Proceedings, Fifteenth International Conference on
Principles of Knowledge Representation and Reasoning (KR 2016)

On the Functional Completeness of Argumentation Semantics

Massimiliano Giacomin
Department of Information Engineering
University of Brescia, Italy
massimiliano.giacomin@unibs.it

Thomas Linsbichler and **Stefan Woltran**
Institute of Information Systems
TU Wien, Vienna, Austria
{linsbich,woltran}@dbai.tuwien.ac.at

Abstract

Abstract argumentation frameworks (AFs) are one of the central formalisms in AI; equipped with a wide range of semantics, they have proven useful in several application domains. We contribute to the systematic analysis of semantics for AFs by connecting two recent lines of research – the work on input/output frameworks and the study of the expressiveness of semantics. We do so by considering the following question: given a function describing an input/output behaviour by mapping extensions (resp. labellings) to sets of extensions (resp. labellings), is there an AF with designated input and output arguments realizing this function under a given semantics? For the major semantics we give exact characterizations of the functions which are realizable in this manner.

1 Introduction

Dung's argumentation frameworks (AFs) have been extensively investigated, mainly because they represent an abstract model unifying a large variety of specific formalisms ranging from nonmonotonic reasoning to logic programming and game theory (Dung 1995). After the development and analysis of different semantics (Verheij 1996; Caminada, Carnielli, and Dunne 2012; Baroni, Caminada, and Giacomin 2011), recent attention has been drawn to their expressive power, i.e. determining which sets of extensions (Dunne et al. 2015) and labellings (Dyrkolbotn 2014) can be enforced in a single AF under a given semantics. Such results have recently been facilitated in order to express AGM-based revision in the context of abstract argumentation (Diller et al. 2015).

In (Baroni et al. 2014), it has been shown that an AF can be viewed as a set of partial interacting sub-frameworks each characterized by an input/output behavior, i.e. a semantics-dependent function which maps each labelling of the "input" arguments (the external arguments affecting the sub-framework) into the set of labellings prescribed for the "output" arguments (the arguments of the sub-framework affecting the external ones). It turns out that under the major semantics, i.e. complete, grounded, stable and (under some mild conditions) preferred semantics, sub-frameworks with the same input/output behavior can be safely exchanged, i.e. replacing a sub-framework with an equivalent one does not affect the justification status of the invariant arguments: semantics of this kind are called *transparent* (Baroni et al. 2014).

As a simple example, consider an argumentation framework including a chain of 4 arguments $a_1, \dots, a_4$ where for $i \in \{2, 3, 4\}$, a_{i-1} attacks a_i and a_i does not receive other attacks, and a_1 is unattacked. This chain can be seen as a sub-framework with input argument a_1 and output argument a_4, which under any transparent semantics can be replaced with any even-length chain without affecting the justification status of the arguments outside the sub-framework.

Then, somewhat resembling functional completeness of a specific set of logic gates, a natural question concerns the expressive power of transparent semantics in the context of an interacting sub-framework: given a so-called *I/O specification*, i.e. a function describing an input/output behaviour by mapping extensions (resp. labellings) to sets of extensions (resp. labellings), is there an AF with designated input and output arguments realizing this function under a given semantics?

Turning to the example above, the sub-framework including the 4-length chain *realizes* the mapping where $\{a_1\}$ is mapped to $\emptyset$ (i.e. if a_1 belongs to an extension then a_4 does not belong to it) and $\emptyset$ is mapped to $\{a_4\}$ (i.e. if a_1 does not belong to an extension then a_4 belongs to it): we call this kind of mapping a two-valued I/O specification. On the other hand, one may want to distinguish between *out* arguments (i.e. attacked by an extension) and *undecided* arguments (i.e. neither belonging to the extension nor attacked by it). Considering the sub-framework above, if a_1 is accepted then a_4 is *out*, if a_1 is *out* then a_4 is accepted, if a_1 is undecided then a_4 is undecided too. We call this kind of mapping a three-valued I/O specification. As it will be shown in the following, not all three-valued I/O specifications are realizable, e.g. there is no sub-framework realizing the variant of the mapping above where a_1 undecided yields a_4 accepted.

In this paper, we answer the question of realizability as follows:

- For the stable, preferred, semi-stable, stage, complete, ideal, and grounded semantics we exactly characterize all realizable two-valued I/O specifications.

- For the preferred and grounded labellings we exactly characterize all realizable three-valued I/O specifications.

Answering this question is essential in many aspects. First, it adds to the analysis and *comparison of semantics* (see e.g. (Caminada and Amgoud 2007; Baroni and Giacomin 2007)), by providing an absolute characterization of their functional expressiveness, which holds independently of how the abstract argumentation framework is instantiated. Second, it lays foundations towards a theory of *dynamic and modular argumentation*. More specifically, a functional characterization provides a common ground for different representations of the same sub-framework, as in metalevel argumentation (Modgil and Bench-Capon 2011) where meta-level arguments making claims about object-level arguments allow for equivalent characterizations of the same framework at different levels of abstraction, or to devise a summarized version of a sub-framework in order to simplify a given argumentation framework. One may also translate a different formalism to an AF or vice versa, e.g. to express a logical system as an AF or provide an argument-free representation of a given AF for human/computer interaction issues. In all of these cases, it is important to know whether an input/output behavior is realizable under a given argumentation semantics. Finally, our results are important in the dynamic setting of *strategic argumentation*, where a player may exploit the fact that for some set of arguments certain labellings are achievable (or non achievable) independently of the labelling of other arguments, or more generally she/he may exploit knowledge on the set of realizable dependencies. For example, an agent may desire to achieve some goal, i.e., ensure that a certain argument is justified. Considering arguments brought up by other agents as input arguments, our results enable the agent to verify whether the goal is achievable and provide one particular way for the agent to bring up further arguments in order to succeed.

The paper is organized as follows. After providing the necessary background in Section 2, Section 3 introduces the notion of an I/O-gadget to represent a sub-framework, and tackles the above problem with extension-based two-valued specifications. Labelling-based three-valued specifications are investigated in Section 4 while Section 5 considers partial specifications where for some inputs the output is not specified. Section 6 concludes the paper.

2 Background

We assume a countably infinite domain of arguments $\mathfrak{A}$. An *argumentation framework* (AF) is a pair $F = (A, R)$ where $A \subseteq \mathfrak{A}$ and $R \subseteq A \times A$. We assume that A is non-empty and finite. For an AF $F = (A, R)$ and a set of arguments $S \subseteq A$, we define $S^+_F = \{a \in A \mid \exists s \in S : (s, a) \in R\}$, $S^\oplus_F = S \cup S^+_F$, and $S^-_F = \{a \in A \mid \exists s \in S : (a, s) \in R\}$.

Given $F = (A, R)$, a set $S \subseteq A$ is *conflict-free* (in F), if there are no arguments $a, b \in S : (a, b) \in R$. An argument $a \in A$ is *defended* (in F) by a set $S \subseteq A$ if $\forall b \in A : (b, a) \in R \Rightarrow b \in S^+_F$. A set $S \subseteq A$ is *admissible* (in F) if it is conflict-free and defends all of its elements. We denote the set of conflict-free and admissible sets in F as $\mathit{cf}(F)$ and $\mathit{ad}(F)$, respectively.

An extension-based *semantics* σ associates to any $F = (A, R)$ the (possibly empty) set $\sigma(F) \subseteq 2^A$ of subsets of A called σ-extensions, where 2^A denotes the powerset of A. In this paper we focus on complete, grounded, preferred, ideal, stable, stage and semi-stable semantics, with extensions defined as follows:

- $S \in \mathit{co}(F)$ iff $S \in \mathit{ad}(F)$ and $a \in S$ for all $a \in A$ defended by S;
- $S \in \mathit{gr}(F)$ iff S is the least element in $\mathit{co}(F)$;
- $S \in \mathit{pr}(F)$ iff $S \in \mathit{ad}(F)$ and $\nexists T \in \mathit{ad}(F)$ s.t. $T \supset S$;
- $S \in \mathit{id}(F)$ iff $S \in \mathit{ad}(F)$, $S \subseteq \bigcap \mathit{pr}(F)$ and $\nexists T \in \mathit{ad}(F)$ s.t. $T \subseteq \bigcap \mathit{pr}(F)$ and $T \supset S$;
- $S \in \mathit{st}(F)$ iff $S \in \mathit{cf}(F)$ and $S^\oplus_F = A$;
- $S \in \mathit{sg}(F)$ iff $S \in \mathit{cf}(F)$ and $\nexists T \in \mathit{cf}(F)$ s.t. $T^\oplus_F \supset S^\oplus_F$;
- $S \in \mathit{se}(F)$ iff $S \in \mathit{ad}(F)$ and $\nexists T \in \mathit{ad}(F)$ s.t. $T^\oplus_F \supset S^\oplus_F$.

Given $F = (A, R)$ and a set $O \subseteq A$, the restriction of $\sigma(F)$ to O, denoted as $\sigma(F)|_O$, is the set $\{E \cap O \mid E \in \sigma(F)\}$.

Given a set of arguments A, a *labelling* L is a function assigning each argument $a \in A$ exactly one label among $\mathbf{t}$, $\mathbf{f}$ and $\mathbf{u}$, i.e. $L : A \mapsto \{\mathbf{t}, \mathbf{f}, \mathbf{u}\}$. If the arguments $A = \{a_1, \dots, a_n\}$ are ordered, then we denote a labelling of A as a sequence of labels, e.g. the labelling $\mathbf{tuf}$ of arguments $\{a_1, a_2, a_3\}$ maps a_1 to $\mathbf{t}$, a_2 to $\mathbf{u}$, and a_3 to $\mathbf{f}$. We denote the set of all possible labellings of A as $\mathcal{L}(A)$. Likewise, given an AF F, we denote the set of all possible labellings of (the arguments of) F as $\mathcal{L}(F)$. Given a labelling L and an argument a, $L(a)$ denotes the labelling of a wrt. L; finally $\mathit{in}(L)$, $\mathit{out}(L)$, and $\mathit{undec}(L)$ denotes the arguments labeled to $\mathbf{t}$, $\mathbf{f}$, and $\mathbf{u}$ by L, respectively. By $\neg L$ we denote the inverse labelling of L, i.e. $\mathit{in}(\neg L) = \mathit{out}(L)$, $\mathit{out}(\neg L) = \mathit{in}(L)$, and $\mathit{undec}(\neg L) = \mathit{undec}(L)$.

Definition 1. Given a set of arguments A and labellings L_1, L_2 thereof, $L_1 \sqsubseteq L_2$ iff $\mathit{in}(L_1) \subseteq \mathit{in}(L_2)$ and $\mathit{out}(L_1) \subseteq \mathit{out}(L_2)$. As usual, $L_1 \sqsubset L_2$ holds iff $L_1 \sqsubseteq L_2$ but $L_2 \not\sqsubseteq L_1$. Moreover, we call L_1 and L_2

- *comparable* if $L_1 \sqsubseteq L_2$ or $L_2 \sqsubseteq L_1$
- *compatible* if $\mathit{in}(L_1) \cap \mathit{out}(L_2) = \mathit{out}(L_1) \cap \mathit{in}(L_2) = \emptyset$.

Note that if L_1 and L_2 are comparable then they are also compatible, while the reverse does not hold in general.

An argumentation semantics can be defined in terms of labellings rather than of extensions, i.e. the labelling-based version of a semantics σ associates to F a set $\mathcal{L}_\sigma(F) \subseteq \mathcal{L}(F)$, where any labelling $L \in \mathcal{L}_\sigma(F)$ corresponds to an extension $S \in \sigma(F)$ as follows: an argument $a \in A$ is labeled to $\mathbf{t}$ iff $a \in S$, is labeled to $\mathbf{f}$ iff $a \in S^+_F$, is labeled to $\mathbf{u}$ if neither of the above conditions holds. Given F and a set $O \subseteq A$, the restriction of $\mathcal{L}_\sigma(F)$ to O, denoted as $\mathcal{L}_\sigma(F)|_O$, is the set $\{L \cap (O \times \{\mathbf{t}, \mathbf{f}, \mathbf{u}\}) \mid L \in \mathcal{L}_\sigma(F)\}$.

The following well-known result can be deduced e.g. from the semantics account given in (Caminada and Gabbay 2009).

Proposition 1. *For an* AF F, $\mathcal{L}_{\mathit{pr}}(F) = \max_\sqsubseteq(\mathcal{L}_{\mathit{co}}(F))$.

3 Extension-based I/O-gadgets

An I/O-gadget represents a (partial) AF where two sets of arguments are identified as input and output arguments, respectively, with the restriction that input arguments do not have any ingoing attacks[1].

Definition 2. Given a set of input arguments $I \subseteq \mathfrak{A}$ and a set of output arguments $O \subseteq \mathfrak{A}$ with $I \cap O = \emptyset$, an *I/O-gadget* is an AF $F = (A, R)$ such that $I, O \subseteq A$ and $I_F^- = \emptyset$.

The injection of a set $J \subseteq I$ to an I/O-gadget F simulates the input J in the way that all arguments in J are accepted (none of them has ingoing attacks since F is an I/O-gadget) and all arguments in $(I \setminus J)$ are rejected (each of them is attacked by the newly introduced argument z, which has no ingoing attacks).

Definition 3. Given an I/O-gadget $F = (A, R)$ and a set of arguments $J \subseteq I$, the *injection* of J to F is the AF

$$\triangleright(F, J) = (A \cup \{z\}, R \cup \{(z, i) \mid i \in (I \setminus J)\}),$$

where z is a newly introduced argument.

An I/O-specification describes a desired input/output behaviour by assigning to each set of input arguments a set of sets of output arguments.

Definition 4. A *two-valued*[2] *I/O-specification* consists of two sets $I, O \subseteq \mathfrak{A}$ and a total[3] function $\mathfrak{f} : 2^I \mapsto 2^{2^O}$.

In order for an I/O-gadget F to satisfy $\mathfrak{f}$ under a semantics σ, the injection of each $J \subseteq I$ to F must have $\mathfrak{f}(J)$ as its σ-extensions restricted to the output arguments. So, informally, with input J applied the set of outputs under σ should be exactly $\mathfrak{f}(J)$.

Definition 5. Given $I, O \subseteq \mathfrak{A}$, a semantics σ and an I/O-specification $\mathfrak{f}$, the I/O-gadget F *satisfies* $\mathfrak{f}$ under σ iff $\forall J \subseteq I : \sigma(\triangleright(F, J))|_O = \mathfrak{f}(J)$.

The following example illustrates these basic concepts.

Example 1. Consider the sets $I = \{a, b\}$ and $O = \{c, d\}$. A possible I/O-specification is the function $\mathfrak{f} : 2^I \mapsto 2^{2^O}$ such that

$$\begin{aligned} \mathfrak{f}(\emptyset) &= \{\{d\}\} \\ \mathfrak{f}(\{a\}) &= \{\{c, d\}\} \\ \mathfrak{f}(\{b\}) &= \{\{c\}, \{d\}\} \\ \mathfrak{f}(\{a, b\}) &= \{\{c, d\}, \{c\}\} \end{aligned}$$

Considering, for instance, the case of input $\{b\}$, the intended meaning of $\mathfrak{f}$ is that if b is accepted and a is not, then either c or d, but not both, should be accepted. On the other hand, in the case of input $\{a\}$, i.e. a is accepted and b is not, both

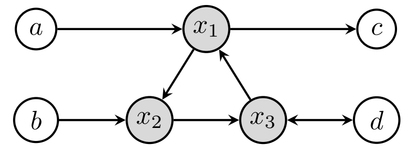

Figure 1: I/O-gadget with $I = \{a, b\}$ and $O = \{c, d\}$ satisfying the I/O-specification given in Example 1 under $\{st, pr, se, sg\}$.

c and d should be accepted. The AF F in Figure 1, where nodes and edges stand for arguments and attacks, respectively, represents an I/O-gadget. It turns out that F satisfies $\mathfrak{f}$ under stable semantics. In order to show this we have to check, for each $J \subseteq I$, whether the injection of J to F, $\triangleright(F, J)$, has exactly $\mathfrak{f}(J)$ as stable extensions restricted to O, i.e. $st(\triangleright(F, J))|_O = \mathfrak{f}(J)$. Considering $J = \emptyset$, we have that $\triangleright(F, \emptyset)$ is F together with a new argument z attacking both a and b. Hence we have $st(\triangleright(F, \emptyset)) = \{\{z, x_1, d\}\}$, meaning that $st(\triangleright(F, \emptyset))|_O = \{\{d\}\}$, which is as specified by $\mathfrak{f}$. Moreover we get $st(\triangleright(F, \{a\})) = \{\{z, a, x_2, c, d\}\}$, $st(\triangleright(F, \{b\})) = \{\{z, b, x_3, c\}, \{z, b, x_1, d\}\}$, and $st(\triangleright(F, \{a, b\})) = \{\{z, a, b, c, d\}, \{z, a, b, x_3, c\}\}$. This shows that for all possible inputs, the extensions restricted to the output-arguments are as specified by $\mathfrak{f}$, hence F satisfies $\mathfrak{f}$ under stable semantics.

While it is easy to verify that F also satisfies $\mathfrak{f}$ under preferred, semi-stable and stage semantics, it does not satisfy $\mathfrak{f}$ under grounded, ideal and complete semantics. For $J = \{b\}$ we have that $gr(\triangleright(F, \emptyset))|_O = id(\triangleright(F, \emptyset))|_O = \{\emptyset\}$ and $co(\triangleright(F, \emptyset))|_O = \{\emptyset, \{c\}, \{d\}\}$, being not in line with $\mathfrak{f}$.

The question we want to address is the following: which conditions must $\mathfrak{f}$ fulfill in order to be satisfiable by some I/O-gadget and how can such an I/O-gadget be constructed? The following generic gadget will be the key concept for the forthcoming characterization results.

Definition 6. Given an I/O-specification $\mathfrak{f}$, let $Y = \{y_i \mid i \in I\}$ and $X = \{x_J^S \mid J \subseteq I, S \in \mathfrak{f}(J)\}$. The *canonical I/O-gadget* (for $\mathfrak{f}$) is defined as

$$\begin{aligned} \mathfrak{C}(\mathfrak{f}) = (&I \cup O \cup Y \cup X \cup \{w\}, \\ &\{(i, y_i) \mid i \in I\} \cup \\ &\{(y_i, x_J^S) \mid x_J^S \in X, i \in J\} \cup \\ &\{(i, x_J^S) \mid x_J^S \in X, i \in (I \setminus J)\} \cup \\ &\{(x, x') \mid x, x' \in X, x \neq x'\} \cup \\ &\{(x, w) \mid x \in X\} \cup \{(w, w)\} \cup \\ &\{(x_J^S, o) \mid x_J^S \in X, o \in (O \setminus S)\}). \end{aligned}$$

Intuitively, the argument x_J^S shall enforce output S for input J. Moreover, w ensures that any stable extension of (an injection to) $\mathfrak{C}(\mathfrak{f})$ must contain an argument in X.

The following theorem shows that any I/O-specification is satisfiable under stable semantics.

Theorem 1. *Every I/O-specification $\mathfrak{f}$ is satisfied by $\mathfrak{C}(\mathfrak{f})$ under st.*

[1] Differently from an I/O-gadget, the notion of *argumentation multipole* in (Baroni et al. 2014) assumes a fixed set of incoming and outgoing attacks rather than of input and output arguments. However, for the purposes of the present paper the two notions are equivalent insofar as input (output) arguments of an I/O-gadget are identified with the sources (destinations) of incoming (outgoing) attacks of the corresponding argumentation multipole.

[2] In the following we omit this specification.

[3] The case of a partial function will be discussed in Section 5.

Proof. Let $I, O \subseteq \mathfrak{A}$ and $\mathfrak{f}$ be an arbitrary I/O-specification. We have to show that $st(\triangleright(\mathfrak{C}(\mathfrak{f}), J))|_O = \mathfrak{f}(J)$ holds for any $J \subseteq I$. Consider such a $J \subseteq I$.

First let $S \in \mathfrak{f}(J)$. We show that $E = \{z\} \cup J \cup \{y_i \mid i \in (I \setminus J)\} \cup \{x_J^S\} \cup S \in st(\triangleright(\mathfrak{C}(\mathfrak{f}), J))$, thus $S \in st(\triangleright(\mathfrak{C}(\mathfrak{f}), J))|_O$. E is conflict-free in $\triangleright(\mathfrak{C}(\mathfrak{f}), J)$ since z only attacks the arguments $(I \setminus J)$, a y_i with $i \in (I \setminus J)$ is only attacked by $i \notin E$, x_J^S is only attacked by other $x \in X$, $i \in (I \setminus J)$ and y_j with $j \in J$, and arguments in S are only attacked by arguments from X but not from x_J^S. E is stable in $\triangleright(\mathfrak{C}(\mathfrak{f}), J)$ since x_J^S attacks w, all other $x \in X$ and all $o \in (O \setminus S)$; z attacks all $i \in (I \setminus J)$; each y_j with $j \in J$ is attacked by j.

It remains to show that there is no $S' \in st(\triangleright(\mathfrak{C}(\mathfrak{f}), J))|_O$ with $S' \notin \mathfrak{f}(J)$. Towards a contradiction assume there is some $S' \in st(\triangleright(\mathfrak{C}(\mathfrak{f}), J))|_O$ with $S' \notin \mathfrak{f}(J)$. Hence there must be some $E' \in st(\triangleright(\mathfrak{C}(\mathfrak{f}), J))$ with $S' \subset E'$. Since w attacks itself, $w \notin E'$, thus by construction of $\mathfrak{C}(\mathfrak{f})$ there must be some $x_{J'}^{S'} \in (X \cap E')$ attacking w, and $x_{J'}^{S'}$ must attack all $o \in (O \setminus S')$. Since $S' \notin \mathfrak{f}(J)$ by assumption, it must hold that $J' \neq J$. Now note that $z \in E'$ and $j \in E'$ for all $j \in J$, since they are not attacked by construction of $\triangleright(\mathfrak{C}(\mathfrak{f}), J)$. Now if $J' \subset J$ then there is some $j \in (J \setminus J')$ attacking $x_{J'}^{S'}$, a contradiction to conflict-freeness of E'. On the other hand if $J' \not\subseteq J$ there is some $j' \in (J' \setminus J)$ which is attacked by z. Therefore also $y_{j'} \in E'$, which attacks $x_{J'}^{S'}$, again a contradiction. $\square$

As to preferred, semi-stable and stage semantics, any I/O-specification is satisfiable, provided that a (possibly empty) output is prescribed for any input.

Proposition 2. *Every I/O-specification $\mathfrak{f}$ such that $\forall J \subseteq I, \mathfrak{f}(J) \neq \emptyset$ is satisfied by $\mathfrak{C}(\mathfrak{f})$ under $\sigma \in \{pr, se, sg\}$.*

Proof (Sketch). Note that $\forall J \subseteq I$, stable, preferred, stage and semi-stable extensions coincide in $\triangleright(\mathfrak{C}(\mathfrak{f}), J)$, thus the result follows from Theorem 1. $\square$

Theorem 2. *An I/O-specification $\mathfrak{f}$ is satisfiable under $\sigma \in \{pr, se, sg\}$ iff $\forall J \subseteq I, \mathfrak{f}(J) \neq \emptyset$.*

Proof. The if-direction is a consequence of Proposition 2. The only-if-direction follows directly by the fact that in any AF, particularly in any injection of some set of arguments to an I/O-gadget, a σ-extension always exists. $\square$

Example 2. Consider the I/O-specification $\mathfrak{f}$ with $I = \{a, b\}$ and $O = \{c, d, e\}$ defined as follows:

$$\begin{aligned} \mathfrak{f}(\emptyset) &= \{\emptyset\} \\ \mathfrak{f}(\{a\}) &= \{\{c, e\}\} \\ \mathfrak{f}(\{b\}) &= \{\{c, d, e\}, \{d, e\}\} \\ \mathfrak{f}(\{a, b\}) &= \{\{c, d, e\}, \{c, d\}\} \end{aligned}$$

The canonical I/O-gadget $\mathfrak{C}(\mathfrak{f})$ is depicted in Figure 2 (without the dotted part)[4]. Let σ be a semantics in $\{st, pr, se, sg\}$. One can verify that for every

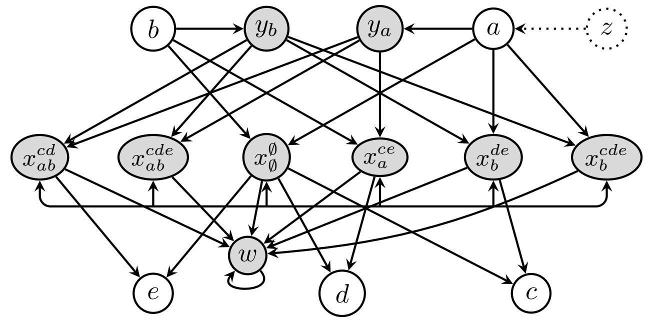

Figure 2: I/O-gadget satisfying the I/O-specification given in Example 2 under $\{st, pr, se, sg\}$. The dotted part illustrates the injection of $\{b\}$.

possible input $J \subseteq I$, the injection of J to $\mathfrak{C}(\mathfrak{f})$ has exactly $\mathfrak{f}(J)$ as σ-extensions restricted to O. As an example let $J = \{b\}$. $\triangleright(\mathfrak{C}(\mathfrak{f}), \{b\})$ adds to $\mathfrak{C}(\mathfrak{f})$ the argument z attacking a. Now $\sigma(\triangleright(\mathfrak{C}(\mathfrak{f}), \{b\})) = \{\{z, b, y_a, x_{\{b\}}^{\{d,e\}}, d, e\}, \{z, b, y_a, x_{\{b\}}^{\{c,d,e\}}, c, d, e\}\}$, hence $\sigma(\triangleright(\mathfrak{C}(\mathfrak{f}), \{b\}))|_O = \{\{d, e\}, \{c, d, e\}\} = \mathfrak{f}(\{b\})$.

Also for complete, grounded and ideal semantics we are able to identify a necessary and sufficient condition for satisfiability. While we show sufficiency of these conditions in more detail, their necessity is by the well-known facts that the intersection of all complete extensions is always a complete extension too and ideal and grounded semantics always yield exactly one extension. We define the former property for I/O-specifications:

Definition 7. An I/O-specification $\mathfrak{f}$ is *closed* iff for each $J \subseteq I$ it holds that $\mathfrak{f}(J) \neq \emptyset$ and $\bigcap \mathfrak{f}(J) \in \mathfrak{f}(J)$.

Example 3. Again considering the I/O-specification $\mathfrak{f}$ given in Example 2, we observe that $\mathfrak{f}$ is closed. For instance, $\bigcap \mathfrak{f}(\{b\}) = \{d, e\}$ and, indeed, $\{d, e\} \in \mathfrak{f}(\{b\})$.

Proposition 3. *Every closed I/O-specification $\mathfrak{f}$ is satisfied by $\mathfrak{C}(\mathfrak{f})$ under co.*

Proof. Let $J \subseteq I$. By construction of $\triangleright(\mathfrak{C}(\mathfrak{f}), J)$, $E^* = \{z\} \cup J \cup \{y_i \mid i \in (I \setminus J)\}$ is contained in all complete extensions, while the elements of $(I \setminus J) \cup \{y_i \mid i \in J\}$ are attacked by E^* and thus by all complete extensions. All $x_{J'}^{S'}$ with $J' \neq J$ are attacked by J or some y_i with $i \in (I \setminus J)$, thus they are attacked by E^*, while all x_J^S with $S \in \mathfrak{f}(J)$ attack each other, and the other attacks they receive come from elements attacked by E^*. Two cases can then be distinguished. If $|\mathfrak{f}(J)| = 1$ then by construction of $\triangleright(\mathfrak{C}(\mathfrak{f}), J)$ there is just one x_J^S defended by E^*, thus the only complete extension is $E^* \cup \{x_J^S\} \cup S$. If, on the other hand, $|\mathfrak{f}(J)| > 1$, any x_J^S with $S \in \mathfrak{f}(J)$ can be included, giving rise to the complete extension $E^* \cup \{x_J^S\} \cup S$, or none of x_J^S can be included, giving rise to the complete extension $E^* \cup \bigcap \mathfrak{f}(J)$ since an x_J^S attacks all $o \in (O \setminus S)$. Taking into account that $\bigcap \mathfrak{f}(J) \in \mathfrak{f}(J)$, in both cases we have that $co(\triangleright(\mathfrak{C}(\mathfrak{f}), J))|_O = \mathfrak{f}(J)$. $\square$

[4] Argument names such as $x_{\{a,b\}}^{\{c,d\}}$ are abbreviated by x_{ab}^{cd}.

Theorem 3. *An I/O-specification $\mathfrak{f}$ is satisfiable under co iff $\mathfrak{f}$ is closed.*

Proof. The if-direction is a consequence of Proposition 3. The only-if-direction follows directly by the fact that in any AF, particularly in any injection of some extension to an I/O-gadget, the intersection of all complete extensions is always a complete extension too. □

Proposition 4. *Every I/O-specification $\mathfrak{f}$ with $|\mathfrak{f}(J)| = 1$ for each $J \subseteq I$ is satisfied by $\mathfrak{C}(\mathfrak{f})$ under gr and id.*

Proof (Sketch). This follows the same idea as the proof of Proposition 3. Since $|\mathfrak{f}(J)| = 1$, $\mathfrak{f}(J)$ is closed and therefore, by Proposition 3, $co(\triangleright(\mathfrak{C}(\mathfrak{f}), J)) = \mathfrak{f}(J)$ for each $J \subseteq I$. By $|co(\triangleright(\mathfrak{C}(\mathfrak{f}), J))| = 1$ we get that complete, grounded and ideal extensions of $\triangleright(\mathfrak{C}(\mathfrak{f}), J)$ coincide, hence the result follows. □

Theorem 4. *An I/O-specification $\mathfrak{f}$ is satisfiable under gr and id iff $|\mathfrak{f}(J)| = 1$ for each $J \subseteq I$.*

Proof. The if-direction is a consequence of Proposition 4. The only-if-direction follows directly by the fact that in any AF, particularly in any injection of some extension to an I/O-gadget, the grounded and ideal extension are uniquely defined. □

4 Labelling-based I/O-gadgets

In the previous section we have dealt with I/O-specifications mapping extensions to sets of extensions. In general, there are two reasons why an argument does not belong to an extension, i.e. either because it is attacked by the extension (i.e. it is labelled $\mathbf{f}$ in a labelling-based version of the semantics) or because it is undecided due to insufficient justification (i.e. it is labelled $\mathbf{u}$). This distinction impacts on the justification status of arguments, since attacks from undecided arguments can prevent attacked arguments from belonging to an extension, while attacks from arguments labelled $\mathbf{f}$ are ineffective. In order to take into account this distinction, we first provide a labelling-based counterpart of the notions introduced in Section 3.

Definition 8. A 3-*valued I/O-specification* consists of two sets $I, O \subseteq \mathfrak{A}$ and a total function $\mathfrak{f} : \mathcal{L}(I) \mapsto 2^{\mathcal{L}(O)}$.

Definition 9. Given an I/O-gadget $F = (A, R)$ and a labelling L of I, the *labelling-injection* of L to F is the AF

$$\blacktriangleright(F, L) = (A \cup \{z\}, R \cup \{(z, a) \mid L(a) = \mathbf{f}\} \cup \{(b, b) \mid L(b) = \mathbf{u}\}),$$

where z is a newly introduced argument.

Definition 10. Given $I, O \subseteq \mathfrak{A}$, a semantics σ and a 3-valued I/O-specification $\mathfrak{f}$, the I/O-gadget F *satisfies* $\mathfrak{f}$ under σ iff $\forall L \in \mathcal{L}(I) : \mathcal{L}_\sigma(\blacktriangleright(F, L))|_O = \mathfrak{f}(L)$.

Example 4. A possible 3-valued I/O-specification for $I = \{a, b\}$ and $O = \{c, d\}$ is the function $\mathfrak{f} : \mathcal{L}(I) \mapsto 2^{\mathcal{L}(O)}$ such that:

$$\mathfrak{f}(\mathbf{uu}) = \{\mathbf{ut}\} \quad \mathfrak{f}(\mathbf{tu}) = \{\mathbf{tt}\} \quad \mathfrak{f}(\mathbf{ut}) = \{\mathbf{ut}, \mathbf{tf}\}$$
$$\mathfrak{f}(\mathbf{fu}) = \{\mathbf{ft}\} \quad \mathfrak{f}(\mathbf{uf}) = \{\mathbf{ut}\} \quad \mathfrak{f}(\mathbf{tt}) = \{\mathbf{tt}, \mathbf{tf}\}$$
$$\mathfrak{f}(\mathbf{tf}) = \{\mathbf{tt}\} \quad \mathfrak{f}(\mathbf{ft}) = \{\mathbf{tf}, \mathbf{ft}\} \quad \mathfrak{f}(\mathbf{ff}) = \{\mathbf{ft}\}$$

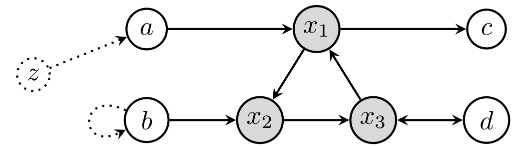

Figure 3: Labelling-injection of $\mathbf{fu}$ to the I/O-gadget from Figure 1.

For instance, setting a to $\mathbf{f}$ and b to $\mathbf{u}$ shall have the effect that c is set to $\mathbf{f}$ and d is set to $\mathbf{t}$. Setting both input-arguments to $\mathbf{t}$ shall have two possible outputs, namely one where both output-arguments are accepted and one with c accepted and d rejected. It turns out that the I/O-gadget F depicted in Figure 1 satisfies $\mathfrak{f}$ under preferred semantics. As an example, the labelling-injection of $\mathbf{fu}$ to F, i.e. $\blacktriangleright(F, \mathbf{fu})$, is depicted in Figure 3. Since $pr(\blacktriangleright(F, \mathbf{fu})) = \{\{z, x_1, d\}\}$ it indeed holds that $\mathcal{L}_{pr}(\blacktriangleright(F, \mathbf{fu}))|_O = \{\mathbf{ft}\} = \mathfrak{f}(\mathbf{fu})$.

By definition of the stable semantics it is clear that in order to be satisfied under *st*, a 3-valued I/O-specification must have empty output for all inputs including a $\mathbf{u}$-labelled argument and, as can be derived from the two-valued case, no output argument labelled to $\mathbf{u}$ for inputs with each argument labelled to $\mathbf{t}$ or $\mathbf{f}$.

Theorem 5. *A* 3-*valued I/O-specification $\mathfrak{f}$ is satisfiable under st iff for each $L \in \mathcal{L}(I)$ it holds that*

- *if $\exists i \in I : L(i) = \mathbf{u}$ then $\mathfrak{f}(L) = \emptyset$, and*
- *otherwise $K(o) \neq \mathbf{u}$ for all $K \in \mathfrak{f}(L)$ and $o \in O$.*

Proof. For the only-if-direction consider the case where $\exists i \in I : L(i) = \mathbf{u}$. Then for any I/O-gadget F, $\blacktriangleright(F, L)$ contains a self-attacking argument otherwise unattacked, hence $\mathcal{L}_{st}(\blacktriangleright(F, L)) = \emptyset$. In the other case, by definition of the stable semantics it is clear that each $o \in O$ must be labelled either $\mathbf{t}$ or $\mathbf{f}$ by any stable labelling.
For the if-direction we get that if $\exists i \in I : L(i) = \mathbf{u}$ then $st(\blacktriangleright(\mathfrak{C}(\mathfrak{f}), L)) = \emptyset$. Otherwise the labelling-injection coincides with the injection from the two-valued case and the result follows from Theorem 1. □

In order to characterize those 3-valued I/O-specifications which are satisfiable under the other semantics we need the following concept of monotonicity.

Definition 11. A 3-valued I/O-specification $\mathfrak{f}$ is *monotonic* iff for all L_1 and L_2 such that $L_1 \sqsubseteq L_2$ it holds that

$$\forall K_1 \in \mathfrak{f}(L_1)\, \exists K_2 \in \mathfrak{f}(L_2) : K_1 \sqsubseteq K_2.$$

The intuitive meaning of monotonicity is the following: if K_1 is an output for input L_1, then for every input which is more committed than L_1 there must be an output more committed than K_1.

Coming to necessary conditions for 3-valued I/O-specifications we start with a rather obvious observation:

Proposition 5. *For every* 3-*valued I/O-specification $\mathfrak{f}$ which is satisfiable under gr, $|\mathfrak{f}(L)| = 1$ for all $L \in \mathcal{L}(I)$.*

Monotonicity is a necessary condition for grounded and preferred semantics

Proposition 6. *Every 3-valued I/O-specification which is satisfiable under $\{gr, pr\}$ is monotonic.*

Proof. Let $\mathfrak{f}$ be a 3-valued I/O-specification and suppose it is satisfied by the I/O-gadget F under *gr* (resp. *pr*). Moreover let $L_1 \sqsubseteq L_2$ be labellings of I. Proposition 7 of (Baroni et al. 2014) says that $\forall K_2 \in \mathcal{L}_{co}(\blacktriangleright(F, L_2))|_O \; \exists K_1 \in \mathcal{L}_{co}(\blacktriangleright(F, L_1))|_O : K_1 \sqsubseteq K_2$ and $\forall K_1 \in \mathcal{L}_{co}(\blacktriangleright(F, L_1))|_O \; \exists K_2 \in \mathcal{L}_{co}(\blacktriangleright(F, L_2))|_O : K_1 \sqsubseteq K_2$. By the facts that $|\mathfrak{f}(L_1)|=|\mathfrak{f}(L_2)|=1$ (cf. Proposition 5) and $\mathcal{L}_{pr}(F) = \max_{\sqsubseteq}(\mathcal{L}_{co}(F))$ for each AF F (cf. Proposition 1), the result follows for *gr* and *pr*, respectively. □

In Propositions 5 and 6 we have given necessary conditions for 3-valued I/O-specifications to be satisfiable under *gr* and *pr*. In the following we show that these conditions are also sufficient in the sense that we can find a satisfying I/O-gadget. The constructions of these I/O-gadgets will depend on the given 3-valued I/O-specification and on the semantics, but they will share the same input- and output-part. The semantics-specific parts, denoted by $X_{\mathfrak{f}}^{\sigma}$ and $R_{\mathfrak{f}}^{\sigma}$ in the following definition, will be given later.

Definition 12. Given a 3-valued I/O-specification $\mathfrak{f}$ we define $I' = \{i' \mid i \in I\}$, $O' = \{o' \mid o \in O\}$, $R_I = \{(i, i') \mid i \in I\}$ and $R_O = \{(o', o'), (o', o) \mid o \in O\}$. The 3-*valued canonical I/O-gadget* for semantics σ and the 3-valued I/O-specification $\mathfrak{f}$ is defined as

$$\mathfrak{D}_{\mathfrak{f}}^{\sigma} = (I \cup I' \cup X_{\mathfrak{f}}^{\sigma} \cup O' \cup O, R_I \cup R_{\mathfrak{f}}^{\sigma} \cup R_O).$$

with $R_{\mathfrak{f}}^{\sigma} \subseteq ((I \cup I') \times X_{\mathfrak{f}}^{\sigma}) \cup (X_{\mathfrak{f}}^{\sigma} \times X_{\mathfrak{f}}^{\sigma}) \cup (X_{\mathfrak{f}}^{\sigma} \times (O' \cup O))$.

The semantics-independent part of $\mathfrak{D}_{\mathfrak{f}}^{\sigma}$ guarantees that the labelling of I coincides with the injected labelling and the labelling of I' is just the inverse.

Lemma 1. *Given a 3-valued I/O-specification $\mathfrak{f}$ and a semantics $\sigma \in \{gr, pr\}$ it holds for every $L \in \mathcal{L}(I)$ that*

$$\sigma(\blacktriangleright(\mathfrak{D}_{\mathfrak{f}}^{\sigma}, L))|_I = L, \text{and}$$
$$\sigma(\blacktriangleright(\mathfrak{D}_{\mathfrak{f}}^{\sigma}, L))|_{I'} = \neg L.$$

Proof. By the fact that arguments in $I \cup I'$ are not allowed to be attacked by the semantics-specific arguments $X_{\mathfrak{f}}^{\sigma}$, it follows that, in $\blacktriangleright(\mathfrak{D}_{\mathfrak{f}}^{\sigma}, L)$, an argument $a \in I$ is unattacked if $L(a) = \mathbf{t}$, attacked by the unattacked argument z if $L(a) = \mathbf{f}$, and self-attacking if $L(a) = \mathbf{u}$. Hence the result for I follows. The result for I' is then immediate by the fact that each $a' \in I'$ is only attacked by $a \in I$ and therefore has the inverse labelling of a. □

Now we turn to the semantics-specific constructions. For grounded semantics we need the concept of *determining* input labellings. An input labelling L is determining for output argument o if L is a minimal (w.r.t. $\sqsubseteq$) input labelling where o gets a concrete value ($\mathbf{t}$ or $\mathbf{f}$) according to $\mathfrak{f}$.

With abuse of notation, in the following we may identify a set including a single labelling with the labelling itself.

Definition 13. Given a 3-valued I/O-specification $\mathfrak{f}$ with $|\mathfrak{f}(L)| = 1$ for all $L \in \mathcal{L}(I)$ and an argument $o \in O$, a labelling L of I is *determining for o* (in $\mathfrak{f}$), if $\mathfrak{f}(L)(o) \neq \mathbf{u}$ and $\forall L' \sqsubset L : \mathfrak{f}(L')(o) = \mathbf{u}$. We denote the set of labellings which are determining for o (in $\mathfrak{f}$) as $\mathfrak{d}_{\mathfrak{f}}(o)$.

Note that for 3-valued I/O-specifications which are monotonic, two different labellings which are determining for a certain output argument cannot be comparable. The following example illustrates the concept of determining labellings.

Example 5. Let $\mathfrak{f}$ be the following 3-valued I/O-specification with $I = \{a, b\}$ and $O = \{c, d\}$:

$$\begin{array}{ccc} \mathfrak{f}(\mathbf{uu}) = \{\mathbf{uu}\} & \mathfrak{f}(\mathbf{tu}) = \{\mathbf{tu}\} & \mathfrak{f}(\mathbf{ut}) = \{\mathbf{ut}\} \\ \mathfrak{f}(\mathbf{uf}) = \{\mathbf{uf}\} & \mathfrak{f}(\mathbf{fu}) = \{\mathbf{uu}\} & \mathfrak{f}(\mathbf{tt}) = \{\mathbf{tt}\} \\ \mathfrak{f}(\mathbf{tf}) = \{\mathbf{tf}\} & \mathfrak{f}(\mathbf{ft}) = \{\mathbf{ut}\} & \mathfrak{f}(\mathbf{ff}) = \{\mathbf{tf}\} \end{array}$$

We have the following sets of determining labellings: $\mathfrak{d}_{\mathfrak{f}}(c) = \{\mathbf{tu}, \mathbf{ff}\}$ and $\mathfrak{d}_{\mathfrak{f}}(d) = \{\mathbf{ut}, \mathbf{uf}\}$. Consider, for instance, the input labelling $\mathbf{ff}$. We have $\mathfrak{f}(\mathbf{ff}) = \mathbf{tf}$. In order to check if $\mathbf{ff}$ is determining for c we have to look at all input labellings being less committed than $\mathbf{ff}$. Now we observe $\mathfrak{f}(\mathbf{uf}) = \mathbf{uf}$, $\mathfrak{f}(\mathbf{fu}) = \mathfrak{f}(\mathbf{uu}) = \mathbf{uu}$. In all of these desired output labellings c has value $\mathbf{u}$, so $\mathbf{ff}$ is determining for c. On the other hand $\mathbf{ff}$ is not determining for d, since $\mathfrak{f}(\mathbf{uf})(d) = \mathbf{f}$.

The semantics-specific construction for grounded semantics is defined as follows.

Definition 14. Given a 3-valued I/O-specification $\mathfrak{f}$ with $|\mathfrak{f}(L)=1|$ for all $L \in \mathcal{L}(I)$, the *gr*-specific part of $\mathfrak{D}_{\mathfrak{f}}^{gr}$ is given by

$$\begin{aligned} X_{\mathfrak{f}}^{gr} = &\{x_o^L \mid o \in O, L \in \mathfrak{d}_{\mathfrak{f}}(o)\}, \text{and} \\ R_{\mathfrak{f}}^{gr} = &\{(i, x_o^L) \mid x_o^L \in X_{\mathfrak{f}}^{gr}, L(i) = \mathbf{f}\} \cup \\ &\{(i', x_o^L) \mid x_o^L \in X_{\mathfrak{f}}^{gr}, L(i) = \mathbf{t}\} \cup \\ &\{(x_o^L, o') \mid x_o^L \in X_{\mathfrak{f}}^{gr}, \mathfrak{f}(L)(o) = \mathbf{t}\} \cup \\ &\{(x_o^L, o) \mid x_o^L \in X_{\mathfrak{f}}^{gr}, \mathfrak{f}(L)(o) = \mathbf{f}\}. \end{aligned}$$

For every $o \in O$ and each input-labelling L which is determining for o, there is the argument x_o^L. This argument can be labelled $\mathbf{t}$ if L is the labelling of I (recall Lemma 1) and intuitively enforces the labelling of o to be as given by $\mathfrak{f}(L)$.

The next results, requiring two preliminary lemmata, characterize satisfiability of grounded semantics.

Lemma 2. *Let $\mathfrak{f}$ be a 3-valued I/O-specification which is monotonic and s.t. $|\mathfrak{f}(L)| = 1$ for each $L \in \mathcal{L}(I)$. Moreover let $o \in O$ and $L, L' \in \mathcal{L}(I)$ be such that $\mathfrak{f}(L)(o) = \mathbf{t}$ and $\mathfrak{f}(L')(o) = \mathbf{f}$. Then L and L' are not compatible.*

Proof. Towards a contradiction assume that L and L' are compatible, and let $V \in \mathcal{L}(I)$ such that for each $i \in I$, $V(i) = \mathbf{t}$ iff $L(i) = \mathbf{t} \vee L'(i) = \mathbf{t}$, $V(i) = \mathbf{f}$ iff $L(i) = \mathbf{f} \vee L'(i) = \mathbf{f}$, $V(i) = \mathbf{u}$ iff $L(i) = L'(i) = \mathbf{u}$. It holds that $L \sqsubseteq V$, thus $\mathfrak{f}(V)(o) = \mathbf{t}$ since $\mathfrak{f}$ is monotonic. However, it also holds that $L' \sqsubseteq V$, thus $\mathfrak{f}(V)(o) = \mathbf{f}$, a contradiction. □

Lemma 3. *Given a 3-valued I/O-specification $\mathfrak{f}$ which is monotonic and s.t. $|\mathfrak{f}(L)| = 1$ for each $L \in \mathcal{L}(I)$, let $o \in O$ and $L, L' \in \mathcal{L}(I)$ be such that L is determining for o. Then $\mathcal{L}_{gr}(\blacktriangleright(\mathfrak{D}_{\mathfrak{f}}^{gr}, L'))(x_o^L)$ is*

1. $\mathbf{t}$ *if* $L \sqsubseteq L'$;

2. $\mathbf{f}$ *if L and L' are not compatible; and*
3. $\mathbf{u}$ *if L and L' are compatible but $L \not\sqsubseteq L'$.*

Proof. Let $G = \mathcal{L}_{gr}(\blacktriangleright(\mathfrak{D}^{gr}_{\mathfrak{f}}, L'))$ and note that, by Lemma 1, we know that $G|_I = L'$ and $G|_{I'} = \neg L'$. (1) If $L \sqsubseteq L'$ then all attackers of x^L_o are $\mathbf{f}$ in G, hence $G(x^L_o) = \mathbf{t}$. (2) If L and L' are not compatible then there is some $i \in I$ such that either $L(i) = \mathbf{t}$ and $L'(i) = \mathbf{f}$ or $L(i) = \mathbf{f}$ and $L'(i) = \mathbf{t}$. In the first case x^L_o is attacked by i' and $G(i') = \mathbf{t}$, in the second case x^L_o is attacked by i and $G(i) = \mathbf{t}$, both entailing $G(x^L_o) = \mathbf{f}$. (3) If L and L' are compatible then, by construction of $\mathfrak{D}^{gr}_{\mathfrak{f}}$ and Lemma 1, all attackers of x^L_o are either $\mathbf{f}$ or $\mathbf{u}$. Moreover, since $L \not\sqsubseteq L'$ there is some $i \in I$ with $L'(i) = \mathbf{u}$ and $L(i) \neq \mathbf{u}$. But then $G(i) = G(i') = \mathbf{u}$ and x^L_o is attacked by either i or i', hence $G(x^L_o) = \mathbf{u}$. ☐

Proposition 7. *Every 3-valued I/O-specification $\mathfrak{f}$ which is monotonic and s.t. $|\mathfrak{f}(L)| = 1$ for each $L \in \mathcal{L}(I)$, is satisfied by $\mathfrak{D}^{gr}_{\mathfrak{f}}$ under gr.*

Proof. Consider some input labelling L. We have to show $\mathcal{L}_{gr}(\blacktriangleright(\mathfrak{D}^{gr}_{\mathfrak{f}}, L))|_O = \mathfrak{f}(L)$. To this end let $o \in O$.

Assume $\mathfrak{f}(L)(o) = \mathbf{u}$. Then, since $\mathfrak{f}$ is monotonic, $\mathfrak{f}(L')(o) = \mathbf{u}$ for all $L' \sqsubseteq L$. Therefore, there is no $L' \sqsubseteq L$ with $L' \in \mathfrak{d}_{\mathfrak{f}}(o)$. By Lemma 3 we get that for all $L'' \in \mathfrak{d}_{\mathfrak{f}}(o)$ it holds that $\mathcal{L}_{gr}(\blacktriangleright(\mathfrak{D}^{gr}_{\mathfrak{f}}, L))(x^{L''}_o) \neq \mathbf{t}$. Since, by construction of $\mathfrak{D}^{gr}_{\mathfrak{f}}$, such $x^{L''}_o$ with $L'' \in \mathfrak{d}_{\mathfrak{f}}(o)$ are the only attackers of o and o', $\mathcal{L}_{gr}(\blacktriangleright(\mathfrak{D}^{gr}_{\mathfrak{f}}, L))(o) = \mathbf{u}$.

Next assume $\mathfrak{f}(L)(o) = \mathbf{t}$. Then there is some $L' \sqsubseteq L$ with $L' \in \mathfrak{d}_{\mathfrak{f}}(o)$ and $\mathfrak{f}(L')(o) = \mathbf{t}$. By Lemma 3 we get $\mathcal{L}_{gr}(\blacktriangleright(\mathfrak{D}^{gr}_{\mathfrak{f}}, L))(x^{L'}_o) = \mathbf{t}$. Moreover, $x^{L'}_o$ attacks o', hence $\mathcal{L}_{gr}(\blacktriangleright(\mathfrak{D}^{gr}_{\mathfrak{f}}, L))(o') = \mathbf{f}$. Towards a contradiction assume there is some $x^{L''}_o$ attacking o with $\mathcal{L}_{gr}(\blacktriangleright(\mathfrak{D}^{gr}_{\mathfrak{f}}, L))(x^{L''}_o) \in \{\mathbf{t}, \mathbf{u}\}$. Then, by Lemma 3, L'' and L are compatible and as $L' \sqsubseteq L$ also L'' and L' are compatible. However, $x^{L''}_o$ attacking o and $x^{L'}_o$ attacking o' means, by construction of $\mathfrak{D}^{gr}_{\mathfrak{f}}$, $\mathfrak{f}(L'')(o) = \mathbf{f}$ and $\mathfrak{f}(L')(o) = \mathbf{t}$, respectively. But then, by Lemma 2, L'' and L' are not compatible, a contradiction. Therefore all attackers of o are labelled $\mathbf{f}$ by $\mathcal{L}_{gr}(\blacktriangleright(\mathfrak{D}^{gr}_{\mathfrak{f}}, L))$, hence $\mathcal{L}_{gr}(\blacktriangleright(\mathfrak{D}^{gr}_{\mathfrak{f}}, L))(o) = \mathbf{t}$.

Finally assume $\mathfrak{f}(L)(o) = \mathbf{f}$. Then there is some $L' \sqsubseteq L$ with $L' \in \mathfrak{d}_{\mathfrak{f}}(o)$ and $\mathfrak{f}(L')(o) = \mathbf{f}$. By Lemma 3 we get $\mathcal{L}_{gr}(\blacktriangleright(\mathfrak{D}^{gr}_{\mathfrak{f}}, L))(x^{L'}_o) = \mathbf{t}$. Moreover, $x^{L'}_o$ attacks o, hence $\mathcal{L}_{gr}(\blacktriangleright(\mathfrak{D}^{gr}_{\mathfrak{f}}, L))(o) = \mathbf{f}$. ☐

Example 6. Again consider the 3-valued I/O-specification $\mathfrak{f}$ from Example 5. We have seen the determining labellings there. The I/O-gadget $\mathfrak{D}^{gr}_{\mathfrak{f}}$ is depicted in Figure 4. Consider for example the labelling-injection of $\mathbf{fu}$ to $\mathfrak{D}^{gr}_{\mathfrak{f}}$, which is indicated by the dotted part of the figure. We get $gr(\blacktriangleright(\mathfrak{D}^{gr}_{\mathfrak{f}}, \mathbf{fu})) = \{z, a'\}$, hence $\mathcal{L}_{gr}(\blacktriangleright(\mathfrak{D}^{gr}_{\mathfrak{f}}, \mathbf{fu}))|_O = \mathbf{uu}$, satisfying $\mathfrak{f}$. One can check that this holds for all possible labelling-injections, hence $\mathfrak{D}^{gr}_{\mathfrak{f}}$ satisfies $\mathfrak{f}$ under the grounded semantics.

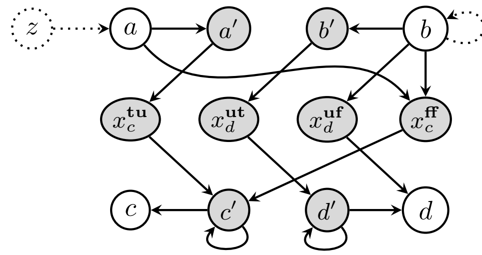

Figure 4: I/O-gadget satisfying the 3-valued I/O-specification $\mathfrak{f}$ given in Example 5 under *gr*, as discussed in Example 6. The dotted part shows the injection of $\mathbf{fu}$.

Theorem 6. *A 3-valued I/O-specification $\mathfrak{f}$ is satisfiable under gr iff $\mathfrak{f}$ is monotonic and for each $L \in \mathcal{L}(I)$, $|\mathfrak{f}(L)|$=1.*

Proof. The if-direction is a direct consequence of Proposition 7. The only-if-direction follows by Propositions 5 and 6. ☐

Now we present the part of the 3-valued canonical I/O-gadget which is specific to the preferred semantics.

Definition 15. Given a 3-valued I/O-specification $\mathfrak{f}$, the *pr*-specific part of $\mathfrak{D}^{pr}_{\mathfrak{f}}$ is given by

$$
\begin{aligned}
X^{pr}_{\mathfrak{f}} =& \{x^L_K \mid L \in \mathcal{L}(I), K \in \mathfrak{f}(L)\}, \text{ and} \\
R^{pr}_{\mathfrak{f}} =& \{(i, x^L_K) \mid x^L_K \in X^{pr}_{\mathfrak{f}}, L(i) = \mathbf{f}\} \cup \\
& \{(i', x^L_K) \mid x^L_K \in X^{pr}_{\mathfrak{f}}, L(i) = \mathbf{t}\} \cup \\
& \{(x^L_K, o') \mid x^L_K \in X^{pr}_{\mathfrak{f}}, K(o) = \mathbf{t}\} \cup \\
& \{(x^L_K, o) \mid x^L_K \in X^{pr}_{\mathfrak{f}}, K(o) = \mathbf{f}\} \cup \\
& \{(x^L_K, x^{L'}_{K'}) \mid \neg(L \sqsubset L' \wedge K \sqsubseteq K') \wedge \\
& \qquad \neg(L' \sqsubset L \wedge K' \sqsubseteq K)\}.
\end{aligned}
$$

Every input-output-combination is represented by an argument x^L_K in $\mathfrak{D}^{pr}_{\mathfrak{f}}$. The way the input-arguments are linked to x^L_K makes sure that, with input-labelling L, x^L_K is not attacked by any argument among $I \cup I'$ which can be $\mathbf{t}$ in a preferred labelling (cf. Lemma 1). Therefore each such argument x^L_K can act as a representative for a preferred labelling, enforcing output-labelling K. We first show a technical lemma, giving sufficient conditions on the labelling-status of the arguments in $X^{pr}_{\mathfrak{f}}$ to get the desired labelling of the output arguments.

Lemma 4. *Given a 3-valued I/O-specification $\mathfrak{f}$ and an input labelling $L \in \mathcal{L}(I)$, it holds for each preferred labelling $P \in pr(\blacktriangleright(\mathfrak{D}^{pr}_{\mathfrak{f}}, L))$ that $P|_O = K$ if*

- *$P(x^L_K) = \mathbf{t}$ and*
- *for all $x^{L'}_{K'} \in X^{pr}_{\mathfrak{f}}$,*
  - *$K \sqsubset K'$ implies $P(x^{L'}_{K'}) \neq \mathbf{t}$ and*
  - *K and K' not comparable implies $P(x^{L'}_{K'}) = \mathbf{f}$.*

Proof. Consider some $K \in \mathfrak{f}(L)$ and an arbitrary $o \in O$. We show that $P(o) = K(o)$.

First assume $K(o) = \mathbf{u}$. By the hypothesis $P(x^{L'}_{K'}) \neq \mathbf{t}$ for all $K' \not\sqsubseteq K$. Moreover, for all $K' \sqsubseteq K$ we have that $K'(o) = \mathbf{u}$ since $K(o) = \mathbf{u}$, thus by construction of $\mathfrak{D}^{pr}_{\mathfrak{f}}$ $x^{L'}_{K'}$ does not attack either o or o'. Summing up, neither o nor o' is attacked by an argument which is $\mathbf{t}$ in P. Hence $P(o) = \mathbf{u}$.

Next let $K(o) = \mathbf{t}$. Since $P(x^L_K) = \mathbf{t}$ we must have that $P(o') = \mathbf{f}$. Besides that, o is attacked by all $x^{L'}_{K'}$ with $K'(o) = \mathbf{f}$. But this means that K and K' are not comparable, hence $P(x^{L'}_{K'}) = \mathbf{f}$ by assumption. Now we know that all attackers of o are $\mathbf{f}$ in P, therefore $P(o) = \mathbf{t}$.

Finally let $K(o) = \mathbf{f}$. Since $P(x^L_K) = \mathbf{t}$ and x^L_K attacks o we get that $P(o) = \mathbf{f}$. □

We proceed by showing that every monotonic function $\mathfrak{f}$ is satisfied by $\mathfrak{D}^{pr}_{\mathfrak{f}}$ under the preferred semantics.

Proposition 8. *Every* 3*-valued* I/O*-specification* $\mathfrak{f}$ *which is monotonic is satisfied by* $\mathfrak{D}^{pr}_{\mathfrak{f}}$ *under pr.*

Proof. Consider an arbitrary input labelling $L \in \mathcal{L}(I)$. We have to show that $\mathcal{L}_{pr}(\blacktriangleright(\mathfrak{D}^{pr}_{\mathfrak{f}}, L))|_O = \mathfrak{f}(L)$.

By construction of $\mathfrak{D}^{pr}_{\mathfrak{f}}$, those $x^{L'}_{K'} \in X^{pr}_{\mathfrak{f}}$ with $L' \sqsubseteq L$ are the only arguments in $X^{pr}_{\mathfrak{f}}$ which can be $\mathbf{t}$ in a preferred labelling of $\blacktriangleright(\mathfrak{D}^{pr}_{\mathfrak{f}}, L)$, since their attackers in $I \cup I'$ are all $\mathbf{f}$, while the other arguments in $X^{pr}_{\mathfrak{f}}$ are attacked by an argument $\mathbf{t}$ of $I \cup I'$. Now the arguments x^L_K with $K \in \mathfrak{f}(L)$ form a clique in $\blacktriangleright(\mathfrak{D}^{pr}_{\mathfrak{f}}, L)$. Moreover each of these x^L_K defends itself, hence there is a preferred labelling of $\blacktriangleright(\mathfrak{D}^{pr}_{\mathfrak{f}}, L)$ for each $K \in \mathfrak{f}(L)$ identified by x^L_K. Let P_K be the preferred labelling with $P_K(x^L_K) = \mathbf{t}$ where $K \in \mathfrak{f}(L)$. All $x^{L'}_{K'}$ with $K' \not\sqsubseteq K \wedge K \not\sqsubseteq K'$ are attacked by x^K_L, hence $P_K(x^{L'}_{K'}) = \mathbf{f}$. Assume $K \sqsubset K'$. If $L \not\sqsubset L'$, then $x^{L'}_{K'}$ is again attacked by x^K_L and $P_K(x^{L'}_{K'}) = \mathbf{f}$. If $L \sqsubset L'$, $P_K(x^{L'}_{K'}) \neq \mathbf{t}$ since it is attacked by some i or i' ($i \in I$) which is $\mathbf{u}$ in P_K. Therefore, by Lemma 4, $P_K|_O = K$.

It remains to show that there is no other preferred labelling besides these P_K with $K \in \mathfrak{f}(L)$. Towards a contradiction, assume that there is a preferred labelling P' where no x^L_K with $K \in \mathfrak{f}(L)$ is $\mathbf{t}$. By our initial considerations, those $x^{L'}_{K'}$ with $L' \sqsubset L$ are the only arguments in $X^{pr}_{\mathfrak{f}}$ which can be $\mathbf{t}$ in P'. It cannot be the case that none of them is $\mathbf{t}$, since P' would not be preferred. Then there is at least one $x^{L'}_{K'}$ which is $\mathbf{t}$ in P', with $L' \sqsubset L$, and without loss of generality we can assume that there is no $x^{L''}_{K''}$ which is $\mathbf{t}$ and $L' \sqsubset L''$. Now, since $\mathfrak{f}$ is monotonic there has to be a $K \in \mathfrak{f}(L)$ such that $K' \sqsubseteq K$. We prove that no argument in $X^{pr}_{\mathfrak{f}}$ attacking x^L_K is $\mathbf{t}$ in P'.

First, the only arguments in $X^{pr}_{\mathfrak{f}}$ that can be $\mathbf{t}$ are those $x^{L''}_{K''}$ with $L'' \sqsubset L$. Note that, according to Definition 15, $x^{L'}_{K'}$ does not attack x^L_K, since $L' \sqsubset L \wedge K' \sqsubseteq K$. If an attacker $x^{L''}_{K''}$ is attacked in turn by $x^{L'}_{K'}$, then it is $\mathbf{f}$, otherwise either $L'' \sqsubset L' \wedge K'' \sqsubseteq K'$ or $L' \sqsubset L'' \wedge K' \sqsubseteq K''$. The first case is impossible, since we would have $L'' \sqsubset L \wedge K'' \sqsubseteq K$, entailing that $x^{L''}_{K''}$ does not attack x^L_K. In the other case, by the assumption on $x^{L'}_{K'}$ it holds that $x^{L''}_{K''}$ is not $\mathbf{t}$.

Now, x^L_K defends itself against all arguments in $X^{pr}_{\mathfrak{f}}$ and none of them is $\mathbf{t}$, moreover by construction of $\blacktriangleright(\mathfrak{D}^{pr}_{\mathfrak{f}}, L)$, all attackers from I and I' are $\mathbf{f}$. But then, consider the labelling P'' obtained from P' by assigning to x^L_K the label $\mathbf{t}$, and by assigning to all the attackers of x^L_K the label $\mathbf{f}$. P'' is admissible and $P' \sqsubseteq P''$, contradicting the maximality of P'. □

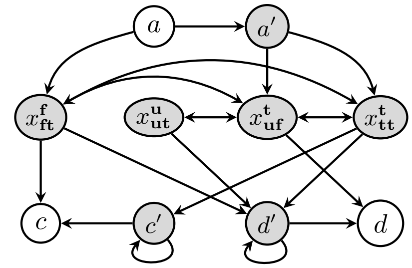

Figure 5: 3-valued canonical I/O-gadget $\mathfrak{D}^{pr}_{\mathfrak{f}}$ for the 3-valued I/O-specification $\mathfrak{f}$ given in Example 7.

Example 7. Consider the 3-valued I/O-specification for $I = \{a\}$ and $O = \{c, d\}$ given by

$$\mathfrak{f}(\mathbf{u}) = \{\mathbf{ut}\} \qquad \mathfrak{f}(\mathbf{t}) = \{\mathbf{tt}, \mathbf{uf}\} \qquad \mathfrak{f}(\mathbf{f}) = \{\mathbf{ft}\}.$$

It is easy to see that $\mathfrak{f}$ is monotonic, since $\mathbf{ut} \in \mathfrak{f}(\mathbf{u})$ has a successor wrt. $\sqsubseteq$ in both $\mathfrak{f}(\mathbf{t})$ and $\mathfrak{f}(\mathbf{f})$.

The AF in Figure 5 depicts the 3-valued canonical I/O-gadget $\mathfrak{D}^{pr}_{\mathfrak{f}}$. Observe the symmetric attacks between arguments $x^L_K, x^{L'}_{K'} \in X^{pr}_{\mathfrak{f}}$ whenever $L = L'$ ($x^{\mathbf{t}}_{\mathbf{uf}}$ and $x^{\mathbf{t}}_{\mathbf{tt}}$), L and L' are not comparable (e.g. $x^{\mathbf{f}}_{\mathbf{ft}}$ and $x^{\mathbf{t}}_{\mathbf{uf}}$), or $L \sqsubset L'$ but $K \not\sqsubseteq K'$ ($x^{\mathbf{u}}_{\mathbf{ut}}$ and $x^{\mathbf{t}}_{\mathbf{uf}}$). However, there is no attack if both $L \sqsubset L'$ and $K \sqsubseteq K'$ holds, as for instance between $x^{\mathbf{u}}_{\mathbf{ut}}$ and $x^{\mathbf{t}}_{\mathbf{tt}}$. To see the motivation behind this consider the injection of $\mathbf{t}$ to $\mathfrak{D}^{pr}_{\mathfrak{f}}$. We get $pr(\blacktriangleright(\mathfrak{D}^{pr}_{\mathfrak{f}}, \mathbf{t})) = \{\{z, a, x^{\mathbf{u}}_{\mathbf{ut}}, x^{\mathbf{t}}_{\mathbf{tt}}, c, d\}, \{z, a, x^{\mathbf{t}}_{\mathbf{uf}}\}\}$, giving rise to $pr(\blacktriangleright(\mathfrak{D}^{pr}_{\mathfrak{f}}, \mathbf{t}))|_O = \{\mathbf{tt}, \mathbf{uf}\}$, therefore satisfying $\mathfrak{f}$ under *pr*. A (symmetric) attack between $x^{\mathbf{u}}_{\mathbf{ut}}$ and $x^{\mathbf{t}}_{\mathbf{tt}}$ would give $\mathbf{ut}$ as output-labelling, which is not as specified by $\mathfrak{f}(\mathbf{t})$.

Theorem 7. *A* 3*-valued* I/O*-specification* $\mathfrak{f}$ *is satisfiable under pr iff* $\mathfrak{f}$ *is monotonic.*

Proof. The if-direction was shown in Proposition 8 while the only-if-direction directly follows from Proposition 6. □

As to the remaining semantics, note that Propositions 7 and 8 also apply to ideal and semi-stable semantics, respectively, since for each L the grounded labelling of $\blacktriangleright(\mathfrak{D}^{gr}_{\mathfrak{f}}, L)$ coincides with the ideal labelling, and the preferred labellings of $\blacktriangleright(\mathfrak{D}^{pr}_{\mathfrak{f}}, L)$ coincide with semi-stable labellings.

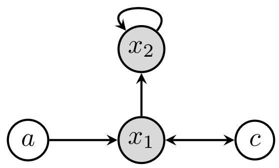

Figure 6: I/O-gadget satisfying a 3-valued I/O-specification under semi-stable (resp. ideal) semantics which is not satisfiable under preferred (resp. grounded) semantics.

However, this does not allow to derive a complete characterization, since there are non-monotonic 3-valued I/O-specifications, satisfiable by ideal and semi-stable semantics, respectively.

Example 8. Consider the AF F in Figure 6 which represents an I/O-gadget with $I = \{a\}$ and $O = \{c\}$. The semi-stable and ideal extensions of the various labelling-injections are $se(\blacktriangleright(F, \mathbf{u})) = id(\blacktriangleright(F, \mathbf{u})) = \{\{c\}\}$, $se(\blacktriangleright(F, \mathbf{t})) = id(\blacktriangleright(F, \mathbf{t})) = \{\{a, c\}\}$, and $se(\blacktriangleright(F, \mathbf{f})) = \{\{z, x_1\}\} \neq id(\blacktriangleright(F, \mathbf{f})) = \{\{z\}\}$. This means that F satisfies the 3-valued I/O-specification given by $\mathbf{u} \mapsto \{\mathbf{t}\}$, $\mathbf{t} \mapsto \{\mathbf{t}\}$, and $\mathbf{f} \mapsto \{\mathbf{u}\}$ under ideal semantics, and the one given by $\mathbf{u} \mapsto \{\mathbf{t}\}$, $\mathbf{t} \mapsto \{\mathbf{t}\}$, and $\mathbf{f} \mapsto \{\mathbf{f}\}$ under semi-stable semantics which are both clearly not monotonic.

An exact characterization of 3-valued I/O-specifications which are satisfiable under semi-stable and ideal semantics, respectively, requires weaker notions of monotonicity. Complete semantics, on the other hand, imposes necessary conditions which are more restrictive. The following is a direct consequence of Proposition 7 in (Baroni et al. 2014).

Proposition 9. *Every* 3*-valued* I/O*-specification* $\mathfrak{f}$ *which is satisfiable under co is monotonic and for all* L_1 *and* L_2 *such that* $L_1 \sqsubseteq L_2$ *it holds that* $\forall K_2 \in \mathfrak{f}(L_2) \exists K_1 \in \mathfrak{f}(L_1) : K_1 \sqsubseteq K_2$.

Exact characterizations of satisfiable 3-valued I/O-specifications for complete, semi-stable and ideal semantics are subject of future work.

5 Partial I/O-specifications

In the previous sections we restricted our considerations to total I/O-specifications, where the output is defined for each input. One can also think of situations where we do not care about the output for some inputs, i.e. we are only interested in satisfiability of a partial function.

The results provided in Section 3 can be directly exploited to handle partial I/O-specifications in the extension-based case: "don't care"-outputs can be assigned arbitrarily, provided that at least one extension is assigned for *pr*, *se* and *sg*, a single extension for *gr* and *id*, and the specification is closed for *co*. Furthermore, all the proofs also work with a partial I/O-specification by considering only the specified inputs in the definition of the canonical I/O-gadget, i.e. neglecting the inputs with undefined output, yielding a considerable simplification. On the other hand, the following example shows some difficulties in the labelling-based case.

Example 9. Consider the partial 3-valued I/O-specification $\mathfrak{f}$ for $I = \{a, b\}$ and $O = \{c\}$ with $\mathfrak{f}(\mathbf{uu}) = \{\mathbf{u}\}$, $\mathfrak{f}(\mathbf{tu}) = \{\mathbf{t}\}$ and undefined, i.e. "don't care", for all other inputs. Clearly, $\mathfrak{f}$ is satisfied by $\mathfrak{D}^{gr}_{\mathfrak{f}}$ or, more easily, by the I/O-gadget $(\{a, b, x, c\}, \{(a, x), (x, c)\})$. Now note that $\mathfrak{f}$ is not monotonic according to Definition 11, since $\nexists K \in \mathfrak{f}(\mathbf{tt}) : \mathbf{t} \sqsubseteq K$. It is monotonic for those inputs for which it is defined though. But this is also the case if we consider $\mathfrak{f}'$ which coincides with $\mathfrak{f}$ on inputs $\mathbf{uu}$ and $\mathbf{tu}$ but also defines $\mathfrak{f}'(\mathbf{ut}) = \{\mathbf{f}\}$. Now one can check that there is no I/O-gadget satisfying $\mathfrak{f}'$ under the grounded semantics. The reason for this is that $\mathfrak{f}$ cannot be extended to a total specification which is still monotonic. In order to be monotonic some $\mathfrak{f}''$ extending $\mathfrak{f}'$ would have to fulfill both $\mathbf{t} \sqsubseteq K$ and $\mathbf{f} \sqsubseteq K$ for the unique output $K \in \mathfrak{f}''(\mathbf{tt})$, which is obviously not possible.

This already leads us to the condition for satisfiability of partial functions.

Definition 16. Given two (partial) 3-valued I/O-specifications $\mathfrak{f}$ and $\mathfrak{f}'$, we say that $\mathfrak{f}'$ *extends* $\mathfrak{f}$ iff for all $L \in \mathcal{L}(I)$ such that $\mathfrak{f}(L)$ is defined, $\mathfrak{f}'(L) = \mathfrak{f}(L)$.

Theorem 8. *A (partial)* 3*-valued* I/O*-specification* $\mathfrak{f}$ *is satisfiable under semantics* σ *iff there is a total function* $\mathfrak{f}'$ *extending* $\mathfrak{f}$ *which is satisfiable under* σ.

It may be noted that the extension-based case covered in Section 3 can be viewed as a particular case of 3-valued partial specification where also the output is partially specified, i.e. for those inputs without undecided arguments we specify a set of extensions (i.e. without distinguishing between $\mathbf{f}$ and $\mathbf{u}$ arguments).

6 Conclusions

To the best of our knowledge, this is the first characterization of the input/output expressive power of argumentation semantics. In (Dunne et al. 2015), expressiveness has been studied as the capability of enforcing sets of extensions. The problem faced in this paper differs in two aspects: on the one hand, we have to enforce a set of extensions for any input rather than a single set of extensions; on the other hand, we can exploit non-output arguments that are not seen outside a sub-framework. Moreover we also consider labellings besides extensions. A labelling-based investigation exploiting hidden arguments is carried out in (Dyrkolbotn 2014), but still in the context of an ordinary AF rather than an I/O-gadget. Pührer (2015) and Strass (2015) have recently investigated the expressiveness of abstract dialectical frameworks (Brewka et al. 2013) under three-valued and two-valued semantics, respectively.

Future work includes the 3-valued I/O-characterization of complete semantics, being the only transparent semantics (Baroni et al. 2014) for which this was left open. Moreover, the investigation of further semantics such as CF2 (Baroni, Giacomin, and Guida 2005) would be of interest. Another issue is the construction of I/O-gadgets from *compact* I/O-specifications where the function is not explicitly stated but, for instance, described as a Boolean (or three-valued) circuit. We conjecture that I/O-gadgets can then be composed from simple building blocks along the lines of the given circuit. A related question in this direction is the identification of minimal I/O-gadgets satisfying a given specification.

Acknowledgements

The authors would like to thank the anonymous reviewers for their helpful comments. This research has been supported by the Austrian Science Fund (FWF) through projects I1102 and P25521.

References

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

Baroni, P.; Boella, G.; Cerutti, F.; Giacomin, M.; van der Torre, L.; and Villata, S. 2014. On the Input/Output behaviour of argumentation frameworks. *Artificial Intelligence* 217:144–197.

Baroni, P.; Caminada, M.; and Giacomin, M. 2011. An introduction to argumentation semantics. *Knowledge Engineering Review* 26(4):365–410.

Baroni, P.; Giacomin, M.; and Guida, G. 2005. SCC-Recursiveness: A general schema for argumentation semantics. *Artificial Intelligence* 168(1-2):162–210.

Brewka, G.; Ellmauthaler, S.; Strass, H.; Wallner, J. P.; and Woltran, S. 2013. Abstract Dialectical Frameworks Revisited. In Rossi, F., ed., *Proceedings of the 23rd International Joint Conference on Artificial Intelligence, IJCAI 2013*, 803–809. AAAI Press / IJCAI.

Caminada, M., and Amgoud, L. 2007. On the evaluation of argumentation formalisms. *Artificial Intelligence* 171(5-6):286–310.

Caminada, M., and Gabbay, D. M. 2009. A logical account of formal argumentation. *Studia Logica* 93(2):109–145.

Caminada, M.; Carnielli, W. A.; and Dunne, P. E. 2012. Semi-stable semantics. *Journal of Logic and Computation* 22(5):1207–1254.

Diller, M.; Haret, A.; Linsbichler, T.; Rümmele, S.; and Woltran, S. 2015. An extension-based approach to belief revision in abstract argumentation. In Yang, Q., and Wooldridge, M., eds., *Proceedings of the 24th International Joint Conference on Artificial Intelligence, IJCAI 2015*, 2926–2932. AAAI Press.

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

Dunne, P. E.; Dvořák, W.; Linsbichler, T.; and Woltran, S. 2015. Characteristics of multiple viewpoints in abstract argumentation. *Artificial Intelligence* 228:153–178.

Dyrkolbotn, S. K. 2014. How to argue for anything: Enforcing arbitrary sets of labellings using AFs. In Baral, C.; De Giacomo, G.; and Eiter, T., eds., *Proceedings of the 14th International Conference on Principles of Knowledge Representation and Reasoning, KR 2014*, 626–629. AAAI Press.

Modgil, S., and Bench-Capon, T. J. M. 2011. Metalevel argumentation. *Journal of Logic and Computation* 21(6):959–1003.

Pührer, J. 2015. Realizability of Three-Valued Semantics for Abstract Dialectical Frameworks. In Yang, Q., and Wooldridge, M., eds., *Proceedings of the 24th International Joint Conference on Artificial Intelligence, IJCAI 2015*, 3171–3177. AAAI Press.

Strass, H. 2015. Expressiveness of two-valued semantics for abstract dialectical frameworks. *J. Artif. Intell. Res. (JAIR)* 54:193–231.

Verheij, B. 1996. Two approaches to dialectical argumentation: admissible sets and argumentation stages. In Meyer, J.-J. C., and van der Gaag, L. C., eds., *Proceedings of the 8th Dutch Conference on Artificial Intelligence, NAIC 1996*, 357–368.