Argumentation Label Functions - Technical Report

Marcos Cramer [a] and Jérémie Dauphin [b]
[a] *TU Dresden*
[b] *University of Luxembourg*

Abstract. An important approach to abstract argumentation is the labeling-based approach, in which one makes use of labelings that assign to each argument one of three labels: `in`, `out` or `und`. In this paper, we address the question, which of the twenty-seven functions from the set of labels to the set of labels can be represented by an argumentation framework. We prove that in preferred, complete and grounded semantics, eleven labeling functions can be represented in this way while sixteen labeling functions cannot be represented by any argumentation framework. We show how this analysis of labeling functions can be applied to prove an impossibility result: Argumentation frameworks extended with a certain kind of weak attack relation cannot be flattened to the standard Dung argumentation frameworks.

Keywords. knowledge representation, abstract argumentation, argumentation semantics, labelings, flattening

1. Introduction

Abstract argumentation frameworks (AFs) [13] are reasoning structures where one aims at extracting sets of jointly acceptable arguments. One of the central methods to do so is the labeling-based approach [2], in which one derives labeling functions which assign to each argument one of three labels: `in`, `out` or `und`. The arguments that are labeled `in` represent the arguments that are jointly acceptable, while the arguments that are `out` represent the ones that are defeated by those. The last label, `und` (*undecided*), represents the cases where one cannot, or decides with proper justification, not to assign either of these two labels. One advantage of the labeling approach is that to verify that an argument is correctly labeled, one only needs to check the labels of its direct ancestors. This allows for a more local evaluation, which is still equivalent to other global approaches such as the extension-based approach.

Many enrichments of abstract argumentation frameworks have been studied, e.g. with bipolar argumentation frameworks which add a second relation of support [9], or with argumentation frameworks with recursive attacks (AFRA) [3] in which attacks may also target other attacks. One methodology for evaluating such enriched frameworks while staying coherent with the basic framework is the flattening approach [6], where the enrichments added to the abstract argumentation frameworks are expressed in terms of extra arguments and attacks, allowing one to evaluate them as abstract argumentation frameworks. An essential concern in the flattening approach is whether the extra

arguments and attacks produce the same behavior as the one intended by the enrichment they flatten. This raises a question: Which relations connecting two arguments can be expressed in terms of arguments and attacks alone?

In this paper we propose to address this research question by studying the representability of label functions, i.e. of functions which map each of the three labels to one of these labels. We prove that in preferred, complete and grounded semantics, eleven labeling functions can be represented by an AF while sixteen labeling functions cannot be represented by any AF. We show how this analysis of labeling functions can be applied to prove an impossibility result: Argumentation frameworks extended with a certain kind of weak attack relation cannot be flattened to the standard Dung argumentation frameworks. Furthermore we also briefly discuss representability of label functions with respect to the stable and semi-stable semantics.

The structure of the paper is as follows: in Section 2, we provide an overview of existing definitions from the literature that we make use of later on. In Section 3 we formally define the notion of labeling function and what it means to represent them as abstract argumentation frameworks. In Section 4 we show which of the twenty-seven labeling functions are representable and which ones are unrepresentable in the context of the complete, grounded and preferred semantics, and briefly mention the case of the stable semantics. In Section 5 we discuss the implications of these impossibility results for the flattening of a particular relation: a weak attack relation that does not propagate the undecided label. We then discuss related work in Section 6 and future work in Section 7, where we also briefly discuss the case of the semi-stable semantics. We provide a short conclusion in Section 8.

Due to space limitations, the proofs of some of the results of this paper are presented in a technical report [11].

2. Preliminaries

In this section we introduce the basic notions of abstract argumentation. We define argumentation semantics using the labeling-based approach [2].

We start by defining the fundamental notion of *argumentation frameworks*.

Definition 1. *An* argumentation framework (AF) $F = (\mathscr{A}, \mathscr{R})$ *is a (finite or infinite) directed graph in which the set $\mathscr{A}$ of vertices is considered to represent arguments and the set $\mathscr{R} \subseteq \mathscr{A} \times \mathscr{A}$ of edges is considered to represent the attack relation between arguments, i.e. the relation between a counterargument and the argument that it counters.*

Given an argumentation framework, we want to make a coherent judgment about which arguments we accept, which arguments we reject and for which arguments we refrain from accepting or rejecting them. For this purpose three labels are used: `in`, `out` and `und`. The set $\{\texttt{in}, \texttt{out}, \texttt{und}\}$ of these three labelings is denoted *Labs*.

Definition 2. *Let $F = (\mathscr{A}, \mathscr{R})$ be an AF. A* labeling *of F is a function $\texttt{Lab} : \mathscr{A} \to Labs$.*

Intuitively a labeling is a judgment about all arguments in an argumentation framework. So far we have not imposed any constraints on the coherence of these judgments. These will be specified later.

Sometimes there may be a choice between multiple coherent points of view. So what we want in the general case is a mapping from argumentation frameworks to sets of labelings. Such a mapping is called an *argumentation semantics*:

Definition 3. *An* argumentation semantics *is a function σ that maps any AF $F = (\mathscr{A}, \mathscr{R})$ to a set $\sigma(F)$ of labelings of F. The elements of $\sigma(F)$ are called σ-labelings of F.*

Note 1. *We usually define an argumentation semantics σ by specifying criteria which a labeling of F has to satisfy in order to be a σ-labeling of F.*

Multiple *argumentation semantics* have been defined in the literature. In this paper we consider the complete, stable, grounded, preferred, and semi-stable semantics:

Definition 4. *Let $F = (\mathscr{A}, \mathscr{R})$ be an AF, and let* Lab *be a labeling of F.*

- *An argument $a \in \mathscr{A}$ is called* legally `in` w.r.t. Lab *iff every argument that attacks a is labeled* `out` *by* Lab *(in other words: for every $b \in \mathscr{A}$ such that $(b,a) \in \mathscr{R}$, we have* $\mathtt{Lab}(b) = \mathtt{out}$*).*
- *An argument $a \in \mathscr{A}$ is called* legally `out` w.r.t. Lab *iff some argument that attacks a is labeled* `in` *by* Lab*.*
- *An argument $a \in \mathscr{A}$ is called* legally `und` w.r.t. Lab *iff no argument that attacks a is labeled* `in` *by* Lab *and some argument that attacks a is labeled* `und` *by* Lab*.*

Definition 5. *Let $F = (\mathscr{A}, \mathscr{R})$ be an AF, and let* Lab *be a labeling of F.*

- Lab *is a* complete labeling *of F iff every arguments that* Lab *labels* `in` *is legally* `in` *w.r.t.* Lab*, every arguments that* Lab *labels* `out` *is legally* `out` *w.r.t.* Lab*, and every arguments that* Lab *labels* `und` *is legally* `und` *w.r.t.* Lab*.*
- Lab *is a* stable labeling *of F iff* Lab *is a complete labeling of F and no argument is labeled* `und` *by* Lab*.*
- Lab *is the* grounded labeling *of F iff* Lab *is the complete labeling of F in which the set of* `in`*-labeled arguments is minimal w.r.t. set inclusion.*
- Lab *is a* preferred labeling *of F iff* Lab *is a complete labeling of F in which the set of* `in`*-labeled arguments is maximal w.r.t. set inclusion.*
- Lab *is a* semi-stable labeling *of F iff* Lab *is a complete labeling of F in which the set of* `und`*-labeled arguments is minimal w.r.t. set inclusion.*

We now present a well-known characterization of the grounded labeling as the least fixpoint of a certain operator $\mathscr{F}_F$ (see [2]):

Definition 6. *Let $F = (\mathscr{A}, \mathscr{R})$ be an AF and let* Lab *be a labeling of F. Then $\mathscr{F}_F(\mathtt{Lab})$ is a labeling of F defined as follows:*

$$\mathscr{F}_F(\mathtt{Lab})(a) = \begin{cases} \mathtt{in} & \text{if every attacker of } a \text{ is labeled } \mathtt{out} \text{by } \mathtt{Lab} \\ \mathtt{out} & \text{if some attacker of } a \text{ is labeled } \mathtt{in} \text{by } \mathtt{Lab} \\ \mathtt{und} & \text{otherwise} \end{cases}$$

Definition 7. *Let $F_1 = (\mathscr{A}_1, \mathscr{R}_1)$ and $F_2 = (\mathscr{A}_2, \mathscr{R}_2)$ be two AFs with $\mathscr{A}_1 \subseteq \mathscr{A}_2$, and let $\mathtt{Lab}_1$ and $\mathtt{Lab}_2$ be labelings of F_1 and F_2 respectively. We say $\mathtt{Lab}_1 \leq \mathtt{Lab}_2$ iff for every argument $a \in \mathscr{A}$, $\mathtt{Lab}_1(a) = \mathtt{in}$ implies $\mathtt{Lab}_2(a) = \mathtt{in}$, and $\mathtt{Lab}_1(a) = \mathtt{out}$ implies $\mathtt{Lab}_2(a) = \mathtt{out}$.*

Proposition 1. *Let $F = (\mathscr{A}, \mathscr{R})$ be an AF. Then the unique grounded labeling of F is the $\leq$-least fixpoint of $\mathscr{F}_F$.*

This least fixpoint of $\mathscr{F}_F$ can be reached by iteratively applying $\mathscr{F}_F$ to the $\leq$-least labeling (i.e. to the labeling that maps every argument to $\mathtt{und}$). In case $\mathscr{A}$ is an infinite set, this might require transfinitely many iterations of $\mathscr{F}_F$, which are defined as follows:

Definition 8. *Let $\mathtt{Lab}_{\mathtt{und}}$ be the labeling that maps every argument to $\mathtt{und}$. For every ordinal α, we define $\mathscr{F}_F^{\alpha}(\mathtt{Lab}_{\mathtt{und}})$ as follows:*

$$\mathscr{F}_F^{\alpha}(\mathtt{Lab}_{\mathtt{und}}) = \begin{cases} \mathtt{Lab}_{\mathtt{und}} & \text{if } \alpha = 0 \\ \mathscr{F}_F(\mathscr{F}_F^{\beta}(\mathtt{Lab}_{\mathtt{und}})) & \text{if } \alpha = \beta + 1 \\ \text{the } \leq\text{-least labeling } \mathtt{Lab} \text{ of } F \text{ such that} & \\ \mathscr{F}_F^{\beta}(\mathtt{Lab}_{\mathtt{und}}) \leq \mathtt{Lab} \text{ for all } \beta \leq \alpha & \text{if } \alpha \text{ is a limit ordinal} \end{cases}$$

Proposition 2. *Let $F = (\mathscr{A}, \mathscr{R})$ be an AF. Then there exists an ordinal α such that the least fixpoint of $\mathscr{F}_F$ is $\mathscr{F}_F^{\alpha}(\mathtt{Lab}_{\mathtt{und}})$.*

The original definition of argumentation semantics proposed by Dung [13] and still widely used to this day is based on *extensions* rather than labelings. While we do not make use of the extension-based definitions of argumentation semantics in this paper, we do require the notion of *admissibility* that is used in the extension-based approach:

Definition 9. *Let $F = (\mathscr{A}, \mathscr{R})$ be an AF, and let $S \subseteq \mathscr{A}$. The set S is called* conflict-free *iff there are no arguments $b, c \in S$ such that b attacks c. Argument $a \in \mathscr{A}$ is* defended *by S iff for every $b \in \mathscr{A}$ such that b attacks a there exists $c \in S$ such that c attacks b. We say that S is* admissible *iff S is conflict-free and every argument in S is defended by S.*

The following two facts directly follow from the correspondence between the labeling-based approach and the extension-based approach as presented in [2]:

Proposition 3. *Let $F = (\mathscr{A}, \mathscr{R})$ be an AF and let $E \subseteq \mathscr{A}$ be admissible. Then there is a preferred labeling of F in which all arguments in E are labeled $\mathtt{in}$.*

Proposition 4. *Let $F = (\mathscr{A}, \mathscr{R})$ be an AF and let $\mathtt{Lab}$ be a preferred labeling of F. Then the set of arguments that are labeled $\mathtt{in}$ by $\mathtt{Lab}$ is admissible.*

In this paper we will make use of the fact that grounded, complete and preferred semantics satisfy the Directionality and SCC-recursivity principles (see [18]).

The intuitive idea behind Directionality is that an argument a can only effect the status of another argument b if there is an $\mathscr{R}$-path from a to b. For formalizing this, we first need some auxiliary notions:

Definition 10. *Let $F = (\mathscr{A}, \mathscr{R})$ be an AF, let $S \subseteq \mathscr{A}$ and let* Lab *be a labeling of F. We write $F|_S$ for the restricted AF $(S, \mathscr{R} \cap (S \times S))$, and we write* Lab$|S$ *for the labeling of $F|_S$ that is identical with* Lab *on all arguments in S.*

Definition 11. *Let $F = \langle \mathscr{A}, \mathscr{R} \rangle$ be an AF. A set $U \subseteq \mathscr{A}$ is* unattacked *iff there exists no $a \in \mathscr{A} \setminus U$ such that a attacks some $b \in U$.*

Definition 12. *A semantics σ satisfies the* Directionality *principle iff for every AF F and every unattacked set U, it holds that $\sigma(F|_U) = \{\texttt{Lab}|_U \mid \texttt{Lab} \in \sigma(F)\}$.*

The intuitive idea behind SCC-recursiveness is that for computing the semantics, we can divide the AF into strongly connected components (SCCs) and compute the labelings by recursively applying the semantics to each SCC, taking into account attacks coming in from previously considered SCCs. For making this precise, we first need some auxiliary notions.

Definition 13. *Let $F = (\mathscr{A}, \mathscr{R})$ be an AF. An $\mathscr{R}$-*path in *F is a sequence $\langle a_0, \ldots, a_n \rangle$ of arguments where $(a_i, a_{i+1}) \in \mathscr{R}$ for $0 \leq i < n$ and where $a_j \neq a_k$ for $0 \leq j < k \leq n$. Let $a, b \in \mathscr{A}$. We define $a \sim b$ iff either $a = b$ or there is an $\mathscr{R}$-path from a to b and there is an $\mathscr{R}$-path from b to a. The equivalence classes under the equivalence relation $\sim$ are called* strongly connected components *(SCCs) of F. We denote the set of SCCs of F by $SCCs(F)$. Given a labeling* Lab *of F, we define $D_F(\texttt{Lab}) := \{b \in \mathscr{A} \mid \exists a \in \mathscr{A} : (a,b) \in \mathscr{R}, \texttt{Lab}(a) = \texttt{in}$ and $a \not\sim b\}$. We define $U_F(S, \texttt{Lab}) := \{a \in S \mid \nexists b : (b,a) \in \mathscr{R}, b \not\sim a$ and $\texttt{Lab}(b) \neq \texttt{out}\}$. We define $\texttt{Lab}_F^{\texttt{out}}$ to be the labeling of F that assigns* out *to every argument in $\mathscr{A}$.*

Definition 14. *A binary function BF is called a* base function *iff for every AF $F = (\mathscr{A}, \mathscr{R})$ such that $|SCCs(F)| = 1$ and every $C \subseteq \mathscr{A}$, $BF(F, C)$ is a set of labelings of F.*

Definition 15. *Given a base function BF, an AF $F = (\mathscr{A}, \mathscr{R})$ and a set $C \subseteq \mathscr{A}$, we recursively define the set $GF(BF, F, C)$ of labelings of F as follows: for every labeling* Lab *of F,* Lab $\in GF(BF, F, C)$ *iff*

- *in case $|SCCs(F)| = 1$,* Lab $\in BF(F, C)$,
- *otherwise, for all $S \in SCCs(F)$, there is a* Lab$' \in GF(BF, F|_{S \setminus D_F(\texttt{Lab})}, U_F(S, \texttt{Lab}) \cap C)$ *such that* Lab$|_S =$ Lab$' \cup \texttt{Lab}^{\texttt{out}}_{F|_{S \cap D_F(\texttt{Lab})}}$.

Definition 16. *A semantics σ satisfies the* SCC-recursiveness *principle iff there is a base function BF such that for every AF $F = (\mathscr{A}, \mathscr{R})$ we have $\sigma(F) = GF(BF, F, \mathscr{A})$.*

3. Label Functions

In this section we define the basic notions of a labeling function, an input-output argumentation framework and the representability of a labeling function.

Definition 17. *A* labeling function *LF is a function from Labs to Labs.*

Definition 18. *Let LF_1 and LF_2 be two labeling functions. Then $LF_1 \circ LF_2$ denotes the composition of these two labeling functions that is defined as $LF_1 \circ LF_2(L) = LF_1(LF_2(L))$.*

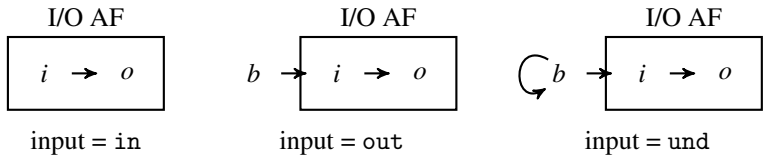

Figure 1. The three standard AFs for the I/O AF that cgp-represents the labeling function $(\texttt{out},\texttt{in},\texttt{und})$.

We use the triplet $(LF(\texttt{in}),LF(\texttt{out}),LF(\texttt{und}))$ to refer to LF in a concise way. For example, the triplet $(\texttt{out},\texttt{und},\texttt{in})$ denotes the labeling function that maps `in` to `out`, `out` to `und` and `und` to `in`.

Definition 19. *An* input-output argumentation framework *(*I/O AF*) is a tuple* $(\mathscr{A},\mathscr{R},i,o)$*, where* $(\mathscr{A},\mathscr{R})$ *is an argumentation framework and* $i,o \in \mathscr{A}$*.*

Definition 20. *Given an input-output argumentation framework* $G = (\mathscr{A},\mathscr{R},i,o)$*, with an argument* $b \notin \mathscr{A}$ *and a label* $L \in Labs$*, the* standard argumentation framework w.r.t. G and L *– denoted* $F_{st}(G,L)$ *– is the argumentation framework* $(\mathscr{A}',\mathscr{R}')$*, where* $\mathscr{A}'$ *and* $\mathscr{R}'$ *are defined through the following case distinction:*

- *If* $L = \texttt{in}$*, then* $\mathscr{A}' = \mathscr{A}$ *and* $\mathscr{R}' = \mathscr{R}$*.*
- *If* $L = \texttt{out}$*, then* $\mathscr{A}' = \mathscr{A} \cup \{b\}$ *and* $\mathscr{R}' = \mathscr{R} \cup \{(b,i)\}$*.*
- *If* $L = \texttt{und}$*, then* $\mathscr{A}' = \mathscr{A} \cup \{b\}$ *and* $\mathscr{R}' = \mathscr{R} \cup \{(b,b),(b,i)\}$*.*

Definition 21. *Let* σ *be an argumentation semantics. An input-output argumentation framework* G represents *a labeling function* LF *w.r.t.* σ *iff for every* $L \in Labs$*,* $\sigma(F_{st}(G,L)) \neq \emptyset$ *and for every labeling* $\texttt{Lab} \in \sigma(F_{st}(G,L))$*,* $Lab(i) = L$ *and* $\texttt{Lab}(o) = LF(L)$*.*

Definition 22. *Let* σ *be an argumentation semantics. A labeling function* LF *is called* σ-representable *iff there is some input-output argumentation framework* G *that represents* LF *w.r.t.* σ*.*

In this work, we shall focus on three of the most well-known semantics, namely complete, grounded and preferred. The principles that these semantics satisfy make them the most appropriate to start with.

Definition 23. *We define* cgp *to be the set of semantics* $\{$*complete, grounded, preferred*$\}$*. If a labeling function can be* σ*-represented for every* $\sigma \in \texttt{cgp}$*, we say that the function is* cgp*-representable. Similarly, if a labeling function cannot be* σ*-represented for any* $\sigma \in \texttt{cgp}$*, we say that the function is* cgp*-unrepresentable.*

Example 1. *Consider the labeling function* $(\texttt{out},\texttt{in},\texttt{und})$ *which maps* `in` *to* `out` *and vice-versa, leaving* `und` *as it is. This function can be* cgp*-represented as depicted in Fig. 1. By having the input directly attack the output, when the input is* `in`*, it forces the output to be* `out`*. Conversely, when the input is* `out`*, there is no attacker of the output left, so it must be* `in`*. And finally when the input is* `und`*, the undecided label propagates to the output.*

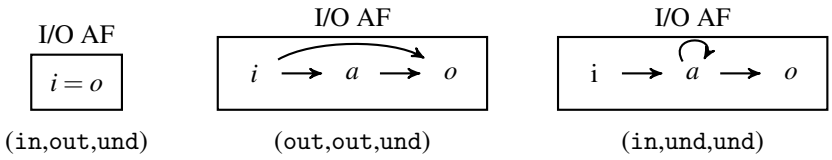

Figure 2. cgp-representation of three labeling functions.

Example 2. *Fig. 2 depicts three I/O AFs that* cgp*-represent the labeling functions* $(\texttt{in},\texttt{out},\texttt{und})$*,* $(\texttt{out},\texttt{out},\texttt{und})$ *and* $(\texttt{in},\texttt{und},\texttt{und})$ *respectively. Note that the I/O AF that represents the identity function* $(\texttt{in},\texttt{out},\texttt{und})$ *consists only of a single argument, so that the input argument i and the output argument o are the same argument.*

We now define how two input-output argumentation frameworks can be composed into a single one. The intuitive idea is that the output of the first I/O AF is used as input for the second I/O AF.

Definition 24. *Let* $G_1 = (\mathscr{A}_1,\mathscr{R}_1,i_1,o_1)$ *and* $G_2 = (\mathscr{A}_2,\mathscr{R}_2,i_2,o_2)$ *be two input-output argumentation frameworks with* $\mathscr{A}_1 \cap \mathscr{A}_2 = \emptyset$*, and let* $c \notin \mathscr{A}_1 \cup \mathscr{A}_2$*. Then we define* $G_1 \oplus G_2$ *to be the input-output argumentation framework* $(\mathscr{A}_1 \cup \mathscr{A}_2 \cup \{c\}, \mathscr{R}_1 \cup \mathscr{R}_2 \cup \{(o_1,c)\} \cup \{(c,i_2)\}, i_1, o_2)$.

The following theorem establishes that composed AFs represent composed label functions with respect to the complete, grounded and preferred semantics.

Theorem 1. *Let* LF_1 *and* LF_2 *be representable labeling functions, and let* $G_1 = (\mathscr{A}_1,\mathscr{R}_1,i_1,o_1)$ *and* $G_2 = (\mathscr{A}_2,\mathscr{R}_2,i_2,o_2)$ *be input-output argumentation frameworks that represent* LF_1 *and* LF_2 *respectively. Then* $G_1 \oplus G_2$ cgp*-represents* $LF_2 \circ LF_1$.

Proof. Let $\sigma \in \texttt{cgp}$. Let $L \in \mathit{Labs}$. Then every σ-labeling of $F_{st}(G_1,L)$ assigns the label $LF_1(L)$ to o_1, and every σ-labeling of $F_{st}(G_2,LF_1(L))$ assigns the label $LF_2 \circ LF_1(L)$ to o_2. We need to show that every σ-labeling of $F_{st}(G_1 \oplus G_2,L)$ assigns the label $LF_2 \circ LF_1(L)$ to o_2. So let $\texttt{Lab}$ be a σ-labeling of $F_{st}(G_1 \oplus G_2,L)$. By the Directionality principle for σ, $\texttt{Lab}|_{F_{st}(G_1,L)} \in \sigma(F_{st}(G_1,L))$, so $\texttt{Lab}(o_1) = LF_1(L)$.

We write $F^* = (\mathscr{A}^*,\mathscr{R}^*)$ for $F_{st}(G_2,LF_1(L))$, and we write $F' = (\mathscr{A}',\mathscr{R}')$ for $F_{st}(G_1 \oplus G_2,L)$. Recall that by Definition 20, $\mathscr{A}^* = \mathscr{A}_2 \cup \{b\}$ if $LF_1(L) = \texttt{out}$ and $\mathscr{A}^* = \mathscr{A}_2$ otherwise. Let $\texttt{Lab}_b$ be the unique σ-labeling of $F^*|_{\mathscr{A}^* \setminus \mathscr{A}_2}$, i.e. $\texttt{Lab}_b = \{(b,\texttt{in})\}$ if $LF_1(L) = \texttt{out}$, and $\texttt{Lab}_b = \emptyset$ otherwise.

By the SCC-recursiveness principle for σ, there is a base function BF_σ such that for every AF $F = (\mathscr{A},\mathscr{R})$, $\sigma(F) = GF(BF_\sigma,F,\mathscr{A})$. In particular, $\sigma(F') = GF(BF_\sigma,F',\mathscr{A}')$, i.e. $\texttt{Lab} \in GF(BF_\sigma,F',\mathscr{A}')$. Clearly $|\mathit{SCCs}(F')| > 1$, so by the SCC-recursiveness principle for σ, for all $S \in \mathit{SCCs}(F')$, we have:

$$\begin{aligned} &\text{There is a } \texttt{Lab}' \in GF(BF_\sigma,F'|_{S \setminus D_{F'}(\texttt{Lab})}, U_{F'}(S,\texttt{Lab})) \\ &\text{such that } \texttt{Lab}|_S = \texttt{Lab}' \cup \texttt{Lab}^{\texttt{out}}_{F'|_{S \cap D_{F'}(\texttt{Lab})}}. \end{aligned} \tag{1}$$

When applying this equation in Case 3 below, we will need to make use of the fact that if $S \subseteq \mathscr{A}_2$, then $S \setminus D_{F'}(\texttt{Lab}) = S \setminus D_{F^*}(\texttt{Lab}_b \cup (\texttt{Lab}|_{\mathscr{A}_2}))$ and $U_{F'}(S,\texttt{Lab}) = U_{F^*}(S,\texttt{Lab}_b \cup$

$(\texttt{Lab}|_{\mathscr{A}_2})$). In the following we show that $U_{F'}(S,\texttt{Lab}) \subseteq U_{F^*}(S,\texttt{Lab}_b \cup (\texttt{Lab}|_{\mathscr{A}_2}))$. The facts that $U_{F^*}(S,\texttt{Lab}_b \cup (\texttt{Lab}|_{\mathscr{A}_2})) \subseteq U_{F'}(S,\texttt{Lab})$, that $S \setminus D_{F'}(\texttt{Lab}) \subseteq S \setminus D_{F^*}(\texttt{Lab}_b \cup (\texttt{Lab}|_{\mathscr{A}_2}))$ and that $S \setminus D_{F^*}(\texttt{Lab}_b \cup (\texttt{Lab}|_{\mathscr{A}_2})) \subseteq S \setminus D_{F'}(\texttt{Lab})$ can be established in a similar way.

> Suppose $x \in U_{F'}(S,\texttt{Lab})$, i.e. $x \in S$ and there is no y such that $(y,x) \in \mathscr{R}'$, $y \not\sim x$ and $\texttt{Lab}(y) \neq \texttt{out}$. Now suppose for a contradiction that $x \notin U_{F^*}(S,\texttt{Lab}_b \cup (\texttt{Lab}|_{\mathscr{A}_2}))$, i.e. there is a z such that $(z,x) \in \mathscr{R}^*$, $z \not\sim x$ and $\texttt{Lab}_b \cup (\texttt{Lab}|_{\mathscr{A}_2})(z) \neq \texttt{out}$. We distinguish two cases:
>
> Case (i): $z \in \mathscr{A}_2$. Choose $y := z$. Note that since $z \not\sim x$, $(z,x) \neq (i,i)$. So $(y,x) = (z,x) \in \mathscr{R}'$, $y \not\sim x$ and $\texttt{Lab}(y) = \texttt{Lab}(z) = \texttt{Lab}_b \cup (\texttt{Lab}|_{\mathscr{A}_2})(z) \neq \texttt{out}$. This contradicts the assumption that there is no such y.
>
> Case (ii): $z = b$. In this case $LF_1(L) = \texttt{out}$. Choose $y := c$. Then $(b,x) \in \mathscr{R}^*$, i.e. $x = i$. Therefore $(y,x) = (c,i) \in \mathscr{R}'$. Additionally $y \not\sim x$. Furthermore, since $\texttt{Lab}(o_1) = LF_1(L) = \texttt{out}$ and o_1 is the only attacker of c, $\texttt{Lab}(c) = \texttt{in}$, i.e. $\texttt{Lab}(y) \neq \texttt{out}$. This contradicts the assumption that there is no such y.
>
> Thus $U_{F'}(S,\texttt{Lab}) \subseteq U_{F^*}(S,\texttt{Lab}_b \cup (\texttt{Lab}|_{\mathscr{A}_2}))$, as required.

Recall that every σ-labeling of F^* assigns the label $LF_2 \circ LF_1(L)$ to o_2. We now establish the required result that $\texttt{Lab}(o_2) = LF_2 \circ LF_1(L)$ by showing that $\texttt{Lab}_b \cup (\texttt{Lab}|_{\mathscr{A}_2}) \in \sigma(F^*)$. By the SCC-recursivity of σ, it is enough to show that $\texttt{Lab}_b \cup (\texttt{Lab}|_{\mathscr{A}_2}) \in GF(BF_\sigma, F^*, \mathscr{A}^*)$. Clearly $|SCCs(F^*)| > 1$, so by the SCC-recursiveness principle for σ it is enough to show that for all $S \in SCCs(F^*)$, there is a $\texttt{Lab}' \in GF(BF_\sigma, F^*|_{S \setminus D_{F^*}(\texttt{Lab}_b \cup (\texttt{Lab}|_{\mathscr{A}_2}))}, U_{F^*}(S, \texttt{Lab}_b \cup (\texttt{Lab}|_{\mathscr{A}_2})))$ such that $(\texttt{Lab}_b \cup (\texttt{Lab}|_{\mathscr{A}_2}))|_S = \texttt{Lab}' \cup \texttt{Lab}^{\texttt{out}}_{F^*|_{S \cap D_{F^*}(\texttt{Lab})}}$. So let $S \in SCCs(F^*)$. We distinguish two cases:

Case 1: $S = \{b\}$ and $LF_1(L) = \texttt{out}$. In this case, $(\texttt{Lab}_b \cup (\texttt{Lab}|_{\mathscr{A}_2}))|_S$ is $\{(b,\texttt{in})\}$, i.e. the unique σ-labeling of the AF $F^*|_S = (\{b\}, \emptyset)$. By the SCC-recursivity of σ, $GF(BF_\sigma, F^*|_S, S) = \sigma(F^*|_S)$, so $\{(b,\texttt{in})\} \in GF(BF_\sigma, F^*|_S, S)$. Note that S is unattacked, i.e. $F'|_{S \cap D_{F'}(\texttt{Lab})}$ is the empty AF and $\texttt{Lab}^{\texttt{out}}_{F'|_{S \cap D_{F'}(\texttt{Lab})}}$ is the empty labeling. So we can choose $\texttt{Lab}' = (\texttt{Lab}_b \cup (\texttt{Lab}|_{\mathscr{A}_2}))|_S = \{(b,\texttt{in})\}$. Furthermore, $F^*|_{S \setminus D_{F^*}(\texttt{Lab}_b \cup (\texttt{Lab}|_{\mathscr{A}_2}))} = F^*|_S$ and $U_{F^*}(S, \texttt{Lab}_b \cup (\texttt{Lab}|_{\mathscr{A}_2})) = S$. Thus $GF(BF_\sigma, F^*|_{S \setminus D_{F^*}(\texttt{Lab}_b \cup (\texttt{Lab}|_{\mathscr{A}_2}))}, U_{F^*}(S, \texttt{Lab}_b \cup (\texttt{Lab}|_{\mathscr{A}_2}))) = GF(BF_\sigma, F^*|_S, S)$, which contains $\texttt{Lab}' = \{(b,\texttt{in})\}$, as required.

Case 2: $S = \{b\}$ and $LF_1(L) = \texttt{und}$. In this case, $(\texttt{Lab}_b \cup (\texttt{Lab}|_{\mathscr{A}_2}))|_S$ is $\{(b,\texttt{und})\}$, i.e. the unique σ-labeling of the AF $F^*|_S = (\{b\}, \{b,b)\})$. By the SCC-recursivity of σ, $GF(BF_\sigma, F^*|_S, S) = \sigma(F^*|_S)$, so $\{(b,\texttt{und})\} \in GF(BF_\sigma, F^*|_S, S)$. Note that S is unattacked, i.e. $F'|_{S \cap D_{F'}(\texttt{Lab})}$ is the empty AF and $\texttt{Lab}^{\texttt{out}}_{F'|_{S \cap D_{F'}(\texttt{Lab})}}$ is the empty labeling. So we can choose $\texttt{Lab}' = (\texttt{Lab}_b \cup (\texttt{Lab}|_{\mathscr{A}_2}))|_S = \{(b,\texttt{und})\}$. Furthermore, $F^*|_{S \setminus D_{F^*}(\texttt{Lab}_b \cup (\texttt{Lab}|_{\mathscr{A}_2}))} = F^*|_S$ and $U_{F^*}(S, \texttt{Lab}_b \cup (\texttt{Lab}|_{\mathscr{A}_2})) = S$. Thus $GF(BF_\sigma, F^*|_{S \setminus D_{F^*}(\texttt{Lab}_b \cup (\texttt{Lab}|_{\mathscr{A}_2}))}, U_{F^*}(S, \texttt{Lab}_b \cup (\texttt{Lab}|_{\mathscr{A}_2}))) = GF(BF_\sigma, F^*|_S, S)$, which contains $\texttt{Lab}' = \{(b,\texttt{und})\}$, as required.

Case 3: $S \subseteq \mathscr{A}_2$. Then $(\texttt{Lab}_b \cup (\texttt{Lab}|_{\mathscr{A}_2}))|_S = \texttt{Lab}|_S$. Furthermore, $S \in SCCs(F')$, so by equation (1), there is a $\texttt{Lab}' \in GF(BF_\sigma, F'|_{S \setminus D_{F'}(\texttt{Lab})}, U_{F'}(S,\texttt{Lab}))$ such that $\texttt{Lab}|_S = \texttt{Lab}' \cup \texttt{Lab}^{\texttt{out}}_{F'|_{S \cap D_{F'}(\texttt{Lab})}}$. Thus $(\texttt{Lab}_b \cup (\texttt{Lab}|_{\mathscr{A}_2}))|_S = \texttt{Lab}' \cup \texttt{Lab}^{\texttt{out}}_{F'|_{S \cap D_{F'}(\texttt{Lab})}}$,

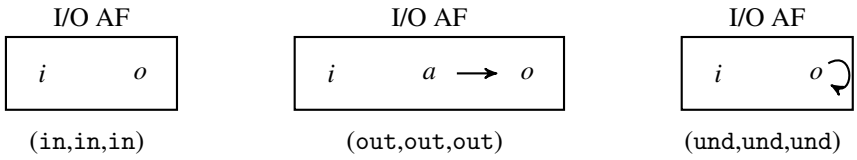

Figure 3. cgp-representation of the three constant labeling functions.

as required. Furthermore, as shown above, $S \setminus D_{F'}(\texttt{Lab}) = S \setminus D_{F^*}(\texttt{Lab}_b \cup (\texttt{Lab}|_{\mathscr{A}_2}))$ and $U_{F'}(S,\texttt{Lab}) = U_{F^*}(S,\texttt{Lab}_b \cup (\texttt{Lab}|_{\mathscr{A}_2}))$, so $\texttt{Lab}' \in GF(BF_\sigma, F^*|_{S\setminus D_{F^*}(\texttt{Lab}_b \cup (\texttt{Lab}|_{\mathscr{A}_2}))}, U_{F^*}(S,\texttt{Lab}_b \cup (\texttt{Lab}|_{\mathscr{A}_2})))$, as required. □

The following corollary directly follows from Theorem 1

Corollary 1. *If LF_1 and LF_2 are* cgp*-representable, then $LF_1 \circ LF_2$ is* cgp*-representable.*

4. Representability of Label Functions

In this section, we will categorize the twenty seven label functions into eleven functions that are cgp-representable and sixteen functions that are not cgp-representable.

As we will show below, a label function is cgp-representable iff it is either a constant function or maps und to und. This motivates the following definition:

Definition 25. *We define the set* Rep *as the following set of labeling functions:*

$$\texttt{Rep} = \{(\texttt{in},\texttt{in},\texttt{in}),(\texttt{out},\texttt{out},\texttt{out})\} \cup \{(l,l',\texttt{und}) \mid l,l' \in \mathit{Labs}\}$$

Theorem 2. *Every function in* Rep *is* cgp*-representable.*

Proof. We have already given the cgp-representations for four of those functions in the Examples 1 and 2. We additionally have representations for the three constant functions $(\texttt{in},\texttt{in},\texttt{in})$, $(\texttt{out},\texttt{out},\texttt{out})$ and $(\texttt{und},\texttt{und},\texttt{und})$ in Figure 3. The missing four functions can be represented by combinations of these seven as follows:

- $(\texttt{in},\texttt{in},\texttt{und}) = (\texttt{out},\texttt{in},\texttt{und}) \circ (\texttt{out},\texttt{out},\texttt{und})$;
- $(\texttt{und},\texttt{in},\texttt{und}) = (\texttt{in},\texttt{und},\texttt{und}) \circ (\texttt{out},\texttt{in},\texttt{und})$;
- $(\texttt{out},\texttt{und},\texttt{und}) = (\texttt{out},\texttt{in},\texttt{und}) \circ (\texttt{in},\texttt{und},\texttt{und})$;
- $(\texttt{und},\texttt{out},\texttt{und}) = (\texttt{out},\texttt{in},\texttt{und}) \circ (\texttt{und},\texttt{in},\texttt{und})$. □

Aside from the widely used semantics included in the set cgp, the stable semantics is another well-known semantics which is also complete-based. Notice however that the stable semantics does not allow for any und arguments, and thus no framework could stable-represent a labeling function as defined in Def. 21, since having und as input would automatically mean there is no extension in the corresponding standard AF, so no output could be given. We can however define a similar notion over 2-valued labelings, i.e. restricting the functions to only two possible inputs and outputs: in and out.

This restriction leaves us with only four different possible labeling functions, and an interesting small result is that all of these are stable-representable. $(\texttt{out},\texttt{in})$ is stable-represented by the I/O AF in Figure 1 and $(\texttt{in},\texttt{out})$ by the I/O AF on the left in Figure 2.

$(\texttt{in},\texttt{in})$ and $(\texttt{out},\texttt{out})$ are stable-represented by the I/O AFs in Figure 3, respectively on the left and in the middle.

Proposition 5. *The four 2-valued labeling functions* $(\texttt{in},\texttt{out}),(\texttt{out},\texttt{in}),(\texttt{in},\texttt{in})$ *and* $(\texttt{out},\texttt{out})$ *are all stable-representable.*

4.1. Unrepresentable Label Functions

In this subsection we establish that the sixteen labeling functions not included in Rep are actually cgp-unrepresentable. We first consider the labeling functions $(\texttt{und},\texttt{und},\texttt{out})$ and $(\texttt{out},\texttt{und},\texttt{out})$ with respect to the preferred and grounded semantics.

Lemma 1. *The labeling functions* $(\texttt{und},\texttt{und},\texttt{out})$ *and* $(\texttt{out},\texttt{und},\texttt{out})$ *are* cgp*-unrepresentable.*

Proof. We show this using a proof by contradiction. Assume $G = (\mathscr{A},\mathscr{R},i,o)$ is an input-output argumentation framework that cgp-represents either $(\texttt{und},\texttt{und},\texttt{out})$ or $(\texttt{out},\texttt{und},\texttt{out})$. This means that in every complete, grounded and preferred labeling of $F_{st}(G,\texttt{und})$, the output argument o is labeled out. We first show how to derive a contradiction in the case of preferred. Let Lab be a preferred labeling of $F_{st}(G,\texttt{und})$, and let E be the set of arguments labeled in by Lab. By Proposition 4 E is admissible w.r.t. $F_{st}(G,\texttt{und})$. This implies that E is admissible w.r.t. $F_{st}(G,\texttt{out})$:

- Conflict-freeness of E w.r.t. $F_{st}(G,\texttt{out})$ follows from the fact that the only attack in $F_{st}(G,\texttt{out})$ that is not present in $F_{st}(G,\texttt{und})$ is the attack from the special argument b to the input argument i, but clearly $b \notin E$.
- Self-defence of E w.r.t. $F_{st}(G,\texttt{out})$ follows from the fact that the only attack in $F_{st}(G,\texttt{und})$ that is not present in $F_{st}(G,\texttt{out})$ is the self-attack on the input argument i, but clearly $i \notin E$.

Now by Proposition 3, there exists a preferred labeling $\texttt{Lab}'$ of $F_{st}(G,\texttt{out})$ in which every argument in E is labeled in. Since $\texttt{Lab}(o) = \texttt{out}$, some argument c labeled in by Lab attacks o. But then $c \in E$, so $\texttt{Lab}'(c) = \texttt{in}$, so $\texttt{Lab}'(o) = \texttt{out}$. But this contradicts the assumption that G represents $(\texttt{und},\texttt{und},\texttt{out})$ or $(\texttt{out},\texttt{und},\texttt{out})$ w.r.t. the preferred semantics, because this would mean that every preferred labeling of $F_{st}(G,\texttt{out})$ labels o as und.

Now we consider the case of the grounded semantics. By a simple transfinite indution one can show that for every ordinal α, $\mathscr{F}^{\alpha}_{F_{st}(G,\texttt{und})}(\texttt{Lab}_{\texttt{und}})(i) = \texttt{und}$ (since i attacks itself in $F_{st}(G,\texttt{und})$). By Propositions 1 and 2, there exists an ordinal α such that the grounded labeling of $F_{st}(G,\texttt{und})$ is $\mathscr{F}^{\alpha}_{F_{st}(G,\texttt{und})}(\texttt{Lab}_{\texttt{und}})$. We now show by transfinite induction that for every ordinal β, $\mathscr{F}^{\beta}_{F_{st}(G,\texttt{und})}(\texttt{Lab}_{\texttt{und}}) \leq \mathscr{F}^{\beta}_{F_{st}(G,\texttt{out})}(\texttt{Lab}_{\texttt{und}})$:

- $\underline{\beta = 0}$: Trivial.
- $\underline{\beta = 1}$: In this case $\mathscr{F}^{\beta}_{F_{st}(G,\texttt{und})}(\texttt{Lab}_{\texttt{und}})$ labels as in all arguments in $\mathscr{A}$ that are unattacked, and $\mathscr{F}^{\beta}_{F_{st}(G,\texttt{out})}(\texttt{Lab}_{\texttt{und}})$ also labels all these arguments as in, and additionally labels the special argument b as in. So clearly $\mathscr{F}^{\beta}_{F_{st}(G,\texttt{und})}(\texttt{Lab}_{\texttt{und}}) \leq \mathscr{F}^{\beta}_{F_{st}(G,\texttt{out})}(\texttt{Lab}_{\texttt{und}})$.

- $\beta = 2$: In this case the in-labeled arguments are the same as in the case $\beta = 1$ for both $\mathscr{F}^{\beta}_{F_{st}(G,\mathtt{und})}(\mathtt{Lab}_{\mathtt{und}})$ and $\mathscr{F}^{\beta}_{F_{st}(G,\mathtt{out})}(\mathtt{Lab}_{\mathtt{und}})$. Additionally, every argument attacked by an unattacked argument from $\mathscr{A}$ is labeled out by both $\mathscr{F}^{\beta}_{F_{st}(G,\mathtt{und})}(\mathtt{Lab}_{\mathtt{und}})$ and $\mathscr{F}^{\beta}_{F_{st}(G,\mathtt{out})}(\mathtt{Lab}_{\mathtt{und}})$, and $\mathscr{F}^{\beta}_{F_{st}(G,\mathtt{out})}(\mathtt{Lab}_{\mathtt{und}})$ additionally labels the special argument i as out. So clearly $\mathscr{F}^{\beta}_{F_{st}(G,\mathtt{und})}(\mathtt{Lab}_{\mathtt{und}}) \leq \mathscr{F}^{\beta}_{F_{st}(G,\mathtt{out})}(\mathtt{Lab}_{\mathtt{und}})$.
- $\beta = \gamma + 1$ for $\gamma \geq 2$: By the inductive hypothesis, we may assume that $\mathscr{F}^{\gamma}_{F_{st}(G,\mathtt{und})}(\mathtt{Lab}_{\mathtt{und}}) \leq \mathscr{F}^{\gamma}_{F_{st}(G,\mathtt{out})}(\mathtt{Lab}_{\mathtt{und}})$. By the definition of $\leq$, every argument labeled in by $\mathscr{F}^{\gamma}_{F_{st}(G,\mathtt{und})}(\mathtt{Lab}_{\mathtt{und}})$ is also labeled in by $\mathscr{F}^{\gamma}_{F_{st}(G,\mathtt{out})}(\mathtt{Lab}_{\mathtt{und}})$. The fact that every argument in $\mathscr{A}$ has the same attackers in $F_{st}(G,\mathtt{out})$ as in $F_{st}(G,\mathtt{und})$ together with the definition of $\mathscr{F}$ imply that every argument in $\mathscr{A}$ that is labeled out by $\mathscr{F}^{\gamma+1}_{F_{st}(G,\mathtt{und})}(\mathtt{Lab}_{\mathtt{und}})$ is also labeled out by $\mathscr{F}^{\gamma+1}_{F_{st}(G,\mathtt{out})}(\mathtt{Lab}_{\mathtt{und}})$. Since $\mathscr{F}^{\gamma+1}_{F_{st}(G,\mathtt{und})}(\mathtt{Lab}_{\mathtt{und}})(i) = \mathtt{und}$, this implies that every argument that is labeled out by $\mathscr{F}^{\gamma+1}_{F_{st}(G,\mathtt{und})}(\mathtt{Lab}_{\mathtt{und}})$ is also labeled out by $\mathscr{F}^{\gamma+1}_{F_{st}(G,\mathtt{out})}(\mathtt{Lab}_{\mathtt{und}})$. Similarly one can show that every argument that is labeled in by $\mathscr{F}^{\gamma+1}_{F_{st}(G,\mathtt{und})}(\mathtt{Lab}_{\mathtt{und}})$ is also labeled in by $\mathscr{F}^{\gamma+1}_{F_{st}(G,\mathtt{out})}(\mathtt{Lab}_{\mathtt{und}})$. Thus $\mathscr{F}^{\beta}_{F_{st}(G,\mathtt{und})}(\mathtt{Lab}_{\mathtt{und}}) \leq \mathscr{F}^{\beta}_{F_{st}(G,\mathtt{out})}(\mathtt{Lab}_{\mathtt{und}})$.
- β is a limit ordinal: Suppose c is an argument such that $\mathscr{F}^{\beta}_{F_{st}(G,\mathtt{und})}(\mathtt{Lab}_{\mathtt{und}})(c) = \mathtt{in}$. This means that there is some $\gamma < \beta$ such that $\mathscr{F}^{\gamma}_{F_{st}(G,\mathtt{und})}(\mathtt{Lab}_{\mathtt{und}})(i) = \mathtt{in}$. By induction hypothesis, $\mathscr{F}^{\gamma}_{F_{st}(G,\mathtt{out})}(\mathtt{Lab}_{\mathtt{und}}) = \mathtt{in}$, so $\mathscr{F}^{\beta}_{F_{st}(G,\mathtt{out})}(\mathtt{Lab}_{\mathtt{und}}) = \mathtt{in}$. Thus every argument that is labeled in by $\mathscr{F}^{\beta}_{F_{st}(G,\mathtt{und})}(\mathtt{Lab}_{\mathtt{und}})$ is also labeled in by $\mathscr{F}^{\beta}_{F_{st}(G,\mathtt{out})}(\mathtt{Lab}_{\mathtt{und}})$. Similarly every argument that is labeled out by $\mathscr{F}^{\beta}_{F_{st}(G,\mathtt{und})}(\mathtt{Lab}_{\mathtt{und}})$ is also labeled out by $\mathscr{F}^{\beta}_{F_{st}(G,\mathtt{out})}(\mathtt{Lab}_{\mathtt{und}})$. Thus $\mathscr{F}^{\beta}_{F_{st}(G,\mathtt{und})}(\mathtt{Lab}_{\mathtt{und}}) \leq \mathscr{F}^{\beta}_{F_{st}(G,\mathtt{out})}(\mathtt{Lab}_{\mathtt{und}})$.

Since G represents either $(\mathtt{und},\mathtt{und},\mathtt{out})$ or $(\mathtt{out},\mathtt{und},\mathtt{out})$ w.r.t. the grounded semantics, the output argument o is labeled out by the grounded labeling of $F_{st}(G,\mathtt{und})$. So $\mathscr{F}^{\alpha}_{F_{st}(G,\mathtt{und})}(\mathtt{Lab}_{\mathtt{und}})(o) = \mathtt{out}$. But since $\mathscr{F}^{\alpha}_{F_{st}(G,\mathtt{und})}(\mathtt{Lab}_{\mathtt{und}}) \leq \mathscr{F}^{\alpha}_{F_{st}(G,\mathtt{out})}(\mathtt{Lab}_{\mathtt{und}})$, this means that $\mathscr{F}^{\alpha}_{F_{st}(G,\mathtt{out})}(\mathtt{Lab}_{\mathtt{und}})(o) = \mathtt{out}$. So the grounded labeling of $F_{st}(G,\mathtt{out})$ labels o as out, in contradiction to the assumption that G represents $(\mathtt{und},\mathtt{und},\mathtt{out})$ or $(\mathtt{out},\mathtt{und},\mathtt{out})$ w.r.t. the grounded semantics.

Finally we consider the case of the complete semantics. Every prefered labeling is a complete labeling and every AF has at least one prefered labeling. These two facts together imply that whenever an input-output argumentation framework G represents an labeling function LF w.r.t. the complete semantics, G also represents LF w.r.t. the prefered semantics. So since the labeling functions $(\mathtt{und},\mathtt{und},\mathtt{out})$ and $(\mathtt{out},\mathtt{und},\mathtt{out})$ are not prefered-representable, they are not complete-representable either. □

Now we extend these results to cover all labeling functions not in Rep.

Theorem 3. *The sixteen labeling functions not in* Rep *are* cgp*-unrepresentable.*

Proof. In Lemma 1 we have already established that the labeling functions $(\texttt{und}, \texttt{und}, \texttt{out})$ and $(\texttt{out}, \texttt{und}, \texttt{out})$ are cgp-unrepresentable. For the other fourteen labeling functions not in Rep we show this result by showing that if one of them was representable, then one of $(\texttt{und}, \texttt{und}, \texttt{out})$ or $(\texttt{out}, \texttt{und}, \texttt{out})$ would be representable too, which would be a contradiction. For this purpose we show in the table below how each of these fourteen labeling functions not in Rep or mentioned in Lemma 1 can be composed with some of the eleven cgp-representable labeling functions from Rep to define either $(\texttt{und}, \texttt{und}, \texttt{out})$ or $(\texttt{out}, \texttt{und}, \texttt{out})$.

The second column of the following table presents for each label function mentioned in the first column a proof in concize notation that shows why the label function in question is cgp-unrepresentable. We explain how these proofs in concize notation should be read through the example of the first proof presented in the table: If $(\texttt{in}, \texttt{in}, \texttt{out})$ were cgp-representable, then the fact that $(\texttt{und}, \texttt{und}, \texttt{out}) = (\texttt{und}, \texttt{out}, \texttt{und}) \circ (\texttt{in}, \texttt{in}, \texttt{out})$ and that $(\texttt{und}, \texttt{out}, \texttt{und})$ is cgp-representable would imply that $(\texttt{und}, \texttt{und}, \texttt{out})$ is cgp-representable by Corollary 1, which would contradict Lemma 1. So we can conclude that $(\texttt{in}, \texttt{in}, \texttt{out})$ is cgp-unrepresentable.

Labeling function	Reason for this labeling function being cgp-unrepresentable
$(\texttt{in}, \texttt{in}, \texttt{out})$	$(\texttt{und}, \texttt{und}, \texttt{out}) = (\texttt{und}, \texttt{out}, \texttt{und}) \circ (\texttt{in}, \texttt{in}, \texttt{out})$
$(\texttt{in}, \texttt{out}, \texttt{in})$	$(\texttt{out}, \texttt{und}, \texttt{out}) = (\texttt{out}, \texttt{und}, \texttt{und}) \circ (\texttt{in}, \texttt{out}, \texttt{in})$
$(\texttt{in}, \texttt{out}, \texttt{out})$	$(\texttt{out}, \texttt{und}, \texttt{out}) = (\texttt{und}, \texttt{out}, \texttt{und}) \circ (\texttt{in}, \texttt{out}, \texttt{out}) \circ (\texttt{out}, \texttt{in}, \texttt{und})$
$(\texttt{in}, \texttt{und}, \texttt{in})$	$(\texttt{out}, \texttt{und}, \texttt{out}) = (\texttt{out}, \texttt{in}, \texttt{und}) \circ (\texttt{in}, \texttt{und}, \texttt{in})$
$(\texttt{in}, \texttt{und}, \texttt{out})$	$(\texttt{out}, \texttt{und}, \texttt{out}) = (\texttt{out}, \texttt{out}, \texttt{und}) \circ (\texttt{in}, \texttt{und}, \texttt{out})$
$(\texttt{out}, \texttt{in}, \texttt{in})$	$(\texttt{out}, \texttt{und}, \texttt{out}) = (\texttt{out}, \texttt{und}, \texttt{und}) \circ (\texttt{out}, \texttt{in}, \texttt{in}) \circ (\texttt{out}, \texttt{in}, \texttt{und})$
$(\texttt{out}, \texttt{in}, \texttt{out})$	$(\texttt{out}, \texttt{und}, \texttt{out}) = (\texttt{und}, \texttt{out}, \texttt{und}) \circ (\texttt{out}, \texttt{in}, \texttt{out})$
$(\texttt{out}, \texttt{out}, \texttt{in})$	$(\texttt{und}, \texttt{und}, \texttt{out}) = (\texttt{out}, \texttt{und}, \texttt{und}) \circ (\texttt{out}, \texttt{out}, \texttt{in})$
$(\texttt{out}, \texttt{und}, \texttt{in})$	$(\texttt{out}, \texttt{und}, \texttt{out}) = (\texttt{out}, \texttt{out}, \texttt{und}) \circ (\texttt{out}, \texttt{und}, \texttt{in})$
$(\texttt{und}, \texttt{in}, \texttt{in})$	$(\texttt{out}, \texttt{und}, \texttt{out}) = (\texttt{out}, \texttt{in}, \texttt{und}) \circ (\texttt{und}, \texttt{in}, \texttt{in}) \circ (\texttt{out}, \texttt{in}, \texttt{und})$
$(\texttt{und}, \texttt{in}, \texttt{out})$	$(\texttt{und}, \texttt{und}, \texttt{out}) = (\texttt{und}, \texttt{out}, \texttt{und}) \circ (\texttt{und}, \texttt{in}, \texttt{out})$
$(\texttt{und}, \texttt{out}, \texttt{in})$	$(\texttt{out}, \texttt{und}, \texttt{out}) = (\texttt{out}, \texttt{out}, \texttt{und}) \circ (\texttt{und}, \texttt{out}, \texttt{in}) \circ (\texttt{out}, \texttt{in}, \texttt{und})$
$(\texttt{und}, \texttt{out}, \texttt{out})$	$(\texttt{out}, \texttt{und}, \texttt{out}) = (\texttt{und}, \texttt{out}, \texttt{out}) \circ (\texttt{out}, \texttt{in}, \texttt{und})$
$(\texttt{und}, \texttt{und}, \texttt{in})$	$(\texttt{und}, \texttt{und}, \texttt{out}) = (\texttt{out}, \texttt{in}, \texttt{und}) \circ (\texttt{und}, \texttt{und}, \texttt{in})$

□

5. Impossibility of Flattening Weak Attacks

Various extensions of argumentation frameworks have been studied in the literature. One fruitful approach to studying such extensions is the flattening methodology, in which extensions of argumentation frameworks are mapped to standard argumentation frameworks through a flattening function that is faithful with respect to the semantics of the extended argumentation frameworks. In this section we show how the theory of label functions can be used prove impossibility results concerning flattenings of certain extensions of argumentation frameworks.

Multiple authors have considered extending argumentation frameworks with a support relation in addition to an attack relation. Frameworks with both an attack and a support relation are called *bipolar argumentation frameworks* (*BAFs*), and multiple ap-

proaches to formalizing their semantics have been studied in the literature, for example *deductive support* [10], *necessary support* [15] and *evidential support* [16]. We briefly sketch the deductive support approach. The intuitive meaning of a deductive support from argument a to argument b is that whenever a is accepted, b must be accepted too. The definition of argumentation semantics for AFs has been adapted to a definition of semantics of BAFs that formalize this intuitive interpretation of deductive support [10]. Later it was shown that the flattening function that replaces every deductive support from a to b by a pair of attacks, namely from b to an auxiliary argument $Z_{(a,b,)}$ and from $Z_{(a,b,)}$ to a, is faithful with respect to these semantics, i.e. that flattening a BAF to an AF and then applying a standard argumentation semantics to the resulting AF gives the same result as directly applying the corresponding deductive support semantics to the BAF [7].

In this section we will study an extension of argumentation frameworks with a weak attack relation. Note that for the formal definition of an extended framework, it is irrelevant whether the second relation that gets added to the standard attack relation is a relation of support or a second attack relation. This motivates the following definitions:

Definition 26. *A* two-relation framework *is a triple* $(\mathscr{A},\mathscr{R},\mathscr{T})$ *such that* $\mathscr{R} \subseteq \mathscr{A} \times \mathscr{A}$ *and* $\mathscr{T} \subseteq \mathscr{A} \times \mathscr{A}$.

Definition 27. *A* two-relation semantics *is a function* σ *that maps any two-relation framework* $B = (\mathscr{A},\mathscr{R},\mathscr{T})$ *to a set* $\sigma(B)$ *of labelings of B. The elements of* $\sigma(B)$ *are called* σ*-labelings of B.*

Definition 28. *Let* σ *be an argumentation semantics and let* σ' *be a two-relation semantics. We say that* σ' extends σ *iff for every two-relation framework* $B = (\mathscr{A},\mathscr{R},\mathscr{T})$ *with* $\mathscr{T} = \emptyset$, $\sigma'(B) = \sigma((\mathscr{A},\mathscr{R}))$.

The semantics that have been defined for BAFs with a deductive support relation (and also the semantics for a necessary support relation) extend the corresponding semantics of standard AFs. This fact directly follows from the fact that BAFs can be flattened to AFs in a way that is faithful with respect to these BAF semantics.

An important feature of the flattening of the deductive support relation that we informally sketched above is that every support between two arguments is flattened in an analogous way and the support relation does not have any additional (potentially nonlocal) effect on the attack relation of the resulting AF. This feature can be formalized as follows:

Definition 29. *Let* $B = (\mathscr{A},\mathscr{R},\mathscr{T})$ *be a two-relation framework, and let* $G = (\mathscr{A}',\mathscr{R}',i,o)$ *be an I/O AF. The* G-flattening of B *is the AF* $flat_G(B) = (\mathscr{A}^*,\mathscr{R}^*)$, *where* $\mathscr{A}^* := \mathscr{A} \cup \{(a,b,c) \mid (a,b) \in \mathscr{T}$ and $c \in \mathscr{A}' \setminus \{i,o\}\}$ *and* $\mathscr{R}^* := \mathscr{R} \cup \{((a,b,c),(a,b,c') \mid (a,b) \in \mathscr{T},(c,c') \in \mathscr{R}'$ and $c,c' \notin \{i,o\}\} \cup \{(a,(a,b,c)) \mid (a,b) \in \mathscr{T}$ and $(i,c) \in \mathscr{R}'\} \cup \{((a,b,c),a) \mid (a,b) \in \mathscr{T}$ and $(c,i) \in \mathscr{R}'\} \cup \{(b,(a,b,c)) \mid (a,b) \in \mathscr{T}$ and $(o,c) \in \mathscr{R}'\} \cup \{((a,b,c),b) \mid (a,b) \in \mathscr{T}$ and $(c,o) \in \mathscr{R}'\}$.

Definition 30. *Let* σ *be an argumentation semantics and let* σ' *be a two-relation semantics that extends* σ. *We say that* σ' admits a uniform local flattening w.r.t. σ *iff there exists an I/O AF G such that for every two-relation argumentation framework B,* $\sigma'(B) = \sigma(flat_G(B))$.

We now consider a way of interpreting two-relation frameworks in which the second relation is not a support relation, but rather a *weak attack* relation. The intention behind our notion of a weak attack is that when an argument a is weakly attacked by an argument b, one can accept a without being able to defend a against the weak attack from b, but that in all other respects (such as conflict-freeness), weak attacks behave like the standard attacks of abstract argumentation, which we from now on call *strong attacks* to distinguish them clearly from weak attacks. In the labeling-based approach this means that the local effect that weak attacks from arguments with certain labels have on other arguments is generally analogous to the local effect of strong attacks, with the only exception that an argument can be labeled `in` even though it is weakly attacked by an argument labeled `und`. So we need the following adaptation of Definition 4 (the abbreviation "s/w" stands for "strong/weak"):

Definition 31. *Let $B = (\mathscr{A}, \mathscr{R}, \mathscr{T})$ be a two-relation framework, and let* `Lab` *be a labeling of B.*

- *An argument $a \in \mathscr{A}$ is called* s/w-legally `in` w.r.t. `Lab` *iff every argument that strongly attacks a is labeled* `out` *by* `Lab` *and every argument that weakly attacks a is labeled either* `out` *or* `und`.
- *An argument $a \in \mathscr{A}$ is called* s/w-legally `out` w.r.t. `Lab` *iff some argument that strongly or weakly attacks a is labeled* `in` *by* `Lab`.
- *An argument $a \in \mathscr{A}$ is called* s/w-legally `und` w.r.t. `Lab` *iff no argument that strongly or weakly attacks a is labeled* `in` *by* `Lab` *and some argument that strongly attacks a is labeled* `und` *by* `Lab`.

Now we define the semantics for two-relation frameworks with strong and weak attacks analogously as for standard AFs:

Definition 32. *Let $B = (\mathscr{A}, \mathscr{R}, \mathscr{T})$ be a two-relation framework, and let* `Lab` *be a labeling of B.*

- `Lab` *is an* s/w-complete labeling *of B iff every argument that* `Lab` *labels* `in` *is s/w-legally* `in` *w.r.t.* `Lab`*, every argument that* `Lab` *labels* `out` *is s/w-legally* `out` *w.r.t.* `Lab`*, and every argument that* `Lab` *labels* `und` *is s/w-legally* `und` *w.r.t.* `Lab`.
- `Lab` *is an* s/w-grounded labeling *of B iff* `Lab` *is an s/w-complete labeling of B in which the set of* `in`*-labeled arguments is minimal w.r.t. set inclusion.*
- `Lab` *is an* s/w-preferred labeling *of B iff* `Lab` *is an s/w-complete labeling of B in which the set of* `in`*-labeled arguments is maximal w.r.t. set inclusion.*

One can easily see that these three semantics extend the corresponding semantics of standard AFs.

The following theorem establishes that the weak attack relation cannot be flattened to the strong attack relation in a uniform local way:

Theorem 4. *Let $\sigma \in$* `cgp`*. Then s/w-σ does not admit a uniform local flattening w.r.t. σ.*

Proof. Suppose for a contradiction that s/w-σ does admit a uniform local flattening w.r.t. σ, i.e. there is an I/O AF G such that for every two-relation argumentation framework B, s/w-$\sigma(B) = \sigma(\mathit{flat}_G(B))$.

Consider the following three two-relation frameworks:

$$B_{\mathtt{in}} := (\{i,o\},\emptyset,\{(i,o)\})$$
$$B_{\mathtt{out}} := (\{i,o,b\},\{(b,i)\},\{(i,o)\})$$
$$B_{\mathtt{und}} := (\{i,o\},\{(b,b),(b,i)\},\{(i,o)\})$$

From Definition 20, one can easily see that for $L \in \mathit{Labs}$, $F_{st}(G,L) = \mathit{flat}_G(B_L)$ (up to isomorphism; auxiliary arguments may have different names in the two frameworks). Now from Definition 32, one can easily see that

$$\sigma(F_{st}(G,\mathtt{in})) = \text{s/w-}\sigma(B_{\mathtt{in}}) = \{\{(i,\mathtt{in}),(o,\mathtt{out})\}\},$$
$$\sigma(F_{st}(G,\mathtt{out})) = \text{s/w-}\sigma(B_{\mathtt{out}}) = \{\{(i,\mathtt{out}),(o,\mathtt{in}),(b,\mathtt{in})\}\}, \text{ and}$$
$$\sigma(F_{st}(G,\mathtt{und})) = \text{s/w-}\sigma(B_{\mathtt{und}}) = \{\{(i,\mathtt{und}),(o,\mathtt{in}),(b,\mathtt{und})\}\}.$$

So G represents $(\mathtt{out},\mathtt{in},\mathtt{in})$ w.r.t. σ, contradicting Theorem 3. □

6. Related Work

In the work of Baroni et al. [1], a similar methodology is introduced, where argumentation frameworks are partitioned, allowing for partitions to be evaluated locally. This local evaluation function needs to condition on the potential statuses of attackers from outside the partition, but does not need to consider the whole rest of the framework. From their results on decomposability of semantics, one could derive a result similar to our Theorem 1 but restricted to finite argumentation frameworks. We however chose to consider infinite argumentation frameworks as well in our work, as it grants more weight to the unrepresentability result derived in Section 4.

The work of Rienstra et al. [17] considers the partitioning of argumentation frameworks such that different semantics are applied to different partitions. In these cases, when evaluating the acceptance status of arguments within a partition, only the outside arguments which are the source of an attack targeting an argument inside that partition need to be considered, using a similar input/output methodology.

Enrichments of argumentation frameworks, such as the AFRA [3] and the BAF [9] have been interpreted in some cases using a flattening approach [7,6] which expresses higher-level relations in terms of auxiliary arguments and attacks, which can replace the original relation in a local fashion. Our results would prove useful when devising flattenings for existing or future enrichments, or showing no such flattening is possible.

7. Future Work

In future work, one could generalize the concept of a label function by dropping the requirement that the output argument always has the same label; these generalized label functions would therefore have a set of possible labels as their output value. Additionally one could drop the distinction between input argument and output argument, thus allow-

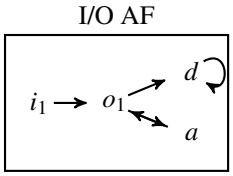

Figure 4. A semi-stable representation of the cgp-unrepresentable function $(\texttt{out}, \texttt{in}, \texttt{out})$.

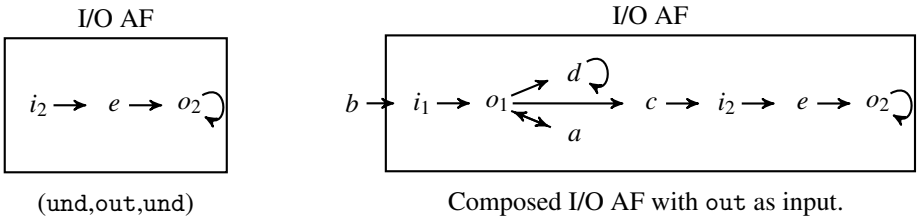

Figure 5. Failure of composition with semi-stable semantics. When the I/O AF from Figure 4 is composed with the I/O AF on the left, we obtained the I/O AF depicted on the right, which however does not produce a consistent label for o_2 when the input is out.

ing an external effect on both arguments and looking at the set of label pairs that these two arguments may take over the different extensions. This would yield to a generalized theory of binary relations between arguments that have a local effect expressible in the 3-label approach. While there are only 27 label functions, the number of such different relations between arguments is 2^{36}, so the classification according to their representability is likely to be much more complex. Such a classification would allow one to extend the impossibility result from Section 5 to other enrichments of abstract argumentation frameworks, or provide insights on how to flatten new enrichments.

Another line of future work would be to investigate the representability with respect to other semantics such as semi-stable [8], stage [19], stage2 [14], CF2 [4], and the more recent SCF2 [12] and weakly complete [5]. We briefly briefly present some preliminary findings for representability with respect to the semi-stable semantics.

The semi-stable semantics [8] has been often criticized for not satisfying a number of standard principles such as directionality [18]. There is however the interesting fact that some functions which are cgp-unrepresentable turn out to be semi-stable-representable.

Example 3. *Consider the I/O AF depicted in Figure 4. When the input argument i_1 is labeled* in*, the output argument o_1 is forced to be* out*, since it is directly attacked by i_1. In the case of* und *input, o_1 cannot be* in*, and by the nature of the semi-stable semantics which minimizes the* und *labeling, the only option is to have a being* in *and thus o_1 is again* out*. Lastly we consider the case that the input is* out*: While in this case the preferred semantics would give us two labelings with either o_1 or a being* in*, the semi-stable semantics will produce a single labeling with the o_1 labeled* in*, because this way the label* und *can be avoided completely, even for the self-attacking argument d. So this I/O AF represents the* $(\texttt{out}, \texttt{in}, \texttt{out})$ *function, which is* cgp*-unrepresentable.*

While the semi-stable semantics might be able to represent more functions, the fact that the semi-stable semantics does not satisfy the directionality principle (see [18])

brings about other issues, notably a lack of compositionality, as illustrated by a counter-example in Figure 5. On the left, we have an I/O AF which semi-stable-represents the function $(\texttt{und}, \texttt{out}, \texttt{und})$. When composed with the I/O AF from Figure 4, we obtain the framework on the right. In that composed framework, when the input is `out`, we now obtain two extensions, namely one where o_1 is `in`, and one where a is `in`. This second extension is now possible, because when o_1 is `in`, we have o_2 labeled `und`, while when a is `in`, we have d labeled `und`, so both of these options minimize the set of `und`-labeled arguments. Since these two labelings have different labels for o_2, this composed I/O AF does not represent any labeling function.

Since our methodology of characterising labeling functions via composition would not work in the case of semi-stable, we leave such an analysis for future work.

8. Conclusion

In this paper, we formally introduce labeling functions, and address the question of which functions are representable with an argumentation framework, focusing on the complete, grounded and preferred semantics, for which the labeling approach has been widely studied. We provide a proof that two representations of labeling functions can be composed to yield the composed labeling function, and use this finding to categorize the twenty seven label functions into eleven label functions that are representable and sixteen that are unrepresentable with respect to these three semantics. We also briefly investigate the case of the stable semantics, which is quite straightforward since it only allows for two different labels. We then discuss how the label function approach can be used to prove an impossibility result about the flattening approach for enrichments of abstract argumentation frameworks. We briefly investigate the case of the semi-stable semantics, as it allows for the representation of some functions which are not representable with respect to the other semantics. However due to the non-directional nature of the semantics, the composability result does not hold, hindering generalizations as done for the other semantics.

References

[1] Pietro Baroni, Guido Boella, Federico Cerutti, Massimiliano Giacomin, Leendert Van Der Torre, and Serena Villata. On the input/output behavior of argumentation frameworks. *Artificial Intelligence*, 217:144–197, 2014.

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

[3] Pietro Baroni, Federico Cerutti, Massimiliano Giacomin, and Giovanni Guida. Afra: Argumentation framework with recursive attacks. *International Journal of Approximate Reasoning*, 52(1):19–37, 2011.

[4] Pietro Baroni, Massimiliano Giacomin, and Giovanni Guida. SCC-recursiveness: a general schema for argumentation semantics. *Artificial Intelligence*, 168(1):162–210, 2005.

[5] Ringo Baumann, Gerhard Brewka, and Markus Ulbricht. Revisiting the foundations of abstract argumentation–semantics based on weak admissibility and weak defense. *Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence (2020)*, 2020.

[6] Guido Boella, Dov M Gabbay, Leendert van der Torre, and Serena Villata. Meta-argumentation modelling I: Methodology and techniques. *Studia Logica*, 93(2-3):297–355, 2009.

[7] Guido Boella, Dov M Gabbay, Leendert WN van der Torre, and Serena Villata. Support in Abstract Argumentation. *COMMA*, 216:111–122, 2010.

[8] Martin Caminada. Semi-stable semantics. In *Proceedings of the 2006 conference on Computational Models of Argument: Proceedings of COMMA 2006*, pages 121–130. IOS Press, 2006.

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

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

[11] Marcos Cramer and Jérémie Dauphin. Argumentation Label Functions - Technical Report. https://orbilu.uni.lu/bitstream/10993/43078/1/report.pdf. University of Luxembourg, 2020.

[12] Marcos Cramer and Leendert van der Torre. SCF2 – an argumentation semantics for rational human judgments on argument acceptability. *Proceedings of the 8th Workshop on Dynamics of Knowledge and Belief (DKB-2019) and the 7th Workshop KI and Kognition (KIK-2019)*, 2019.

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

[14] Wolfgang Dvořák and Sarah Alice Gaggl. Stage semantics and the scc-recursive schema for argumentation semantics. *Journal of Logic and Computation*, 26(4):1149–1202, 2014.

[15] Farid Nouioua and Vincent Risch. Argumentation frameworks with necessities. In *International Conference on Scalable Uncertainty Management*, pages 163–176. Springer, 2011.

[16] Nir Oren and Timothy J Norman. Semantics for evidence-based argumentation. *Computational Models of Argument*, 2008.

[17] Tjitze Rienstra, Alan Perotti, Serena Villata, Dov M Gabbay, and Leendert van der Torre. Multi-sorted argumentation. In *International Workshop on Theorie and Applications of Formal Argumentation*, pages 215–231. Springer, 2011.

[18] Leendert van der Torre and Srdjan Vesic. The principle-based approach to abstract argumentation semantics. In Pietro Baroni, Dov Gabbay, Massimiliano Giacomin, and Leendert van der Torre, editors, *Handbook of Formal Argumentation*. College Publications, 2018.

[19] Bart Verheij. Two Approaches to Dialectical Argumentation: Admissible Sets and Argumentation Stages. In *In Proceedings of the biannual International Conference on Formal and Applied Practical Reasoning (FAPR) workshop*, pages 357–368. Universiteit, 1996.