SF}$
4. $\mathtt{out}(\mathfrak{ALab})$ *is minimal (w.r.t. set inclusion) among all complete arrow labellings of* $\mathfrak{SF}$
5. $\mathtt{undec}(\mathfrak{ALab})$ *is maximal (w.r.t. set inclusion) among all complete arrow labellings of* $\mathfrak{SF}$

Furthermore, it holds that the complete arrow labelling with minimal in *is unique.*

Proof. **from 1 to 2** Suppose $\mathtt{in}(\mathfrak{ALab})$ is maximal among all complete arrow labellings of $\mathfrak{SF}$. That is, there is no complete arrow labelling $\mathfrak{ALab}'$ of $\mathfrak{SF}$ such that $\mathtt{in}(\mathfrak{ALab}) \subsetneq \mathtt{in}(\mathfrak{ALab}')$. Suppose, towards a contradiction, that $\mathtt{out}(\mathfrak{ALab})$ is not maximal among all complete arrow labellings of $\mathfrak{SF}$. Then there exists a complete arrow labelling $\mathfrak{ALab}'$ such that $\mathtt{out}(\mathfrak{ALab}) \subsetneq \mathtt{out}(\mathfrak{ALab}')$. It then follows from Lemma 62 (point 3) that $\mathtt{in}(\mathfrak{ALab}) \subsetneq \mathtt{in}(\mathfrak{ALab}')$. Contradiction.
from 2 to 1 Similar to the previous point.
from 3 to 4 Suppose $\mathtt{in}(\mathfrak{ALab})$ is minimal among all complete arrow labellings of $\mathfrak{SF}$. That is, there is no complete arrow labelling $\mathfrak{ALab}'$ of $\mathfrak{SF}$ such that $\mathtt{in}(\mathfrak{ALab}') \subsetneq \mathtt{in}(\mathfrak{ALab})$. Suppose, towards a contradiction, that $\mathtt{out}(\mathfrak{ALab})$ is not minimal among all complete arrow labellings of $\mathfrak{SF}$. Then there exists a complete arrow labelling $\mathfrak{ALab}'$ such that $\mathtt{out}(\mathfrak{ALab}') \subsetneq \mathtt{out}(\mathfrak{ALab})$. It then follows from Lemma 62 (point 3) that $\mathtt{in}(\mathfrak{ALab}') \subsetneq \mathtt{in}(\mathfrak{ALab})$. Contradiction.
from 4 to 3 Similar to the previous point.
from 5 to 3 Suppose $\mathtt{undec}(\mathfrak{ALab})$ is maximal among all complete arrow labellings of $\mathfrak{SF}$. That is, there is no complete arrow labelling $\mathfrak{ALab}'$ of $\mathfrak{SF}$ such that $\mathtt{undec}(\mathfrak{ALab}) \subsetneq \mathtt{undec}(\mathfrak{ALab}')$. Suppose, towards a contradiction, that $\mathtt{in}(\mathfrak{ALab})$ is not maximal among all complete arrow labellings of $\mathfrak{SF}$. Then there exists a complete arrow labelling $\mathfrak{ALab}'$ such that $\mathtt{in}(\mathfrak{ALab}) \subsetneq \mathtt{in}(\mathfrak{ALab}')$. It then follows from Lemma 63 (point 3) that $\mathtt{undec}(\mathfrak{ALab}') \subsetneq \mathtt{undec}(\mathfrak{ALab})$. Contradiction.

As for the the last point to be proved (from 3 to 5), a particular difficulty is that we cannot just use the same proof strategy as the previous point (from 5 to 3). This is because point 3 of Lemma 63 only goes one-way (it's an "if" instead of an "iff"). To overcome this, we will need to make use of the uniqueness of the grounded arrow labelling.

uniqueness grounded arrow labelling Suppose $\mathfrak{ALab}_1$ and $\mathfrak{ALab}_2$ are complete arrow labellings of $\mathfrak{SF}$ with minimal $\texttt{in}$. That is, they are grounded arrow labellings of $\mathfrak{SF}$. From Theorem 21 (point 3)[12] it follows that $\mathfrak{NLab}_1 = \mathfrak{ALab2NLab}(\mathfrak{ALab}_1)$ and $\mathfrak{NLab}_2 = \mathfrak{ALab2NLab}(\mathfrak{ALab}_2)$ are grounded node labellings of $\mathfrak{SF}$. However, since the grounded node labelling is unique [15] it follows that $\mathfrak{NLab}_1 = \mathfrak{NLab}_2$, so also $\mathfrak{NLab2ALab}(\mathfrak{NLab}_1) = \mathfrak{NLab2ALab}(\mathfrak{NLab}_2)$. However, since $\mathfrak{NLab2ALab}(\mathfrak{NLab}_1) = \mathfrak{ALab}_1$ and $\mathfrak{NLab2ALab}(\mathfrak{NLab}_2) = \mathfrak{ALab}_2$ (Theorem 21 point 2) it follows that $\mathfrak{ALab}_1 = \mathfrak{ALab}_2$.

Using the uniqueness of the grounded arrow labelling, we can then proceed to show that point 3 implies point 5.

from 3 to 5 Suppose $\mathfrak{ALab}_1$ is a complete arrow labelling of $\mathfrak{SF}$ with minimal $\texttt{in}$. That is, $\mathfrak{ALab}_1$ is a minimal element of the set of complete arrow labellings of $\mathfrak{SF}$ (when applying an ordering based on set-inclusion on the $\texttt{in}$-labelled part of the labellings). As this minimal element is unique, it is also the smallest element, meaning that it is less or equal to each element of the set. That is, for each complete arrow labelling $\mathfrak{ALab}'$ of $\mathfrak{SF}$, it holds that $\texttt{in}(\mathfrak{ALab}) \subseteq \texttt{in}(\mathfrak{ALab}')$. From Lemma 63 (point 1) it then follows that $\texttt{undec}(\mathfrak{ALab}') \subseteq \texttt{undec}(\mathfrak{ALab})$. It then follows that there is there is no complete arrow labelling $\mathfrak{ALab}'$ of $\mathfrak{SF}$ such that $\texttt{undec}(\mathfrak{ALab}) \subsetneq \texttt{undec}(\mathfrak{ALab}')$. That is, $\mathfrak{ALab}$ is a complete arrow labelling with maximal $\texttt{undec}$.

□

From Theorem 64 it follows that the grounded, preferred and semi-stable arrow labellings cover all possibilities regarding the maximisation and minimisation of a particular label (among the complete arrow labellings).

Appendix F. Equivalence of Node Labellings and Arrow Labellings for SETAFs

Lemma 65. *Let $\mathfrak{SF} = (\mathfrak{N}, \mathfrak{arr})$ be a SETAF. If $\mathfrak{NLab}$ is a complete node labelling of $\mathfrak{SF}$ then $\mathfrak{ALab} = \mathfrak{NLab2ALab}(\mathfrak{NLab})$ is a complete arrow labelling of $\mathfrak{SF}$.*

Proof. We need to prove that the three bullet points of Definition 20 are satisfied. Let $(\mathfrak{M}, A) \in \mathfrak{arr}$. We distinguish three cases:

(1) $\mathfrak{ALab}((\mathfrak{M}, A)) = \texttt{in}$. From the definition of $\mathfrak{NLab2ALab}$ it then follows that $\mathfrak{NLab}(B) = \texttt{in}$ for each $B \in \mathfrak{M}$. From the fact that $\mathfrak{NLab}$ is a complete node labelling, it then follows that for every $(\mathfrak{M}', B) \in \mathfrak{arr}$ with $B \in \mathfrak{M}$ there is a $C \in \mathfrak{M}'$ such that $\mathfrak{NLab}(C) = \texttt{out}$. From the definition of $\mathfrak{NLab2ALab}$ it then follows that $\mathfrak{ALab}((\mathfrak{M}', B)) = \texttt{out}$.

[12] Note that the proof of Theorem 21 does not depend on this result or its consequences.

(2) $\mathfrak{ALab}((\mathfrak{M}, A)) = \texttt{out}$. From the definition of $\mathfrak{NLab2ALab}$ it then follows that $\mathfrak{NLab}(B) = \texttt{out}$ for some $B \in \mathfrak{M}$. From the fact that $\mathfrak{NLab}$ is a complete node labelling, it then follows that there is an $(\mathfrak{M}', B) \in \mathfrak{arr}$ such that $\mathfrak{NLab}(C) = \texttt{in}$ for each $C \in \mathfrak{M}'$. From the definition of $\mathfrak{NLab2ALab}$ it then follows that $\mathfrak{ALab}((\mathfrak{M}', B)) = \texttt{in}$.

(3) $\mathfrak{ALab}((\mathfrak{M}, A)) = \texttt{undec}$. From the definition of $\mathfrak{NLab2ALab}$ it then follows that not for all $B \in \mathfrak{M} : \mathfrak{NLab}(B) = \texttt{in}$ and there is no $B \in \mathfrak{M} : \mathfrak{NLab}(B) = \texttt{out}$. Since for each $B \in \mathfrak{M}$ the label $\mathfrak{NLab}(B)$ can only be $\texttt{in}$, $\texttt{out}$ or $\texttt{undec}$, there is some $B \in \mathfrak{M}$ such that $\mathfrak{NLab}(B) = \texttt{undec}$. From the fact that $\mathfrak{NLab}$ is a complete node labelling, it then follows that not for each $\mathfrak{M}' \subseteq \mathfrak{N}$ such that $(\mathfrak{M}', B) \in \mathfrak{arr}$ it holds that there is a $C \in \mathfrak{M}' : \mathfrak{NLab}(C) = \texttt{out}$ and there does not exist an $\mathfrak{M}' \subseteq \mathfrak{N}$ such that $(\mathfrak{M}', B) \in \mathfrak{arr}$ and $\forall C \in \mathfrak{M}' : \mathfrak{NLab}(C) = \texttt{in}$. From the definition of $\mathfrak{NLab2ALab}$ it then follows that $\mathfrak{ALab}((\mathfrak{M}', B)) = \texttt{undec}$, which means that not for each $(\mathfrak{M}', B) \in \mathfrak{arr}$ with $B \in \mathfrak{M}$ it holds that $\mathfrak{ALab}((\mathfrak{M}', B)) = \texttt{out}$. From the fact that there is $B \in \mathfrak{M} : \mathfrak{NLab}(B) = \texttt{out}$ and the fact that $\mathfrak{NLab}$ is a complete node labelling it follows that there is no $(\mathfrak{M}', B) \in \mathfrak{arr}$ with $B \in \mathfrak{M}$ such that $\mathfrak{ALab}((\mathfrak{M}', B)) = \texttt{in}$.

□

Lemma 66. *Let $\mathfrak{SF} = (\mathfrak{N}, \mathfrak{arr})$ be a SETAF. If $\mathfrak{ALab}$ is a complete arrow labelling of $\mathfrak{SF}$ then $\mathfrak{NLab} = \mathfrak{ALab2NLab}(\mathfrak{ALab})$ is a complete node labelling of $\mathfrak{SF}$.*

Proof. We need to prove that the three bullet points of Definition 17 are satisfied. Let $A \in \mathfrak{N}$. We distinguish three cases:

(1) $\mathfrak{NLab}(A) = \texttt{in}$. We need to show that for every $(\mathfrak{M}, A) \in \mathfrak{arr}$ it holds that $\exists B \in \mathfrak{M} : \mathfrak{NLab}(B) = \texttt{out}$. Let $(\mathfrak{M}, A) \in \mathfrak{arr}$ be an arbitrary arrow towards A. From the definition of $\mathfrak{ALab2NLab}$ and the fact that $\mathfrak{NLab}(A) = \texttt{in}$ it follows that $\mathfrak{ALab}((\mathfrak{M}, A)) = \texttt{out}$. From the fact that $\mathfrak{ALab}$ is a complete arrow labelling it then follows that there exists a $(\mathfrak{M}', B) \in \mathfrak{arr}$ with $B \in \mathfrak{M}$ such that $\mathfrak{ALab}((\mathfrak{M}', B)) = \texttt{in}$. From the definition of $\mathfrak{ALab2NLab}$ it then follows that $\mathfrak{NLab}(B) = \texttt{out}$.

(2) $\mathfrak{NLab}(A) = \texttt{out}$. We need to show that there exists a $(\mathfrak{M}, A) \in \mathfrak{arr}$ such that $\forall B \in \mathfrak{M} : \mathfrak{NLab}(B) = \texttt{in}$. From the definition of $\mathfrak{ALab2NLab}$ and the fact that $\mathfrak{NLab}(A) = \texttt{out}$ it follows that there is a $(\mathfrak{M}, A) \in \mathfrak{arr}$ such that $\mathfrak{ALab}((\mathfrak{M}, A)) = \texttt{in}$. From the fact that $\mathfrak{ALab}$ is a complete arrow labelling, it then follows that for each $(\mathfrak{M}, B) \in \mathfrak{arr}$ such that $B \in \mathfrak{M}$, it holds that $\mathfrak{ALab}((\mathfrak{M}, B)) = \texttt{out}$. From the definition of $\mathfrak{ALab2NLab}$, it then follows that $\mathfrak{NLab}(B) = \texttt{in}$.

(3) $\mathfrak{NLab}(A) = \texttt{undec}$. We need to show that not for all $(\mathfrak{M}, A) \in \mathfrak{arr}$ it holds that $\exists B \in \mathfrak{M} : \mathfrak{NLab}(B) = \texttt{out}$, and that there is no $(\mathfrak{M}, A) \in \mathfrak{arr}$ such that $\forall B \in \mathfrak{M} : \mathfrak{NLab}(B) = \texttt{in}$. From the definition of $\mathfrak{ALab2NLab}$ and the fact that $\mathfrak{NLab}(A) = \texttt{undec}$ it follows that (i) not for all $(\mathfrak{M}, A) \in \mathfrak{arr}$ it holds that $\mathfrak{ALab}((\mathfrak{M}, A)) = \texttt{out}$, and (ii) there is no $(\mathfrak{M}, A) \in \mathfrak{arr}$ such that $\mathfrak{ALab}((\mathfrak{M}, A)) = \texttt{in}$. From (i) it follows that there is a $(\mathfrak{M}, A) \in \mathfrak{arr}$ such that $\mathfrak{ALab}((\mathfrak{M}, A)) \neq \texttt{out}$. From the fact that $\mathfrak{ALab}$ is a complete arrow labelling, it then follows that there is no $(\mathfrak{M}', B) \in \mathfrak{arr}$ such that $\mathfrak{ALab}((\mathfrak{M}', B)) = \texttt{in}$, which from the definition of $\mathfrak{ALab2NLab}$ implies that $\mathfrak{NLab}(B) \neq \texttt{out}$. From (ii) it follows that for each $(\mathfrak{M}, A) \in \mathfrak{arr}$ it holds that $\mathfrak{ALab}((\mathfrak{M}, A)) \neq \texttt{in}$. Let $(\mathfrak{M}, A) \in \mathfrak{arr}$ be an arbitrary arrow towards A. It directly follows that $\mathfrak{ALab}((\mathfrak{M}, A)) \neq \texttt{in}$. From the fact that $\mathfrak{ALab}$ is a complete arrow labelling, it follows that there is a $(\mathfrak{M}', B) \in \mathfrak{arr}$ with $B \in \mathfrak{M}$ such that $\mathfrak{ALab}((\mathfrak{M}', B)) \neq \texttt{out}$. From the definition of $\mathfrak{ALab2NLab}$, it then follows that $\mathfrak{NLab}(B) \neq \texttt{in}$.

□

Lemma 67. *Let* $\mathfrak{NLab}$ *be a complete node labelling of SETAF* $\mathfrak{SF} = (\mathfrak{N}, \mathfrak{arr})$. *It holds that* $\mathfrak{ALab2NLab}(\mathfrak{NLab2ALab}(\mathfrak{NLab})) = \mathfrak{NLab}$.

Proof. Let $\mathfrak{ALab} = \mathfrak{NLab2ALab}(\mathfrak{NLab})$. It suffices to prove the following three properties, for an arbitrary $A \in \mathfrak{N}$.

(1) If $\mathfrak{NLab}(A) = \mathtt{in}$ then $\mathfrak{ALab2NLab}(\mathfrak{ALab})(A) = \mathtt{in}$.
Suppose $\mathfrak{NLab}(A) = \mathtt{in}$. Then from $\mathfrak{NLab}$ being a complete node labelling, it follows that for each $(\mathfrak{M}, A) \in \mathfrak{arr}$ it holds that $\mathfrak{NLab}(B) = \mathtt{out}$ for some $B \in \mathfrak{M}$, which from the definition of $\mathfrak{NLab2ALab}$ implies that $\mathfrak{ALab}((\mathfrak{M}, A)) = \mathtt{out}$. From the definition of $\mathfrak{ALab2NLab}$ it then follows that $\mathfrak{ALab2NLab}(\mathfrak{ALab})(A) = \mathtt{in}$.

(2) If $\mathfrak{NLab}(A) = \mathtt{out}$ then $\mathfrak{ALab2NLab}(\mathfrak{ALab})(A) = \mathtt{out}$.
Suppose $\mathfrak{NLab}(A) = \mathtt{out}$. Then from $\mathfrak{NLab}$ being a complete node labelling, it follows that there is a $(\mathfrak{M}, A) \in \mathfrak{arr}$ such that $\forall B \in \mathfrak{M} : \mathfrak{NLab}(B) = \mathtt{in}$, which from the definition of $\mathfrak{NLab2ALab}$ implies that $\mathfrak{ALab}((\mathfrak{M}, A)) = \mathtt{in}$. From the definition of $\mathfrak{ALab2NLab}$ it then follows that $\mathfrak{ALab2NLab}(\mathfrak{ALab})(A) = \mathtt{out}$.

(3) If $\mathfrak{NLab}(A) = \mathtt{undec}$ then $\mathfrak{ALab2NLab}(\mathfrak{ALab})(A) = \mathtt{undec}$.
Suppose $\mathfrak{NLab}(A) = \mathtt{undec}$. Then from $\mathfrak{NLab}$ being a complete node labelling, it follows that (i) not for each $(\mathfrak{M}, A) \in \mathfrak{arr}$ it holds that $\mathfrak{NLab}(B) = \mathtt{out}$ for some $B \in \mathfrak{M}$, and (ii) there is no $(\mathfrak{M}, A) \in \mathfrak{arr}$ such that $\forall B \in \mathfrak{M} : \mathfrak{NLab}(B) = \mathtt{in}$. From (i) it directly follows that there exists a $(\mathfrak{M}, A) \in \mathfrak{arr}$ such that $\mathfrak{NLab}(B) \neq \mathtt{out}$ for some $B \in \mathfrak{M}$, thus $\mathfrak{ALab}((\mathfrak{M}, A)) \neq \mathtt{out}$. Then, from the definition of $\mathfrak{ALab2NLab}$ it follows that $\mathfrak{ALab2NLab}(\mathfrak{ALab})(A) \neq \mathtt{in}$. From (ii) it directly follows that for each $(\mathfrak{M}, A) \in \mathfrak{arr}$ it holds that $\neg\forall B \in \mathfrak{M} : \mathfrak{NLab}(B) = \mathtt{in}$. Then, from the definition of $\mathfrak{NLab2ALab}$ it follows that $\mathfrak{ALab}((\mathfrak{M}, A)) \neq \mathtt{in}$. Then, from the definition of $\mathfrak{ALab2NLab}$ it follows that $\mathfrak{ALab2NLab}(\mathfrak{ALab})(A) \neq \mathtt{out}$. This, together with the earlier obtained fact that $\mathfrak{ALab2NLab}(\mathfrak{ALab})(A) \neq \mathtt{in}$ implies that $\mathfrak{ALab2NLab}(\mathfrak{NLab})(A) = \mathtt{undec}$.

□

Lemma 68. *Let* $\mathfrak{ALab}$ *be a complete arrow labelling of SETAF* $\mathfrak{SF} = (\mathfrak{N}, \mathfrak{arr})$. *It holds that* $\mathfrak{NLab2ALab}(\mathfrak{ALab2NLab}(\mathfrak{ALab})) = \mathfrak{ALab}$.

Proof. Let $\mathfrak{NLab} = \mathfrak{ALab2NLab}(\mathfrak{ALab})$. It suffices to prove the following three properties, for an arbitrary $(\mathfrak{M}, A) \in \mathfrak{arr}$.

(1) If $\mathfrak{ALab}((\mathfrak{M}, A)) = \mathtt{in}$ then $\mathfrak{NLab2ALab}(\mathfrak{NLab})((\mathfrak{M}, A)) = \mathtt{in}$.
Suppose $\mathfrak{ALab}((\mathfrak{M}, A)) = \mathtt{in}$. Then from the fact that $\mathfrak{ALab}$ is a complete arrow labelling, it follows that for each $(\mathfrak{M}', B) \in \mathfrak{arr}$ with $B \in \mathfrak{M}$, $\mathfrak{ALab}((\mathfrak{M}', B)) = \mathtt{out}$. From the definition of $\mathfrak{ALab2NLab}$ it then follows that $\mathfrak{NLab}(B) = \mathtt{in}$. From the definition of $\mathfrak{NLab2ALab}$ it then follows that $\mathfrak{NLab2ALab}(\mathfrak{NLab})((\mathfrak{M}, A)) = \mathtt{in}$.

(2) If $\mathfrak{ALab}((\mathfrak{M}, A)) = \mathtt{out}$ then $\mathfrak{NLab2ALab}(\mathfrak{NLab})((\mathfrak{M}, A)) = \mathtt{out}$.
Suppose $\mathfrak{ALab}((\mathfrak{M}, A)) = \mathtt{out}$. Then, from the fact that $\mathfrak{ALab}$ is a complete arrow labelling, it follows that there exists a $(\mathfrak{M}', B) \in \mathfrak{arr}$ with $B \in \mathfrak{M}$ such that $\mathfrak{ALab}((\mathfrak{M}', B)) = \mathtt{in}$. From the definition of $\mathfrak{ALab2NLab}$ it then follows that $\mathfrak{NLab}(B) = \mathtt{out}$. From the definition of $\mathfrak{NLab2ALab}$ it then follows that $\mathfrak{NLab2ALab}(\mathfrak{NLab})((\mathfrak{M}, A)) = \mathtt{out}$.

(3) If $\mathfrak{ALab}((\mathfrak{M}, A)) = \mathtt{undec}$ then $\mathfrak{NLab2ALab}(\mathfrak{NLab})((\mathfrak{M}, A)) = \mathtt{undec}$.
Suppose $\mathfrak{ALab}((\mathfrak{M}, A)) = \mathtt{undec}$. Then from the fact that $\mathfrak{ALab}$ is a complete arrow labelling,

it follows that (i) not for each $(\mathfrak{M}', B) \in \mathfrak{arr}$ with $B \in \mathfrak{M}$ it holds that $\mathfrak{ALab}((\mathfrak{M}', B)) = \texttt{out}$, and (ii) there is no $(\mathfrak{M}', B) \in \mathfrak{arr}$ with $B \in \mathfrak{M}$ such that $\mathfrak{ALab}((\mathfrak{M}', B)) = \texttt{in}$. From (i) it directly follows that there is a $(\mathfrak{M}', B) \in \mathfrak{arr}$ with $B \in \mathfrak{M}$ such that $\mathfrak{ALab}((\mathfrak{M}', B)) \neq \texttt{out}$. From the definition of $\mathfrak{ALab2NLab}$ it then follows that $\mathfrak{NLab}(B) \neq \texttt{in}$. From the definition of $\mathfrak{NLab2ALab}$ it then follows that $\mathfrak{NLab2ALab}(\mathfrak{NLab})((\mathfrak{M}, A)) \neq \texttt{in}$. From (ii) it directly follows that for each $(\mathfrak{M}', B) \in \mathfrak{arr}$ with $B \in \mathfrak{M}$ it holds that $\mathfrak{ALab}((\mathfrak{M}', B)) \neq \texttt{in}$. From the definition of $\mathfrak{ALab2NLab}$ it then follows that $\mathfrak{NLab}(B) \neq \texttt{out}$. From the definition of $\mathfrak{NLab2ALab}$ it then follows that $\mathfrak{NLab2ALab}(\mathfrak{NLab})((\mathfrak{M}, A)) \neq \texttt{out}$. From this, together with the earlier observed fact that $\mathfrak{NLab2ALab}(\mathfrak{NLab})((\mathfrak{M}, A)) \neq \texttt{in}$, it follows that $\mathfrak{NLab2ALab}(\mathfrak{NLab})((\mathfrak{M}, A)) = \texttt{undec}$.

□

Lemma 69. *Let* $\mathfrak{NLab}_1$ *and* $\mathfrak{NLab}_2$ *be complete node labellings of a SETAF* $\mathfrak{SF} = (\mathfrak{N}, \mathfrak{arr})$. *Let* $\mathfrak{ALab}_1 = \mathfrak{NLab2ALab}(\mathfrak{NLab}_1)$ *and* $\mathfrak{ALab}_2 = \mathfrak{NLab2ALab}(\mathfrak{NLab}_2)$. *It holds that:*

(1) $\texttt{in}(\mathfrak{NLab}_1) \subseteq \texttt{in}(\mathfrak{NLab}_2)$ *iff* $\texttt{in}(\mathfrak{ALab}_1) \subseteq \texttt{in}(\mathfrak{ALab}_2)$.
(2) $\texttt{in}(\mathfrak{NLab}_1) = \texttt{in}(\mathfrak{NLab}_2)$ *iff* $\texttt{in}(\mathfrak{ALab}_1) = \texttt{in}(\mathfrak{ALab}_2)$.
(3) $\texttt{in}(\mathfrak{NLab}_1) \subsetneq \texttt{in}(\mathfrak{NLab}_2)$ *iff* $\texttt{in}(\mathfrak{ALab}_1) \subsetneq \texttt{in}(\mathfrak{ALab}_2)$.
(4) $\texttt{out}(\mathfrak{NLab}_1) \subseteq \texttt{out}(\mathfrak{NLab}_2)$ *iff* $\texttt{out}(\mathfrak{ALab}_1) \subseteq \texttt{out}(\mathfrak{ALab}_2)$.
(5) $\texttt{out}(\mathfrak{NLab}_1) = \texttt{out}(\mathfrak{NLab}_2)$ *iff* $\texttt{out}(\mathfrak{ALab}_1) = \texttt{out}(\mathfrak{ALab}_2)$.
(6) $\texttt{out}(\mathfrak{NLab}_1) \subsetneq \texttt{out}(\mathfrak{NLab}_2)$ *iff* $\texttt{out}(\mathfrak{ALab}_1) \subsetneq \texttt{out}(\mathfrak{ALab}_2)$.

Proof. (1) "$\Rightarrow$": Suppose $\texttt{in}(\mathfrak{NLab}_1) \subseteq \texttt{in}(\mathfrak{NLab}_2)$. Let $(\mathfrak{M}, A) \in \texttt{in}(\mathfrak{ALab}_1)$. Then, from the definition of $\mathfrak{NLab2ALab}$ it follows that $\forall B \in \mathfrak{M} : B \in \texttt{in}(\mathfrak{NLab}_1)$. The fact that $\texttt{in}(\mathfrak{NLab}_1) \subseteq \texttt{in}(\mathfrak{NLab}_2)$ then implies that $\forall B \in \mathfrak{M} : B \in \texttt{in}(\mathfrak{NLab}_2)$. From the definition of $\mathfrak{NLab2ALab}$ it then follows that $(\mathfrak{M}, A) \in \texttt{in}(\mathfrak{ALab}_2)$.
"$\Leftarrow$": Suppose $\texttt{in}(\mathfrak{ALab}_1) \subseteq \texttt{in}(\mathfrak{ALab}_2)$. Let $A \in \texttt{in}(\mathfrak{NLab}_1)$. First assume that there is a set of nodes $\mathfrak{M} \subseteq \texttt{in}(\mathfrak{NLab}_1) \subseteq \mathfrak{N}$ and a node $B \in \mathfrak{N}$ such that $(\mathfrak{M}, B) \in \mathfrak{arr}$, i.e., A is involved in an outgoing arrow that is "in". Then, from the definition of $\mathfrak{NLab2ALab}$ it follows that $(\mathfrak{M}, B) \in \texttt{in}(\mathfrak{ALab}_1)$. The fact that $\texttt{in}(\mathfrak{ALab}_1) \subseteq \texttt{in}(\mathfrak{ALab}_2)$ then implies that $(\mathfrak{M}, B) \in \texttt{in}(\mathfrak{ALab}_2)$. From the definition of $\mathfrak{NLab2ALab}$ it then follows that $A \in \texttt{in}(\mathfrak{NLab}_2)$ and we are done. Now on the other hand assume that there is no arrow $(\mathfrak{M}, B) \in \mathfrak{arr}$ with $A \in \mathfrak{M} \subseteq \texttt{in}(\mathfrak{NLab}_1)$, i.e., A is not involved in an outgoing arrow that is "in". Towards contradiction assume that $A \notin \texttt{in}(\mathfrak{NLab}_2)$. This means either A is not defended by $\texttt{in}(\mathfrak{NLab}_2)$ or there is some $\mathfrak{M} \subseteq \texttt{in}(\mathfrak{NLab}_2)$ such that $(\mathfrak{M}, A) \in \mathfrak{arr}$. Since we assume $A \in \texttt{in}(\mathfrak{NLab}_1)$ and $\mathfrak{NLab}_1$ is complete we know that for each $(\mathfrak{M}', A) \in \mathfrak{arr}$ towards A there is a counter-attack $(\mathfrak{M}'', C) \in \mathfrak{arr}$ with $C \in \mathfrak{M}'$ and $\mathfrak{M}'' \subseteq \texttt{in}(\mathfrak{NLab}_1)$. From the definition of $\mathfrak{NLab2ALab}$ it then follows that $(\mathfrak{M}'', C) \in \texttt{in}(\mathfrak{ALab}_1)$. From $\texttt{in}(\mathfrak{ALab}_1) \subseteq \texttt{in}(\mathfrak{ALab}_2)$ we then get $(\mathfrak{M}'', C) \in \texttt{in}(\mathfrak{ALab}_2)$ and, hence, from the definition of $\mathfrak{NLab2ALab}$ we get $D \in \texttt{in}(\mathfrak{NLab}_2)$ for each $D \in \mathfrak{M}''$. Since we chose an arbitrary attacker $\mathfrak{M}'$ of A, we know that A is defended by $\texttt{in}(\mathfrak{NLab}_2)$. It remains to handle the case where there is some $\mathfrak{M} \subseteq \texttt{in}(\mathfrak{NLab}_2)$ such that $(\mathfrak{M}, A) \in \mathfrak{arr}$, i.e., A is not in $\texttt{in}(\mathfrak{NLab}_2)$ due to a conflict. However, since $\mathfrak{NLab}_1$ is complete this means that there is some $B \in \mathfrak{M}$ such that $B \in \texttt{out}(\mathfrak{NLab}_1)$, and therefore $(\mathfrak{M}, A) \in \texttt{out}(\mathfrak{ALab}_1)$. From this and the fact that $\texttt{in}(\mathfrak{ALab}_1) \subseteq \texttt{in}(\mathfrak{ALab}_2)$ and from Lemma 62 (point 1) we get $(\mathfrak{M}, A) \in \texttt{out}(\mathfrak{ALab}_2)$, which gives us from the definition of $\mathfrak{NLab2ALab}$ that for some $B \in \mathfrak{M}$ it holds $B \in \texttt{out}(\mathfrak{NLab}_2)$, a contradiction to our assumption $\mathfrak{M} \subseteq \texttt{in}(\mathfrak{NLab}_2)$.

(2) This follows directly from point 1.
(3) This follows directly from point 1 and point 2.
(4) "⇒": Suppose $\mathtt{out}(\mathfrak{NLab}_1) \subseteq \mathtt{out}(\mathfrak{NLab}_2)$. Let $(\mathfrak{M}, A) \in \mathtt{out}(\mathfrak{ALab}_1)$. Then, from the definition of $\mathfrak{NLab2ALab}$ it follows that $B \in \mathtt{out}(\mathfrak{NLab}_1)$ for some $B \in \mathfrak{M}$. The fact that $\mathtt{out}(\mathfrak{NLab}_1) \subseteq \mathtt{out}(\mathfrak{NLab}_2)$ then implies that $B \in \mathtt{out}(\mathfrak{NLab}_2)$. From the definition of $\mathfrak{NLab2ALab}$ it then follows that $(\mathfrak{M}, A) \in \mathtt{out}(\mathfrak{ALab}_2)$.
"⇐": Suppose $\mathtt{out}(\mathfrak{ALab}_1) \subseteq \mathtt{out}(\mathfrak{ALab}_2)$. Let $A \in \mathtt{out}(\mathfrak{NLab}_1)$. Then, since $\mathfrak{NLab}_1$ is complete, there is a $(\mathfrak{M}, A) \in \mathfrak{att}$ such that $\mathfrak{M} \subseteq \mathtt{in}(\mathfrak{NLab}_1)$. From the definition of $\mathfrak{NLab2ALab}$ it follows that $(\mathfrak{M}, A) \in \mathtt{in}(\mathfrak{ALab}_1)$. From our assumption $\mathtt{out}(\mathfrak{ALab}_1) \subseteq \mathtt{out}(\mathfrak{ALab}_2)$ and Lemma 62 (point 1) we get $\mathtt{in}(\mathfrak{ALab}_1) \subseteq \mathtt{in}(\mathfrak{ALab}_2)$, and therefore $(\mathfrak{M}, A) \in \mathtt{in}(\mathfrak{ALab}_2)$. From the definition of $\mathfrak{NLab2ALab}$ it follows that $\mathfrak{M} \subseteq \mathtt{in}(\mathfrak{NLab}_2)$, and since $\mathfrak{NLab}_2$ is complete we get $A \in \mathtt{out}(\mathfrak{NLab}_2)$.
(5) This follows directly from point 4.
(6) This follows directly from point 4 and point 5.
□

Note that the following similar statements **do not hold**. While properties 7', 8', 9, and 9' already do not hold for AFs (the same counter-example applies in this case), the properties 7 and 8 do hold for AFs, but do not hold for SETAFs.

7 If $\mathtt{undec}(\mathfrak{NLab}_1) \subseteq \mathtt{undec}(\mathfrak{NLab}_2)$ then $\mathtt{undec}(\mathfrak{ALab}_1) \subseteq \mathtt{undec}(\mathfrak{ALab}_2)$. A counter example is $\mathfrak{SF}$ in Example 70: We have $\mathtt{undec}(\mathfrak{NLab}_2) \subseteq \mathtt{undec}(\mathfrak{NLab}_3)$ but $\mathtt{undec}(\mathfrak{ALab}_2) \not\subseteq \mathtt{undec}(\mathfrak{ALab}_3)$.

7' If $\mathtt{undec}(\mathfrak{ALab}_1) \subseteq \mathtt{undec}(\mathfrak{ALab}_2)$ then $\mathtt{undec}(\mathfrak{NLab}_1) \subseteq \mathtt{undec}(\mathfrak{NLab}_2)$. A counter example is *AF* in Example 9: We have $\mathtt{undec}(ALab_2) \subseteq \mathtt{undec}(ALab_3)$ but $\mathtt{undec}(NLab_2) \not\subseteq \mathtt{undec}(NLab_3)$.

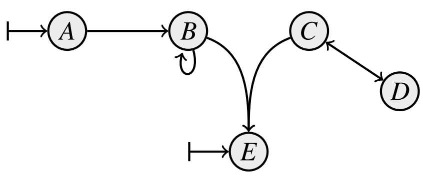
8 If $\mathtt{undec}(\mathfrak{NLab}_1) = \mathtt{undec}(\mathfrak{NLab}_2)$ then $\mathtt{undec}(\mathfrak{ALab}_1) = \mathtt{undec}(\mathfrak{ALab}_2)$. A counter example is $\mathfrak{SF}$ in Example 70: We have $\mathtt{undec}(\mathfrak{NLab}_2) = \mathtt{undec}(\mathfrak{NLab}_3)$ but $\mathtt{undec}(\mathfrak{ALab}_2) \neq \mathtt{undec}(\mathfrak{ALab}_3)$.

8' If $\mathtt{undec}(\mathfrak{ALab}_1) = \mathtt{undec}(\mathfrak{ALab}_2)$ then $\mathtt{undec}(\mathfrak{NLab}_1) = \mathtt{undec}(\mathfrak{NLab}_2)$. A counter example is *AF* in Example 9: We have $\mathtt{undec}(ALab_2) = \mathtt{undec}(ALab_3)$ but $\mathtt{undec}(NLab_2) \neq \mathtt{undec}(NLab_3)$.

9 If $\mathtt{undec}(\mathfrak{NLab}_1) \subsetneq \mathtt{undec}(\mathfrak{NLab}_2)$ then $\mathtt{undec}(\mathfrak{ALab}_1) \subsetneq \mathtt{undec}(\mathfrak{ALab}_2)$. A counter example is *AF* in Example 9: We have $\mathtt{undec}(NLab_3) \subsetneq \mathtt{undec}(NLab_2)$ but $\mathtt{undec}(NLab_3) \not\subsetneq \mathtt{undec}(NLab_2)$.

9' If $\mathtt{undec}(\mathfrak{ALab}_1) \subsetneq \mathtt{undec}(\mathfrak{ALab}_2)$ then $\mathtt{undec}(\mathfrak{NLab}_1) \subsetneq \mathtt{undec}(\mathfrak{NLab}_2)$. A counter example is *AF* in Example 10: We have $\mathtt{undec}(ALab_3) \subsetneq \mathtt{undec}(ALab_2)$ but $\mathtt{undec}(NLab_3) \not\subsetneq \mathtt{undec}(NLab_2)$.

Example 70. *Let* $\mathfrak{SF} = (\mathfrak{A}, \mathfrak{att})$ *be a SETAF with* $\mathfrak{A} = \{A, B, C, D, E\}$ *and*
$\mathfrak{att} = \{(\emptyset, A), (\{A\}, B), (\{B\}, B), (\{B, C\}, E), (\{C\}, D), (\{D\}, C), (\emptyset, E)\}$.



$\mathfrak{SF}$ has three complete node labellings:

$$\mathfrak{NLab}_1 = (\emptyset, \{A, E\}, \{B, C, D\})$$
$$\mathfrak{NLab}_2 = (\{C\}, \{A, D, E\}, \{B\})$$
$$\mathfrak{NLab}_3 = (\{D\}, \{A, C, E\}, \{B\})$$

and three complete arrow labellings:

$$\mathfrak{ALab}_1 = (\{(\emptyset, A), (\emptyset, E)\}, \{(\{A\}, B)\}, \{(\{B\}, B), (\{B, C\}, E), (\{C\}, D), (\{D\}, C)\})$$
$$\mathfrak{ALab}_2 = (\{(\emptyset, A), (\{C\}, D), (\emptyset, E)\}, \{(\{A\}, B), (\{D\}, C)\}, \{(\{B\}, B), (\{B, C\}, E)\})$$
$$\mathfrak{ALab}_3 = (\{(\emptyset, A), (\{D\}, C), (\emptyset, E)\}, \{(\{A\}, B), (\{B, C\}, E), (\{C\}, D)\}, \{(\{B\}, B)\})$$

These node labellings and arrow labellings correspond to each other through the functions $\mathfrak{NLab2ALab}$ and $\mathfrak{ALab2NLab}$.

Lemma 71. *Let $\mathfrak{ALab}_1$ and $\mathfrak{ALab}_2$ be complete arrow labellings of SETAF $\mathfrak{SF} = (\mathfrak{N}, \mathfrak{arr})$. Let $\mathfrak{NLab}_1 = \mathfrak{ALab2NLab}(\mathfrak{ALab}_1)$ and $\mathfrak{NLab}_2 = \mathfrak{ALab2NLab}(\mathfrak{ALab}_2)$. It holds that:*

(1) $\mathtt{in}(\mathfrak{ALab}_1) \subseteq \mathtt{in}(\mathfrak{ALab}_2)$ *iff* $\mathtt{in}(\mathfrak{NLab}_1) \subseteq \mathtt{in}(\mathfrak{NLab}_2)$.
(2) $\mathtt{in}(\mathfrak{ALab}_1) = \mathtt{in}(\mathfrak{ALab}_2)$ *iff* $\mathtt{in}(\mathfrak{NLab}_1) = \mathtt{in}(\mathfrak{NLab}_2)$.
(3) $\mathtt{in}(\mathfrak{ALab}_1) \subsetneq \mathtt{in}(\mathfrak{ALab}_2)$ *iff* $\mathtt{in}(\mathfrak{NLab}_1) \subsetneq \mathtt{in}(\mathfrak{NLab}_2)$.
(4) $\mathtt{out}(\mathfrak{ALab}_1) \subseteq \mathtt{out}(\mathfrak{ALab}_2)$ *iff* $\mathtt{out}(\mathfrak{NLab}_1) \subseteq \mathtt{out}(\mathfrak{NLab}_2)$.
(5) $\mathtt{out}(\mathfrak{ALab}_1) = \mathtt{out}(\mathfrak{ALab}_2)$ *iff* $\mathtt{out}(\mathfrak{NLab}_1) = \mathtt{out}(\mathfrak{NLab}_2)$.
(6) $\mathtt{out}(\mathfrak{ALab}_1) \subsetneq \mathtt{out}(\mathfrak{ALab}_2)$ *iff* $\mathtt{out}(\mathfrak{NLab}_1) \subsetneq \mathtt{out}(\mathfrak{NLab}_2)$.

Proof. (1) "$\Rightarrow$": Suppose $\mathtt{in}(\mathfrak{ALab}_1) \subseteq \mathtt{in}(\mathfrak{ALab}_2)$. Let $A \in \mathtt{in}(\mathfrak{NLab}_1)$. Then, from the definition of $\mathfrak{ALab2NLab}$ it follows that for every $(\mathfrak{M}, A) \in \mathfrak{arr}$, $(\mathfrak{M}, A) \in \mathtt{out}(\mathfrak{ALab}_1)$. From the fact that $\mathtt{in}(\mathfrak{ALab}_1) \subseteq \mathtt{in}(\mathfrak{ALab}_2)$ it follows that (Lemma 62) $\mathtt{out}(\mathfrak{ALab}_1) \subseteq \mathtt{out}(\mathfrak{ALab}_2)$, so $(\mathfrak{M}, A) \in \mathtt{out}(\mathfrak{ALab}_2)$. From the definition of $\mathfrak{ALab2NLab}$ it then follows that $\mathfrak{NLab}_2(A) = \mathtt{in}$. That is, $A \in \mathtt{in}(\mathfrak{NLab}_2)$.
"$\Leftarrow$": Suppose $\mathtt{in}(\mathfrak{NLab}_1) \subseteq \mathtt{in}(\mathfrak{NLab}_2)$. Let $(\mathfrak{M}, A) \in \mathtt{in}(\mathfrak{ALab}_1)$. Since we assume that $\mathfrak{ALab}_1$ is complete, it follows that for every $(\mathfrak{M}', B) \in \mathfrak{arr}$ towards some $B \in \mathfrak{M}$ it holds $(\mathfrak{M}', B) \in \mathtt{out}(\mathfrak{ALab}_1)$, which in turn means for each of these $(\mathfrak{M}', B) \in \mathfrak{arr}$ there is some $(\mathfrak{M}'', C) \in \mathfrak{arr}$ with $C \in \mathfrak{M}'$ and $(\mathfrak{M}'', C) \in \mathtt{in}(\mathfrak{ALab}_1)$. Then, from the definition of $\mathfrak{ALab2NLab}$ it then follows that $C \in \mathtt{out}(\mathfrak{NLab}_1)$. From $\mathtt{in}(\mathfrak{NLab}_1) \subseteq \mathtt{in}(\mathfrak{NLab}_2)$ and Lemma 59 (point 1) we then get $\mathtt{out}(\mathfrak{NLab}_1) \subseteq \mathtt{out}(\mathfrak{NLab}_2)$. From this and $C \in \mathtt{out}(\mathfrak{NLab}_1)$ we get $C \in \mathtt{out}(\mathfrak{NLab}_2)$. Then from the definition of $\mathfrak{ALab2NLab}$ it follows $(\mathfrak{M}'', C) \in \mathtt{in}(\mathfrak{ALab}_2)$, and then since $\mathfrak{ALab}_2$ is complete we get $(\mathfrak{M}', B) \in \mathtt{out}(\mathfrak{ALab}_2)$ and consequently $(\mathfrak{M}, A) \in \mathtt{in}(\mathfrak{ALab}_2)$.
(2) This follows directly from point 1.
(3) This follows directly from point 1 and point 2.
(4) "$\Rightarrow$": Suppose $\mathtt{out}(\mathfrak{ALab}_1) \subseteq \mathtt{out}(\mathfrak{ALab}_2)$. Let $A \in \mathtt{out}(\mathfrak{NLab}_1)$. Then, from the definition of $\mathfrak{ALab2NLab}$ it follows that there exists a $(\mathfrak{M}, A) \in \mathfrak{arr}$ such that $(\mathfrak{M}, A) \in \mathtt{in}(\mathfrak{ALab}_1)$. From the fact that $\mathtt{out}(\mathfrak{ALab}_1) \subseteq \mathtt{out}(\mathfrak{ALab}_2)$ it follows that (Lemma 62) $\mathtt{in}(\mathfrak{ALab}_1) \subseteq \mathtt{in}(\mathfrak{ALab}_2)$, so

$(\mathfrak{M}, A) \in \mathtt{in}(\mathfrak{ALab}_2)$. From the definition of $\mathfrak{ALab2NLab}$ it then follows that $\mathfrak{NLab}_2(A) = \mathtt{out}$. That is, $A \in \mathtt{out}(\mathfrak{NLab}_2)$.

"$\Leftarrow$": Suppose $\mathtt{out}(\mathfrak{NLab}_1) \subseteq \mathtt{out}(\mathfrak{NLab}_2)$. Let $(\mathfrak{M}, A) \in \mathtt{out}(\mathfrak{ALab}_1)$. Since we assume that $\mathfrak{ALab}_1$ is complete, it follows that there exists a $(\mathfrak{M}', B)$ towards some $B \in \mathfrak{M}$ such that $(\mathfrak{M}', B) \in \mathtt{in}(\mathfrak{ALab}_1)$. By completeness this means that for every $(\mathfrak{M}'', C) \in \mathfrak{arr}$ towards some $C \in \mathfrak{M}'$ it holds $(\mathfrak{M}'', C) \in \mathtt{out}(\mathfrak{ALab}_1)$. Then, from the definition of $\mathfrak{ALab2NLab}$ it follows that $C \in \mathtt{in}(\mathfrak{NLab}_1)$, and since this is the case for every $C \in \mathfrak{M}'$ we get $\mathfrak{M}' \subseteq \mathtt{in}(\mathfrak{NLab}_1)$. From $\mathtt{out}(\mathfrak{NLab}_1) \subseteq \mathtt{out}(\mathfrak{NLab}_2)$ and Lemma 62 we then get $\mathtt{in}(\mathfrak{NLab}_1) \subseteq \mathtt{in}(\mathfrak{NLab}_2)$. From this and $\mathfrak{M}' \subseteq \mathtt{in}(\mathfrak{NLab}_1)$ we get $\mathfrak{M}' \subseteq \mathtt{in}(\mathfrak{NLab}_2)$. Then from the definition of $\mathfrak{ALab2NLab}$ it follows that for each $B \in \mathfrak{M}$ there is a $C \in \mathfrak{M}'$ such that $(\mathfrak{M}'', C) \in \mathtt{out}(\mathfrak{ALab}_2)$ and hence $(\mathfrak{M}', B) \in \mathtt{in}(\mathfrak{ALab}_2)$, and then since $\mathfrak{ALab}_2$ is complete we get $(\mathfrak{M}, A) \in \mathtt{out}(\mathfrak{ALab}_2)$.

(5) This follows directly from point 4.
(6) This follows directly from point 4 and point 5.

□

Notice that the respective missing cases do not hold—analogous to Lemma 69 (the same counter examples apply in this case). In fact, the similarities between Lemma 69 and Lemma 71 are no coincidence, they are a direct consequence of the fact that the functions $\mathfrak{NLab2ALab}$ and $\mathfrak{ALab2NLab}$ are each others inverses (see Theorem 21, point 2).

Lemma 72. *Let $\mathfrak{SF} = (\mathfrak{N}, \mathfrak{arr})$ be a SETAF.*

(1) *If $\mathfrak{NLab}$ is a grounded node labelling of $\mathfrak{SF}$ then $\mathfrak{NLab2ALab}(\mathfrak{NLab})$ is a grounded arrow labelling of $\mathfrak{SF}$.*
(2) *If $\mathfrak{ALab}$ is a grounded arrow labelling of $\mathfrak{SF}$ then $\mathfrak{ALab2NLab}(\mathfrak{ALab})$ is a grounded node labelling of $\mathfrak{SF}$.*

Proof. (1) Let $\mathfrak{NLab}$ be a grounded node labelling of $\mathfrak{SF}$. Since a grounded node labelling is also a complete node labelling, it follows (Lemma 65) that $\mathfrak{ALab} = \mathfrak{NLab2ALab}(\mathfrak{NLab})$ is a complete arrow labelling. Suppose, towards a contradiction, that $\mathfrak{ALab}$ does not have minimal $\mathtt{in}$. That is, there exists a complete arrow labelling $\mathfrak{ALab}'$ such that $\mathtt{in}(\mathfrak{ALab}') \subsetneq \mathtt{in}(\mathfrak{ALab})$. From point 3 of Lemma 71 it then follows that $\mathtt{in}(\mathfrak{ALab2NLab}(\mathfrak{ALab}')) \subsetneq \mathtt{in}(\mathfrak{ALab2NLab}(\mathfrak{ALab}))$. Let $\mathfrak{NLab}' = \mathfrak{ALab2NLab}(\mathfrak{ALab}')$. It follows (Lemma 66) that $\mathfrak{NLab}'$ is a complete node labelling. Furthermore, it follows from Lemma 67 that $\mathfrak{ALab2NLab}(\mathfrak{ALab}) = \mathfrak{NLab}$. Hence, we obtain $\mathtt{in}(\mathfrak{NLab}') \subsetneq \mathtt{in}(\mathfrak{NLab})$. But this is impossible since $\mathfrak{NLab}$ is a grounded node labelling and therefore has minimal $\mathtt{in}$ among all complete node labellings.

(2) Let $\mathfrak{ALab}$ be a grounded arrow labelling of $\mathfrak{SF}$. Since a grounded arrow labelling is also a complete arrow labelling, it follows (Lemma 66) that $\mathfrak{NLab} = \mathfrak{ALab2NLab}(\mathfrak{ALab})$ is a complete node labelling. Suppose, towards a contradiction, that $\mathfrak{NLab}$ does not have minimal $\mathtt{in}$. That is, there exists a complete node labelling $\mathfrak{NLab}'$ such that $\mathtt{in}(\mathfrak{NLab}') \subsetneq \mathtt{in}(\mathfrak{NLab})$. From point 3 of Lemma 69 it then follows that $\mathtt{in}(\mathfrak{NLab2ALab}(\mathfrak{NLab}')) \subsetneq \mathtt{in}(\mathfrak{NLab2ALab}(\mathfrak{NLab}))$. Let $\mathfrak{ALab}' = \mathfrak{NLab2ALab}(\mathfrak{NLab}')$. It follows (Lemma 65) that $\mathfrak{ALab}'$ is a complete arrow labelling. Furthermore, it follows from lemma 68 that $\mathfrak{NLab2ALab}(\mathfrak{NLab}) = \mathfrak{ALab}$. Hence, we obtain $\mathtt{in}(\mathfrak{ALab}') \subsetneq \mathtt{in}(\mathfrak{ALab})$. But this is impossible since $\mathfrak{ALab}$ is a grounded arrow labelling and therefore has minimal $\mathtt{in}$ among all complete arrow labellings.

□

Lemma 73. *Let $\mathfrak{SF} = (\mathfrak{N}, \mathfrak{arr})$ be a SETAF.*

(1) *If $\mathfrak{NLab}$ is a preferred node labelling of $\mathfrak{SF}$ then $\mathfrak{NLab2ALab}(\mathfrak{NLab})$ is a preferred arrow labelling of $\mathfrak{SF}$.*
(2) *If $\mathfrak{ALab}$ is a preferred arrow labelling of $\mathfrak{SF}$ then $\mathfrak{ALab2NLab}(\mathfrak{ALab})$ is a preferred node labelling of $\mathfrak{SF}$.*

Proof. Similar to the proof of Lemma 72 □

Lemma 74. *Let $\mathfrak{SF} = (\mathfrak{N}, \mathfrak{arr})$ be a SETAF.*

(1) *If $\mathfrak{NLab}$ is a stable node labelling of $\mathfrak{SF}$ then $\mathfrak{NLab2ALab}(\mathfrak{NLab})$ is a stable arrow labelling of $\mathfrak{SF}$.*
(2) *If $\mathfrak{ALab}$ is a stable arrow labelling of $\mathfrak{SF}$ then $\mathfrak{ALab2NLab}(\mathfrak{ALab})$ is a stable node labelling of $\mathfrak{SF}$.*

Proof. (1) Let $\mathfrak{NLab}$ be a stable node labelling of $\mathfrak{SF}$. Since a stable node labelling is also a complete node labelling, it follows (Lemma 65) that $\mathfrak{ALab} = \mathfrak{NLab2ALab}(\mathfrak{NLab})$ is a complete arrow labelling. In order to prove that $\mathfrak{ALab}$ is also a stable arrow labelling, we need to show that no arrow is labelled `undec`. Let $(\mathfrak{M}, A) \in \mathfrak{arr}$. The fact that $\mathfrak{NLab}$ is a stable node labelling implies that all $B \in \mathfrak{M}$ are labelled either `in` or `out`. We distinguish two cases (i) and (ii). (i) In case for all $B \in \mathfrak{M}$ it holds $\mathfrak{NLab}(B) = \texttt{in}$, it follows from the definition of $\mathfrak{NLab2ALab}$ that $\mathfrak{ALab}((\mathfrak{M}, A)) = \texttt{in}$. (ii) In case $\mathfrak{NLab}(B) = \texttt{out}$ for at least one $B \in \mathfrak{M}$, it follows from the definition of $\mathfrak{NLab2ALab}$ that $\mathfrak{ALab}((\mathfrak{M}, A)) = \texttt{out}$. In both cases, $\mathfrak{ALab}((\mathfrak{M}, A)) \neq \texttt{undec}$.

(2) Let $\mathfrak{ALab}$ be a stable arrow labelling of $\mathfrak{SF}$. Since a stable arrow labelling is also a complete arrow labelling, it follows (Lemma 66) that $\mathfrak{NLab} = \mathfrak{ALab2NLab}(\mathfrak{ALab})$ is a complete node labelling. In order to prove that $\mathfrak{NLab}$ is also a stable node labelling, we need to show that no node is labelled `undec`. Let $A \in \mathfrak{N}$ and let $\mathfrak{a}$ be the set of arrows towards A. The fact that $\mathfrak{ALab}$ is a stable arrow labelling means that for every $(\mathfrak{M}, A) \in \mathfrak{a}$ it holds that $\mathfrak{ALab}((\mathfrak{M}, A))$ is either `in` or `out`. This implies that either (i) there exists a $(\mathfrak{M}, A) \in \mathfrak{a}$ such that $\mathfrak{ALab}((\mathfrak{M}, A)) = \texttt{in}$, or (ii) for each $(\mathfrak{M}, A) \in \mathfrak{a}$ it holds that $\mathfrak{ALab}((\mathfrak{M}, A)) = \texttt{out}$. In the case (i), it follows from the definition of $\mathfrak{ALab2NLab}$ that $\mathfrak{NLab}(A) = \texttt{out}$. In case (ii), it follows from the definition of $\mathfrak{ALab2NLab}$ that $\mathfrak{NLab}(A) = \texttt{in}$. In both cases it holds that $\mathfrak{NLab}(A) \neq \texttt{undec}$.
□

Remark 75. *Notice that it does* not *hold that if $\mathfrak{NLab}$ is a semi-stable node labelling of $\mathfrak{SF}$ then $\mathfrak{NLab2ALab}(\mathfrak{NLab})$ is a semi-stable arrow labelling of $\mathfrak{SF}$. Example 10 provides a counter example: while NLab$_2$ is a semi-stable node labelling, ALab$_2$ is not a semi-stable arrow labelling. Neither does the other direction hold (i.e., if $\mathfrak{ALab}$ is a semi-stable arrow labelling of $\mathfrak{SF}$ then $\mathfrak{ALab2NLab}(\mathfrak{ALab})$ is a semi-stable node labelling of $\mathfrak{SF}$), Example 9 provides a counter example: while ALab$_2$ is a semi-stable arrow labelling, NLab$_2$ is not a semi-stable node labelling. As SETAFs generalise AFs and the functions $\mathfrak{NLab2ALab}$ and $\mathfrak{ALab2NLab}$ generalise the functions NLab2ALab and ALab2NLab respectively, the same counter examples apply in the case of SETAFs as with AFs.*

We recall the following Theorem 21 from Section 3 that sums up our findings regarding the connections between arrow labellings and node labellings on SETAFs.

Theorem 21. *Let $\mathfrak{SF} = (\mathfrak{N}, \mathfrak{arr})$ be a SETAF and let $\mathfrak{NLab}$ and $\mathfrak{ALab}$ be a node labelling and an arrow labelling of $\mathfrak{SF}$, respectively. It holds that:*

(1) *If $\mathfrak{NLab}$ is a complete node labelling, then $\mathfrak{NLab2ALab}(\mathfrak{NLab})$ is a complete arrow labelling. If $\mathfrak{ALab}$ is a complete arrow labelling, then $\mathfrak{ALab2NLab}(\mathfrak{ALab})$ is a complete node labelling.*
(2) *When restricted to complete node labellings and complete arrow labellings, the functions $\mathfrak{ALab2NLab}$ and $\mathfrak{NLab2ALab}$ become bijections and each other's inverses.*
(3) *If $\mathfrak{NLab}$ is a grounded node labelling, then $\mathfrak{NLab2ALab}(\mathfrak{NLab})$ is a grounded arrow labelling. If $\mathfrak{ALab}$ is a grounded arrow labelling, then $\mathfrak{ALab2NLab}(\mathfrak{ALab})$ is a grounded node labelling.*
(4) *If $\mathfrak{NLab}$ is a preferred node labelling, then $\mathfrak{NLab2ALab}(\mathfrak{NLab})$ is a preferred arrow labelling. If $\mathfrak{ALab}$ is a preferred arrow labelling, then $\mathfrak{ALab2NLab}(\mathfrak{ALab})$ is a preferred node labelling.*
(5) *If $\mathfrak{NLab}$ is a stable node labelling, then $\mathfrak{NLab2ALab}(\mathfrak{NLab})$ is a stable arrow labelling. If $\mathfrak{ALab}$ is a stable arrow labelling, then $\mathfrak{ALab2NLab}(\mathfrak{ALab})$ is a stable node labelling.*

Proof. (1) This follows from Lemma 65 and Lemma 66.
(2) This follows from Lemma 67 and Lemma 68.
(3) This follows from Lemma 72.
(4) This follows from Lemma 73.
(5) This follows from Lemma 74.
□

Appendix G. AFs with collective attacks vs. SETAFs

The concept of a SETAF (Definition 11) is very close to that of an Argumentation System in the sense of [14], which allows for collective attacks. However, where the SETAF arrows ($\mathfrak{arr}$) are a subset of $2^{\mathfrak{N}} \times \mathfrak{N}$ in Definition 11, they are a subset of $(2^{\mathfrak{N}} \setminus \emptyset) \times \mathfrak{N}$ in [14, Definition 1]. It is not completely clear why Nielsen and Parsons decided to rule out the empty set as a basis of an attack, since the well-definedness and correctness of their work does not seem to depend on it. An argument that is attacked by the empty set is always rejected (in labelling terms: it is always `out`) and has the same effects regarding the complete, grounded, preferred, semi-stable and stable extensions as if it does not exist at all. However, it can still have advantages to be able to represent such attacks, especially when it comes to the ability to model ABA. It is perfectly possible for an ABA-derivation not to use any assumptions at all, but still attack another ABA-derivation. Hence, if we want SETAFs to be abstractions of ABA, we need to be able to take into account attacks originating from the empty set.

In the following, we show that the difference between AFs with collective attacks and SETAFs is marginal. We show that each SETAF can be represented as AF with collective attacks without affecting the semantics. First, let us recall both definitions.

Definition 11. *An* argumentation framework with set attacks (SETAF) *is a tuple $\mathfrak{SF} = (\mathfrak{N}, \mathfrak{arr})$ where $\mathfrak{N}$ is a finite set of nodes, whose structure can be left implicit, and $\mathfrak{arr} \subseteq 2^{\mathfrak{N}} \times \mathfrak{N}$.*

Definition 76 ([14])**.** *An* argumentation framework with collective attacks *is a tuple $\mathfrak{SF} = (\mathfrak{N}, \mathfrak{arr})$ where $\mathfrak{N}$ is a finite set of nodes, whose structure can be left implicit, and $\mathfrak{arr} \subseteq (2^{\mathfrak{N}} \setminus \emptyset) \times \mathfrak{N}$.*

By definition, each AF with collective attacks is a SETAF.

To map each SETAF to an AF with collective attacks, we simply delete all nodes that are attacked by the empty set (and all attacks they were involved in).

Definition 77. *Let SETAF2collAF* : $\{\mathfrak{SF} \mid \mathfrak{SF}$ *is a SETAF*$\} \to \{\mathfrak{SF} \mid \mathfrak{SF}$ *is a AF with collective attacks*$\}$ *be defined as follows: for a SETAF* $\mathfrak{SF} = (\mathfrak{N}, \mathfrak{arr})$, *we obtain the corresponding AF with collective attacks* $\mathit{SETAF2collAF}(\mathfrak{SF}) = (\mathfrak{N}', \mathfrak{arr}')$ *with*

$$\begin{aligned}\mathfrak{N}' &= \{A \in \mathfrak{N} \mid \nexists(\emptyset, A) \in \mathfrak{arr}\},\\ \mathfrak{arr}' &= \{(\mathfrak{M}, A) \in \mathfrak{arr} \mid \mathfrak{M} \neq \emptyset, A \in \mathfrak{N}', \mathfrak{M} \subseteq \mathfrak{N}'\} = \mathfrak{arr}|_{(2^{\mathfrak{N}'} \setminus \emptyset) \times \mathfrak{N}'}.\end{aligned}$$

The translation partitions the class of all SETAF into equivalence classes where each corresponds to a single AF with collective attacks.

Definition 78. *Two SETAFs* $\mathfrak{SF}_1, \mathfrak{SF}_2$ *are equivalent to each other* ($\mathfrak{SF}_1 \equiv \mathfrak{SF}_2$) *iff*

$$\mathit{SETAF2collAF}(\mathfrak{SF}_1) = \mathit{SETAF2collAF}(\mathfrak{SF}_2).$$

By $C_{\mathit{SETAF2collAF}(\mathfrak{SF})} = \{\mathfrak{SF}' \mid \mathfrak{SF} \equiv \mathfrak{SF}'\}$, *we denote the equivalence class of SETAF* $\mathfrak{SF}$.

Each AF with collective attacks $\mathfrak{SF}$ corresponds to an equivalence class $C_{\mathfrak{SF}}$ that contains each SETAF with additional arguments that are attacked by the empty set and additional attacks that involve arguments attacked by the empty set. Note that $\mathfrak{SF}$ is contained in the equivalence class $C_{\mathfrak{SF}}$ (it is the smallest SETAF contained in the class). For instance, the empty AF with collective attacks $(\emptyset, \emptyset)$ is the minimal representative of the equivalence class

$$C_{(\emptyset,\emptyset)} = \{(\mathfrak{N}, \mathfrak{arr}) \mid \forall A \in \mathfrak{N} : (\emptyset, A) \in \mathfrak{arr}\}.$$

Interestingly, it can be the case that the empty set is stable in a given SETAF $\mathfrak{SF} = (\mathfrak{N}, \mathfrak{arr})$ although it contains nodes, i.e., $\mathfrak{N} \neq \emptyset$. For instance, the SETAF $\mathfrak{SF} = (\mathfrak{N}, \mathfrak{arr})$ with $\mathfrak{N} = \{A, B\}$ and arrows $\mathfrak{arr} = \{(\emptyset, A), (\emptyset, B), (\{A, B\}, A)\}$ admits an empty stable extension. We observe that the considered SETAF belongs to the equivalence class $C_{(\emptyset,\emptyset)}$ which gets mapped to the empty AF with collective attacks (it is well known that the empty set is stable in the empty framework).

We show that AF with collective attacks and SETAF semantics coincide.

Below, we make use of the following notation. For a SETAF $\mathfrak{SF} = (\mathfrak{N}, \mathfrak{arr})$, we write

$$cf(\mathfrak{SF}) = \{\mathfrak{M} \subseteq \mathfrak{N} \mid \mathfrak{N} \text{ is conflict-free in } \mathfrak{SF}\}.$$

Proposition 79. *Let* $\mathfrak{SF} = (\mathfrak{N}, \mathfrak{arr})$ *be a SETAF and let* $\mathit{SETAF2collAF}(\mathfrak{SF}) = \mathfrak{SF}' = (\mathfrak{N}', \mathfrak{arr}')$ *be its corresponding AF with collective attacks. Then*

(1) $cf(\mathfrak{SF}) = cf(\mathfrak{SF}')$;
(2) $\mathfrak{M}^+_{\mathfrak{SF}}|_{\mathfrak{N}'} = \mathfrak{M}^+_{\mathfrak{SF}'}|_{\mathfrak{N}'}$ *for all* $\mathfrak{M} \in cf(\mathfrak{SF}) = cf(\mathfrak{SF}')$;
(3) $F_{\mathfrak{SF}}(\mathfrak{M}) = F_{\mathfrak{SF}'}(\mathfrak{M})$ *for all* $\mathfrak{M} \in cf(\mathfrak{SF}) = cf(\mathfrak{SF}')$.

Proof. By definition, we have $\mathfrak{N}' \subseteq Nh$ and $\mathfrak{arr}' \subseteq \mathfrak{arr}$. Let $\mathcal{A}_\emptyset = \mathfrak{N} \setminus \mathfrak{N}'$ denote all nodes that are attacked by the empty set in $\mathfrak{SF}$.

(1) We show $cf(\mathfrak{SF}) = cf(\mathfrak{SF}')$. First, consider a conflict-free set $\mathfrak{M} \in cf(\mathfrak{SF})$. By definition, $\mathfrak{M}$ is not attacked by any subset of $\mathfrak{M}$; therefore, $(\emptyset, A) \notin \mathfrak{arr}$ for any $A \in \mathfrak{M}$. Hence $\mathfrak{M} \subseteq \mathfrak{N}'$. Since $\mathfrak{arr}' \subseteq \mathfrak{arr}$, i.e., we do not add new attacks in the corresponding AF with collective attacks, we conclude that $\mathfrak{M}$ is conflict-free in $\mathfrak{SF}'$, that is, $\mathfrak{M} \in cf(\mathfrak{SF}')$.
Now, consider a set $\mathfrak{M} \in cf(\mathfrak{SF}')$. We have $\mathfrak{M} \in \mathfrak{N}'$, hence $(\emptyset, A) \notin \mathfrak{arr}$ for all $A \in \mathfrak{M}$, moreover, for all attacks $(b, A) \in \mathfrak{arr} \setminus \mathfrak{arr}'$ it holds that $b \cap \mathcal{A}_\emptyset \neq \emptyset$ (in the translation, we delete only attacks that involve arguments that are attacked by the empty set). Therefore, $\mathfrak{M} \in cf(\mathfrak{SF})$.

(2) Consider a conflict-free set $\mathfrak{M} \in cf(\mathfrak{SF}) = cf(\mathfrak{SF}')$. We show that $\mathfrak{M}^+_{\mathfrak{SF}}|_{\mathfrak{N}'} = \mathfrak{M}^+_{\mathfrak{SF}'}|_{\mathfrak{N}'}$, i.e., $\mathfrak{M}$ attacks the same nodes in $\mathfrak{N}'$ in both $\mathfrak{SF}$ and $\mathfrak{SF}'$. From $\mathfrak{arr}' \subseteq \mathfrak{arr}$ we obtain $\mathfrak{M}^+_{\mathfrak{SF}'}|_{\mathfrak{N}'} \subseteq \mathfrak{M}^+_{\mathfrak{SF}}|_{\mathfrak{N}'}$. For the other direction, consider a node $A \in \mathfrak{M}^+_{\mathfrak{SF}}|_{\mathfrak{N}'}$. Then there is $(\mathfrak{M}', A) \in \mathfrak{arr}$ with $\mathfrak{M}' \subseteq \mathfrak{M}$. It holds that $\mathfrak{M}' \in \mathfrak{N}'$ since $\mathfrak{M} \in \mathfrak{N}'$, therefore $(\mathfrak{M}', A) \in \mathfrak{arr}'$. Hence $A \in \mathfrak{M}^+_{\mathfrak{SF}'}|_{\mathfrak{N}'}$. We obtain $\mathfrak{M}^+_{\mathfrak{SF}}|_{\mathfrak{N}'} = \mathfrak{M}^+_{\mathfrak{SF}'}|_{\mathfrak{N}'}$, as desired.

(3) Consider a conflict-free set $\mathfrak{M} \in cf(\mathfrak{SF}) = cf(\mathfrak{SF}')$. We show that $\mathfrak{M}$ defends the same nodes in $\mathfrak{SF}$ and $\mathfrak{SF}'$.
First, we observe that $\mathfrak{M}$ can only defend nodes in $\mathfrak{N} \setminus \mathcal{A}_\emptyset = \mathfrak{N}'$ since the empty set cannot be counter-attacked.
Now, consider a node $A \in \mathfrak{N}'$. As shown above (cf. 2.), $\mathfrak{M}$ attacks the same nodes in $\mathfrak{N}'$ in both frameworks $\mathfrak{SF}$ and $\mathfrak{SF}'$. Hence, A is defended against the same attacks (b, A) with $b \subseteq Nh'$ in both $\mathfrak{SF}$ and $\mathfrak{SF}'$. It remains to argue that A is defended against all attacks in $\mathfrak{arr} \setminus \mathfrak{arr}'$: consider an attack (b, A) with $b \not\subseteq \mathfrak{N}'$. That is, b contains some argument $B \in \mathcal{A}_\emptyset$ that is attacked by the empty set. Since each set contains the empty set, A is defended against this attack in $\mathfrak{SF}$. □

We obtain the following.

Theorem 80. *Let $\mathfrak{SF} = (\mathfrak{N}, \mathfrak{arr})$ be a SETAF and let $SETAF2collAF(\mathfrak{SF}) = \mathfrak{SF}' = (\mathfrak{N}', \mathfrak{arr}')$ be its corresponding AF with collective attacks. Then $cf(\mathfrak{SF}) = cf(\mathfrak{SF}')$, moreover, for all $\mathfrak{M} \in cf(\mathfrak{SF})$,*

- *$\mathfrak{M}$ is admissible in $\mathfrak{SF}$ iff $\mathfrak{M}$ is admissible in $\mathfrak{SF}'$;*
- *$\mathfrak{M}$ is complete in $\mathfrak{SF}$ iff $\mathfrak{M}$ is complete in $\mathfrak{SF}'$;*
- *$\mathfrak{M}$ is grounded in $\mathfrak{SF}$ iff $\mathfrak{M}$ is grounded in $\mathfrak{SF}'$;*
- *$\mathfrak{M}$ is preferred in $\mathfrak{SF}$ iff $\mathfrak{M}$ is preferred in $\mathfrak{SF}'$;*
- *$\mathfrak{M}$ is stable in $\mathfrak{SF}$ iff $\mathfrak{M}$ is stable in $\mathfrak{SF}'$;*
- *$\mathfrak{M}$ is semi-stable in $\mathfrak{SF}$ iff $\mathfrak{M}$ is semi-stable in $\mathfrak{SF}'$.*

Proof. As shown above, the conflict-free sets coincide, and the characteristic function restricted to the conflict-free sets yields the same output. Hence, by definition, the statement follows for admissible, complete, preferred, grounded semantics. Regarding range-based semantics, it suffices to observe that for all sets $\mathfrak{M} \subseteq \mathfrak{N}$, it holds that $\mathfrak{N} \setminus \mathfrak{N}' \subseteq Mh^+_{\mathfrak{SF}}$ (since $\emptyset \subseteq \mathfrak{M}$). That is, all arguments attacked by the empty set are contained in the range of $\mathfrak{M}$. We obtain that semi-stable and stable semantics coincide. □

Differently phrased, all SETAFs in the same equivalence class coincide on the semantics.

Corollary 81. *Let $\mathfrak{SF}$ be a SETAF and let $C_{SETAF2collAF(\mathfrak{SF})}$ be its corresponding equivalence class. Then $\sigma(\mathfrak{SF}_1) = \sigma(\mathfrak{SF}_2)$ for any two SETAFs $\mathfrak{SF}_1, \mathfrak{SF}_2 \in C_{SETAF2collAF(\mathfrak{SF})}$, for any semantics σ considered in this paper.*

From the one-to-one correspondence between extension semantics and labellings (cf. Theorem 58) and from Theorem 80, we obtain the following result.

Theorem 82. *Let $\mathfrak{SF} = (\mathfrak{N}, \mathfrak{arr})$ be a SETAF, let SETAF2collAF$(\mathfrak{SF}) = \mathfrak{SF}' = (\mathfrak{N}', \mathfrak{arr}')$ be its corresponding AF with collective attacks, and let $\mathcal{A}_\emptyset = \mathfrak{N} \setminus \mathfrak{N}'$ denote all nodes that are attacked by the empty set in $\mathfrak{SF}$. Then,*

- *for each SETAF node labelling $\mathfrak{NLab}$ for $\mathfrak{SF}$, let $\mathfrak{NLab}' = \mathfrak{NLab}|_{Nh'}$ denote the SETAF node labelling restricted to $\mathfrak{SF}'$. It holds that*
  - $*$ *$\mathfrak{NLab}$ is complete on $\mathfrak{SF}$ iff $\mathfrak{NLab}'$ is complete on $\mathfrak{SF}'$;*
  - $*$ *$\mathfrak{NLab}$ is grounded on $\mathfrak{SF}$ iff $\mathfrak{NLab}'$ is grounded on $\mathfrak{SF}'$;*
  - $*$ *$\mathfrak{NLab}$ is preferred on $\mathfrak{SF}$ iff $\mathfrak{NLab}'$ is preferred on $\mathfrak{SF}'$;*
  - $*$ *$\mathfrak{NLab}$ is stable on $\mathfrak{SF}$ iff $\mathfrak{NLab}'$ is stable on $\mathfrak{SF}'$;*
  - $*$ *$\mathfrak{NLab}$ is semi-stable on $\mathfrak{SF}$ iff $\mathfrak{NLab}'$ is semi-stable on $\mathfrak{SF}'$.*
- *for each SETAF node labelling $\mathfrak{NLab}'$ for $\mathfrak{SF}'$, let $\mathfrak{NLab} = \mathfrak{NLab} \cup \{(A, \texttt{out}) \mid A \in \mathcal{A}_\emptyset\}$ denote the extension of $\mathfrak{NLab}'$ to the set of nodes in $\mathcal{A}_\emptyset$ (observe that each such node is assigned `out`). It holds that*
  - $*$ *$\mathfrak{NLab}'$ is complete on $\mathfrak{SF}'$ iff $\mathfrak{NLab}$ is complete on $\mathfrak{SF}$;*
  - $*$ *$\mathfrak{NLab}'$ is grounded on $\mathfrak{SF}'$ iff $\mathfrak{NLab}$ is grounded on $\mathfrak{SF}$;*
  - $*$ *$\mathfrak{NLab}'$ is preferred on $\mathfrak{SF}'$ iff $\mathfrak{NLab}$ is preferred on $\mathfrak{SF}$;*
  - $*$ *$\mathfrak{NLab}'$ is stable on $\mathfrak{SF}'$ iff $\mathfrak{NLab}$ is stable on $\mathfrak{SF}$;*
  - $*$ *$\mathfrak{NLab}'$ is semi-stable on $\mathfrak{SF}'$ iff $\mathfrak{NLab}$ is semi-stable on $\mathfrak{SF}$.*

Proof. We provide a proof for complete semantics. The remaining proofs are analogous.

First, let $\mathfrak{NLab}$ be a complete labelling for $\mathfrak{SF}$, and let $\mathfrak{NLab}' = \mathfrak{NLab}|_{Nh'}$ denote the SETAF node labelling restricted to $\mathfrak{SF}'$. We show that $\mathfrak{NLab}$ is complete on $\mathfrak{SF}$ iff $\mathfrak{NLab}'$ is complete on $\mathfrak{SF}'$:

from left to right By Theorem 58, $\texttt{in}(\mathfrak{NLab})$ is a complete node extension of $\mathfrak{SF}$. Moreover, by Theorem 80, $\texttt{in}(\mathfrak{NLab})$ is a complete node extension of $\mathfrak{M}'$. Applying Theorem 58 again, we obtain that $\texttt{in}(\mathfrak{NLab})$ induces a complete node labelling of $\mathfrak{SF}'$.

from right to left Analogous to the other direction.

Now, let $\mathfrak{NLab}'$ be a complete node labelling for $\mathfrak{SF}'$ and let $\mathfrak{NLab} = \mathfrak{NLab} \cup \{(A, \texttt{out}) \mid A \in \mathcal{A}_\emptyset\}$ denote the extension of $\mathfrak{NLab}'$ to the set of nodes in $\mathcal{A}_\emptyset$. We show that $\mathfrak{NLab}'$ is complete on $\mathfrak{SF}'$ iff $\mathfrak{NLab}$ is complete on $\mathfrak{SF}$:

from left to right By Theorem 58, $\texttt{in}(\mathfrak{NLab}')$ is a complete node extension of $\mathfrak{SF}'$. By Theorem 80, $\texttt{in}(\mathfrak{NLab}')$ is a complete node extension of $\mathfrak{M}$. Applying Theorem 58 again, we obtain that $\texttt{in}(\mathfrak{NLab}')$ induces a complete node labelling of $\mathfrak{SF}$. Moreover, since each node $A \in \mathcal{A}_\emptyset$ is attacked (by the empty set), we obtain that A is labelled `out`, as desired.

from right to left Analogous to the first case above (the restriction of a complete node labelling of $\mathfrak{SF}$ to $\mathfrak{SF}'$ is complete).

$\square$

Appendix H. Equivalence of Arrow Extensions and Arrow Labellings for Argumentation Frameworks

Definition 83. *Let (N, arr) be an argumentation framework and let $a \subseteq arr$ be conflict-free. We define the function $a2ALab(a) = (a, a^+, arr \setminus (a \cup a^+))$.*

Definition 84. *Let (N, arr) be an argumentation framework and let ALab be an arrow labelling. We define the function $ALab2a(ALab) = \mathtt{in}(ALab)$.*

Theorem 85. *Let (N, arr) be an argumentation framework.*

(1) *If $a \subseteq arr$ is a complete extension then $a2ALab(a)$ is a complete labelling.*
(2) *If ALab is a complete labelling then $ALab2a(ALab)$ is a complete extension.*

Proof. (1) Let $a2ALab(a) = ALab$. Let a be a complete extension. We show that $a2ALab(a) = ALab$ is a complete labelling:

- We show that $ALab((A, B)) = \mathtt{in}$ implies $ALab((C, A)) = \mathtt{out}$ for each arrow $(C, A) \in arr$. Let $(A, B) \in \mathtt{in}(ALab)$ and consider an arrow $(C, A) \in arr$. By Definition 84, $\mathtt{in}(ALab) = a$. Since a is complete, the arrow (A, B) is defended by a (by Definition 6). Hence, $(C, A) \in a^+$. By Definition 84, we obtain that $(C, A) \in \mathtt{out}(ALab)$.
- We show that $ALab((A, B)) = \mathtt{out}$ implies $ALab((C, A)) = \mathtt{in}$ for some $(C, A) \in arr$. Let $(A, B) \in \mathtt{out}(ALab)$. By Definition 84, $(A, B) \in a^+$. That is, (A, B) is attacked by a. Hence, there is some arrow $(C, A) \in arr$ such that $(C, A) \in a$. By Definition 83, we obtain $(C, A) \in \mathtt{in}(ALab)$.
- We show that if $ALab((A, B)) = \mathtt{undec}$ then not for each $(C, A) \in arr$ it holds that $ALab((C, A)) = \mathtt{out}$, and there does not exist a $(C, A) \in arr$ such that $ALab((C, A)) = \mathtt{in}$.
  Let $(A, B) \in \mathtt{undec}(ALab)$. We provide a proof by contradiction.
  First assume for each $(C, A) \in arr$ it holds that $ALab((C, A)) = \mathtt{out}$. By Definition 83, for each $(C, A) \in arr$ it holds that $(C, A) \in a^+$. Hence, each attacker of (A, B) is attacked. By Definition 5, we obtain that $(A, B) \in F(a)$. By the fundamental lemma and by definition of complete extension semantics, we obtain that $(A, B) \in a$. Therefore, by definition of *a2ALab*, we obtain $ALab((A, B)) = \mathtt{in}$, contradiction to the assumption $(A, B) \in \mathtt{undec}(ALab)$. Hence, we obtain that not for each $(C, A) \in arr$ it holds that $ALab((C, A)) = \mathtt{out}$.
  Now assume that there exists $(C, A) \in arr$ such that $ALab((C, A)) = \mathtt{in}$. By Definition 83, $(C, A) \in a$. Since (C, A) attacks (A, B) it furthermore holds that $(A, B) \in a^+$. Hence, $ALab((A, B)) = \mathtt{out}$, contradiction to our initial assumption. Hence we obtain that there does not exist a $(C, A) \in arr$ such that $ALab((C, A)) = \mathtt{in}$.

(2) Let $ALab2a(ALab) = a$. Let *ALab* be a complete labelling. We show that $ALab2a(ALab) = a$ is a complete extension.

- We show that a is conflict-free. Let $(A, B), (C, D) \in a$. By Definition 84, $(A, B), (C, D) \in \mathtt{in}(ALab)$. Towards a contradiction, assume (A, B) attacks (C, D), that is, $B = C$. By definition of a complete labelling, we obtain $ALab((A, B)) = \mathtt{out}$. Hence we obtain a contradiction.
- We show that $a = F(a)$.
  First, let $(A, B) \in a$. By Definition 84, $ALab((A, B)) = \mathtt{in}$. By definition of a complete labelling, for all $(C, A) \in arr$ it holds that $ALab((C, A)) = \mathtt{out}$. Hence, there exists $(D, C) \in arr$ such

that $ALab((D,C)) = \texttt{in}$. Hence, (A,B) is defended by a against each attack. We obtain $(A,B) \in F(a)$.

For the other direction, let $(A,B) \in F(a)$. Hence, each attacker of (A,B) is attacked by a: for all $(C,A) \in arr$ it holds that $(C,A) \in a^+$. Towards a contradiction, we assume $ALab((A,B)) \neq \texttt{in}$. Then $ALab((A,B)) \in \{\texttt{out}, \texttt{undec}\}$. We proceed by case distinction.

First assume, $ALab((A,B)) = \texttt{out}$. By Definition 7, there exists a $(C,A) \in arr$ such that $ALab((C,A)) = \texttt{in}$. By Definition 84, $(C,A) \in a$. We obtain a contradiction since a is conflict-free.

Now assume, $ALab((A,B)) = \texttt{undec}$. By Definition 7, there is some $(C,A) \in arr$ such that $ALab((C,A)) \neq \texttt{out}$, and there does not exist a $(C,A) \in arr$ such that $ALab((C,A)) = \texttt{in}$. By the first condition, there is some $(C,A) \in arr$ such that $ALab((C,A)) \in \{\texttt{in}, \texttt{undec}\}$. By the second condition, $ALab((C,A)) \neq \texttt{in}$. Hence, there is some $(C,A) \in arr$ such that $ALab((C,A)) = \texttt{undec}$. Since $(C,A) \in a^+$ (by our initial assumption), there is some $(D,C) \in a$. By Definition 84, $ALab((D,C)) = \texttt{in}$. By Definition 7, each arrow which is attacked by (D,C) is labelled out. Hence $ALab((C,A)) = \texttt{out}$. Contradiction to the assumption that $ALab((C,A)) = \texttt{undec}$.

We obtain $ALab((A,B)) = \texttt{in}$ and therefore, $(A,B) \in a$, as desired.

□

Theorem 86. *Let (N, arr) be an argumentation framework and let $a \subseteq arr$. Then*

(1) *if a is a grounded extension then $a2ALab(a)$ is a grounded labelling;*
(2) *if a is a preferred extension then $a2ALab(a)$ is a preferred labelling;*
(3) *if a is a semi-stable extension then $a2ALab(a)$ is a semi-stable labelling;*
(4) *if a is a stable extension then $a2ALab(a)$ is a stable labelling.*

Proof. Let $a2ALab(a) = ALab$.

(1) Let a be the grounded extension. By Theorem 85, it holds that $ALab$ is a complete labelling. We show that $\texttt{in}(ALab)$ is $\subseteq$-minimal among all complete arrow labellings. Towards a contradiction, assume there is a complete labelling $AbLab'$ such that $\texttt{in}(ALab') \subsetneq \texttt{in}(ALab)$. By Theorem 85, $a' = \texttt{in}(ALab')$ is a complete extension. By Definition 83, $\texttt{in}(ALab) = a$, hence there is a complete extension a' such that $a' \subsetneq a$, contradiction to $\subseteq$-minimality of a. We obtain that $ALab$ is the grounded labelling.
(2) Let a be a preferred extension. By Theorem 85, it holds that $ALab$ is a complete labelling. We show that $\texttt{in}(ALab)$ is $\subseteq$-maximal among all complete arrow labellings. Towards a contradiction, assume there is a complete labelling $AbLab'$ such that $\texttt{in}(ALab) \subsetneq \texttt{in}(ALab')$. By Theorem 85, $a' = \texttt{in}(ALab')$ is a complete extension, contradiction to $\subseteq$-maximality of a. We obtain that $ALab$ is a preferred labelling.
(3) Let a be a semi-stable extension. By Theorem 85, it holds that $ALab$ is a complete labelling. Moreover, by definition of semi-stable semantics, $a \cup a^+$ is $\subseteq$-maximal among all complete extensions. We show that $\texttt{undec}(ALab)$ is $\subseteq$-minimal among all complete labellings.
Towards a contradiction, suppose there is a complete labelling $ALab'$ such that $\texttt{undec}(ALab') \subsetneq \texttt{undec}(ALab)$. By Theorem 85, $a' = \texttt{in}(ALab')$ is a complete extension. By Definition 83, $\texttt{undec}(ALab') = arr \setminus (a' \cup (a')^+)$. Hence, $arr \setminus (a' \cup (a')^+) \subsetneq arr \setminus (a \cup a^+)$, therefore,

$a' \cup (a')^+ \supsetneq a \cup a^+$, that is, we have found a complete extension a' showing that a is not semi-stable, contradiction to our initial assumption. We obtain that *ALab* is a semi-stable labelling.

(4) Let a be a stable extension. By Theorem 85, it holds that *ALab* is a complete labelling. Moreover, by definition of stable semantics, $a \cup a^+ = arr$. Hence, by Definition 83, $\texttt{undec}(ALab) = \emptyset$. We obtain that *ALab* is a stable labelling. $\square$

Theorem 87. *Let (N, arr) be an argumentation framework and let ALab be an arrow labelling. Then*

(1) *if ALab is a grounded labelling then $ALab2a(ALab)$ is a grounded extension;*
(2) *if ALab is a preferred labelling then $ALab2a(ALab)$ is a preferred extension;*
(3) *if ALab is a semi-stable labelling then $ALab2a(ALab)$ is a semi-stable extension;*
(4) *if ALab is a stable labelling then $ALab2a(ALab)$ is a stable extension.*

Proof. Let $ALab2a(ALab) = a$.

(1) Let *ALab* be the grounded labelling. By Theorem 85, it holds that a is a complete extension. We show that a is $\subseteq$-minimal among all complete extensions. Towards a contradiction, assume there is a complete extension a' such that $a' \subsetneq a$. By Theorem 85, $a2ALab(a')$ is a complete arrow labelling. By Definition 83, $a = \texttt{in}(ALab)$ and $a' = \texttt{in}(a2ALab(a'))$. Therefore, $\texttt{in}(a2ALab(a')) \subsetneq \texttt{in}(ALab)$, contradiction to the assumption that *ALab* is the grounded labelling. We obtain that a is the grounded extension.

(2) Let *ALab* be a preferred labelling. By Theorem 85, it holds that a is a complete extension. We show that a is $\subseteq$-maximal among all complete extensions. Towards a contradiction, assume there is a complete extension a' such that $a' \supsetneq a$. By Theorem 85, $a2ALab(a')$ is a complete arrow labelling. By Definition 83, $a = \texttt{in}(ALab)$ and $a' = \texttt{in}(a2ALab(a'))$. Therefore, $\texttt{in}(a2ALab(a')) \supsetneq \texttt{in}(ALab)$, contradiction to the assumption that *ALab* is $\subseteq$-maximal. We obtain that a is a preferred extension.

(3) Let *ALab* be a semi-stable labelling. By Theorem 85, it holds that a is a complete extension. Moreover, by definition of semi-stable semantics, $\texttt{undec}(ALab)$ is $\subseteq$-minimal among all complete labellings. We show that $a \cup a^+$ is $\subseteq$-maximal among all complete extensions.
Towards a contradiction, suppose there is a complete extension a' such that $a' \cup (a')^+ \supsetneq a \cup a^+$. By Theorem 85, $ALab' = a2ALab(a')$ is a complete labelling. By Definition 83, $\texttt{undec}(ALab') = arr \setminus (a' \cup (a')^+)$. Hence, $arr \setminus (a' \cup (a')^+) \subsetneq arr \setminus (a \cup a^+)$, and therefore, $\texttt{undec}(ALab') \subsetneq \texttt{undec}(ALab)$. Consequently, *ALab* cannot be a semi-stable labelling; hence, we conclude that a is a semi-stable extension.

(4) Let *ALab* be a stable labelling. By Theorem 85, it holds that a is a complete extension. By definition of stable labellings, $\texttt{undec}(ALab) = \emptyset$. Hence, each attack is either labelled $\texttt{in}$ or labelled $\texttt{out}$. By Definition 84, $\texttt{in}(ALab) = a$. By Definition 7, if an attack (A, B) is labelled $\texttt{out}$ then there is $(C, A) \in arr$ such that $ALab((C, A)) = \texttt{in}$. Hence, $a^+ = arr \setminus a$. We obtain that a is a stable labelling. $\square$

Lemma 88. *Let $AF = (N, arr)$ be an argumentation framework.*

(1) *For a complete arrow labelling ALab it holds that $a2ALab(ALab2a(ALab)) = ALab$.*
(2) *For a complete arrow extension a it holds that $ALab2a(a2ALab(a)) = a$.*

Proof. (1) Let $ALab2a(ALab) = a$. We prove the following three properties, for an arbitrary arrow $(A, B) \in arr$.

- If $AbLab((A,B)) = \texttt{in}$ then $a2ALab(a)((A,B)) = \texttt{in}$.
  Suppose $AbLab((A,B)) = \texttt{in}$. By Definition 84, $(A,B) \in a$. By Definition 83, we obtain $a2ALab(a)((A,B)) = \texttt{in}$.
- If $ALab((A,B)) = \texttt{out}$ then $a2ALab(a)((A,B)) = \texttt{out}$.
  Suppose $ALab((A,B)) = \texttt{out}$. Then by Definition of a complete arrow labelling, it follows that there exists a $(C,A) \in arr$ such that $ALab((C,A)) = \texttt{in}$. By Definition 84 $(C,A) \in a$. By Definition 83, we obtain $a2ALab(a)((C,A)) = \texttt{in}$. We proceed by case distinction.
  First assume $a2ALab(a)((A,B)) = \texttt{in}$. Then $a2ALab(a)((C,A)) = \texttt{out}$, by definition of a complete arrow labelling. Hence we obtain a contradiction.
  Next assume $a2ALab(a)((A,B)) = \texttt{undec}$. By definition of a complete arrow labelling, there is some $(C,A) \in arr$ such that $a2ALab(a)((C,A)) \neq \texttt{out}$, and there does not exist a $(C,A) \in arr$ such that $a2ALab(a)((C,A)) = \texttt{in}$. The latter condition contradicts our assumption.
  We obtain $a2ALab(a)((A,B)) = \texttt{out}$, as desired.
- If $ALab((A,B)) = \texttt{undec}$ then $a2ALab(a)((A,B)) = \texttt{undec}$.
  Suppose $AbLab((A,B)) = \texttt{undec}$. By definition of a complete arrow labelling, there is some $(C,A) \in arr$ such that $ALab((C,A)) \neq \texttt{out}$, and there does not exist a $(C,A) \in arr$ such that $ALab((C,A)) = \texttt{in}$.
  To show that $a2ALab(a)((A,B)) = \texttt{undec}$ we provide a proof by contradiction. Towards a contradiction, assume $a2ALab(a)((A,B)) \neq \texttt{undec}$. Then either $a2ALab(a)((A,B)) = \texttt{in}$ or $a2ALab(a)((A,B)) = \texttt{out}$. We proceed by case distinction.
  First assume $a2ALab(a)((A,B)) = \texttt{in}$. By Definition 83, $(A,B) \in a$. By Definition 84, we obtain $ALab((A,B)) = \texttt{in}$. This is a contradiction to $ALab((A,B)) = \texttt{undec}$.
  Next assume $a2ALab(a)((A,B)) = \texttt{out}$. By Definition 83, $(A,B) \in a^+$. By definition of a complete arrow extension, there is some $(C,A) \in arr$ with $(C,A) \in a$. By Definition 84, $ALab((C,A)) = \texttt{in}$. This is a contradiction to the assumption that there does not exist a $(C,A) \in arr$ such that $ALab((C,A)) = \texttt{in}$.
  Hence we can conclude $a2ALab(a)((A,B)) = \texttt{undec}$.

(2) Consider a complete extension a. By Definition 83, $\texttt{in}(a2ALab(a)) = a$. By Definition 84, we obtain $ALab2a(a2ALab(a)) = \texttt{in}(a2ALab(a)) = a$.
□

Theorem 89. *When restricted to complete node labellings and complete arrow labellings, the functions ALab2a and a2ALab become bijections and each other's inverses.*

Proof. This follows directly from Lemma 88. □

Appendix I. Equivalence of Node Extensions and Arrow Extensions for Argumentation Frameworks

We define the functions

$$Args2a = ALab2a \circ NLab2ALab \circ Args2NLab$$

and

$$a2Args = Args2NLab \circ ALab2NLab \circ ALab2a.$$

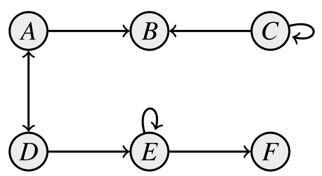
Theorem 90. *Let $AF = (N, arr)$ be an argumentation framework and let $M \subseteq N$ and $a \subseteq arr$. It holds that:*

(1) *If M is complete node extension then then $Args2a(M)$ is a complete arrow extension.*
If a is a complete arrow extension, then $a2Args(a)$ is a complete node extension.
(2) *When restricted to complete node labellings and complete arrow labellings, the functions $ALab2NLab$ and $NLab2ALab$ become bijections and each other's inverses.*
(3) *If M is a grounded node extension, then $Args2a(M)$ is a grounded arrow extension.*
If a is a grounded arrow extension, then $a2Args(a)$ is a grounded node extension.
(4) *If M is a preferred node extension, then $Args2a(M)$ is a preferred arrow extension.*
If a is a preferred arrow extension, then $a2Args(a)$ is a preferred node extension.
(5) *If M is a stable node extension, then $Args2a(M)$ is a stable arrow extension.*
If a is a stable arrow extension, then $a2Args(a)$ is a stable node extension.

Proof. (1) First, let M is complete node extension. By results from [33] and as summarised in Table 2, $Args2NLab(M)$ is a complete node labelling. By Theorem 8, $NLab2ALab(Args2NLab(M))$ is a complete arrow labelling. Finally, by Theorem 87, $ALab2a(NLab2ALab(Args2NLab(M))) = Args2a(M)$ is a complete arrow extension.
Now, let a a complete arrow extension. By Theorem 86, $a2ALab(a)$ is a complete arrow labelling. By Theorem 8, $ALab2NLab(a2ALab(a))$ is a complete node labelling. Finally, by results from [33] and as summarised in Table 2, $NLab2Args(ALab2NLab(a2ALab(a))) = a2Args(a)$ is a complete node extensions.
(2) By results from [33], Theorem 8, and Theorem 89.
(3) Analogous to point 1.
(4) Analogous to point 1.
(5) Analogous to point 1. □

For semi-stable semantics, we consider the following counter-example:

Example 91. *Let us recall the $AF = (N, arr)$ from Example 10 with $N = \{A, B, C, D, E, F\}$ and $arr = \{(A, B), (C, B), (C, C), (A, D), (D, A), (D, E), (E, E), (E, F)\}$.*



The AF has three complete node extensions $M_1 = \emptyset$, $M_2 = \{A\}$, and $M_3 = \{D, F\}$ that are one-to-one related to the complete arrow extensions $a_1 = \emptyset$, $a_2 = \{(A, B), (A, D)\}$, and $a_3 = \{(D, A), (D, E)\}$ via the functions Args2a and a2Args.

Let us see the function Args2a = ALab2a ∘ NLab2ALab ∘ Args2NLab at work. Let us compute Args2a(M_2) in three steps:

$$\begin{aligned}
Args2NLab(M_2) &= (\{A\}, \{B, D\}, \{C, E, F\}) \\
NLab2ALab(Args2NLab(M_2)) &= (\{(A,B),(A,D)\}, \{(D,A),(D,E)\}, \{(C,B),(C,C),\\
&\qquad (E,E),(E,F)\}) \\
ALab2a(NLab2ALab(Args2NLab(M_2))) &= \{(A,B),(A,D)\}
\end{aligned}$$

Hence, we obtain Args2a(M_2) = a_2.

The set M_2 is the only semi-stable node extension of AF. However, the set a_2 is not a semi-stable arrow extension since $a_3 \cup a_3^+ \supset a_2 \cup a_2^+$. Indeed, a_3 is the unique semi-stable arrow extension of AF. On the other hand, $M_3 = a2Args(a_3)$ is not a semi-stable node extension.

Appendix J. Equivalence of Arrow Extensions and Arrow Labellings for SETAFs

Definition 92. *Let $(\mathfrak{N}, \mathfrak{arr})$ be a SETAF and let $\mathfrak{a} \subseteq \mathfrak{arr}$. We define the function $\mathfrak{a2ALab}(\mathfrak{a}) = (\mathfrak{a}, \mathfrak{a}^+, \mathfrak{arr} \setminus (\mathfrak{a} \cup \mathfrak{a}^+))$.*

Definition 93. *Let $(\mathfrak{N}, \mathfrak{arr})$ be a SETAF and let $\mathfrak{ALab}$ be an arrow labelling. We define the function $\mathfrak{ALab2a}(\mathfrak{ALab}) = \mathtt{in}(\mathfrak{ALab})$.*

Theorem 94. *Let $(\mathfrak{N}, \mathfrak{arr})$ be a SETAF.*

(1) *If $\mathfrak{a} \subseteq \mathfrak{arr}$ is a complete extension then $\mathfrak{a2ALab}(\mathfrak{a})$ is a complete labelling.*
(2) *If $\mathfrak{ALab}$ is a complete labelling then $\mathfrak{ALab2a}(\mathfrak{ALab})$ is a complete extension.*

Proof. (1) Let $\mathfrak{a}$ be a complete extension and let $\mathfrak{a2ALab}(\mathfrak{a}) = \mathfrak{ALab}$. We show that $\mathfrak{a2ALab}(\mathfrak{a}) = \mathfrak{ALab}$ is a complete labelling:

- We show that if $\mathfrak{ALab}((\mathfrak{M}, A)) = \mathtt{in}$ then for each $(\mathfrak{M}', B) \in \mathfrak{arr}$ with $B \in \mathfrak{M}$ it holds that $\mathfrak{ALab}((\mathfrak{M}', B)) = \mathtt{out}$. Let $(\mathfrak{M}, A) \in \mathtt{in}(\mathfrak{ALab})$ and consider an arrow $(\mathfrak{M}', B) \in \mathfrak{arr}$ with $B \in \mathfrak{M}$. By Definition 93, $\mathfrak{M} \in \mathfrak{a}$. Since $\mathfrak{a}$ is complete, the arrow $(\mathfrak{M}, A)$ is defended by $\mathfrak{a}$. Hence, $(\mathfrak{M}', B) \in \mathfrak{a}^+$. By Definition 93, we obtain that $(\mathfrak{M}', B) \in \mathtt{out}(\mathfrak{ALab})$.
- We show that if $\mathfrak{ALab}((\mathfrak{M}, A)) = \mathtt{out}$ then there exists an $(\mathfrak{M}', B) \in \mathfrak{arr}$ such that $B \in \mathfrak{M}$ and $\mathfrak{ALab}((\mathfrak{M}', B)) = \mathtt{in}$. Let $(Mh, A) \in \mathtt{out}(\mathfrak{ALab})$. By Definition 93, $(\mathfrak{M}, A) \in \mathfrak{a}^+$. Hence, there is some arrow $(\mathfrak{M}', B) \in \mathfrak{a}$ such that $B \in \mathfrak{M}$. By Definition 92, we obtain $(\mathfrak{M}', B) \in \mathtt{in}(\mathfrak{ALab})$.
- We show that if $\mathfrak{ALab}((\mathfrak{M}, A)) = \mathtt{undec}$ then not for each $(\mathfrak{M}', B) \in \mathfrak{arr}$ such that $B \in \mathfrak{M}$ it holds that $\mathfrak{ALab}((\mathfrak{M}', B)) = \mathtt{out}$ and there does not exist an $(\mathfrak{M}', B) \in \mathfrak{arr}$ such that $B \in \mathfrak{M}$ and $\mathfrak{ALab}((\mathfrak{M}', B)) = \mathtt{in}$.
  Let $(\mathfrak{M}, A) \in \mathtt{undec}(\mathfrak{ALab})$. We provide a proof by contradiction.
  First assume for each $(\mathfrak{M}', B) \in \mathfrak{arr}$ with $B \in \mathfrak{M}$ it holds that $\mathfrak{ALab}((\mathfrak{M}', B)) = \mathtt{out}$. By Definition 92, for each $(\mathfrak{M}', B) \in \mathfrak{arr}$ with $B \in \mathfrak{M}$ it holds that $(\mathfrak{M}', B) \in \mathfrak{a}^+$. By Definition 18, we obtain that $(\mathfrak{M}, A) \in F(\mathfrak{a})$. By the fundamental lemma and by definition of complete semantics,

we obtain that $(\mathfrak{M}, A) \in \mathfrak{a}$. Therefore, by definition of $\mathfrak{a}2\mathfrak{ALab}$, we obtain $\mathfrak{ALab}((\mathfrak{M}, A)) = \mathtt{in}$, contradiction to the assumption $(\mathfrak{M}, A) \in \mathtt{undec}(\mathfrak{ALab})$.
Now assume that there exists $(\mathfrak{M}', B) \in \mathfrak{arr}$ with $B \in \mathfrak{M}$ such that $\mathfrak{ALab}((\mathfrak{M}', B)) = \mathtt{in}$. By Definition 92, $(\mathfrak{M}', B) \in \mathfrak{a}$. Hence $(\mathfrak{M}, A) \in \mathfrak{a}^+$. Therefore we obtain $\mathfrak{ALab}((\mathfrak{M}, A)) = \mathtt{out}$, contradiction to our initial assumption. This concludes the proof of the statement.

(2) Let $\mathfrak{ALab}$ be a complete labelling and let $\mathfrak{ALab}2\mathfrak{a}(\mathfrak{ALab}) = \mathfrak{a}$. We show that $\mathfrak{ALab}2\mathfrak{a}(\mathfrak{ALab}) = \mathfrak{a}$ is a complete extension.

- We show that $\mathfrak{a}$ is conflict-free. Let $(\mathfrak{M}, A), (\mathfrak{M}', B) \in \mathfrak{a}$. By Definition 93, $(\mathfrak{M}, A), (C, D) \in \mathtt{in}(\mathfrak{ALab})$. Towards a contradiction, assume $(\mathfrak{M}, A)$ attacks $(\mathfrak{M}', B)$, that is, $B \in \mathfrak{M}$. By definition of a complete labelling, we obtain $\mathfrak{ALab}((\mathfrak{M}, A)) = \mathtt{out}$. Hence we obtain a contradiction.
- We show that $\mathfrak{a} = F(\mathfrak{a})$.
First, let $(\mathfrak{M}, A) \in \mathfrak{a}$. By Definition 93, $\mathfrak{ALab}((\mathfrak{M}, A)) = \mathtt{in}$. By definition of a complete labelling, for all $(\mathfrak{M}', B) \in arr$ with $B \in \mathfrak{M}$ it holds that $\mathfrak{ALab}((\mathfrak{M}', B)) = \mathtt{out}$. Hence, there exists $(\mathfrak{M}'', C) \in \mathfrak{arr}$ with $C \in \mathfrak{M}'$ such that $\mathfrak{ALab}((\mathfrak{M}'', C)) = \mathtt{in}$. Hence, $(\mathfrak{M}, A)$ is defended by $\mathfrak{a}$ against each attack. We obtain $(\mathfrak{M}, A) \in F(\mathfrak{a})$.
For the other direction, let $(\mathfrak{M}, A) \in F(\mathfrak{a})$. Each attacker of $(\mathfrak{M}, A)$ is attacked by $\mathfrak{a}$: for all $(\mathfrak{M}', B) \in arr$ with $B \in \mathfrak{M}$ it holds that $(\mathfrak{M}', B) \in \mathfrak{a}^+$. Towards a contradiction, we assume $\mathfrak{ALab}((\mathfrak{M}, A)) \neq \mathtt{in}$. Then $\mathfrak{ALab}((\mathfrak{M}, A)) \in \{\mathtt{out}, \mathtt{undec}\}$. We proceed by case distinction.
First assume, $\mathfrak{ALab}((\mathfrak{M}, A)) = \mathtt{out}$. By Definition 17, there exists a $(\mathfrak{M}', B) \in \mathfrak{arr}$ with $B \in \mathfrak{M}$ such that $\mathfrak{ALab}((\mathfrak{M}', B)) = \mathtt{in}$. By Definition 93, $(\mathfrak{M}', B) \in \mathfrak{a}$. We obtain a contradiction since $\mathfrak{a}$ is conflict-free.
Now assume, $\mathfrak{ALab}((\mathfrak{M}, A)) = \mathtt{undec}$. By Definition 17, there is some $(\mathfrak{M}', B) \in \mathfrak{arr}$ with $B \in \mathfrak{M}$ such that $\mathfrak{ALab}((\mathfrak{M}', B)) \neq \mathtt{out}$, and there does not exist a $(\mathfrak{M}', B) \in \mathfrak{arr}$ with $B \in \mathfrak{M}$ such that $\mathfrak{ALab}((\mathfrak{M}', B)) = \mathtt{in}$. By the first condition, there is some $(\mathfrak{M}', B) \in \mathfrak{arr}$ such that $\mathfrak{ALab}((\mathfrak{M}', B)) \in \{\mathtt{in}, \mathtt{undec}\}$. By the second condition, $\mathfrak{ALab}((\mathfrak{M}', B)) \neq \mathtt{in}$. Hence, there is some $(\mathfrak{M}', B) \in \mathfrak{arr}$ such that $\mathfrak{ALab}((\mathfrak{M}', B)) = \mathtt{undec}$. Since $(\mathfrak{M}', B) \in \mathfrak{a}^+$ (by our initial assumption), there is some $(\mathfrak{M}'', C) \in \mathfrak{a}$ with $C \in \mathfrak{M}$. By Definition 93, $\mathfrak{ALab}((\mathfrak{M}'', C)) = \mathtt{in}$. By Definition 17, each arrow which is attacked by $(\mathfrak{M}'', C)$ is labelled $\mathtt{out}$. Hence $\mathfrak{ALab}((\mathfrak{M}', B)) = \mathtt{out}$. Contradiction to the assumption that $\mathfrak{ALab}((\mathfrak{M}', B)) = \mathtt{undec}$.
We obtain $\mathfrak{ALab}((\mathfrak{M}, A)) = \mathtt{in}$ and therefore, $(\mathfrak{M}, A) \in \mathfrak{a}$, as desired.

□

Theorem 95. *Let* $(\mathfrak{N}, \mathfrak{arr})$ *be a SETAF and let* $\mathfrak{a} \subseteq \mathfrak{arr}$. *Then*

(1) *if* $\mathfrak{a}$ *is a grounded extension then* $\mathfrak{a}2\mathfrak{ALab}(\mathfrak{a})$ *is a grounded labelling;*
(2) *if* $\mathfrak{a}$ *is a preferred extension then* $\mathfrak{a}2\mathfrak{ALab}(\mathfrak{a})$ *is a preferred labelling;*
(3) *if* $\mathfrak{a}$ *is a semi-stable extension then* $\mathfrak{a}2\mathfrak{ALab}(\mathfrak{a})$ *is a semi-stable labelling;*
(4) *if* $\mathfrak{a}$ *is a stable extension then* $\mathfrak{a}2\mathfrak{ALab}(\mathfrak{a})$ *is a stable labelling.*

Proof. Let $\mathfrak{a}2\mathfrak{ALab}(\mathfrak{a}) = \mathfrak{ALab}$.

(1) Let $\mathfrak{a}$ be the grounded extension. By Theorem 94, it holds that $\mathfrak{ALab}$ is a complete labelling. We show that $\mathtt{in}(\mathfrak{ALab})$ is $\subseteq$-minimal among all complete arrow labellings. Towards a contradiction, assume there is a complete labelling $AbLab'$ such that $\mathtt{in}(\mathfrak{ALab}') \subsetneq \mathtt{in}(\mathfrak{ALab})$. By Theorem 94,

$\mathfrak{a}' = \texttt{in}(\mathfrak{ALab}')$ is a complete extension. By Definition 92, $\texttt{in}(\mathfrak{ALab}) = \mathfrak{a}$, hence there is a complete extension $\mathfrak{a}'$ such that $\mathfrak{a}' \subsetneq \mathfrak{a}$, contradiction to $\subseteq$-minimality of $\mathfrak{a}$. We obtain that $\mathfrak{ALab}$ is the grounded labelling.

(2) Let $\mathfrak{a}$ be a preferred extension. By Theorem 94, it holds that $\mathfrak{ALab}$ is a complete labelling. We show that $\texttt{in}(\mathfrak{ALab})$ is $\subseteq$-maximal among all complete arrow labellings. Towards a contradiction, assume there is a complete labelling *AbLab*$'$ such that $\texttt{in}(\mathfrak{ALab}) \subsetneq \texttt{in}(\mathfrak{ALab}')$. By Theorem 94, $\mathfrak{a}' = \texttt{in}(\mathfrak{ALab}')$ is a complete extension, contradiction to $\subseteq$-maximality of $\mathfrak{a}$. We obtain that $\mathfrak{ALab}$ is a preferred labelling.

(3) Let $\mathfrak{a}$ be a semi-stable extension. By Theorem 94, it holds that $\mathfrak{ALab}$ is a complete labelling. Moreover, by definition of semi-stable semantics, $\mathfrak{a} \cup \mathfrak{a}^+$ is $\subseteq$-maximal among all complete extensions. We show that $\texttt{undec}(\mathfrak{ALab})$ is $\subseteq$-minimal among all complete labellings.
Towards a contradiction, suppose there is a complete labelling $\mathfrak{ALab}'$ such that $\texttt{undec}(\mathfrak{ALab}') \subsetneq \texttt{undec}(\mathfrak{ALab})$. By Theorem 94, $\mathfrak{a}' = \texttt{in}(\mathfrak{ALab}')$ is a complete extension. By Definition 92, $\texttt{undec}(\mathfrak{ALab}') = \mathfrak{arr} \setminus (\mathfrak{a}' \cup (\mathfrak{a}')^+)$. Hence, $\mathfrak{arr} \setminus (\mathfrak{a}' \cup (\mathfrak{a}')^+) \subsetneq \mathfrak{arr} \setminus (\mathfrak{a} \cup \mathfrak{a}^+)$, therefore, $\mathfrak{a}' \cup (\mathfrak{a}')^+ \supsetneq \mathfrak{a} \cup \mathfrak{a}^+$, that is, we have found a complete extension $\mathfrak{a}'$ showing that $\mathfrak{a}$ is not semi-stable, contradiction to our initial assumption. We obtain that $\mathfrak{ALab}$ is a semi-stable labelling.

(4) Let $\mathfrak{a}$ be a stable extension. By Theorem 94, it holds that $\mathfrak{ALab}$ is a complete labelling. Moreover, by definition of stable semantics, $\mathfrak{a} \cup \mathfrak{a}^+ = \mathfrak{arr}$. Hence, by Definition 92, $\texttt{undec}(\mathfrak{ALab}) = \emptyset$. We obtain that $\mathfrak{ALab}$ is a stable labelling. $\square$

Theorem 96. *Let* $(\mathfrak{N}, \mathfrak{arr})$ *be a SETAF and let* $\mathfrak{ALab}$ *be an arrow labelling. Then*

(1) *if* $\mathfrak{ALab}$ *is a grounded labelling then* $\mathfrak{ALab2a}(\mathfrak{ALab})$ *is a grounded extension;*
(2) *if* $\mathfrak{ALab}$ *is a preferred labelling then* $\mathfrak{ALab2a}(\mathfrak{ALab})$ *is a preferred extension;*
(3) *if* $\mathfrak{ALab}$ *is a semi-stable labelling then* $\mathfrak{ALab2a}(\mathfrak{ALab})$ *is a semi-stable extension;*
(4) *if* $\mathfrak{ALab}$ *is a stable labelling then* $\mathfrak{ALab2a}(\mathfrak{ALab})$ *is a stable extension.*

Proof. Let $\mathfrak{ALab2a}(\mathfrak{ALab}) = \mathfrak{a}$.

(1) Let $\mathfrak{ALab}$ be the grounded labelling. By Theorem 94, it holds that $\mathfrak{a}$ is a complete extension. We show that $\mathfrak{a}$ is $\subseteq$-minimal among all complete extensions. Towards a contradiction, assume there is a complete extension $\mathfrak{a}'$ such that $\mathfrak{a}' \subsetneq \mathfrak{a}$. By Theorem 95, Theorem 94, $\mathfrak{a2ALab}(\mathfrak{a}')$ is a complete arrow labelling. By Definition 92, $\mathfrak{a} = \texttt{in}(\mathfrak{ALab})$ and $\mathfrak{a}' = \texttt{in}(\mathfrak{a2ALab}(\mathfrak{a}'))$. Therefore, $\texttt{in}(\mathfrak{a2ALab}(\mathfrak{a}')) \subsetneq \texttt{in}(\mathfrak{ALab})$, contradiction to the assumption that $\mathfrak{ALab}$ is the grounded labelling. We obtain that $\mathfrak{a}$ is grounded.

(2) Let $\mathfrak{ALab}$ be a preferred labelling. By Theorem 94, it holds that $\mathfrak{a}$ is a complete extension. We show that $\mathfrak{a}$ is $\subseteq$-maximal among all complete extensions. Towards a contradiction, assume there is a complete extension $\mathfrak{a}'$ such that $\mathfrak{a}' \supsetneq \mathfrak{a}$. By Theorem 95, Theorem 94, $\mathfrak{a2ALab}(\mathfrak{a}')$ is a complete arrow labelling. By Definition 92, $\mathfrak{a} = \texttt{in}(\mathfrak{ALab})$ and $\mathfrak{a}' = \texttt{in}(\mathfrak{a2ALab}(\mathfrak{a}'))$. Therefore, $\texttt{in}(\mathfrak{a2ALab}(\mathfrak{a}')) \supsetneq \texttt{in}(\mathfrak{ALab})$, contradiction to the assumption that $\mathfrak{ALab}$ is $\subseteq$-maximal. We obtain that $\mathfrak{a}$ is preferred.

(3) Let $\mathfrak{ALab}$ be a semi-stable labelling. By Theorem 94, it holds that $\mathfrak{ALab}$ is a complete extension. Moreover, by definition of semi-stable semantics, $\texttt{undec}(\mathfrak{ALab})$ is $\subseteq$-minimal among all complete labellings. We show that $\mathfrak{a} \cup \mathfrak{a}^+$ is $\subseteq$-maximal among all complete extensions.
Towards a contradiction, suppose there is a complete extension $\mathfrak{a}'$ such that $\mathfrak{a}' \cup (\mathfrak{a}')^+ \supsetneq \mathfrak{a} \cup \mathfrak{a}^+$. By Theorem 95, Theorem 94, $\mathfrak{ALab}' = \mathfrak{a2ALab}(\mathfrak{a}')$ is a complete labelling. By Definition 92,

$\texttt{undec}(\mathfrak{ALab}') = \mathfrak{arr} \setminus (\mathfrak{a}' \cup (\mathfrak{a}')^+)$. Hence, $\mathfrak{arr} \setminus (\mathfrak{a}' \cup (\mathfrak{a}')^+) \subsetneq \mathfrak{arr} \setminus (\mathfrak{a} \cup \mathfrak{a}^+)$, and therefore, $\texttt{undec}(\mathfrak{ALab}') \subsetneq \texttt{undec}(\mathfrak{ALab})$. That is, $\mathfrak{ALab}$ is not a semi-stable labelling, contradiction to our initial assumption. We conclude that $\mathfrak{a}$ is semi-stable.

(4) Let $\mathfrak{ALab}$ be a stable labelling. By Theorem 94, it holds that $\mathfrak{a}$ is a complete extension. By definition of stable labellings, $\texttt{undec}(\mathfrak{ALab}) = \emptyset$. Hence, each attack is either labelled $\in$ or labelled $\texttt{out}$. By Definition 93, $\texttt{in}(\mathfrak{ALab}) = \mathfrak{a}$. By Definition 17, if an attack $(\mathfrak{M}, A)$ is labelled $\texttt{out}$ then there is $(\mathfrak{M}', B) \in \mathfrak{arr}$ such that $\mathfrak{ALab}((\mathfrak{M}', B)) = \texttt{in}$. Hence, $\mathfrak{a}^+ = \mathfrak{arr} \setminus \mathfrak{a}$. Consequently, $\mathfrak{a} \cup \mathfrak{a}^+ = \mathfrak{arr}$. We obtain that $\mathfrak{a}$ is stable. $\square$

Lemma 97. *Let* $(\mathfrak{N}, \mathfrak{arr})$ *be a SETAF.*

(1) *For a complete arrow labelling* $\mathfrak{ALab}$ *it holds that* $\mathfrak{a2ALab}(\mathfrak{ALab2a}(\mathfrak{ALab})) = \mathfrak{ALab}$.
(2) *For a complete arrow extension* $\mathfrak{a}$ *it holds that* $\mathfrak{ALab2a}(\mathfrak{a2ALab}(\mathfrak{a})) = \mathfrak{a}$.

Proof. (1) Let $\mathfrak{ALab2a}(\mathfrak{ALab}) = \mathfrak{a}$. We prove the following three properties, for an arbitrary arrow $(\mathfrak{M}, A) \in \mathfrak{arr}$.

- If $AbLab((\mathfrak{M}, A)) = \texttt{in}$ then $\mathfrak{a2ALab}(\mathfrak{a})((\mathfrak{M}, A)) = \texttt{in}$.
  Suppose $AbLab((\mathfrak{M}, A)) = \texttt{in}$. By Definition 93, $(\mathfrak{M}, A) \in \mathfrak{a}$. By Definition 92, we obtain $\mathfrak{a2ALab}(\mathfrak{a})((\mathfrak{M}, A)) = \texttt{in}$.
- If $\mathfrak{ALab}((\mathfrak{M}, A)) = \texttt{out}$ then $\mathfrak{a2ALab}(\mathfrak{a})((\mathfrak{M}, A)) = \texttt{out}$.
  Suppose $\mathfrak{ALab}((\mathfrak{M}, A)) = \texttt{out}$. Then by Definition of a complete arrow labelling, it follows that there exists a $(\mathfrak{M}', B) \in \mathfrak{arr}$ with $B \in \mathfrak{M}$ such that $\mathfrak{ALab}((\mathfrak{M}', B)) = \texttt{in}$. By Definition 93 $(\mathfrak{M}', B) \in \mathfrak{a}$. By Definition 92, we obtain $\mathfrak{a2ALab}(\mathfrak{a})((\mathfrak{M}', B)) = \texttt{in}$. We proceed by case distinction.
  First assume $\mathfrak{a2ALab}(\mathfrak{a})((\mathfrak{M}, A)) = \texttt{in}$. Then $\mathfrak{a2ALab}(\mathfrak{a})((\mathfrak{M}', B)) = \texttt{out}$, by definition of a complete arrow labelling. Hence we obtain a contradiction.
  Next assume $\mathfrak{a2ALab}(\mathfrak{a})((\mathfrak{M}, A)) = \texttt{undec}$. By definition of a complete arrow labelling, there is some $(\mathfrak{M}', B) \in \mathfrak{arr}$ with $B \in \mathfrak{M}$ such that $\mathfrak{a2ALab}(\mathfrak{a})((\mathfrak{M}', B)) \neq \texttt{out}$, and there does not exist a $(\mathfrak{M}', B) \in \mathfrak{arr}$ with $B \in \mathfrak{M}$ such that $\mathfrak{a2ALab}(\mathfrak{a})((Mh', B)) = \texttt{in}$. The latter condition contradicts our assumption.
  We obtain $\mathfrak{a2ALab}(\mathfrak{a})((\mathfrak{M}, A)) = \texttt{out}$, as desired.
- If $\mathfrak{ALab}((\mathfrak{M}, A)) = \texttt{undec}$ then $\mathfrak{a2ALab}(\mathfrak{a})((\mathfrak{M}, A)) = \texttt{undec}$.
  Suppose $AbLab((\mathfrak{M}, A)) = \texttt{undec}$. By definition of a complete arrow labelling, there is some $(\mathfrak{M}', B) \in \mathfrak{arr}$ with $B \in \mathfrak{M}$ such that $\mathfrak{a2ALab}(\mathfrak{a})((\mathfrak{M}', B)) \neq \texttt{out}$, and there does not exist a $(\mathfrak{M}', B) \in \mathfrak{arr}$ with $B \in \mathfrak{M}$ such that $\mathfrak{a2ALab}(\mathfrak{a})((Mh', B)) = \texttt{in}$.
  To show that $\mathfrak{a2ALab}(\mathfrak{a})((\mathfrak{M}, A)) = \texttt{undec}$ we provide a proof by contradiction. Towards a contradiction, assume $\mathfrak{a2ALab}(\mathfrak{a})((\mathfrak{M}, A)) \neq \texttt{undec}$. Then either $\mathfrak{a2ALab}(\mathfrak{a})((\mathfrak{M}, A)) = \texttt{in}$ or $\mathfrak{a2ALab}(\mathfrak{a})((\mathfrak{M}, A)) = \texttt{out}$. We proceed by case distinction.
  First assume $\mathfrak{a2ALab}(\mathfrak{a})((\mathfrak{M}, A)) = \texttt{in}$. By Definition 92, $(\mathfrak{M}, A) \in \mathfrak{a}$. By Definition 93, we obtain $\mathfrak{ALab}((\mathfrak{M}, A)) = \texttt{in}$. This is a contradiction to $\mathfrak{ALab}((\mathfrak{M}, A)) = \texttt{undec}$.
  Next assume $\mathfrak{a2ALab}(\mathfrak{a})((\mathfrak{M}, A)) = \texttt{out}$. By Definition 92, $(\mathfrak{M}, A) \in \mathfrak{a}^+$. By definition of a complete arrow extension, there is some $(\mathfrak{M}', B) \in \mathfrak{arr}$ with $B \in \mathfrak{M}$ and $(\mathfrak{M}', B) \in \mathfrak{a}$. By Definition 93, $\mathfrak{ALab}((\mathfrak{M}', B)) = \texttt{in}$. This is a contradiction to the assumption that there does not exist a $(\mathfrak{M}', B) \in \mathfrak{arr}$ with $B \in \mathfrak{M}$ such that $\mathfrak{ALab}(\mathfrak{M}', B)) = \texttt{in}$.
  Hence we can conclude $\mathfrak{a2ALab}(\mathfrak{a})((\mathfrak{M}, A)) = \texttt{undec}$.

(2) Consider a complete arrow extension $\mathfrak{a}$. By Definition 92, $\texttt{in}(\mathfrak{a2ALab}(\mathfrak{a})) = \mathfrak{a}$. By Definition 93, $\mathfrak{ALab2a}(\mathfrak{a2ALab}(\mathfrak{a})) = \texttt{in}(\mathfrak{a2ALab}(\mathfrak{a}))$. Therefore, we obtain $\mathfrak{ALab2a}(\mathfrak{a2ALab}(\mathfrak{a})) = \mathfrak{a}$. □

Theorem 98. *When restricted to complete node labellings and complete arrow labellings, the functions* $\mathfrak{ALab2a}$ *and* $\mathfrak{a2ALab}$ *become bijections and each other's inverses.*

Proof. This follows directly from Lemma 97. □

Appendix K. Equivalence of Node Extensions and Arrow Extensions for SETAFs

In this section, we will provide proofs regarding the equivalence of node and arrow extensions. We recall the functions

$$\mathfrak{Args2a} = \mathfrak{ALab2a} \circ \mathfrak{NLab2ALab} \circ \mathfrak{Args2NLab}$$

and

$$\mathfrak{a2Args} = \mathfrak{Args2NLab} \circ \mathfrak{ALab2NLab} \circ \mathfrak{ALab2a}.$$

Theorem 99. *Let* $(\mathfrak{N}, \mathfrak{arr})$ *be a SETAF and let* $\mathfrak{M} \subseteq \mathfrak{N}$ *and* $\mathfrak{a} \subseteq \mathfrak{arr}$. *It holds that:*

(1) *If* $\mathfrak{M}$ *is complete node extension then then* $\mathfrak{Args2a}(\mathfrak{M})$ *is a complete arrow extension.*
If $\mathfrak{a}$ *is a complete arrow extension, then* $\mathfrak{a2Args}(\mathfrak{a})$ *is a complete node extension.*
(2) *When restricted to complete node labellings and complete arrow labellings, the functions* $\mathfrak{Args2a}$ *and* $\mathfrak{a2Args}$ *become bijections and each other's inverses.*
(3) *If* $\mathfrak{M}$ *is a grounded node extension, then* $\mathfrak{Args2a}(\mathfrak{M})$ *is a grounded arrow extension.*
If $\mathfrak{a}$ *is a grounded arrow extension, then* $\mathfrak{a2Args}(\mathfrak{a})$ *is a grounded node extension.*
(4) *If* $\mathfrak{M}$ *is a preferred node extension, then* $\mathfrak{Args2a}(\mathfrak{M})$ *is a preferred arrow extension.*
If $\mathfrak{a}$ *is a preferred arrow extension, then* $\mathfrak{a2Args}(\mathfrak{a})$ *is a preferred node extension.*
(5) *If* $\mathfrak{M}$ *is a stable node extension, then* $\mathfrak{Args2a}(\mathfrak{M})$ *is a stable arrow extension.*
If $\mathfrak{a}$ *is a stable arrow extension, then* $\mathfrak{a2Args}(\mathfrak{a})$ *is a stable node extension.*

Proof. (1) First, let $\mathfrak{M}$ is complete node extension. By [15, Theorem 5.10, Theorem 5.11] (and as summarised in Table 8), $\mathfrak{Args2NLab}(\mathfrak{M})$ is a complete node labelling. By Theorem 21, $\mathfrak{NLab2ALab}(\mathfrak{Args2NLab}(\mathfrak{M}))$ is a complete arrow labelling.
Finally, by Theorem 96, $\mathfrak{ALab2a}(\mathfrak{NLab2ALab}(\mathfrak{Args2NLab}(\mathfrak{M}))) = \mathfrak{Args2a}(\mathfrak{M})$ is a complete arrow extension.
Now, let $\mathfrak{a}$ a complete arrow extension. By Theorem 95, $\mathfrak{a2ALab}(\mathfrak{a})$ is a complete arrow labelling. By Theorem 21, $\mathfrak{ALab2NLab}(\mathfrak{a2ALab}(\mathfrak{a}))$ is a complete node labelling. Finally, by [15, Theorem 5.10, Theorem 5.11], $\mathfrak{NLab2Args}(\mathfrak{ALab2NLab}(\mathfrak{a2ALab}(\mathfrak{a}))) = \mathfrak{a2Args}(\mathfrak{a})$ is a complete node extension.
(2) By [15, Theorem 5.10, Theorem 5.11], Theorem 21, and Theorem 98.
(3) Analogous to point 1.
(4) Analogous to point 1.
(5) Analogous to point 1. □

We note that the above result does not apply to semi-stable semantics. As a counter-example, we refer to Example 91.

Appendix L. Connection Between Argumentation Frameworks and SETAFs

Recall Definition 22. As with AFs (see Definition 43), we can turn SETAF "inside out" as well. We want to emphasise that the resulting framework is still an AF, even if we turn a SETAF inside out. Moreover, note that the following result subsumes Theorem 44, as every AF can be seen as a SETAF.

The following theorem is a slightly reformulated version of Theorem 23 from Section 4, i.e., the following proof establishes Theorem 23.

Theorem 100. *Let $\mathfrak{SF} = (\mathfrak{N}, \mathfrak{arr})$ be a SETAF and $AF_{\mathfrak{SF}} = (N, arr)$ be its inside-out framework.*

(1) *If $\mathfrak{ALab}$ is a complete arrow labelling of $\mathfrak{SF}$, then $\mathfrak{ALab}$ is a complete node labelling of $AF_{\mathfrak{SF}}$.*
If NLab is a complete node labelling of $AF_{\mathfrak{SF}}$, then NLab is a complete arrow labelling of $\mathfrak{SF}$.

(2) *If $\mathfrak{ALab}$ is a preferred (resp. grounded or stable) arrow labelling of $\mathfrak{SF}$, then $\mathfrak{ALab}$ is a preferred (resp. grounded or stable) node labelling of $AF_{\mathfrak{SF}}$.*
If NLab is a preferred (resp. grounded, stable or semi-stable) node labelling of $AF_{\mathfrak{SF}}$, then NLab is a preferred (resp. grounded, stable or semi-stable) arrow labelling of $\mathfrak{SF}$.

Proof. (1) This follows directly from the definition of a complete node labelling (Definition 17, first three bullet points), the definition of a complete arrow labelling (Definition 20, first three bullet points) and the definition of an inside out argumentation framework (Definition 22).

(2) This follows directly from point 1, together with the definition of a preferred (resp. grounded, stable or semi-stable) node labelling (Definition 17) and the definition of a preferred (resp. grounded, stable or semi-stable) arrow labelling (Definition 20).
□

Because of Theorem 100 (point 3) the well-behavedness of arrow labellings carries over from argumentation frameworks to SETAF: as arrow labellings are essentially node labellings (of the inside out argumentation framework) they satisfy the standard properties of node labellings described in the literature. Hence, Theorem 61 follows from [33, Definition 5, Definition 6 and Theorem 1], Lemma 62 follows from [33, Lemma 1], Lemma 63 follows from [11, Lemma 2] and Theorem 64 follows from [33, Theorem 6, Theorem 7].

Appendix M. ABA semantics reformulated

The way arguments are defined in Definition 27 is slightly different from the original definition in [30]. This means that even though the definition of a set of assumptions attacking an assumption (Definition 30 is the same as the notion of attack in the ABA literature [30], our definition of attack refers to a different kind of argument and is therefore slightly different. We now show that nevertheless the two notions of attack coincide and that therefore the derived concepts of defence and semantics are equivalent no matter which notion of argument is used.

Since arguments in [30] are derivation trees as given in Definition 26, the equivalence results will be given in terms of arguments and derivations as given in Definitions 27 and 26, respectively.

Lemma 101. *Let $D = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^{-})$ be an ABAF.*

(1) *The notion of a set of assumptions Asms $\subseteq \mathcal{A}$ attacking an assumption $\alpha \in \mathcal{A}$ is equivalent when using arguments or derivations in Definition 30.*
(2) *The notion of a set of assumptions $Asms_1 \subseteq \mathcal{A}$ attacking a set of assumption $Asms_2 \subseteq \mathcal{A}$ is equivalent when using arguments or derivations in Definition 30.*
(3) *The set $Asms^+$ of a set of assumptions Asms $\subseteq \mathcal{A}$ is equivalent when using arguments or derivations in Definition 30.*
(4) *The notion of a set of assumptions Asms $\subseteq \mathcal{A}$ being conflict-free is equivalent when using arguments or derivations in Definition 30.*
(5) *The notion of a set of assumptions Asms $\subseteq \mathcal{A}$ defending an assumption $\alpha \in \mathcal{A}$ is equivalent when using arguments or derivations in Definition 30.*
(6) *The set $F(Asms)$ of a set of assumption Asms $\subseteq \mathcal{A}$ is equivalent when using arguments or derivations in Definition 30.*

Proof. We prove all items.

(1) There exists an ABA-argument $(Asms', \bar{\alpha})$ iff there exists at least one ABA-derivation for $\bar{\alpha}$ supported by $Asms'$. Thus, *Asms* attacks α no matter whether the notion of argument or derivation is used in the definition.
(2) $Asms_1 \subseteq \mathcal{A}$ attacks $Asms_2 \subseteq \mathcal{A}$ iff $Asms_1$ attacks some $\alpha \in Asms_2$. Thus, the claim follows by the first item.
(3) For a set of assumptions $Asms \subseteq \mathcal{A}$, $Asms^+ = \{\alpha \in \mathcal{A} \mid Asms \text{ attacks } \alpha\}$. Thus, the claim follows by the first item.
(4) By the third item a set of assumptions $Asms \subseteq \mathcal{A}$ is conflict-free no matter whether the notion of argument or derivation is used in the definition.
(5) $Asms \subseteq \mathcal{A}$ defends $\alpha \in \mathcal{A}$ iff each $Asms'$ attacking α is attacked by *Asms*. Thus, the claim follows by the first three items.
(6) For a set of assumptions $Asms \subseteq \mathcal{A}$, $F(Asms) = \{\alpha \in \mathcal{A} \mid \alpha \text{ is defended by } Asms\}$. Thus, the claim follows by the fourth item. $\square$

We now proceed to prove that the semantics in ABA are equivalent no matter whether the notion of ABA arguments or derivation trees is used in the definition.

Theorem 102. *Let $D = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^{-})$ be an ABAF.*

(1) *The notion of a set of assumptions Asms $\subseteq \mathcal{A}$ being an admissible assumption set is equivalent when using arguments or derivations in Definition 31.*
(2) *The notion of a set of assumptions Asms $\subseteq \mathcal{A}$ being a complete (resp. grounded, preferred, semi-stable, stable) assumption extension is equivalent when using arguments or derivations in Definition 31.*

Proof. We prove both statements

(1) admissible: By Lemma 101.
(2) • complete: By Lemma 101.
  • grounded: By the equivalence of complete assumption extensions.
  • preferred: By the equivalence of complete assumption extensions.

- semi-stable: By the equivalence of complete assumption extensions and because by Lemma 101 $Asms^+$ is equivalent independent of the underlying notion.
- stable: Same as semi-stable. ☐

As mentioned in Section 5, the way preferred and stable semantics in the context of ABAFs were defined in Definition 31 is slightly different from the way these were originally defined in [12, 30]. We have chosen to describe all ABA semantics in a uniform way, based on the notion of complete semantics. This has been done to allow for easy conversion between extensions and labellings, as well as to provide uniformity with the rest of the paper.

We will now proceed to show that our description of the ABA semantics in Definition 31 is equivalent to the original description in [12, 30]. Since the notion of admissible sets and complete extensions are simply reformulations of the definitions in [12, 30] in terms of the function F introduced here, we start with proving equivalence for the preferred semantics.

Theorem 103. *Let $D = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^-)$ be an ABAF and $Asms \subsetneq \mathcal{A}$. The following two statements are equivalent:*

(1) *The set Asms is a maximal admissible assumption set of D*
(2) *The set Asms is a maximal complete assumption extension of D*

Proof. (From 1 to 2) Let *Asms* be a maximal admissible assumption set. From [12, Corollary 5.8] it follows that *Asms* is a complete assumption extension. Suppose *Asms* is not *maximal* complete. Then there exists a complete assumption extension $Asms'$ with $Asms \subsetneq Asms'$. However, since by definition, every complete assumption extension is also an admissible assumption set, it holds that $Asms'$ is an admissible assumption set. But this would mean that *Asms* is not a *maximal* admissible assumption set; contradiction.

(From 2 to 1) Let *Asms* be a maximal complete assumption extension. Then by definition, *Asms* is also an admissible assumption set. We now need to prove that it is also a *maximal* admissible assumption set. Suppose this is not the case, then there exists a maximal admissible assumption set $Asms'$ with $Asms \subsetneq Asms'$. It follows from [12, Corollary 5.8] that $Asms'$ is also a complete assumption extension. But this would mean that *Asms* is not a *maximal* complete assumption extension; contradiction. ☐

The next thing to show is that our description of stable semantics (Definition 31) is equivalent with the way stable semantics was originally defined in [12].

Theorem 104. *Let $D = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^-)$ be an ABAF and $Asms \subsetneq \mathcal{A}$. The following two statements are equivalent:*

(1) *Asms is a conflict-free assumption set that attacks every assumption in $\mathcal{A} \setminus Asms$*
(2) *Asms is a complete assumption extension with $Asms \cup Asms^+ = \mathcal{A}$*

Proof. (From 1 to 2) Let *Asms* is a conflict-free assumption set attacking every assumption in $\mathcal{A} \setminus Asms$. Thus, if $\alpha \notin Asms$, then $\alpha \in Asms^+$. Since *Asms* is conflict-free, if $\alpha \in Asms$, then $\alpha \notin Asms^+$. It follows that if $\alpha \in \mathcal{A}$, then $\alpha \in Asms$ or $\alpha \in Asms^+$, i.e. $\mathcal{A} = Asms \cup Asms^+$. Clearly, if $Asms'$ attacks *Asms*, then *Asms* counter-attacks $Asms'$, so *Asms* is admissible. Since no superset of *Asms* is conflict-free, it even be complete.

(From 2 to 1) Let *Asms* is a complete assumption extension with $Asms \cup Asms^+ = \mathcal{A}$. Thus, if $\alpha \in \mathcal{A}$, then $\alpha \in Asms$ or $\alpha \in Asms^+$. It follows that for every assumption $\alpha \in \mathcal{A} \setminus Asms$, it holds that $\alpha \in Asms^+$, so α is attacked by *Asms*. Since *Asms* is complete, it is conflict-free. □

Using the lemmas and theorems provided in this section, we will now prove Theorem 32 from Section 5.

Theorem 32. *Let $D = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^-)$ be an ABAF.*

(1) *The set $Asms \subseteq \mathcal{A}$ is an admissible assumption set according to Definitions 26-31 iff Asms is an admissible assumption set according to the definitions in [12, 30].*
(2) *The set $Asms \subseteq \mathcal{A}$ is a complete (resp. grounded, preferred, semi-stable or stable) assumption extension according to Definitions 26-31 iff Asms is a complete (resp. grounded, preferred, semi-stable or stable) assumption extension according to the definitions in [12, 29, 30]*

Proof. We prove both items.

(1) By Theorem 102 and because the notion of admissible assumption set used here is a simple reformulation of the one in [12, 30]: In [30], a set of assumption *Asms* is defined as an admissible assumption set iff it does not attack itself and it attacks every set of assumptions $Asms'$ that attacks *Asms*. However, by definition, *Asms* attacks every set of assumptions $Asms'$ that attacks *Asms* iff $Asms \subseteq F(Asms)$.
(2) By Theorems 102, 103, and 104 and because by the same reasoning as for the first item, the notion of complete assumption extension is a reformulation of the definitions in [12, 30] in terms of the function F. □

In the following, we investigate how the slightly altered notion of ABA-arguments (Definition 27 influences the associated AF as compared to [30]. Since arguments in [30] are equivalent to derivation trees in Definition 26, we will prove the results in terms of derivation trees. However, exactly the same results hold with respect to arguments as defined in [30].

Definition 105. *Given an ABAF $D = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^-)$, we say that a derivation tree for c_1 supported by* $Asms_1$ derivation-attacks *a derivation tree for c_2 supported by $Asms_2$ iff $c_1 = \bar{\gamma}$ for some $\gamma \in Asms_2$.*

Note that Definition 105 is equivalent to the notion of attack in [30].

Definition 106. *Given an ABAF $D = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^-)$, the associated derivation AF AF_{derD} is defined as (N_{der}, arr_{der}) with N_{der} being the set of derivation trees, and arr_{der} being the derivation-attack relation among derivation trees.*

Definition 106 is equivalent to the associated AF as defined in [30].

Lemma 107. *Let $D = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^-)$ be an ABAF, AF_D the associated AF, and AF_{derD} the associated derivation AF. The following are equivalent:*

(1) *There exists an ABA-argument $(Asms, c)$ in AF_D.*
(2) *There exists a derivation tree for c supported by Asms in AF_{derD}.*

Proof. It follows directly from Definitions 26 and 27 that an ABA-argument $(Asms, c)$ represents the set of all derivation trees supported by the set $Asms$ of assumptions and having conclusion c. □

Lemma 108. *Let $D = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^{-})$ be an ABAF, $AF_D = (N, arr)$ the associated AF, and AF_{derD} the associated derivation AF. The following are equivalent:*

(1) *The ABA-argument $(Asms_1, c_1)$ attacks the ABA-argument $(Asms_2, c_2)$ in AF_D.*
(2) *A derivation tree for c_1 supported by $Asms_1$ derivation-attacks a derivation tree for c_2 supported by $Asms_2$ in AF_{derD}.*
(3) *All ABA-derivations for c_1 supported by $Asms_1$ derivation-attack all derivation trees for c_2 supported by $Asms_2$ in AF_{derD}.*

Proof. (1 to 2) Assume that the ABA-argument $(Asms_1, c_1)$ attacks the ABA-argument $(Asms_2, c_2)$ in AF_D. By Lemma 107 there exists a derivation tree for c_1 supported by $Asms_1$ and a derivation tree for c_2 supported by $Asms_2$ in AF_{derD}. Since $(Asms_1, c_1)$ attacks $(Asms_2, c_2)$, by Definition 27 $c_1 = \bar{\gamma}$ for some $\gamma \in Asms_2$. Thus, by Definition 105 the derivation tree for c_1 supported by $Asms_1$ derivation-attacks the derivation tree for c_2 supported by $Asms_2$.

(2 to 1) Analogously.

(2 to 3) Assume that a derivation tree for c_1 supported by $Asms_1$ derivation-attacks a derivation tree for c_2 supported by $Asms_2$. By Definition 105, $c_1 = \bar{\gamma}$ for some $\gamma \in Asms_2$. Since this holds for all derivation trees for c_1 supported by $Asms_1$ and all derivation tree for c_2 supported by $Asms_2$, by Definition 105 all derivation trees for c_1 supported by $Asms_1$ derivation-attack all derivation trees for c_2 supported by $Asms_2$.

(3 to 2) Analogously. □

Theorem 109. *Let $D = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^{-})$ be an ABAF, $AF_D = (N, arr)$ the associated AF, and $AF_{derD} = (N_{der}, arr_{der})$ the associated derivation AF. Let $M \subseteq N$ be a set of ABA-arguments and let $Mder \subseteq N_{der}$ be a set of derivations such that all derivations for c supported by $Asms$ are in $Mder$ iff $(Asms, c) \in M$. Then M is an admissible set of AF_D iff $Mder$ is an admissible set of AF_{derD}.*

Proof. (left to right) Assume M is an admissible set of AF_D and assume that $Mder$ is not an admissible set of AF_{derD}. Then either $Mder$ is not conflict-free or $F(Mder) \not\subseteq Mder$.

Assume first that $Mder$ is not conflict-free, i.e. there exists some ABA-derivation for c_2 supported by $Asms_2$ contained in $Mder$ and contained in $Mder^+$. This means that there exists some derivation tree for c_1 supported by $Asms_1$ in $Mder$ which derivation-attacks the derivation tree for c_2 supported by $Asms_2$ in $Mder$. By the definition of M and by Lemma 107, there exist ABA-arguments $(Asms_1, c_1)$ and $(Asms_2, c_2)$ in M, and by Lemma 108 $(Asms_1, c_1)$ attacks $(Asms_2, c_2)$. Thus, M is not conflict free; contradiction.

Assume now that $F(Mder) \not\subseteq Mder$, i.e. there exists some derivation tree for c_3 from $Asms_3$ in $Mder$ which is not defended by $Mder$. This means that there exists a derivation tree for c_4 from $Asms_4$ derivation-attacking the derivation tree for c_3 from $Asms_3$ and the derivation tree for c_4 from $Asms_4$ is not derivation-attacked by any derivation tree in $Mder$. By the definition of M and by Lemma 107 there exists an ABA-argument $(Asms_3, c_3)$ in M and there exists an ABA-argument $(Asms_4, c_4)$ which by Lemma 108 attacks $(Asms_3, c_3)$ and $(Asms_4, c_4)$ is not attacked by any argument in $Mder$. Thus, $(Asms_3, c_3)$ is not defended by M and therefore $(Asms_3, c_3) \notin F(M)$; contradiction.

(right to left) Assume that $Mder$ is an admissible set of AF_{derD} and that M is not an admissible set of AF_D. Then either M is not conflict-free or $F(M) \not\subseteq M$. Assume first that M is not conflict-free, i.e. there exists some ABA-argument $(Asms_1, c_1)$ contained in M and contained in M^+. Consequently, $(Asms_1, c_1)$ is attacked by some ABA-argument $(Asms_2, c_2) \in M$. By Lemma 108, all derivation trees for c_1 supported by $Asms_1$ are derivation-attacked by all derivation trees for c_2 supported by $Asms_2$. By definition of $Mder$, all derivation trees for c_1 supported by $Asms_1$ and all derivation trees for c_2 supported by $Asms_2$ are contained in $Mder$. Thus, $Mder \cap Mder^+ \neq \emptyset$ which contradicts the assumption that $Mder$ is an admissible set.

Assume now that $F(M) \not\subseteq M$, i.e. there exists some ABA-argument for $(Asms_3, c_3) \in M$ which is not defended by M. This means that there exists an ABA-argument $(Asms_4, c_4)$ attacking $(Asms_3, c_3)$ which is not attacked by any argument in M. By Lemma 108, all derivation trees for c_4 from $Asms_4$ derivation-attack all derivation trees for c_3 from $Asms_3$. Furthermore, by the definition of $Mder$ all derivation trees for c_4 from $Asms_4$ are not derivation-attacked by any derivation tree in $Mder$ and all derivation trees for c_3 from $Asms_3$ are contained in $Mder$. This means that all derivation trees for c_3 from $Asms_3$ are not defended by $Mder$; contradiction. $\square$

Theorem 110. *Let $D = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^-)$ be an ABAF, $AF_D = (N, arr)$ the associated AF, and $AF_{derD} = (N_{der}, arr_{der})$ the associated derivation AF. Let $M \subseteq N$ be a set of ABA-arguments and let $Mder \subseteq N_{der}$ be a set of derivations such that all derivations for c supported by $Asms$ are in $Mder$ iff $(Asms, c) \in M$. Then M is a complete (resp. grounded, preferred, semi-stable, stable) extension of AF_D iff $Mder$ is a complete (resp. grounded, preferred, stable) extension of AF_{derD}.*

Proof. We prove the statement for all listed semantics.

- complete: Analogue to the proof of Theorem 109.
- grounded: Follows from complete.
- preferred: Follows from complete.
- semi-stable:
  (left to right) Assume M is a semi-stable extension of AF_D and assume that $Mder$ is not a semi-stable extension of AF_{derD}, i.e. there exists some $Mder'$ such that

$$Mder \cup Mder^+ \subsetneq Mder' \cup Mder'^+.$$

  Thus, there exists some derivation tree for some c supported by some $Asms$ in $Mder'$ or in $Mder'^+$ which is not contained in $Mder$ and is not contained in $Mder^+$. By the definition of $Mder$, the argument $(Asms, c)$ it not contained in M. By Lemma 108 since the derivation tree for c supported by $Asms$ is not in $Mder^+$, there is no ABA-argument in M attacking $(Asms, c)$ and therefore $(Asms, c) \notin M^+$. Since $M \cup M^+$ is maximal, there exists no M' such that $(Asms, c)$ is contained in M' or M'^+. Consequently by the definition of $Mder$ and by Lemma 107, there exists no $Mder'$ such that $Mder'$ contains an derivation tree for c supported by $Asms$. Furthermore, by Lemma 108, no derivation tree for c supported by $Asms$ is contained in $Mder'^+$; contradiction.
  (right to left) Assume $Mder$ is a semi-stable extension of AF_{derD} and assume that M is not a semi-stable extension of AF_D, i.e. there exists some M' such that

$$M \cup M^+ \subsetneq M' \cup M'^+.$$

Thus, there exists some ABA-argument $(Asms, c)$ in M' or in M'^{+} which is not contained in M and is not contained in M^{+}. By the definition of $Mder$, no derivation tree for c supported by $Asms$ is contained in $Mder$. However, by Lemma 107 there exists at least one derivation tree for c supported by $Asms$ in N_{der}. By Lemma 108 since $(Asms, c) \notin M^{+}$ there is no derivation tree in $Mder$ which derivation-attacks the derivation tree for c supported by $Asms$ and therefore the derivation tree for c supported by $Asms$ is not in $Mder^{+}$. Since $Mder \cup Mder^{+}$ is maximal, there exists no $Mder'$ such that the derivation tree for c supported by $Asms$ is contained in $Mder'$ or $Mder'^{+}$. Consequently by the definition of $Mder$ and by Lemma 107, there exists no M' such that M' contains $(Asms, c)$. Furthermore, by Lemma 108, $(Asms, c)$ is not contained in M'^{+}; contradiction.

- stable:
  (left to right) Assume M is a stable extension of AF_D and assume that $Mder$ is not a stable extension of AF_{derD}, i.e. $Mder \cup Mder^{+} \neq N_{der}$. This means there exists some derivation tree for some conclusion c supported by some $Asms$ in N_{der} which is not contained in $Mder$ and not contained in $Mder^{+}$. By definition of $Mder$, the ABA-argument $(Asms, c)$ is not contained in M. Since M is a stable extension, $(Asms, c) \in M^{+}$. This means that $(Asms, c)$ is attacked by an ABA-argument $(Asms_1, c_1) \in M$. By Lemma 107 there is at least one derivation tree for c_1 supported by $(Asms_1)$ in N_{der}. By definition of $Mder$, all derivation trees for c_1 supported by $(Asms_1)$ are contained in $Mder$ and by Lemma 108 they all attack the derivation of c supported by $Asms$. Thus, the derivation of c supported by $Asms$ is contained in $Mder^{+}$; contradiction.
  (right to left) Assume $Mder$ is a stable extension of AF_{derD} and assume that M is not a stable extension of AF_D, i.e. $M \cup M^{+} \neq N_{der}$. This means that there exists some ABA-argument $(Asms, c) \in N$ which is not contained in M and not contained in M^{+}. By definition of $Mder$, no derivation tree for c supported by $Asms$ is contained in $Mder$, but by Lemma 107 there exists at least one derivation for c supported by $Asms$ in AF_{derD}. Since $Mder$ is a stable extension, all derivation trees for c supported by $Asms$ are contained in $Mder^{+}$. This means that there exist some derivation trees for some c_1 supported by some $Asms_1$ which attack all derivation trees for c supported by $Asms$. By definition of $Mder$, the argument $(Asms_1, c_1$ is contained in $Mder$ and by Lemma 108 $(Asms_1, c_1)$ attacks $(Asms, c)$. Thus, $(Asms, c) \in M^{+}$; contradiction. □

Theorem 35. *Let $D = (\mathcal{L}, \mathcal{R}, \mathcal{A}, ^{-})$ be an ABAF, $\mathfrak{SF}_D$ be the associated SETAF, and $Asms \subseteq \mathcal{A}$. It holds that*

(1) *Asms is a complete extension of $\mathfrak{SF}_D$ iff Asms is a complete extension of D in the sense of [12, 30];*
(2) *Asms is a grounded extension of $\mathfrak{SF}_D$ iff Asms is a grounded extension of D in the sense of [12, 30];*
(3) *Asms is a preferred extension of $\mathfrak{SF}_D$ iff Asms is a preferred extension of D in the sense of [12, 30];*
(4) *Asms is a semi-stable extension of $\mathfrak{SF}_D$ iff Asms is a semi-stable extension of D in the sense of [35];*
(5) *Asms is a stable extension of $\mathfrak{SF}_D$ iff Asms is a stable extension of D in the sense of [12, 30].*

Proof. This follows directly from definitions 30, 31, 15, 16 and 33. □

Proposition 36. *Let $AF_1 = (N_1, arr_1)$ and $AF_2 = (N_2, arr_2)$ be two AFs such that conditions (1) and (2) are met.*

(i) If $NLab_1$ is a complete (resp. grounded, preferred or stable) node labelling of AF_1, then

$$NLab_2 = NLab_1 \cup \{(A, \texttt{in}) \mid A \in N_2 \setminus N_1,\ \forall (B, A) \in arr_2 : NLab_1(B) = \texttt{out}\}$$

$$\cup \{(A, \texttt{out}) \mid A \in N_2 \setminus N_1,\ \exists (B,A) \in arr_2 : NLab_1(B) = \texttt{in}\}$$
$$\cup \{(A, \texttt{undec}) \mid A \in N_2 \setminus N_1,\ \neg\forall (B,A) \in arr_2 : NLab_1(B) = \texttt{out},$$
$$\neg\exists (B,A) \in arr_2 : NLab_1(B) = \texttt{in}\}$$

is a complete (resp. grounded, preferred or stable) node labelling of AF_2.

(ii) If $NLab_2$ is a complete (resp. grounded, preferred or stable) node labelling of AF_2, then $NLab_1$ is a complete (resp. grounded, preferred or stable) node labelling of AF_1.

Proof. Let $AF_1 = (N_1, arr_1)$ and $AF_2 = (N_2, arr_2)$ be two AFs such that conditions (1) and (2) are met. For complete, grounded, and preferred semantics, the result is an immediate consequence of the directionality principle. It remains to prove the proposition for stable semantics.

(i) Let $NLab_1$ be stable, i.e., $NLab_1$ is a complete labelling such that $\texttt{undec}(NLab_1) = \emptyset$. By directionality, it holds that $NLab_2$ is a complete node labelling of AF_2. To show that $NLab_2$ is a stable node labelling it suffices to prove that $\texttt{undec}(NLab_2) = \emptyset$.
Let $A \in N_2$. In case $A \in N_1$, we have either $NLab_2(A) = \texttt{in}$ or $NLab_2(A) = \texttt{out}$ since $NLab_1$ is stable. Therefore, $NLab_2(A) \neq \texttt{undec}$ in this case.
Now, let $A \in N_2 \setminus N_1$. By condition (2), A receives no incoming attacks from nodes in $N_2 \setminus N_1$. Therefore, all attackers of A are either labelled $\texttt{in}$ or $\texttt{out}$. We proceed by case distinction: (a) all incoming attackers of A are labelled $\texttt{out}$. Then A is labelled $\texttt{in}$, by definition of $NLab_2$. (b) there is an incoming attacker of A that is labelled $\texttt{in}$. Then A is labelled $\texttt{out}$, by definition of $NLab_2$. Since no node in N_1 is labelled $\texttt{undec}$, the case distinction is exhaustive. This proves that $NLab_2$ is a stable node labelling of AF_2.

(ii) Let $NLab_2$ be a stable node labelling of AF_2. It holds that $NLab_1$ is a complete node labelling of AF_1 (by directionality); moreover, $\texttt{undec}(NLab_1) = \emptyset$. Therefore, we obtain that $NLab_1$ is a stable node labelling of AF_1. □

Proposition 37. *Let D be an ABAF. Let $\mathfrak{S}\mathfrak{F}_D$ be the associated SETAF and let $AF_{\mathfrak{S}\mathfrak{F}_D}$ be its inside-out AF. Let AF_D be the AF associated with D. Then the relation between $AF_1 := AF_{\mathfrak{S}\mathfrak{F}_D}$ and $AF_2 := AF_D$ is as described in (1) and (2) (up to argument names).*

Proof. Let $AF_D = AF_2 = (N_2, arr_2)$ be the AF associated with D. Suppose A is an node that has some out-going arrow. Then, by definition, A is some ABA-argument of the form $A = (Asms, \overline{\gamma})$ where γ is some assumption in D. Again by definition the SETAF $\mathfrak{S}\mathfrak{F}_D$ contains some arrow $(Asms, \gamma)$ and consequently the inside-out AF $AF_{\mathfrak{S}\mathfrak{F}_D} = AF_1 = (N_1, arr_1)$ contains an argument $B = (Asms, \gamma)$ corresponding to this arrow. Thus we can assign to any node A in AF_D with an out-going arrow a corresponding node B in $AF_{\mathfrak{S}\mathfrak{F}_D}$. In the same vein, the arrow relation is preserved: If there is an arrow from $A_1 = (Asms_1, \overline{\gamma_1})$ to $A_2 = (Asms_2, \overline{\gamma_2})$ in AF_D, then there is an arrow from $B_1 = (Asms_1, \gamma_1)$ to $B_2 = (Asms_2, \gamma_2)$ in $AF_{\mathfrak{S}\mathfrak{F}_D}$.

Vice versa, if $B = (Asms, \gamma)$ is a node in the inside-out AF $AF_{\mathfrak{S}\mathfrak{F}_D}$, then we can similarly argue that there is some ABA-argument of the form $A = (Asms, \overline{\gamma})$ in AF_D. This node must have at least one out-going arrow in AF_D since it entails the contrary of γ. The arrow relation is preserved analogously. □