Labellings for Assumption-Based and Abstract Argumentation

Claudia Schulz, Francesca Toni

Department of Computing, Imperial College London, London SW7 2AZ, UK

Abstract

The semantics of Assumption-Based Argumentation (ABA) frameworks are traditionally characterised as assumption extensions, i.e. sets of accepted assumptions. *Assumption labellings* are an alternative way to express the semantics of *flat* ABA frameworks, where one of the labels IN, OUT, or UNDEC is assigned to each assumption. They are beneficial for applications where it is important to distinguish not only between accepted and non-accepted assumptions, but further divide the non-accepted assumptions into those which are clearly rejected and those which are neither accepted nor rejected and thus undecided. We prove one-to-one correspondences between assumption labellings and extensions for the admissible, grounded, complete, preferred, ideal, semi-stable and stable semantics. We also show how the definition of assumption labellings for flat ABA frameworks can be extended to assumption labellings for *any* (flat and *non-flat*) ABA framework, enabling reasoning with a wider range of scenarios. Since flat ABA frameworks are structured instances of Abstract Argumentation (AA) frameworks, we furthermore investigate the relation between assumption labellings for flat ABA frameworks and argument labellings for AA frameworks. Building upon prior work on complete assumption and argument labellings, we prove one-to-one correspondences between grounded, preferred, ideal, and stable assumption and argument labellings, and a one-to-many correspondence between admissible assumption and argument labellings. Inspired by the notion of admissible assumption labellings we introduce *committed admissible argument labellings* for AA frameworks, which correspond more closely to admissible assumption labellings of ABA frameworks than admissible argument labellings do.

Keywords: Assumption-Based Argumentation, Labelling Semantics, Abstract Argumentation

1. Introduction

Argumentation provides an intuitive way of modelling human reasoning and decision making and has thus received considerable attention in AI research [1, 2]. Argumentation frameworks formalise the notions of arguments and conflicts between them, and specify semantics to determine which arguments are accepted in a debate of conflicting arguments. Two main types of argumentation frameworks can be distinguished: In *Abstract Argumentation* (AA) frameworks [3] a set of abstract entities, called arguments, is given along with

Email addresses: claudia.schulz@imperial.ac.uk (Claudia Schulz), f.toni@imperial.ac.uk (Francesca Toni)

an attack relation between these arguments. In structured argumentation frameworks (e.g. [4, 5, 6, 7], see [8] for an overview) structured knowledge is given from which arguments are constructed and attacks are derived.

Here, we deal with a certain type of structured argumentation frameworks, namely *Assumption-Based Argumentation* (ABA) frameworks [4, 9, 10], which have proven useful in a variety of applications including agent negotiation and dialogue [11, 12, 13, 14, 15, 16], decision making [17, 18, 19], web reasoning [20], and explanation [21, 22]. In an ABA framework structured knowledge is given in terms of inference rules expressed in an underlying logical language. A subset of sentences is defined to be the set of *assumptions*, each of which has a *contrary* sentence in the language. As an example, consider an ABA framework with the following inference rules[1] about the excellence of Imperial College London (ICL) and the withdrawal of the UK from the EU, where assumptions are indicated in italic:

ICL is an excellent university ← *ICL takes many international students and staff*
Fewer EU citizens apply to ICL ← *EU citizens need a visa to move to the UK*
EU citizens can move to the UK without a visa ← *The UK remains in the EU*

The contraries of the assumptions are as follows:

- contrary of *ICL takes many international students and staff*: Fewer EU citizens apply to ICL;
- contrary of *EU citizens need a visa to move to the UK*: EU citizens can move to the UK without a visa;
- contrary of *The UK remains in the EU*: The UK leaves the EU.

The semantics of ABA frameworks are defined in terms of sets of accepted assumptions, called *assumption extensions*, which are determined based on the contraries of assumptions and derivations from assumptions using the inference rules [4]. In the above example, one of the assumption extensions is {*The UK remains in the EU*}, expressing that the assumption *The UK remains in the EU* is accepted and all other assumptions are not.

Given a *flat* ABA framework, where assumptions do not occur on the left-hand side of inference rules, arguments and attacks between them can be constructed. A flat ABA framework can thus instantiate an AA framework comprising all arguments and attacks constructable from the flat ABA framework [23]. It follows that the semantics of a flat ABA framework can also be expressed in terms of the semantics of AA frameworks [23], i.e. as sets of accepted arguments called *argument extensions* [3]. Importantly, the assumption and argument extensions of a flat ABA framework correspond for nearly all semantics defined for flat ABA frameworks[2]: an argument extension contains all arguments supported by the assumptions in an assumption extension, and conversely an assumption extension contains all assumptions supporting arguments in an argument extension [23, 20, 24].

The semantics of AA frameworks can be alternatively formulated as *argument labellings*, which assign one of the labels `in` (accepted), `out` (rejected), or `undec` (undecided) to each argument [25, 26]. This notion of semantics has the advantage over argument extensions

[1] Rules are adapted from www.imperial.ac.uk/newsandeventspggrp/imperialcollege/newssummary/news_13-1-2016-18-10-57 and www.independent.co.uk/student/news/eu-referendum-result-brexit-leave-remain-higher-education-sector-students-a7100106.html

[2] The only exception is the semi-stable semantics [24].

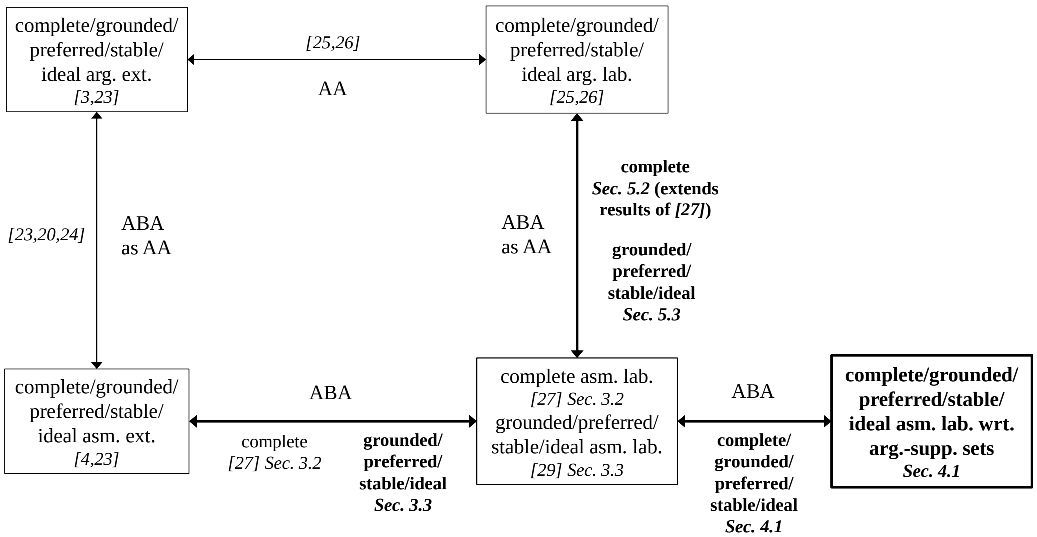

Figure 1: A summary of results concerning the complete, grounded, preferred, stable, and ideal semantics in the different sections of this paper, where applicable, in the context of previous work. Bidirectional arrows indicate semantic correspondence, and bold indicates novel work in this paper.

that it does not only distinguish between accepted and non-accepted arguments, but further divides the non-accepted arguments into rejected and undecided ones. Since argument labellings and argument extensions correspond [25, 26], argument labellings can also be used to characterise the semantics for flat ABA frameworks in terms of arguments. Recently, the idea of argument labellings was transferred to assumptions [27], yielding a new characterisation of the semantics of flat ABA frameworks. In contrast to argument labellings which label whole arguments, *assumption labellings* label each assumption as IN (accepted), OUT (rejected), or UNDEC (undecided). Assumption labellings have the advantage over assumption extensions that rejected (OUT) assumptions and assumptions which are neither accepted nor rejected (UNDEC) are distinguished.

This distinction can be important in applications such as decision making: undecided assumptions can for example provide an indication that further information from an expert is required in order to decide their acceptability for sure. For instance, in the ICL example one of the assumption labellings labels *The UK remains in the EU* as IN, *EU citizens need a visa to move to the UK* as OUT, and *ICL takes many international students and staff* as UNDEC. This expresses that it is not certain whether or not *ICL takes many international students and staff* and thus provides a more detailed interpretation than the previously given assumption extension.

This work considerably extends [27, 28], where we considered only two semantics of flat ABA frameworks[3]. Here, we investigate all semantics defined for flat ABA frameworks, i.e. admissible, grounded, complete, preferred, ideal, semi-stable, and stable semantics, making

[3]Caminada and Schulz [29] introduce grounded, preferred, ideal, and stable assumption labellings for ABA but do neither prove correspondence with the respective assumption extensions nor investigate their relation with argument labellings.

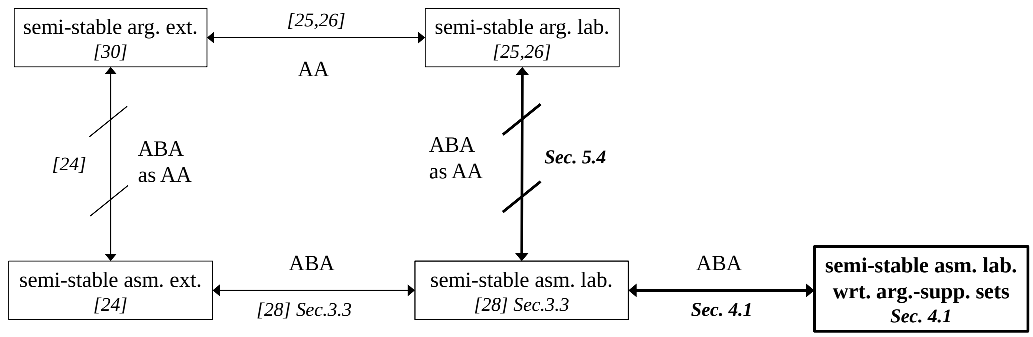

Figure 2: Results concerning the semi-stable semantics in the different sections of this paper, where applicable, in the context of previous work. Bidirectional arrows indicate semantic correspondence, crossed out arrows denote non-correspondence, and bold indicates novel work in this paper.

this paper a self-contained reference for assumption labellings and their relation with both assumption extensions and argument labellings. We prove that there is a one-to-one correspondence between assumption labellings and extensions for all aforementioned semantics. We also investigate the relation between assumption and argument labellings for flat ABA frameworks, showing a one-to-one correspondence for the grounded, complete, preferred, ideal, and stable semantics, as summarised in Figure 1. Since semi-stable argument and assumption extensions do not correspond [24], it is unsurprising that the respective labellings do not correspond either, as shown in Figure 2. Concerning the admissible semantics we prove a one-to-many correspondence between assumption and argument labellings. Based on this dissimilarity, we introduce a variant of admissible argument labellings for AA frameworks, called *committed admissible argument labellings*, which correspond more closely to admissible assumption labellings than the original admissible argument labellings, as illustrated in Figure 3. We furthermore extend [27, 28], where only flat ABA frameworks have been considered, by introducing assumption labellings for *any* ABA framework. Our results are summarised in Figure 4. In ABA frameworks which may not be flat, assumptions can occur on the left-hand side of an inference rule and can thus constitute facts, expressing statements such as "I firmly believe that ICL will always be an excellent university".

The paper is organised as follows. In Section 2 we give background on flat ABA frameworks and AA frameworks. In Section 3 we introduce assumption labellings for the different semantics of flat ABA frameworks and prove their correspondence with assumption extensions of flat ABA frameworks, building upon prior work in [27, 28]. In Section 4 we simplify the definition of assumption labellings for flat ABA frameworks by considering only certain sets of assumptions as attackers of assumptions. We furthermore introduce a graphical representation of flat ABA frameworks and illustrate how assumption labellings can be easily determined and represented using these graphs. In Section 5 we investigate the correspondence between assumption and argument labellings of flat ABA frameworks (considerably extending preliminary work in [27]) and introduce committed admissible argument labellings as a variant of admissible argument labellings for AA frameworks. In Section 6 we extend the definition of assumption labellings from flat ABA frameworks to any ABA framework, and in Section 7 we conclude and discuss future research.

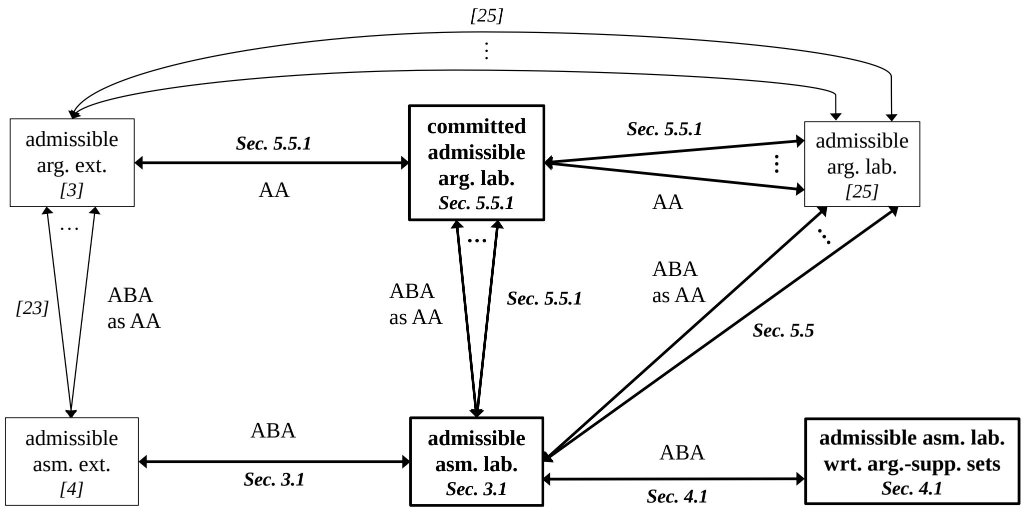

Figure 3: Results concerning the admissible semantics in the different sections of this paper, where applicable, in the context of previous work. Bidirectional arrows indicate semantic correspondence (arrows with the same starting point but different end points indicate one-to-many correspondence), and bold indicates novel work in this paper.

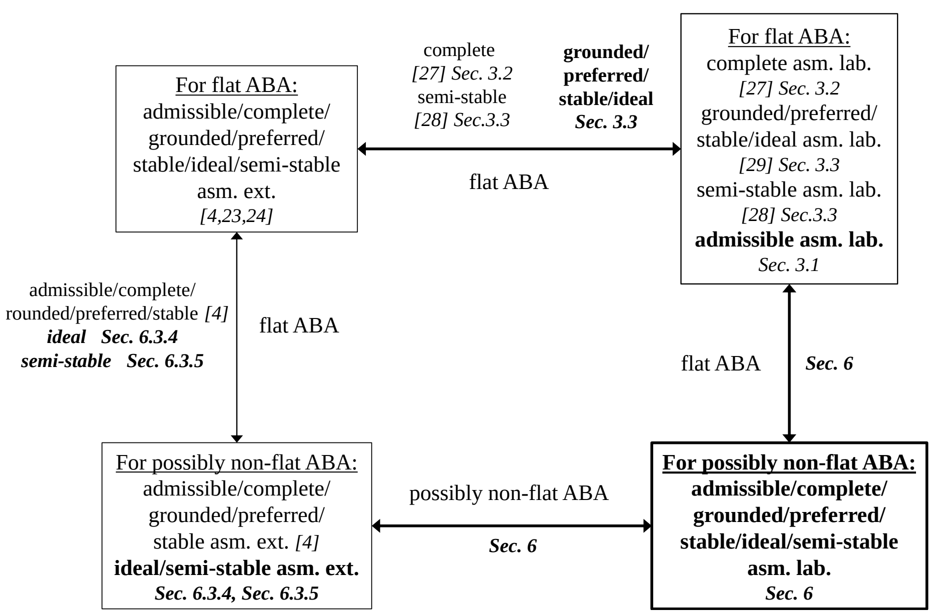

Figure 4: Results for any (possibly non-flat) ABA framework in the different sections of this paper, where applicable, in the context of previous work. Bidirectional arrows indicate semantic correspondence, and bold indicates novel work in this paper.

2. Background

2.1. Abstract Argumentation

An *Abstract Argumentation (AA) framework* [3] is a pair $\langle Ar, Att \rangle$, where Ar is a set of arguments and $Att \subseteq Ar \times Ar$ is a binary attack relation between arguments. A pair $(A, B) \in Att$ expresses that argument A *attacks* argument B, or equivalently that B is attacked by A. A set of arguments $Args \subseteq Ar$ attacks an argument $B \in Ar$ if and only if there is $A \in Args$ such that A attacks B. $Args^+ = \{A \in Ar \mid Args \textit{ attacks } A\}$ denotes the set of all arguments attacked by $Args$ [26].

Let $Args \subseteq Ar$ be a set of arguments.

- $Args$ is *conflict-free* if and only if $Args \cap Args^+ = \emptyset$.
- $Args$ *defends* $A \in Ar$ if and only if $Args$ attacks every $B \in Ar$ attacking A.

The semantics of an AA framework are defined in terms of *argument extensions*, i.e. sets of accepted arguments [3, 23, 30]. A set of arguments $Args \subseteq Ar$ is

- an *admissible argument extension* if and only if $Args$ is conflict-free and defends all arguments $A \in Args$;
- a *complete argument extension* if and only if $Args$ is conflict-free and consists of all arguments it defends;
- a *grounded argument extension* if and only if $Args$ is a minimal (w.r.t. $\subseteq$) complete argument extension;
- a *preferred argument extension* if and only if $Args$ is a maximal (w.r.t. $\subseteq$) complete argument extension;
- an *ideal argument extension* if and only if $Args$ is a maximal (w.r.t. $\subseteq$) admissible argument extension satisfying that for all preferred argument extensions $Args'$, $Args \subseteq Args'$;
- a *semi-stable argument extension* if and only if $Args$ is a complete argument extension and for all complete argument extensions $Args'$, $Args \cup Args^+ \not\subset Args' \cup Args'^+$;
- a *stable argument extension* if and only if $Args$ is a complete argument extension and $Args \cup Args^+ = Ar$.

Note that these definitions of argument extensions are not the original ones introduced in [3] but are equivalent formulations [26].

Another way of expressing the semantics of an AA framework is in terms of argument labellings [31, 25]. An *argument labelling* is a total function $LabArg : Ar \rightarrow \{\texttt{in}, \texttt{out}, \texttt{undec}\}$. The set of arguments labelled `in` by $LabArg$ is $\texttt{in}(LabArg) = \{A \in Ar \mid LabArg(A) = \texttt{in}\}$; the sets of arguments labelled `out` and `undec` are denoted $\texttt{out}(LabArg)$ and $\texttt{undec}(LabArg)$, respectively.

An argument labelling $LabArg$ is an *admissible argument labelling* if and only if for each argument $A \in Ar$ it holds that:

- if $LabArg(A) = \texttt{in}$ then for each $B \in Ar$ attacking A, $LabArg(B) = \texttt{out}$;

- if $LabArg(A) = \texttt{out}$ then there exists some $B \in Ar$ attacking A such that $LabArg(B) = \texttt{in}$.

An argument labelling $LabArg$ is a *complete argument labelling* if and only if it is an admissible argument labelling and for each argument $A \in Ar$ it holds that:

- if $LabArg(A) = \texttt{undec}$ then there exists some $B \in Ar$ attacking A such that $LabArg(B) = \texttt{undec}$ and there exists no $C \in Ar$ attacking A such that $LabArg(C) = \texttt{in}$.

In order to define argument labellings according to other semantics, we first recall how to compare the *commitment* of argument labellings [26].

> Let $LabArg_1$ and $LabArg_2$ be argument labellings. $LabArg_2$ is *more or equally committed* than $LabArg_1$, denoted $LabArg_1 \sqsubseteq LabArg_2$, if and only if $\texttt{in}(LabArg_1) \subseteq \texttt{in}(LabArg_2)$ and $\texttt{out}(LabArg_1) \subseteq \texttt{out}(LabArg_2)$.

A complete argument labelling $LabArg$ is [25, 32]

- a *grounded argument labelling* if and only if $\texttt{in}(LabArg)$ is minimal (w.r.t. $\subseteq$) among all complete argument labellings;
- a *preferred argument labelling* if and only if $\texttt{in}(LabArg)$ is maximal (w.r.t. $\subseteq$) among all complete argument labellings;
- an *ideal argument labelling* if and only if $LabArg$ is a maximal (w.r.t. $\sqsubseteq$) admissible argument labelling which satisfies that for all preferred argument labellings $LabArg'$, $LabArg \sqsubseteq LabArg'$;
- a *semi-stable argument labelling* if and only if $\texttt{undec}(LabArg)$ is minimal (w.r.t. $\subseteq$) among all complete argument labellings;
- a *stable argument labelling* if and only if $\texttt{undec}(LabArg) = \emptyset$.

Complete, grounded, preferred, ideal, semi-stable, and stable argument extensions correspond one-to-one to the sets of arguments labelled in by the complete, grounded, preferred, ideal, semi-stable, and stable argument labellings, respectively [25, 32]. In contrast, an admissible argument extension may correspond to various admissible argument labellings [25].

2.2. Assumption-Based Argumentation

An *Assumption-Based Argumentation (ABA) framework* [4, 9, 10] is a tuple $\langle \mathcal{L}, \mathcal{R}, \mathcal{A}, \bar{\ } \rangle$, where

- $(\mathcal{L}, \mathcal{R})$ is a deductive system, with $\mathcal{L}$ a language of countably many sentences and $\mathcal{R}$ a set of inference rules of the form $s_0 \leftarrow s_1, \ldots, s_n$ $(n \geq 0)$ where $s_0, \ldots, s_n \in \mathcal{L}$; s_0 is the *head* of the inference rule and $s_1, \ldots, s_n$ is its *body*;
- $\mathcal{A} \subseteq \mathcal{L}$ is a non-empty set of *assumptions*;
- $\bar{\ }$ is a total mapping from $\mathcal{A}$ into $\mathcal{L}$ defining the *contrary* of assumptions, where $\overline{\alpha}$ denotes the contrary of $\alpha \in \mathcal{A}$.

An ABA framework is *flat* if assumptions only occur in the body of inference rules [33]. From here onwards, until Section 6, we assume as given a flat ABA framework $\langle \mathcal{L}, \mathcal{R}, \mathcal{A}, \overline{} \rangle$.

An *argument* [9] with conclusion $s \in \mathcal{L}$ and *premises* $Asms \subseteq \mathcal{A}$, denoted $Asms \vdash s$, is a finite tree where every node holds a sentence in $\mathcal{L}$ or the sentence τ (where $\tau \notin \mathcal{L}$ stands for "true"), such that

- the root node holds s;
- for every node N
  - if N is a leaf then N holds either an assumption or τ;
  - if N is not a leaf and N holds the sentence s_0, then there is an inference rule $s_0 \leftarrow s_1, \ldots, s_m \in \mathcal{R}$ and either $m = 0$ and the only child node of N holds τ or $m > 0$ and N has m children holding $s_1, \ldots, s_m$;
- $Asms$ is the set of all assumptions held by leaf nodes.

We sometimes name arguments with capital letters, e.g. $A\!: Asms \vdash s$ is an argument with name A. With an abuse of notation, the name of an argument is also used to refer to the whole argument. Note that for every assumption $\alpha \in \mathcal{A}$ there exists an *assumption-argument* $\{\alpha\} \vdash \alpha$.

Let $Asms, Asms_1, Asms_2 \subseteq \mathcal{A}$ be sets of assumptions and let $\alpha \in \mathcal{A}$ be an assumption.

- $Asms$ *attacks* α if and only if there exists an argument $Asms' \vdash \overline{\alpha}$ such that $Asms' \subseteq Asms$. Equivalently, we say that α is attacked by $Asms$.
- $Asms_1$ *attacks* $Asms_2$ if and only if $Asms_1$ attacks some $\alpha \in Asms_2$.
- $Asms^+ = \{\alpha \in \mathcal{A} \mid Asms \textit{ attacks } \alpha\}$.
- $Asms$ *defends* α if and only if $Asms$ attacks all sets of assumptions attacking α.
- $Asms$ is *conflict-free* if and only if $Asms \cap Asms^+ = \emptyset$.

The semantics of an ABA framework are defined as *assumption extensions*, i.e. sets of accepted assumptions [4, 23, 24]. A set of assumptions $Asms \subseteq \mathcal{A}$ is

- an *admissible assumption extension* if and only if $Asms$ is conflict-free and defends every $\alpha \in Asms$;
- a *complete assumption extension* if and only if $Asms$ is conflict-free and consists of all assumptions it defends;
- a *grounded assumption extension* if and only if $Asms$ is a minimal (w.r.t. $\subseteq$) complete assumption extension;
- a *preferred assumption extension* if and only if $Asms$ is a maximal (w.r.t. $\subseteq$) complete assumption extension;
- an *ideal assumption extension* if and only if $Asms$ is a maximal (w.r.t. $\subseteq$) complete assumption extension satisfying that for all preferred assumption extensions $Asms'$, $Asms \subseteq Asms'$;

- a *semi-stable assumption extension* if and only if $Asms$ is a complete assumption extension and for all complete assumption extensions $Asms'$, $Asms \cup Asms^+ \not\subset Asms' \cup Asms'^+$;

- a *stable assumption extension* if and only if $Asms$ is a complete assumption extension and $Asms \cup Asms^+ = \mathcal{A}$.

Note that some of these definitions are not the original ones introduced in [4, 23] but are equivalent formulations as proven in [24].

2.3. Correspondence between ABA and AA

A flat ABA framework $\langle \mathcal{L}, \mathcal{R}, \mathcal{A}, \bar{\ } \rangle$ can be mapped onto a *corresponding AA framework* $\langle Ar_{ABA}, Att_{ABA} \rangle$ [23], where

- Ar_{ABA} is the set of all arguments $Asms \vdash s$;

- $(Asms_1 \vdash s_1,\ Asms_2 \vdash s_2) \in Att_{ABA}$ if and only if $\exists \alpha \in Asms_2$ such that $s_1 = \overline{\alpha}$.

Given an admissible/complete/grounded/preferred/ideal/stable assumption extension $Asms$ of $\langle \mathcal{L}, \mathcal{R}, \mathcal{A}, \bar{\ } \rangle$, the set of all arguments whose premises are a subset of $Asms$ is an admissible/complete/grounded/preferred/ideal/stable argument extension of $\langle Ar_{ABA}, Att_{ABA} \rangle$ [23, 20, 24]. Conversely, given an admissible/complete/grounded/preferred/ideal/stable argument extension $Args$ of $\langle Ar_{ABA}, Att_{ABA} \rangle$, the union of all premises of arguments in $Args$ is an admissible/complete/grounded/preferred/ideal/stable assumption extension of $\langle \mathcal{L}, \mathcal{R}, \mathcal{A}, \bar{\ } \rangle$ [23, 20, 24]. Note that this correspondence does not hold for semi-stable assumption and argument extensions.

3. Assumption Labellings

Inspired by argument labellings for AA frameworks, we build upon recently introduced labellings for ABA frameworks [27]. In contrast to labellings in AA, which assign labels to whole arguments, assumption labellings assign labels to single assumptions. The three labels used throughout this paper are IN, indicating that an assumption is accepted, OUT, indicating that an assumption is rejected, and UNDEC, indicating that an assumption is neither accepted nor rejected and thus undecided.

Definition 1. An *assumption labelling* is a total function $LabAsm : \mathcal{A} \rightarrow \{\text{IN}, \text{OUT}, \text{UNDEC}\}$.

If $LabAsm(\alpha) = \text{IN}$, we say that α *is labelled* IN by $LabAsm$, or equivalently that $LabAsm$ labels α (as) IN. Analogous terminology is used for assumptions labelled OUT and UNDEC. The set of all assumptions labelled IN by $LabAsm$ is $\text{IN}(LabAsm) = \{\alpha \in \mathcal{A} \mid LabAsm(\alpha) = \text{IN}\}$, and the sets of all assumptions labelled OUT and UNDEC are denoted $\text{OUT}(LabAsm)$ and $\text{UNDEC}(LabAsm)$, respectively.

We frequently represent an assumption labelling as a set of ordered pairs, for example for an ABA framework with $\mathcal{A} = \{\phi, \psi, \chi\}$ and the labelling $LabAsm$ such that $LabAsm(\phi) = \text{IN}$, $LabAsm(\psi) = \text{OUT}$, and $LabAsm(\chi) = \text{UNDEC}$, we equivalently represent $LabAsm$ as $\{(\phi, \text{IN}), (\psi, \text{OUT}), (\chi, \text{UNDEC})\}$.

3.1. Admissible Semantics

An admissible assumption extension is a set of *accepted* assumptions which is able to defend itself. In other words, if an assumption α is contained in an admissible assumption extension, then all sets of assumptions attacking α contain some assumption attacked by this admissible assumption extension. In an *admissible assumption labelling* the concept of defence is mirrored by requiring that if an assumption α is accepted (labelled IN) then all sets of assumptions attacking α contain a rejected assumption (labelled OUT), which in turn is attacked by a set of accepted assumptions (all labelled IN). In addition, we require that an undecided assumption (labelled UNDEC) is not attacked by a set of accepted assumptions (all labelled IN), since an assumption attacked by accepted assumptions can clearly not be accepted (due to the conflict-freeness property of the admissible semantics) and should thus be rejected rather than undecided.

Definition 2. Let *LabAsm* be an assumption labelling. *LabAsm* is an *admissible assumption labelling* if and only if for each assumption $\alpha \in \mathcal{A}$ it holds that:

- if $LabAsm(\alpha) = \text{IN}$ then for each set of assumptions $Asms$ attacking α there exists some $\beta \in Asms$ such that $LabAsm(\beta) = \text{OUT}$;
- if $LabAsm(\alpha) = \text{OUT}$ then there exists a set of assumptions $Asms$ attacking α such that for all $\beta \in Asms$, $LabAsm(\beta) = \text{IN}$;
- if $LabAsm(\alpha) = \text{UNDEC}$ then for each set of assumptions $Asms$ attacking α there exists some $\beta \in Asms$ such that $LabAsm(\beta) \neq \text{IN}$.

Example 1. Consider the following ABA framework, which we call ABA_1:

$$\mathcal{L} = \{r, p, x, \rho, \psi, \chi\},$$
$$\mathcal{R} = \{p \leftarrow \rho; x \leftarrow \psi\},$$
$$\mathcal{A} = \{\rho, \psi, \chi\},$$
$$\overline{\rho} = r\ ,\ \overline{\psi} = p\ ,\ \overline{\chi} = x.$$

ABA_1 has three admissible assumption labellings:

- $LabAsm_1 = \{(\rho, \text{UNDEC}), (\psi, \text{UNDEC}), (\chi, \text{UNDEC})\}$,
- $LabAsm_2 = \{(\rho, \text{IN}), (\psi, \text{OUT}), (\chi, \text{UNDEC})\}$, and
- $LabAsm_3 = \{(\rho, \text{IN}), (\psi, \text{OUT}), (\chi, \text{IN})\}$.

These assumption labellings demonstrate two important points: first, an assumption which is not attacked by any set of assumption (ρ in ABA_1) cannot be labelled OUT; second, an assumption attacked by a set of assumptions containing only IN labelled assumptions (ψ in $LabAsm_2$ and $LabAsm_3$) is always labelled OUT.

It is important to note that the *empty set* of assumptions has a special role as an attacking set of assumptions: any assumption attacked by the empty set is labelled OUT by all admissible assumption labellings since the empty set stands for a (non-refutable) fact, so the attacked assumption clearly has to be rejected, as illustrated in Example 2.

Example 2. Let ABA_2 be ABA_1 from Example 1 with the additional sentences ϕ and f in $\mathcal{L}$, where ϕ is an assumption with $\overline{\phi} = f$, and with the additional inference rule $f \leftarrow$.

Since $\{\} \vdash f$ is an argument, ϕ is attacked by the empty set of assumptions as well as by all other sets of assumptions. Thus, ϕ cannot be labelled IN since the attacking empty set does not contain an assumption labelled OUT, and ϕ cannot be labelled UNDEC since the attacking empty set does not contain an assumption not labelled IN. Consequently, ϕ is labelled OUT by all admissible assumption labellings.

ABA_2 has thus three admissible assumption labellings:

- $LabAsm_1 = \{(\phi, \text{OUT}), (\rho, \text{UNDEC}), (\psi, \text{UNDEC}), (\chi, \text{UNDEC})\}$
- $LabAsm_2 = \{(\phi, \text{OUT}), (\rho, \text{IN}), (\psi, \text{OUT}), (\chi, \text{UNDEC})\}$
- $LabAsm_3 = \{(\phi, \text{OUT}), (\rho, \text{IN}), (\psi, \text{OUT}), (\chi, \text{IN})\}$

Note that these are the same admissible assumption labellings as for ABA_1, but with the additional assumption ϕ which is always labelled OUT. The number of admissible assumption labellings is thus not influenced by assumptions attacked by the empty set since these assumptions do not have alternative labels in different admissible assumption labellings.

The following theorem shows that there is a one-to-one correspondence between the admissible semantics in terms of assumption labellings and extensions.

Theorem 1.

1. *Let Asms be an admissible assumption extension. Then LabAsm with* $\text{IN}(LabAsm) = Asms$, $\text{OUT}(LabAsm) = Asms^+$, *and* $\text{UNDEC}(LabAsm) = \mathcal{A} \setminus (Asms \cup Asms^+)$ *is an admissible assumption labelling.*
2. *Let LabAsm be an admissible assumption labelling. Then* $Asms = \text{IN}(LabAsm)$ *is an admissible assumption extension with* $Asms^+ = \text{OUT}(LabAsm)$ *and* $\mathcal{A} \setminus (Asms \cup Asms^+) = \text{UNDEC}(LabAsm)$.

The proof of Theorem 1 as well as all other results can be found in the Appendix.

Example 3. ABA_2 from Example 2 has three admissible assumption extensions: $Asms_1 = \{\}$, $Asms_2 = \{\rho\}$, and $Asms_3 = \{\rho, \chi\}$, corresponding to the three admissible assumption labellings $LabAsm_1$, $LabAsm_2$, and $LabAsm_3$, respectively.

Note that without the third condition in Definition 2, the second item in Theorem 1 would not hold. For example, $LabAsm_4 = \{(\phi, \text{OUT}), (\rho, \text{IN}), (\psi, \text{UNDEC}), (\chi, \text{UNDEC})\}$ would be an admissible assumption labelling of ABA_2 (see Example 2), but even though $Asms_4 = \text{IN}(LabAsm_4) = \{\rho\}$ is an admissible assumption extension of ABA_2, it does not hold that $Asms_4^+ = \text{OUT}(LabAsm_4)$ as stated in the second item of Theorem 1 since $\psi \in Asms_4^+$ but $\psi \notin \text{OUT}(LabAsm_4)$.

If an assumption is defended by an admissible assumption extension then adding this assumption to the extension yields another admissible assumption extension [4] (similar to the Fundamental Lemma in AA [3]). Due to the one-to-one correspondence between admissible assumption labellings and extensions, an analogue property holds for admissible assumption labellings.

Lemma 2. *Let LabAsm be an admissible assumption labelling and let* α *be such that for each set of assumptions Asms attacking* α *there exists some* $\beta \in Asms$ *such that* $LabAsm(\beta) = \text{OUT}$. *Let* $\alpha^\star = \{\gamma \in \mathcal{A} \mid \exists Asms \subseteq \mathcal{A}$ *such that* $\alpha \in Asms, Asms$ *attacks* γ, $\forall \delta \in Asms : \delta \neq \alpha \rightarrow LabAsm(\delta) = \text{IN}\}$. *Then* $LabAsm'$ *with* $\text{IN}(LabAsm') = \text{IN}(LabAsm) \cup \{\alpha\}$, $\text{OUT}(LabAsm') = \text{OUT}(LabAsm) \cup \alpha^\star$ *and* $\text{UNDEC}(LabAsm') = \text{UNDEC}(LabAsm) \setminus (\{\alpha\} \cup \alpha^\star)$ *is an admissible assumption labelling.*

The lemma states that if an assumption α is defended by an admissible assumption labelling, i.e. all attacking sets of assumptions $Asms$ contain an assumption β labelled $\textsc{out}$, then changing the label of α to $\textsc{in}$ and changing the label of all assumptions γ which now need to be rejected (due to the change of label of α) to $\textsc{out}$ yields another admissible assumption labelling.

Example 4. Let ABA_3 be an ABA framework with:

$$\begin{aligned}
&\mathcal{L} = \{f, p, r, x, \phi, \psi, \rho, \chi\},\\
&\mathcal{R} = \{r \leftarrow \phi, \chi;\ p \leftarrow \rho\},\\
&\mathcal{A} = \{\phi, \psi, \rho, \chi\},\\
&\overline{\phi} = f,\ \overline{\psi} = p,\ \overline{\rho} = r,\ \overline{\chi} = x.
\end{aligned}$$

$LabAsm_1 = \{(\phi, \textsc{undec}), (\psi, \textsc{undec}), (\rho, \textsc{undec}), (\chi, \textsc{in})\}$ is an admissible assumption labellings of ABA_3. Since ϕ is not attacked by any set of assumptions, it holds that each set of assumptions attacking ϕ contains an assumption labelled $\textsc{out}$, and $\phi^+ = \{\rho\}$. As stated in Lemma 2, $LabAsm_2$ with $\textsc{in}(LabAsm_2) = \{\chi, \phi\}$, $\textsc{out}(LabAsm_2) = \{\rho\}$, and $\textsc{undec}(LabAsm_2) = \{\psi\}$ is an admissible assumption labelling of ABA_3. Since with respect to $LabAsm_2$ it holds that each set of assumptions attacking ψ contains an assumption labelled $\textsc{out}$, $LabAsm_3$ with $\textsc{in}(LabAsm_3) = \{\chi, \phi, \psi\}$, $\textsc{out}(LabAsm_3) = \{\rho\}$, and $\textsc{undec}(LabAsm_3) = \{\}$ is also an admissible assumption labelling of ABA_3.

3.2. Complete Semantics

In addition to defending itself against attackers, a complete assumption extension contains every assumption it defends. This additional condition is mirrored in *complete assumption labellings* by requiring that an assumption which is defended has to be labelled $\textsc{in}$. This can be achieved by modifying the definition of admissible assumption labellings in various ways.

In an admissible assumption labelling a defended assumption may be labelled $\textsc{in}$ or $\textsc{undec}$. Thus, one way of modifying the definition of admissible assumption labellings is to prohibit labelling defended assumptions as $\textsc{undec}$. In other words, an assumption labelled $\textsc{undec}$ has to be attacked by at least one set of assumptions which does not contain any assumption labelled $\textsc{out}$.

Definition 3. Let $LabAsm$ be an assumption labelling. $LabAsm$ is a *complete assumption labelling* if and only if $LabAsm$ is an admissible assumption labelling and for each assumption $\alpha \in \mathcal{A}$ it holds that:

- if $LabAsm(\alpha) = \textsc{undec}$ then there exists a set of assumptions $Asms$ attacking α such that for all $\gamma \in Asms$, $LabAsm(\gamma) \neq \textsc{out}$.

Note that the new condition for $\textsc{undec}$ assumptions implies that there exists a set of assumptions $Asms$ attacking α such that for some $\gamma \in Asms$, $LabAsm(\gamma) = \textsc{undec}$, since by the definition of admissible assumption labellings some $\beta \in Asms$ is not labelled $\textsc{in}$ and by the new condition no $\gamma \in Asms$ is labelled $\textsc{out}$. However, the fact that there exists a set of assumptions $Asms$ attacking α such that for some $\gamma \in Asms$, $LabAsm(\gamma) = \textsc{undec}$, is not a sufficient condition for characterising admissible assumption labellings which are also complete assumption labellings.

Example 5. Consider again ABA_1 from Example 1 and its three admissible assumption labellings. In $LabAsm_1$, ρ does not satisfy the new condition for `undec` assumptions, and in $LabAsm_2$, χ does not satisfy the new condition. The only admissible assumption labelling satisfying the new condition is $LabAsm_3$, which is thus the only complete assumption labelling of ABA_1.

The second way to modify the definition of admissible assumption labellings to express the complete semantics is to add a condition which explicitly states that if an assumption α is defended, i.e. if all sets of assumptions attacking α contain some assumption labelled OUT, then α has to be labelled IN. This condition adds the "opposite direction" of the first condition of an admissible assumption labelling. To make this way of defining complete assumption labellings more uniform, the "opposite direction" of the second condition of an admissible assumption labelling is added, too. This renders the third condition of an admissible assumption labelling superfluous and thus leaves two "if and only if" conditions to be satisfied by each $\alpha \in \mathcal{A}$:

- $LabAsm(\alpha) =$ IN if and only if for each set of assumptions $Asms$ attacking α there exists some $\beta \in Asms$ such that $LabAsm(\beta) =$ OUT;
- $LabAsm(\alpha) =$ OUT if and only if there exists a set of assumptions $Asms$ attacking α such that for all $\beta \in Asms$, $LabAsm(\beta) =$ IN.

Since $LabAsm$ is an assumption labelling and thus labels each assumption in an ABA framework, assumptions which do not satisfy the right hand side of either of the above conditions are "automatically" labelled UNDEC by $LabAsm$.

A third way to define complete assumption labellings reverses all three conditions of Definition 3, thus specifying which label an assumption satisfying a certain condition should have.

Theorem 3. *Let LabAsm be an assumption labelling. The following statements are equivalent:*

1. *LabAsm is a complete assumption labelling.*
2. *LabAsm is such that for each $\alpha \in \mathcal{A}$ it holds that:*
   - $LabAsm(\alpha) =$ IN *if and only if for each set of assumptions Asms attacking α there exists some $\beta \in Asms$ such that $LabAsm(\beta) =$* OUT*;*
   - $LabAsm(\alpha) =$ OUT *if and only if there exists a set of assumptions Asms attacking α such that for all $\beta \in Asms$, $LabAsm(\beta) =$* IN*.*
3. *LabAsm is such that for each $\alpha \in \mathcal{A}$ it holds that:*
   - *if for each set of assumptions Asms attacking α there exists some $\beta \in Asms$ such that $LabAsm(\beta) =$* OUT*, then $LabAsm(\alpha) =$* IN*;*
   - *if there exists a set of assumptions Asms attacking α such that for all $\beta \in Asms$, $LabAsm(\beta) =$* IN*, then $LabAsm(\alpha) =$* OUT*;*
   - *if for each set of assumptions $Asms_1$ attacking α there exists some $\beta \in Asms_1$ such that $LabAsm(\beta) \neq$* IN*, and there exists a set of assumptions $Asms_2$ attacking α such that for all $\gamma \in Asms_2$, $LabAsm(\gamma) \neq$* OUT*, then $LabAsm(\alpha) =$* UNDEC*.*

Example 6. Consider again ABA_1 from Example 1 and its three admissible assumption labellings. $LabAsm_1$ does not satisfy the second item in Theorem 3 since ρ violates the first condition. Similarly, $LabAsm_1$ does not satisfy the third item in Theorem 3 since ρ violates the first condition. $LabAsm_2$ does not satisfy the second or third item in Theorem 3 since χ violates the first condition of both items. Only $LabAsm_3$ satisfies the second as well as the third item in Theorem 3, and is thus the only complete assumption labelling of ABA_1.

All three ways of defining complete assumption labellings are useful in their own rights. Definition 3 is particularly suitable to verify whether a given assumption labelling is indeed a complete assumption labelling. In contrast, the third item in Theorem 3 is more suitable for determining which assumptions should have which label. Since the second item in Theorem 3 can be considered as the "union" of the two other definitions, it lends itself to either of the two tasks.

Note that the definition of admissible assumption labellings cannot be equivalently expressed by reversing of the conditions in Definition 2 since they are not mutually exclusive. In particular, an unattacked assumption would satisfy both the condition to be labelled IN and to be labelled UNDEC, so not matter which of the two labels was assigned to the assumption, one of the two conditions would be violated.

The following theorem proves that there is a one-to-one correspondence between complete assumption labellings and extensions, just as between admissible assumption labellings and extensions.

Theorem 4.

1. *Let $Asms$ be a complete assumption extension. Then $LabAsm$ with* $\text{IN}(LabAsm) = Asms$, $\text{OUT}(LabAsm) = Asms^+$, *and* $\text{UNDEC}(LabAsm) = \mathcal{A} \setminus (Asms \cup Asms^+)$ *is a complete assumption labelling.*
2. *Let $LabAsm$ be a complete assumption labelling. Then* $Asms = \text{IN}(LabAsm)$ *is a complete assumption extension with* $Asms^+ = \text{OUT}(LabAsm)$ *and* $\mathcal{A} \setminus (Asms \cup Asms^+) = \text{UNDEC}(LabAsm)$.

3.3. Grounded, preferred, ideal, semi-stable, and stable semantics

Based on the notion of complete assumption labellings, the grounded, preferred, ideal, semi-stable, and stable semantics can be defined in terms of assumption labellings.

Definition 4. A complete assumption labelling $LabAsm$ is:

- a *grounded assumption labelling* if and only if $\text{IN}(LabAsm)$ is minimal (w.r.t. $\subseteq$) among all complete assumption labellings;
- a *preferred assumption labelling* if and only if $\text{IN}(LabAsm)$ is maximal (w.r.t. $\subseteq$) among all complete assumption labellings;
- an *ideal assumption labelling* if and only if $\text{IN}(LabAsm)$ is maximal (w.r.t. $\subseteq$) among all complete assumption labellings satisfying that for all preferred assumption labellings $LabAsm'$, $\text{IN}(LabAsm) \subseteq \text{IN}(LabAsm')$;
- a *semi-stable assumption labelling* if and only if $\text{UNDEC}(LabAsm)$ is minimal (w.r.t. $\subseteq$) among all complete assumption labellings;
- a *stable assumption labelling* if and only if $\text{UNDEC}(LabAsm) = \emptyset$.

Example 7. Let ABA_4 be the ABA framework with:

$\mathcal{L} = \{r, p, x, \rho, \psi, \chi\}$,
$\mathcal{R} = \{r \leftarrow \psi \,;\, p \leftarrow \rho \,;\, p \leftarrow \chi \,;\, x \leftarrow \psi \,;\, x \leftarrow \chi\}$,
$\mathcal{A} = \{\rho, \psi, \chi\}$,
$\overline{\rho} = r$, $\overline{\psi} = p$, $\overline{\chi} = x$.

ABA_4 has three complete assumption labellings:

- $LabAsm_1 = \{(\rho, \text{UNDEC}), (\psi, \text{UNDEC}), (\chi, \text{UNDEC})\}$,
- $LabAsm_2 = \{(\rho, \text{OUT}), (\psi, \text{IN}), (\chi, \text{OUT})\}$, and
- $LabAsm_3 = \{(\rho, \text{IN}), (\psi, \text{OUT}), (\chi, \text{UNDEC})\}$.

$LabAsm_1$ is the grounded assumption labelling, $LabAsm_2$ and $LabAsm_3$ are both preferred assumption labellings, $LabAsm_1$ is the ideal assumption labelling, and $LabAsm_2$ is the only stable as well as the only semi-stable assumption labelling.

The following theorem proves that the grounded, preferred, ideal, semi-stable, and stable assumption labellings correspond one-to-one to the respective assumption extensions.

Theorem 5.

1. *Let Asms be a grounded/preferred/ideal/semi-stable/stable assumption extension. Then LabAsm with* $\text{IN}(LabAsm) = Asms$*,* $\text{OUT}(LabAsm) = Asms^+$*, and* $\text{UNDEC}(LabAsm) = \mathcal{A} \setminus (Asms \cup Asms^+)$ *is a grounded/preferred/ideal/semi-stable/stable assumption labelling.*
2. *Let LabAsm be a grounded/preferred/ideal/semi-stable/stable assumption labelling. Then* $Asms = \text{IN}(LabAsm)$ *is a grounded/preferred/ideal/semi-stable/stable assumption extension with* $Asms^+ = \text{OUT}(LabAsm)$ *and* $\mathcal{A} \setminus (Asms \cup Asms^+) = \text{UNDEC}(LabAsm)$*.*

Corollary 6 follows straightaway from the correspondence of grounded and ideal assumption labellings and extensions and the uniqueness of grounded and ideal assumption extensions [4, 23].

Corollary 6. *The grounded and ideal assumption labelling are both unique.*

We now show that preferred, ideal, semi-stable, and stable assumption labellings can be redefined in terms of admissible (rather than complete) assumption labellings.

Proposition 7. *Let LabAsm be an admissible assumption labelling.*

- *LabAsm is a preferred assumption labelling if and only if* $\text{IN}(LabAsm)$ *is maximal (w.r.t. $\subseteq$) among all admissible assumption labellings.*
- *LabAsm is an ideal assumption labelling if and only if* $\text{IN}(LabAsm)$ *is maximal (w.r.t. $\subseteq$) among all admissible assumption labellings satisfying that for all preferred assumption labellings* $LabAsm'$*,* $\text{IN}(LabAsm) \subseteq \text{IN}(LabAsm')$*.*
- *LabAsm is a semi-stable assumption labelling if and only if* $\text{UNDEC}(LabAsm)$ *is minimal (w.r.t. $\subseteq$) among all admissible assumption labellings.*
- *LabAsm is a stable assumption labelling if and only if* $\text{UNDEC}(LabAsm) = \emptyset$*.*

Example 8. The results from Proposition 7 are illustrated by ABA_1 (see Examples 1 and 5). For example, the only maximal admissible assumption labelling of ABA_1 is $LabAsm_3$, which is also the only maximal complete, and thus preferred, assumption labelling of ABA_1.

4. Argument-supporting sets of assumptions

Determining admissible or complete assumption labellings of an ABA framework as well as checking whether an assumption labelling is admissible or complete can be cumbersome since some conditions specifying the label of an assumption α require to consider *every* set of assumptions attacking α. In particular, not only the set of premises of an argument whose conclusion is the contrary of α attacks α, but also every *superset* thereof.

Example 9. To verify whether χ is correctly labelled in the admissible assumption labelling $LabAsm_3 = \{(\rho, \text{IN}), (\psi, \text{OUT}), (\chi, \text{IN})\}$ of ABA_1 (see Example 1), not only the set of assumptions $\{\psi\}$, which forms the premises of an argument with conclusion x (the contrary of χ), but also every superset thereof, i.e. $\{\rho, \psi\}$, $\{\psi, \chi\}$, and $\{\rho, \psi, \chi\}$, has to be checked.

In this section, we show that considering only sets of assumptions which form the premises of some argument, which we call *argument-supporting* sets of assumptions, when determining or checking assumption labellings is equivalent to considering all sets of assumptions. This is inspired by the fact that assumption extensions can be determined and checked by considering either all or only argument-supporting sets of assumptions [33].

4.1. Assumption labellings with respect to argument-supporting sets of assumptions

A set of assumptions is *argument-supporting* if it forms the premises of some argument.

Definition 5. Let $Asms \subseteq \mathcal{A}$ be a set of assumptions. $Asms$ is an *argument-supporting* set of assumptions if and only if there exists some $s \in \mathcal{L}$ such that $Asms \vdash s$ is an argument.

Note that all singleton sets of assumptions are argument-supporting, i.e. for every assumption $\alpha \in \mathcal{A}$, $\{\alpha\}$ is an argument-supporting set of assumptions since $\{\alpha\} \vdash \alpha$ is an argument.

Notation 6. The set of all argument-supporting sets of assumptions is $\mathcal{S}_{arg} = \{Asms \subseteq \mathcal{A} \mid Asms \textit{ is an argument-supporting set of assumptions}\}$.

We define a variant of admissible assumption labellings where only argument-supporting, rather than all, sets of assumptions attacking an assumption are taken into account.

Definition 7. Let $LabAsm$ be an assumption labelling. $LabAsm$ is an *admissible assumption labelling with respect to argument-supporting sets* if and only if for each assumption $\alpha \in \mathcal{A}$ it holds that:

- if $LabAsm(\alpha) = \text{IN}$ then for each argument-supporting set of assumptions $Asms$ attacking α there exists some $\beta \in Asms$ such that $LabAsm(\beta) = \text{OUT}$;
- if $LabAsm(\alpha) = \text{OUT}$ then there exists an argument-supporting set of assumptions $Asms$ attacking α such that for all $\beta \in Asms$, $LabAsm(\beta) = \text{IN}$;
- if $LabAsm(\alpha) = \text{UNDEC}$ then for each argument-supporting set of assumptions $Asms$ attacking α there exists some $\beta \in Asms$ such that $LabAsm(\beta) \neq \text{IN}$.

To check whether χ in ABA_1 is correctly labelled according to admissible assumption labellings with respect to argument-supporting sets, only the set $\{\psi\}$ has to be taken into account (compare Example 9).

The following Lemma shows that the original definition of admissible assumption labellings (Definition 2) and the new definition of admissible assumption labellings with respect to argument-supporting sets can be used interchangeably. This extends the result of [33] that admissible assumption extensions can be equivalently defined in terms of all sets of assumptions or argument-supporting sets of assumptions.

Lemma 8. *Let LabAsm be an assumption labelling. LabAsm is an admissible assumption labelling if and only if LabAsm is an admissible assumption labelling with respect to argument-supporting sets.*

Analogously to admissible assumption labellings, we define a variant of complete assumption labellings, where only argument-supporting sets of assumptions attacking an assumption in question are taken into account.

Definition 8. Let *LabAsm* be an assumption labelling. *LabAsm* is a *complete assumption labelling with respect to argument-supporting sets* if and only if *LabAsm* is an admissible assumption labelling with respect to argument-supporting sets and for each assumption $\alpha \in \mathcal{A}$ it holds that:

- if $LabAsm(\alpha) = \text{UNDEC}$ then there exists an argument-supporting set of assumptions $Asms$ attacking α such that for all $\gamma \in Asms$, $LabAsm(\gamma) \neq \text{OUT}$.

As for admissible assumption labellings, the notions of complete assumption labellings and complete assumption labellings with respect to argument-supporting sets are equivalent.

Proposition 9. *Let LabAsm be an assumption labelling. LabAsm is a complete assumption labelling if and only if LabAsm is a complete assumption labelling with respect to argument-supporting sets.*

Since the grounded, preferred, ideal, semi-stable, and stable assumption labellings are based on complete/admissible assumption labellings, it follows that they can be equivalently defined in terms of complete/admissible assumption labellings with respect to argument-supporting sets.

Note that here we do not address the issue of computing argument-supporting sets of assumptions, as this is beyond the scope of this paper. However, we note that in order to determine complete assumption labellings, arguments have to be constructed when considering argument-supporting sets as well as when considering all sets of assumptions. Once arguments have been constructed for all sets of assumptions or argument-supporting sets, using argument-supporting sets may be beneficial for determining complete assumption labellings. The reason is that, depending on the ABA framework and in particular on the set of inference rules $\mathcal{R}$, the set of all argument-supporting sets of assumptions $\mathcal{S}_{arg}$ may be much smaller than or equal to the set of all sets of assumptions $\wp(\mathcal{A})$. For example, in ABA_1 from Example 1 the set of all argument-supporting sets of assumptions consists only of the singleton sets, i.e. $\{\{\rho\}, \{\psi\}, \{\chi\}\}$, whereas the set of all sets of assumptions is $\wp(\{\rho, \psi, \chi\}) = \{\{\}, \{\rho\}, \{\psi\}, \{\chi\}, \{\rho, \psi\}, \{\rho, \chi\}, \{\psi, \chi\}, \{\rho, \psi, \chi\}\}$. Therefore, considering only argument-supporting sets of assumptions when determining complete assumption labellings may in the best case require to check only a fraction of all sets of assumptions, but in the worst case it is exactly the same.

Observation 10. *Let $\mathcal{S}_{all} = \wp(\mathcal{A})$ be the set of all sets of assumptions, so $|\mathcal{S}_{all}| = 2^{|\mathcal{A}|}$.*

- *In the* best case, *$|\mathcal{S}_{arg}| = |\mathcal{A}|$. This is for example the case if $\mathcal{R} = \emptyset$, since the only argument-supporting sets of assumptions are the singleton sets.*
- *In the* worst case, *$|\mathcal{S}_{arg}| = |\mathcal{S}_{all}| = 2^{|\mathcal{A}|}$. This is for example the case if $\mathcal{R}$ is such that for each $Asms_i \in \mathcal{S}_{all}$ there exists some inference rule $s_i \leftarrow s_{i_1}, \ldots, s_{i_n} \in \mathcal{R}$ such that $Asms_i = \{s_{i_1}, \ldots, s_{i_n}\}$.*

Example 10. Let ABA_5 be the ABA framework with:

$$\mathcal{L} = \{p, r, \psi, \rho\},$$
$$\mathcal{R} = \{p \leftarrow;\ p \leftarrow \rho;\ r \leftarrow \psi;\ r \leftarrow \psi, \rho\},$$
$$\mathcal{A} = \{\psi, \rho\},$$
$$\overline{\psi} = p\ ,\ \overline{\rho} = r.$$

Here, the set of all argument-supporting sets of assumptions is $\mathcal{S}_{arg} = \{\{\}, \{\psi\}, \{\rho\}, \{\psi, \rho\}\}$, which is the same as the set of all sets of assumptions.

4.2. ABA graphs

In most of the ABA literature (e.g. [33, 23, 12, 34, 10, 35]), ABA frameworks are not displayed graphically; they are simply given as tuples, as done in the Examples presented so far. We introduce *ABA graphs*, where nodes are argument-supporting sets of assumptions and edges are attacks between these argument-supporting sets of assumptions.

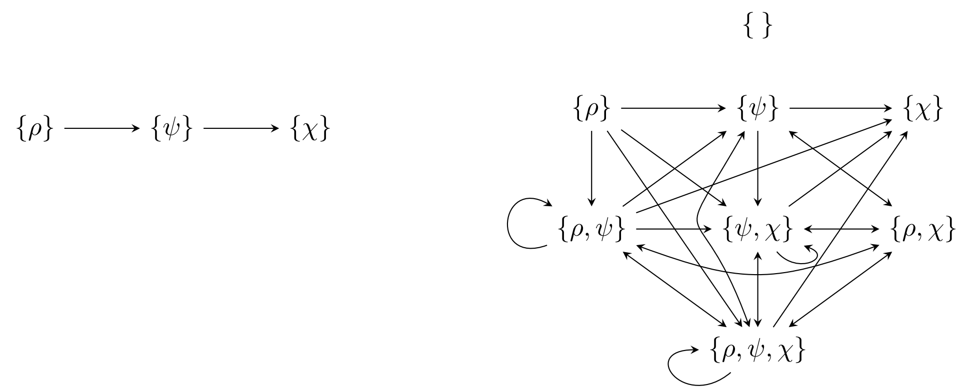

Figure 5: Left: the ABA graph of ABA_1. Right: the graph illustrating all sets of assumptions of ABA_1 and all attacks between them.

Definition 9. The *ABA graph* $\mathcal{G} = (V, E)$ is a directed graph with $V = \mathcal{S}_{arg}$ and $E = \{(Asms_1, Asms_2) \mid Asms_1, Asms_2 \in V$ *and* $Asms_1$ *attacks* $Asms_2\}$.

The ABA graph of ABA_1 from Example 1 has only three nodes, namely the singleton sets of assumptions, as shown on the left of Figure 5. As a comparison, the right of Figure 5 illustrates the graph of all sets of assumptions and attacks between them. Since an ABA graph illustrates all argument-supporting sets of assumptions and attacks between them, an ABA graph can be used to determine the semantics of an ABA framework.

Example 11. The ABA graph of ABA_4 (see Example 7) is displayed on the left of Figure 6. It illustrates which (argument-supporting) sets of assumptions have to be taken into account when determining complete or admissible assumption labellings (with respect to argument-supporting sets). For example, for ρ to be labelled IN by a complete assumption labelling (with respect to argument-supporting sets), all (argument-supporting) sets of assumptions attacking ρ have to contain an assumption labelled OUT. Since the only set of assumptions attacking ρ in the ABA graph is $\{\psi\}$, we deduce that ψ has to be labelled OUT by any complete assumption labelling that labels ρ as IN. It is then easy to verify, based on the two sets of assumptions attacking χ, that with ψ labelled OUT, χ can only be labelled UNDEC. This complete assumption labelling of ABA_4 is illustrated in the ABA graph on the right of Figure 6.

Figure 6: The ABA graph of ABA_4 (see Example 11). The right version also indicates one of the complete assumption labellings of ABA_4.

Another graphical representation of an ABA framework is the *attack relationship graph* [4], which was introduced to characterise different types of ABA frameworks. The question thus arises whether attack relationship graphs can also be used to determine the semantics, in particular the assumption labellings, of an ABA framework.

> The *attack relationship graph* $\mathcal{G}_{att} = (V, E)$ is a directed graph with $V = \mathcal{A}$ and $E = \{(\alpha, \beta) \mid \alpha, \beta \in V$ *and* $\alpha \in Asms$ *such that* $Asms$ *attacks* β *and* $\nexists Asms' \subset Asms$ *such that* $Asms'$ *attacks* $\beta\}$.

The main difference between an ABA graph and an attack relationship graph is that the vertices of an ABA graph are *sets* of assumptions, including all the singleton sets, whereas the vertices of an attack relationship graph are single assumptions. The following example demonstrates that attack relationship graphs do not capture enough information to determine the semantics of an ABA framework.

Example 12. Let ABA_6 be the ABA framework with:

$\mathcal{L} = \{\rho, \psi, \chi, \phi, \omega, r, p, x\}$,
$\mathcal{R} = \{r \leftarrow \phi\ ;\ r \leftarrow \omega\ ;\ r \leftarrow \psi, \chi\}$,
$\mathcal{A} = \{\rho, \psi, \chi, \phi, \omega\}$,
$\overline{\rho} = r$, $\overline{\psi} = p$, $\overline{\chi} = x$, $\overline{\phi} = \psi$, $\overline{\omega} = \psi$.

Furthermore, let ABA_7 have the same $\mathcal{L}$, $\mathcal{A}$, and contraries as ABA_6, but with $\mathcal{R} = \{r \leftarrow \phi\ ;\ r \leftarrow \psi, \omega\ ;\ r \leftarrow \chi, \omega\}$. The ABA graphs of ABA_6 and ABA_7, which are structurally different, are displayed in Figure 7. In contrast, the attack relationship graphs of ABA_6 and ABA_7 are the same, as illustrated in Figure 8. Thus, based on the attack relationship graph it is impossible to distinguish ABA_6 and ABA_7. However, the two ABA frameworks have different complete labellings, as indicated in Figure 7. It is therefore not possible to determine the complete or admissible assumption labellings of ABA_6 and ABA_7, and more generally of any ABA framework, based on their attack relationship graphs.

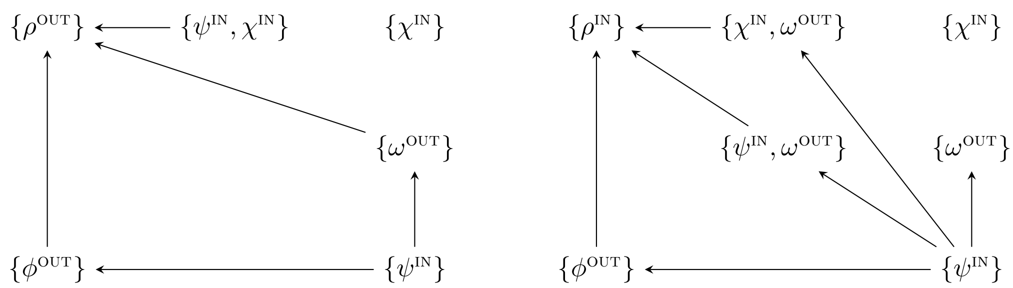

Figure 7: The ABA graphs of ABA_6 (left) and ABA_7 (right) from Example 12, each with their only complete assumption labelling.

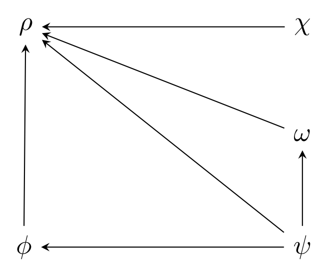

Figure 8: The attack relationship graph of both ABA_6 and ABA_7 from Example 12.

Conversely, ABA graphs cannot be (straightforwardly) used for the purpose of attack relationship graphs, i.e. to characterise different types of ABA frameworks. For example, an ABA framework is *stratified* if and only if its attack relationship graph does not have an infinite sequence of edges [4]. However, ABA graphs may have an infinite sequence of edges even if the attack relationship graph does not, as demonstrated in Example 13.

Example 13. Let ABA_8 be the following ABA framework:

$$\mathcal{L} = \{p, r, x, \psi, \rho, \chi\},$$
$$\mathcal{R} = \{r \leftarrow \psi\ ;\ x \leftarrow \rho\ ;\ r \leftarrow \psi, \chi\},$$
$$\mathcal{A} = \{\psi, \rho, \chi\},$$
$$\overline{\psi} = p\ ,\ \overline{\rho} = r\ ,\ \overline{\chi} = x.$$

The attack relationship graph and the ABA graph of ABA_8 are displayed in Figure 9. Since the attack relationship graph does not have any infinite sequence of edges, ABA_8 is stratified. However, the ABA graph does have an infinite sequence of edges since it comprises a cycle.

Figure 9: The attack relationship graph (left) and the ABA graph (right) of ABA_8 from Example 13.

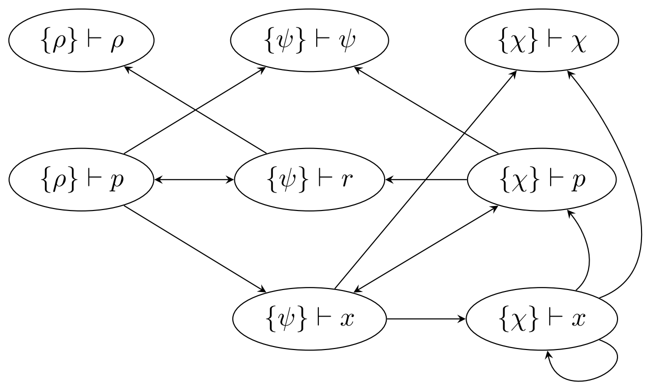

Figure 10: AA graph of the corresponding AA framework of ABA_4.

The example illustrates that an infinite sequence of edges in an ABA graph does not indicate that the ABA framework is not stratified.

Proposition 11. *Let $\langle \mathcal{L}, \mathcal{R}, \mathcal{A}, ^{-} \rangle$ be an ABA framework and let $\mathcal{G}$ be its ABA graph and $\mathcal{G}_{att}$ its attack relationship graph. If $\mathcal{G}_{att}$ has an infinite sequence of edges then $\mathcal{G}$ has an infinite sequence of edges, but not vice versa.*

Another way to graphically represent an ABA framework is in terms of the *AA graph* of its corresponding AA framework, as for example done in [24, 22]. Interestingly, even though nodes in an ABA graph are *argument*-supporting sets of assumptions, an ABA graph does not generally have the same number of nodes as the AA graph of the corresponding AA framework, where nodes are *arguments*. In particular, an AA graph may have more nodes than an ABA graph since the same set of assumptions may form the set of premises of various arguments. As an example, compare the ABA graph of ABA_4 shown in Figure 6 with the AA graph of its corresponding AA framework illustrated in Figure 10.

Recently a new way to represent arguments of an ABA framework has been introduced with the purpose of eliminating redundancies in arguments [36], namely as *argument graphs* rather than tree-structured arguments. Various argument graphs can furthermore be combined to form a larger argument graph, which represents a set of arguments without redundancies. Since the semantics of ABA frameworks in terms of argument graphs is slightly different from the semantics of ABA frameworks in terms of assumption and argument extensions [36], a detailed comparison between argument graphs and ABA graphs goes beyond the scope of this paper.

5. Assumption labellings versus argument labellings

In this section, we examine the relationship between assumption labellings of an ABA framework and argument labellings of its corresponding AA framework. In the remainder, and if clear from the context, we assume as given a flat ABA framework $\langle \mathcal{L}, \mathcal{R}, \mathcal{A}, ^{-} \rangle$ and its corresponding AA framework $\langle Ar_{ABA}, Att_{ABA} \rangle$ (see Section 2).

5.1. Translating between assumption and argument labellings

Before going into detail about the (non-) correspondence between assumption and argument labellings according to the various semantics, we examine the relationship between assumption and argument labellings in general.

Notation 10. $\mathscr{L}_{Asm}$ denotes the set of all assumption labellings of $\langle \mathcal{L}, \mathcal{R}, \mathcal{A}, ^- \rangle$ and $\mathscr{L}_{Arg}$ the set of all argument labellings of $\langle Ar_{ABA}, Att_{ABA} \rangle$.

First, we observe that the number of all possible assumption labellings of an ABA framework is smaller than or equal to the number of all possible argument labellings of its corresponding AA framework since an assumption labelling labels only assumptions, i.e. $|\mathcal{A}|$ elements, whereas an argument labelling labels every assumption-argument as well as every argument constructed using inference rules in $\mathcal{R}$.

Observation 12. *Since assumption labellings assign one of three labels to each assumption,* $|\mathscr{L}_{Asm}| = 3^{|\mathcal{A}|}$. *Since argument labellings assign one of three labels to each argument,* $|\mathscr{L}_{Arg}| = 3^{|Ar_{ABA}|}$.

- *In the* best case $|\mathscr{L}_{Arg}| = |\mathscr{L}_{Asm}| = 3^{|\mathcal{A}|}$. *This is the case if the only arguments are assumption-arguments, so* $|Ar_{ABA}| = |\mathcal{A}|$*, for example if* $\mathcal{R} = \emptyset$.

- *In* all other cases $|\mathscr{L}_{Arg}| > |\mathscr{L}_{Asm}|$. *This is the case if there exists at least one argument which is not an assumption-argument, so* $|Ar_{ABA}| > |\mathcal{A}|$*, for example if there exists an inference rule* $s \leftarrow \in \mathcal{R}$.

As an example, ABA_4 from Example 7 has three assumptions, so there are $|\mathscr{L}_{Asm}| = 3^3 = 27$ possible assumption labellings. In contrast, the corresponding AA framework of ABA_4 has eight arguments (see Figure 10), so there are $|\mathscr{L}_{Arg}| = 3^8 = 6561$ possible argument labellings. We will see in the following sections that even though the number of possible assumption labellings of an ABA framework may be less than the number of possible argument labellings of its corresponding AA framework, the number of assumption and argument labellings according to various semantics is the same.

In order to compare assumption and argument labellings, we define two functions for translating between the two types of labellings. The first translation, `LabAsm2LabArg`, determines the labels of arguments based on the given labels of premises of these arguments.

Definition 11. `LabAsm2LabArg` : $\mathscr{L}_{Asm} \rightarrow \mathscr{L}_{Arg}$ maps an assumption labelling $LabAsm$ to an argument labelling $LabArg$ such that:

- $\texttt{in}(LabArg) = \{Asms \vdash s \in Ar_{ABA} \mid Asms \subseteq \textsc{in}(LabAsm)\}$,
- $\texttt{out}(LabArg) = \{Asms \vdash s \in Ar_{ABA} \mid \exists \alpha \in Asms : \alpha \in \textsc{out}(LabAsm)\}$,
- $\texttt{undec}(LabArg) = \{Asms \vdash s \in Ar_{ABA} \mid \exists \alpha \in Asms : \alpha \in \textsc{undec}(LabAsm),$ $Asms \cap \textsc{out}(LabAsm) = \emptyset\}$.

`LabAsm2LabArg` mirrors the correspondence between assumption and argument extensions (see Section 2.3) through the mapping from IN labelled assumption to `in` labelled arguments. In addition, `LabAsm2LabArg` maps OUT and UNDEC labelled assumptions to `out` and `undec` labelled arguments, respectively. An argument is labelled `out` if one of its premises is labelled OUT, independently of the labels of its other premises. The intuition of this translation is that an assumption α which is labelled OUT is attacked by a set of assumptions labelled IN. Since this set gives rise to an `in` labelled argument, any argument which has α as its premise is attacked by an `in` labelled argument and should thus be labelled `out`. Arguments labelled `undec` are simply those whose premises fulfil neither the condition for `in` nor for `out` labelled arguments.

Lemma 13. `LabAsm2LabArg` *is an injective function but not generally a surjective function.*

The second translation, `LabArg2LabAsm`, determines the labels of assumptions based on the given labels of assumption-arguments.

Definition 12. `LabArg2LabAsm` $: \mathscr{L}_{Arg} \rightarrow \mathscr{L}_{Asm}$ maps an argument labelling *LabArg* to an assumption labelling *LabAsm* such that:

- $\text{IN}(LabAsm) = \{\alpha \in \mathcal{A} \mid \{\alpha\} \vdash \alpha \in \texttt{in}(LabArg)\}$,
- $\text{OUT}(LabAsm) = \{\alpha \in \mathcal{A} \mid \{\alpha\} \vdash \alpha \in \texttt{out}(LabArg)\}$,
- $\text{UNDEC}(LabAsm) = \{\alpha \in \mathcal{A} \mid \{\alpha\} \vdash \alpha \in \texttt{undec}(LabArg)\}$.

In contrast to `LabAsm2LabArg`, the translation from `in` labelled arguments to IN labelled assumptions in terms of `LabArg2LabAsm` does not mirror the correspondence between argument and assumption extensions (see Section 2). In particular, the set of IN labelled assumptions consists of all assumptions whose *assumption-arguments* are labelled `in`, rather than of all assumptions occurring as the premise of *any argument* labelled `in` (which would mirror the correspondence between argument and assumption extensions). This is to ensure that the translation of *any* argument labelling results in a well-defined assumption labelling.

Example 14. Let ABA_9 be the ABA framework with:

$$\mathcal{L} = \{\rho, \psi, r, p\},$$
$$\mathcal{R} = \{r \leftarrow \rho\},$$
$$\mathcal{A} = \{\rho, \psi\},$$
$$\overline{\rho} = \psi,\ \overline{\psi} = p.$$

The corresponding AA framework of ABA_9 has three arguments, $A_1 : \{\rho\} \vdash \rho$, $A_2 : \{\psi\} \vdash \psi$, and $A_3 : \{\rho\} \vdash r$. Let *LabArg* be the argument labelling $\{(A_1, \texttt{out}), (A_2, \texttt{out}), (A_3, \texttt{in})\}$. Then $\texttt{LabArg2LabAsm}(LabArg) = \{(\psi, \text{IN}), (\rho, \text{OUT})\}$ is a well-defined assumption labelling. However, if the set of assumptions labelled IN was defined in such a way that it mirrors the correspondence between argument and assumption extensions, i.e. $\text{IN}(LabAsm) = \{\alpha \in \mathcal{A} \mid \exists Asms \vdash s \in \texttt{in}(LabArg), \alpha \in Asms\}$, then $\rho \in \text{IN}$ because $A_3 \in \texttt{in}(LabArg)$ but also $\rho \in \text{OUT}$ because $A_1 \in \texttt{out}(LabArg)$.

Note that if `LabArg2LabAsm` was a function between admissible or *complete*, rather than arbitrary, argument and assumption labellings, the translation from `in` labelled arguments to IN labelled assumptions could mirror the correspondence between argument and assumption extensions [27].

Lemma 14. `LabArg2LabAsm` *is a surjective function but not generally an injective function.*

5.2. Complete semantics

Due to the one-to-one correspondence between complete assumption labellings and extensions (Theorem 4), between complete assumption and argument extensions [20, 24], and between complete argument extensions and labellings [25], there is also a one-to-one correspondence between complete assumption labellings and complete argument labellings. Theorem 15 below characterises the complete argument labelling corresponding to a given complete assumption labelling in terms of the mapping `LabAsm2LabArg`.

Theorem 15. *Let LabAsm be an assumption labelling. LabAsm is a complete assumption labelling if and only if* `LabAsm2LabArg`$(LabAsm)$ *is a complete argument labelling.*

Note that in addition to proving that every complete assumption labelling is translated to a corresponding complete argument labelling by `LabAsm2LabArg`, analogous to the correspondence between complete assumption and argument extensions [24], Theorem 15 also proves that for every complete argument labelling which is the translation of some assumption labelling *LabAsm* in terms of `LabAsm2LabArg`, *LabAsm* is a complete assumption labelling.

Since `LabAsm2LabArg` is injective but not generally surjective (see Lemma 13), there may be an argument labelling *LabArg* which is not the translation of any assumption labelling in terms of `LabAsm2LabArg`, so a natural question is whether *LabArg* may be a complete argument labelling. The following Proposition shows that this is not the case, i.e. every complete argument labelling is the translation of some assumption labelling in terms of `LabAsm2LabArg`.

Proposition 16. *Let LabArg be a complete argument labelling. Then there exists a unique assumption labelling LabAsm such that* `LabAsm2LabArg`$(LabAsm) = LabArg$*.*

It follows directly from Theorem 15 that *LabAsm* is a complete assumption labelling.

We now examine the translation from argument to assumption labellings in terms of `LabArg2LabAsm`. Theorem 17 below shows that the translation of a complete argument labelling yields a complete assumption labelling.

Theorem 17. *Let LabArg be an argument labelling. If LabArg is a complete argument labelling then* `LabArg2LabAsm`$(LabArg)$ *is a complete assumption labelling.*

Note that since `LabArg2LabAsm` is surjective (see Lemma 14) the converse of Theorem 17 does not hold, i.e. a complete assumption labelling *LabAsm* may be the translation of some argument labelling in terms of `LabArg2LabAsm` which is not a complete argument labelling, as illustrated by the following example.

Example 15. ABA_9 from Example 14 has only one complete assumption labelling $LabAsm_1 = \{(\rho, \text{OUT}), (\psi, \text{IN})\}$. The corresponding AA framework of ABA_9 has three arguments, $A_1 : \{\rho\} \vdash \rho$, $A_2 : \{\psi\} \vdash \psi$, and $A_3 : \{\rho\} \vdash r$, and one complete argument labelling, $LabArg_1 = \{(A_1, \texttt{out}), (A_2, \texttt{in}), (A_3, \texttt{out})\}$. It holds that `LabArg2LabAsm`$(LabArg_1) = LabAsm_1$, but also that for $LabArg_2 = \{(A_1, \texttt{out}),$
$(A_2, \texttt{in}), (A_3, \texttt{undec})\}$ `LabArg2LabAsm`$(LabArg_2) = LabAsm_1$, where $LabArg_2$ is not a complete argument labelling.

However, a weaker version of the converse of Theorem 17 holds: Every complete assumption labelling is the translation of some complete argument labelling in terms of `LabArg2LabAsm`.

Lemma 18. *Let LabAsm be a complete assumption labelling. Then there exists a complete argument labelling LabArg such that* $\texttt{LabArg2LabAsm}(LabArg) = LabAsm$.

Even though there may be multiple argument labellings which are translated to the same complete assumption labelling in terms of `LabArg2LabAsm`, there are no two *complete* argument labellings which are translated to the same assumption labelling.

Lemma 19. *Let* $LabArg_1 \neq LabArg_2$ *be two complete argument labellings. Then* $\texttt{LabArg2LabAsm}(LabArg_1) \neq \texttt{LabArg2LabAsm}(LabArg_2)$.

Lemmas 18 and 19 imply that every complete assumption labelling is the translation of a *unique* complete argument labelling in terms of `LabArg2LabAsm`.

Corollary 20. *Let LabAsm be a complete assumption labelling. Then there exists a unique complete argument labelling LabArg such that* $\texttt{LabArg2LabAsm}(LabArg) = LabAsm$.

From Theorems 15 and 17, and Proposition 16, and Lemmas 18, and 19 it follows that there is a one-to-one correspondence between complete argument labellings and complete assumption labellings in terms of both `LabArg2LabAsm` and `LabAsm2LabArg`. Thus, when restricting `LabArg2LabAsm` and `LabAsm2LabArg` to complete argument and assumption labellings, they are bijective functions as well as the inverse of one another (see [24] for the analogous result about complete argument and assumption extensions).

Corollary 21. *Let* $\mathscr{L}_{AsmComp}$ *be the set of all complete assumption labellings of* $\langle \mathcal{L}, \mathcal{R}, \mathcal{A}, ^{-} \rangle$ *and* $\mathscr{L}_{ArgComp}$ *the set of all complete argument labellings of* $\langle Ar_{ABA}, Att_{ABA} \rangle$*. Let*

- $\texttt{LabArg2LabAsm}' : \mathscr{L}_{ArgComp} \to \mathscr{L}_{AsmComp}$ *such that* $\forall LabArg \in \mathscr{L}_{ArgComp}$: $\texttt{LabArg2LabAsm}'(LabArg) = \texttt{LabArg2LabAsm}(LabArg)$
- $\texttt{LabAsm2LabArg}' : \mathscr{L}_{AsmComp} \to \mathscr{L}_{ArgComp}$ *such that* $\forall LabAsm \in \mathscr{L}_{AsmComp}$: $\texttt{LabAsm2LabArg}'(LabAsm) = \texttt{LabAsm2LabArg}(LabAsm)$.

$\texttt{LabArg2LabAsm}'$ *and* $\texttt{LabAsm2LabArg}'$ *are bijective functions and each other's inverse.*

5.3. Grounded, preferred, ideal, and stable semantics

Due to existing correspondence results between grounded, preferred, ideal, and stable argument labellings and extensions [25, 32], argument and assumption extensions [23, 20, 24], and assumption extensions and labellings (see Section 3.3), the one-to-one correspondence between grounded, preferred, ideal, and stable assumption and argument labellings can be proven in a similar way as for complete assumption and argument labellings.

Theorem 22 states the relationship between a given grounded/preferred/ideal/stable assumption labelling and a grounded/preferred/ideal/stable argument labelling in terms of `LabAsm2LabArg`.

Theorem 22. *Let LabAsm be an assumption labelling. LabAsm is a grounded/preferred/ideal/stable assumption labelling if and only if* $\texttt{LabAsm2LabArg}(LabAsm)$ *is a grounded/preferred/ideal/stable argument labelling.*

Theorem 23 states the relationship between a given grounded/preferred/ideal/stable argument labelling and a grounded/preferred/ideal/stable assumption labelling in terms of `LabArg2LabAsm`.

Theorem 23. *Let LabArg be an argument labelling. If LabArg is a grounded/preferred/ideal/stable argument labelling then* `LabArg2LabAsm`($LabArg$) *is a grounded/preferred/ideal/stable assumption labelling.*

Note that as for the complete semantics the converse of Theorem 23 does not hold. A counter-example is ABA_9 from Example 15 whose only grounded/preferred/ideal/stable assumption labelling is $LabAsm_1$, which is the translation of the argument labelling $LabArg_2$ in terms of `LabArg2LabAsm`, but $LabArg_2$ is not a grounded/preferred/ideal/stable argument labelling.

Due to the one-to-one correspondence between complete assumption and argument labellings (see Corollary 21), it is straightforward that there is also a one-to-one correspondence between grounded/preferred/ideal/stable argument and assumption labellings in terms of `LabAsm2LabArg` and `LabArg2LabAsm`.

5.4. Semi-stable semantics

In contrast to the grounded, preferred, ideal, and stable semantics, semi-stable assumption and argument extensions are not in a one-to-one correspondence [24]. Since semi-stable assumption labellings correspond to semi-stable assumption extensions (Theorem 5) and semi-stable argument labellings to semi-stable argument extensions [25], it follows that there is no one-to-one correspondence between semi-stable assumption and argument labellings in terms of `LabAsm2LabArg` and `LabArg2LabAsm`. However, since semi-stable assumption and argument labellings are complete labellings, the translation of a semi-stable assumption labelling in terms of `LabAsm2LabArg` is of course a complete argument labelling and the translation of a semi-stable argument labelling in terms of `LabArg2LabAsm` is a complete assumption labelling.

The following example illustrates an ABA framework where all semi-stable argument labellings are translated to semi-stable assumption labellings by `LabArg2LabAsm`, but not all semi-stable assumption labellings are translated to semi-stable argument labellings by `LabAsm2LabArg`.

Example 16. Let ABA_{10} be the ABA framework with:

$$\mathcal{L} = \{\rho, \psi, \chi, x\},$$
$$\mathcal{R} = \{x \leftarrow \psi, \chi\},$$
$$\mathcal{A} = \{\rho, \psi, \chi\},$$
$$\overline{\rho} = \psi,\ \overline{\psi} = \rho,\ \overline{\chi} = \chi.$$

ABA_{10} has three complete assumption labellings: $LabAsm_1$ labels all assumptions as UNDEC, and $LabAsm_2$ and $LabAsm_3$ are as illustrated in the ABA graphs in Figure 11. Both $LabAsm_2$ and $LabAsm_3$ are semi-stable assumption labellings of ABA_{10}.

The corresponding AA framework of ABA_{10} is shown in Figure 12, along with two of its complete argument labellings $LabArg_2$ and $LabArg_3$. The third complete argument labelling $LabArg_1$ labels all arguments as `undec`. Only $LabArg_2$ is a semi-stable argument labelling. Thus, `LabArg2LabAsm` translates all semi-stable argument labellings to semi-stable assumption labellings, namely `LabArg2LabAsm`($LabArg_2$) $= LabAsm_2$, but `LabAsm2LabArg` does not translate all semi-stable assumption labellings to semi-stable argument labellings since `LabAsm2LabArg`($LabAsm_3$) $= LabArg_3$.

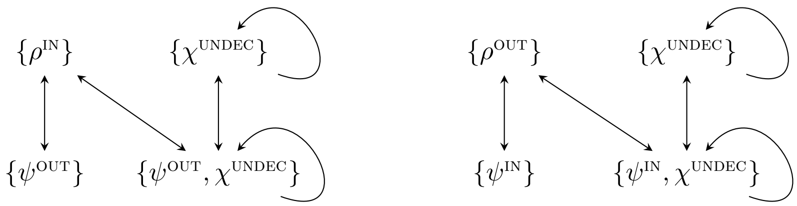

Figure 11: The ABA graph of ABA_{10} with two of its complete assumption labellings $LabAsm_2$ (left) and $LabAsm_3$ (right), which are both semi-stable assumption labellings (see Example 16).

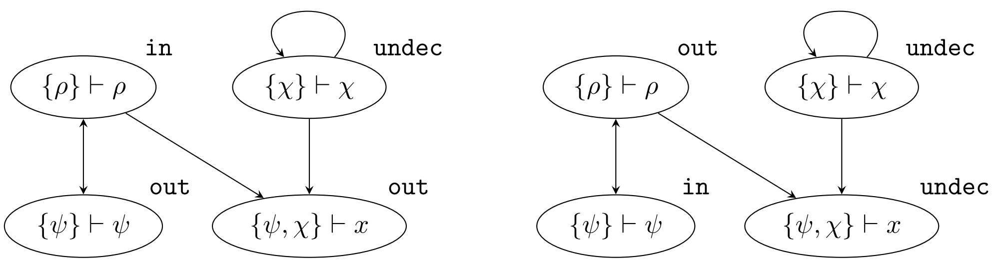

Figure 12: The AA graph of the corresponding AA framework of ABA_{10} with two of its complete argument labellings $LabArg_2$ (left) and $LabArg_3$ (right). Only $LabArg_2$ is a semi-stable argument labelling (see Example 16).

The next example illustrates an ABA framework where all semi-stable assumption labellings are translated to semi-stable argument labellings by `LabAsm2LabArg`, but not all semi-stable argument labellings are translated to semi-stable assumption labellings by `LabArg2LabAsm`.

Example 17. Let ABA_{11} be the following ABA framework:

$$\mathcal{L} = \{\rho, \psi, \chi, \omega, x, w\},$$
$$\mathcal{R} = \{x \leftarrow \psi, \chi; w \leftarrow \omega; w \leftarrow \psi\},$$
$$\mathcal{A} = \{\rho, \psi, \chi, \omega\},$$
$$\overline{\rho} = \psi,\ \overline{\psi} = \rho,\ \overline{\chi} = \chi,\ \overline{\omega} = w.$$

ABA_{11} has three complete assumption labellings: $LabAsm_1$ labels all assumptions as UNDEC, and $LabAsm_2$ and $LabAsm_3$ are as illustrated in the ABA graphs in Figure 13. Only $LabAsm_3$ is a semi-stable assumption labelling.

The corresponding AA framework of ABA_{11} is shown in Figure 14, along with two of its complete argument labellings $LabArg_2$ and $LabArg_3$. The third complete argument labelling $LabArg_1$ labels all arguments as `undec`. Both $LabArg_2$ and $LabArg_3$ are semi-stable argument labellings. Thus, `LabAsm2LabArg` translates all semi-stable assumption labellings to semi-stable argument labellings, namely `LabAsm2LabArg`$(LabAsm_3) = LabArg_3$, but `LabArg2LabAsm` does not translate all semi-stable argument labellings to semi-stable assumption labellings since `LabArg2LabAsm`$(LabArg_2) = LabAsm_2$.

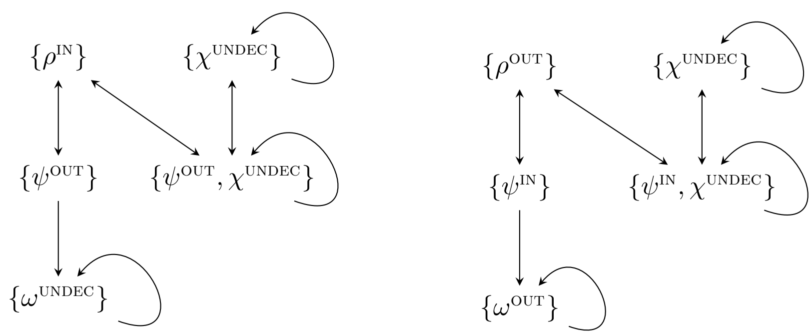

Figure 13: The ABA graph of ABA_{11} with two of its complete assumption labellings $LabAsm_2$ (left) and $LabAsm_3$ (right). Only $LabAsm_3$ is a semi-stable assumption labelling (see Example 17).

Note that ABA_{10} and ABA_{11} are special cases illustrating that semi-stable assumption and argument labellings do not correspond in general. However, there are also cases where semi-stable argument and assumption labellings correspond as demonstrated by the following example.

Example 18. Let ABA_{12} be the same as ABA_{11} but with $\overline{\chi} = x$. Then $LabAsm_1$ and $LabAsm_3$ are complete assumption labellings as before, but in $LabAsm_2$ χ is labelled IN rather than UNDEC, so both $LabAsm_2$ and $LabAsm_3$ are semi-stable assumption labellings. The corresponding AA framework of ABA_{12} has the same complete argument labellings $LabArg_1$ and $LabArg_3$ as he corresponding AA framework of ABA_{11}, but in $LabArg_2$ the argument $\{\chi\} \vdash \chi$ is labelled `in` rather than `undec`. Thus, both $LabArg_2$ and $LabArg_3$ are semi-stable argument labellings, corresponding to the two semi-stable assumption labellings of ABA_{12} in terms of `LabAsm2LabArg` and `LabArg2LabAsm`.

5.5. Admissible semantics

We have shown in Theorem 1 that admissible assumption extensions and labellings are in a one-to-one correspondence. Furthermore, we know that admissible assumption and argument extensions correspond [23], but this correspondence is not one-to-one as for the complete, grounded, preferred, ideal, and stable semantics, but one-to-many as illustrated by the following example.

Example 19. Let ABA_{13} be the ABA framework with

$$\mathcal{L} = \{\rho, \psi, p\},$$
$$\mathcal{R} = \{p \leftarrow \psi\},$$
$$\mathcal{A} = \{\rho, \psi\},$$
$$\overline{\rho} = \psi,\ \overline{\psi} = \rho.$$

The admissible assumption extensions of ABA_{13} are $Asms_1 = \{\}$, $Asms_2 = \{\rho\}$ and $Asms_3 = \{\psi\}$. The corresponding AA framework has three arguments, $A_1 : \{\rho\} \vdash \rho$, $A_2 : \{\psi\} \vdash \psi$, and $A_3 : \{\psi\} \vdash p$, and four admissible argument extensions, $Args_1 = \{\}$, $Args_2 = \{A_1\}$, $Args_3 = \{A_2\}$, and $Args_4 = \{A_2, A_3\}$. $Args_3$ and $Args_4$ both correspond to the admissible assumption extension $Asms_3$ in the sense that $Asms_3$ is the set of all

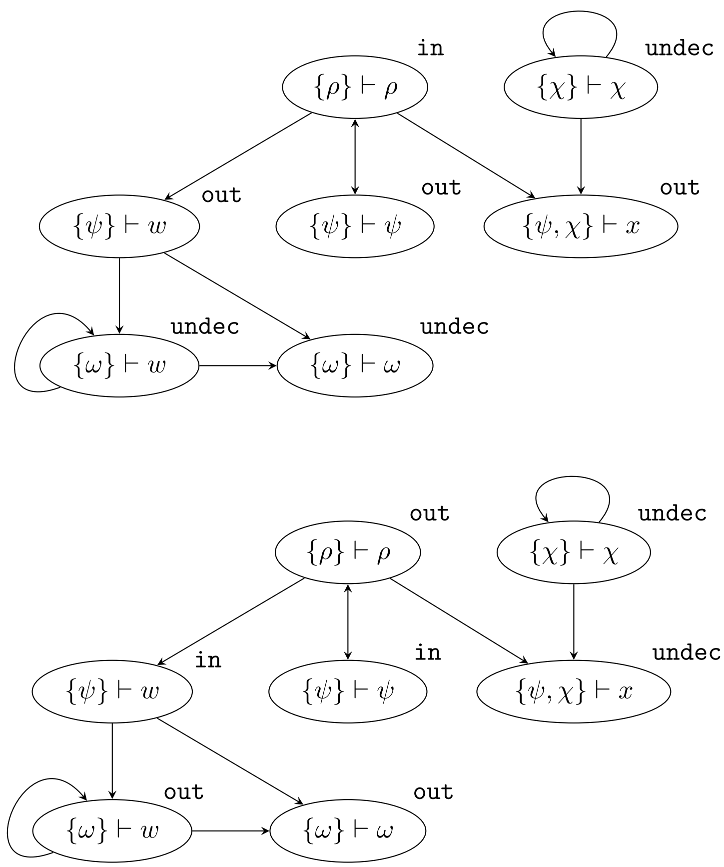

Figure 14: The AA graph of the corresponding AA framework of ABA_{11} with two of its complete argument labellings $LabArg_2$ (top) and $LabArg_3$ (bottom), which are both semi-stable argument labellings (see Example 17).

assumptions occurring in the premises of arguments in both $Args_3$ and $Args_4$ [23]. Conversely, only $Args_4$ corresponds to $Asms_3$ in the sense that it is the set of all arguments whose premises are contained in $Asms_3$ [23].

In addition, the correspondence between admissible argument extensions and labellings is one-to-many rather than one-to-one [25]. This implies that the correspondence between admissible assumption and argument labellings is one-to-many rather than one-to-one. Thus, only some of the correspondence results analogous to those for complete semantics hold for admissible semantics.

Theorem 24. *Let LabAsm be an assumption labelling. If LabAsm is an admissible assumption labelling then* `LabAsm2LabArg`$(LabAsm)$ *is an admissible argument labelling.*

Example 20. Consider again ABA_{13} from Example 19. ABA_{13} has the same number of admissible assumption labellings and extensions, which correspond one-to-one:

- $LabAsm_1 = \{(\rho, \text{UNDEC}), (\psi, \text{UNDEC})\}$ corresponds to $Asms_1$;

- $LabAsm_2 = \{(\rho, \text{IN}), (\psi, \text{OUT})\}$ corresponds to $Asms_2$; and
- $LabAsm_3 = \{(\rho, \text{OUT}), (\psi, \text{IN})\}$ corresponds to $Asms_3$.

In contrast, the corresponding AA framework of ABA_{13} has eight admissible argument labellings, even though it has only four admissible argument extensions:

- $LabArg_{11} = \{(A_1, \texttt{undec}), (A_2, \texttt{undec}), (A_3, \texttt{undec})\}$ corresponds to $Args_1$;
- $LabArg_{21} = \{(A_1, \texttt{in}), (A_2, \texttt{undec}), (A_3, \texttt{undec})\}$,
  $LabArg_{22} = \{(A_1, \texttt{in}), (A_2, \texttt{undec}), (A_3, \texttt{out})\}$,
  $LabArg_{23} = \{(A_1, \texttt{in}), (A_2, \texttt{out}), (A_3, \texttt{undec})\}$, and
  $LabArg_{24} = \{(A_1, \texttt{in}), (A_2, \texttt{out}), (A_3, \texttt{out})\}$ all correspond to $Args_2$;
- $LabArg_{31} = \{(A_1, \texttt{undec}), (A_2, \texttt{in}), (A_3, \texttt{undec})\}$, and
  $LabArg_{32} = \{(A_1, \texttt{out}), (A_2, \texttt{in}), (A_3, \texttt{undec})\}$ both correspond to $Args_3$; and
- $LabArg_{41} = \{(A_1, \texttt{out}), (A_2, \texttt{in}), (A_3, \texttt{in})\}$ corresponds to $Args_4$.

The translation of each admissible assumption labelling in terms of `LabAsm2LabArg` is an admissible argument labelling, i.e. $\texttt{LabAsm2LabArg}(LabAsm_1) = LabArg_{11}$, $\texttt{LabAsm2LabArg}(LabAsm_2) = LabArg_{24}$, and $\texttt{LabAsm2LabArg}(LabAsm_3) = LabArg_{41}$.

The following example shows that the converse of Theorem 24 does however not hold.

Example 21. In ABA_{13} from Example 20 $\texttt{LabAsm2LabArg}(\{(\rho, \text{IN}), (\psi, \text{UNDEC})\}) = LabArg_{21}$, which is an admissible argument labelling, but $\{(\rho, \text{IN}), (\psi, \text{UNDEC})\}$ is not an admissible assumption labelling.

It is furthermore not the case that every admissible argument labelling is the translation of some admissible assumption labelling in terms of `LabAsm2LabArg` (i.e. the analogous result of Proposition 16 for the admissible semantics does not hold).

Example 22. Consider the admissible argument labelling $LabArg_{22}$ of ABA_{13} (see Example 20). There exists no admissible assumption labelling such that $LabArg_{22}$ is the translation in terms of `LabAsm2LabArg` since the arguments A_2 and A_3 have different labels even though their premises are the same.

Concerning `LabArg2LabAsm`, it is surprisingly not the case that the translation of every admissible argument labelling in terms of `LabArg2LabAsm` is an admissible assumption labelling (i.e. the analogous result of Theorem 17 for admissible semantics does not hold), as illustrated by the following example.

Example 23. Consider the admissible argument labelling $LabArg_{31}$ of ABA_{13} (see Example 20). $\texttt{LabArg2LabAsm}(LabArg_{31}) = \{(\rho, \text{UNDEC}), (\psi, \text{IN})\}$, which is not an admissible assumption labelling.

However, it holds that every admissible assumption labelling is the translation of some admissible argument labelling in terms of `LabArg2LabAsm`.

Proposition 25. *Let LabAsm be an admissible assumption labelling. Then there exists an admissible argument labelling LabArg such that* $\texttt{LabArg2LabAsm}(LabArg) = LabAsm$.

As in the case of complete assumption and argument labellings, an admissible assumption labelling may also be the translation of some argument labelling in terms of `LabArg2LabAsm` which is not an admissible argument labelling. For example, $\texttt{LabArg2LabAsm}(\{(A_1, \texttt{undec}), (A_2, \texttt{undec}), (A_3, \texttt{in})\}) = LabAsm_2$, where $LabAsm_2$ is an admissible assumption labelling but $\{(A_1, \texttt{undec}), (A_2, \texttt{undec}), (A_3, \texttt{in})\}$ is not an admissible argument labelling of ABA_{13} (see Example 20).

5.5.1. Committed admissible argument labellings

One of the reasons for the one-to-many correspondence between admissible assumption and argument labellings is the one-to-many correspondence between admissible argument extensions and labellings. This arises since admissible argument labellings make no restriction on arguments labelled `undec`, so any argument can be labelled `undec` in an admissible argument labelling. In contrast, admissible assumption labellings and extensions are in a one-to-one correspondence since admissible assumption labellings make restrictions on assumptions labelled UNDEC (see Section 3.1). We now introduce a variant of admissible argument labellings, which follows the spirit of admissible assumption labellings by restricting `undec` arguments to arguments which are not attacked by any `in` labelled arguments.

Definition 13. Let $\langle Ar, Att\rangle$ be an AA framework and let $LabArg$ be an argument labelling of $\langle Ar, Att\rangle$. $LabArg$ is a *committed admissible argument labelling* of $\langle Ar, Att\rangle$ if and only if for each argument $A \in Ar$ it holds that:

- if $LabArg(A) = \texttt{in}$ then for each $B \in Ar$ attacking A, $LabArg(B) = \texttt{out}$;
- if $LabArg(A) = \texttt{out}$ then there exists some $B \in Ar$ attacking A such that $LabArg(B) = \texttt{in}$;
- if $LabArg(A) = \texttt{undec}$ then there exists no $B \in Ar$ attacking A such that $LabArg(B) = \texttt{in}$.

From Definition 13 it follows directly that each committed admissible argument labelling is an admissible argument labelling.

Corollary 26. *Let $\langle Ar, Att\rangle$ be an AA framework and let LabArg be an argument labelling of $\langle Ar, Att\rangle$. If LabArg is a committed admissible argument labelling of $\langle Ar, Att\rangle$ then it is an admissible argument labelling of $\langle Ar, Att\rangle$, but not vice versa.*

Example 24. The AA framework $\langle Ar_{ABA_{13}}, Att_{ABA_{13}}\rangle$ (see Examples 19 and 20) has four committed admissible argument labellings, namely $LabArg_{11}$, $LabArg_{24}$, $LabArg_{32}$, and $LabArg_{41}$. The other admissible argument labellings are not committed admissible since they violate the third condition in Definition 13. For example, $LabArg_{21}$ is not a committed admissible argument labelling since argument A_2 is labelled `undec`, but there exists an argument attacking A_2 which is labelled `in`, namely A_1.

Differently from admissible argument labellings, committed admissible argument labellings are in a one-to-one correspondence with admissible argument extensions.

Theorem 27. *Let $\langle Ar, Att\rangle$ be an AA framework.*

1. *Let $Args \subseteq Ar$ be an admissible argument extension of $\langle Ar, Att\rangle$. Then LabArg with $\texttt{in}(LabArg) = Args$, $\texttt{out}(LabArg) = Args^+$, and $\texttt{undec}(LabArg) = Ar \setminus (Args \cup Args^+)$ is a committed admissible argument labelling of $\langle Ar, Att\rangle$.*

2. *Let LabArg be a committed admissible argument labelling of* $\langle Ar, Att\rangle$. *Then* $Args = \texttt{in}(LabArg)$ *is an admissible argument extension of* $\langle Ar, Att\rangle$ *with* $Args^+ = \texttt{out}(LabArg)$, *and* $Ar \setminus (Args \cup Args^+) = \texttt{undec}(LabArg)$.

Note that the way arguments are labelled `in`, `out`, and `undec` in the first item of Theorem 27 mirrors the *Ext2Lab* operator in [25]. On the other hand, the second item of Theorem 27 extends the *Lab2Ext* operator from [25] as it not only defines an argument extension based on an argument labelling, but also the set of arguments attacked by the argument extension and the set of arguments which are neither contained in nor attacked by the argument extension.

Note also that committed admissible argument labellings are different from other variations of the admissible semantics such as strongly admissible argument labellings (and extensions) [37, 38], which require that an accepted argument is defended by accepted arguments other than itself, and related admissible argument extensions [16], which require that all accepted arguments are "relevant" for defending some accepted argument.

Given this one-to-one correspondence between committed admissible argument labellings and admissible argument labellings, we now show that there is a "more refined" one-to-many correspondence between admissible assumption labellings and committed admissible argument labellings as compared to admissible argument labellings, i.e. some additional correspondence results hold. Firstly, the converse of Theorem 24 is satisfied for committed admissible argument labellings.

Theorem 28. *Let LabAsm be an assumption labelling. LabAsm is an admissible assumption labelling if and only if* `LabAsm2LabArg`$(LabAsm)$ *is a committed admissible argument labelling.*

Secondly, the translation of a committed admissible argument labelling in terms of `LabArg2LabAsm` is an admissible assumption labelling.

Theorem 29. *Let LabArg be an argument labelling. If LabArg is a committed admissible argument labelling then* `LabArg2LabAsm`$(LabArg)$ *is an admissible assumption labelling.*

Furthermore, Proposition 25 also holds for committed admissible argument labellings.

Proposition 30. *Let LabAsm be an admissible assumption labelling. Then there exists a committed admissible argument labelling LabArg such that* `LabArg2LabAsm`$(LabArg) = LabAsm$.

The following example illustrates that due to the additional correspondence results, the one-to-many correspondence of admissible assumption labellings with committed admissible argument labellings is a "more refined" than with admissible argument labellings.

Example 25. Consider again ABA_{13} from Examples 19, 20, and 24. $LabAsm_2$ is the translation of only one committed admissible argument labelling in terms of `LabArg2LabAsm`, namely $LabArg_{24}$, rather than of two different admissible argument labellings $LabArg_{23}$ and $LabArg_{24}$. Furthermore, the translations of all committed admissible argument labellings in terms of `LabArg2LabAsm` are admissible assumption labellings. In contrast, the translations of the three admissible argument labellings $LabArg_{21}$, $LabArg_{22}$, and $LabArg_{31}$ in terms of `LabArg2LabAsm` are not admissible assumption labellings.

The reason that despite the additional correspondence results there is no one-to-one correspondence between admissible assumption labellings and committed admissible argument labellings is that a committed admissible argument labelling may not be the translation of any admissible assumption labelling in terms of `LabAsm2LabArg`. For example, the committed admissible argument labelling $LabArg_{32}$ of ABA_{13} is not the translation of any admissible assumption labelling in terms of `LabAsm2LabArg` (see Examples 19, 20, and 24).

Note that it would be straightforward to define a new notion of admissible assumption labellings, which corresponds more closely to admissible argument labellings. This can be achieved by deleting the restriction on `undec` assumptions from the definition of admissible assumption labellings. However, we do not examine this possible variation further since we believe that the restriction on `undec` assumptions is intuitive: it seems reasonable that any assumption attacked by accepted assumptions cannot be accepted and should thus be rejected (OUT) rather than neither accepted nor rejected (UNDEC).

6. Non-flat ABA frameworks

So far, we only considered flat ABA frameworks. In general however, ABA frameworks may not be flat, for example the instance of ABA corresponding to auto-epistemic logic [39] is never flat [4]. For possibly non-flat ABA frameworks assumption extensions are defined more generally than for flat ABA frameworks: they are *closed* sets of assumptions and they are based on a more general notion of defence [4]. A set of assumption $Asms \subseteq \mathcal{A}$

- is *closed* if and only if $Asms = \{\alpha \in \mathcal{A} \mid \exists Asms' \subseteq Asms : Asms' \vdash \alpha\}$;
- defends $\alpha \in \mathcal{A}$ if and only if $Asms$ attacks all closed sets of assumptions attacking α.

Note that in flat ABA frameworks every set of assumptions is closed since in these frameworks assumptions do not occur as the head of inference rules and therefore the more general notion of defence coincides with the notion of defence introduced in Section 2. For flat ABA frameworks, the general definition of assumption extensions for possibly non-flat ABA frameworks (introduced in the following) thus coincides with the definitions given in Section 2.

From here onwards, and if not specified otherwise, we assume as given a possibly non-flat ABA framework $\langle \mathcal{L}, \mathcal{R}, \mathcal{A}, ^{-} \rangle$. Furthermore, "defence" refers to the more general notion.

6.1. Admissible semantics

We recall the definition of admissible assumption extensions for possibly non-flat ABA frameworks.

> A set of assumption $Asms \subseteq \mathcal{A}$ is an *admissible assumption extension* if and only if $Asms$ is closed, conflict-free, and defends every $\alpha \in Asms$.

We first illustrate that admissible assumption labellings as introduced for flat ABA frameworks (Definition 2) do not correctly express the semantics of non-flat ABA frameworks.

Example 26. Let ABA_{14} be the non-flat ABA framework with:

$\mathcal{L} = \{\rho, \psi, \chi, p, x\}$,
$\mathcal{R} = \{\rho \leftarrow \chi\}$,
$\mathcal{A} = \{\rho, \psi, \chi\}$,
$\overline{\rho} = \psi$, $\overline{\psi} = p$, $\overline{\chi} = x$.

According to Definition 2, ABA_{14} has four admissible assumption labellings:

- $LabAsm_1 = \{(\rho, \text{UNDEC}), (\psi, \text{UNDEC}), (\chi, \text{UNDEC})\}$,
- $LabAsm_2 = \{(\rho, \text{OUT}), (\psi, \text{IN}), (\chi, \text{UNDEC})\}$,
- $LabAsm_3 = \{(\rho, \text{UNDEC}), (\psi, \text{UNDEC}), (\chi, \text{IN})\}$, and
- $LabAsm_4 = \{(\rho, \text{OUT}), (\psi, \text{IN}), (\chi, \text{IN})\}$.

However, ABA_{14} has only two admissible assumption extensions (according to the definition for possibly non-flat ABA frameworks): $Asms_1 = \{\}$, and $Asms_2 = \{\psi\}$. $Asms_1$ corresponds to $LabAsm_1$ and $Asms_2$ to $LabAsm_2$ (in terms of Theorem 1). The corresponding sets of assumptions (in terms of Theorem 1) of $LabAsm_3$ and $LabAsm_4$ are $Asms_3 = \{\chi\}$ and $Asms_4 = \{\psi, \chi\}$, respectively. Neither of them is an admissible assumption extension of ABA_{14}, since neither of them is a closed set of assumptions. Thus, $LabAsm_3$ and $LabAsm_4$ should not be admissible assumption labellings of ABA_{14}.

As illustrated in Example 26, a reason that the definition of admissible assumption labellings of flat ABA frameworks does not correctly express the semantics of non-flat ABA frameworks is that the set of IN labelled assumptions may not be closed. A straightforward way of revising the definition of admissible assumption labellings is thus to explicitly add the condition "IN($LabAsm$) is a closed set of assumptions". However, this condition expresses a restriction on the whole set of IN labelled assumptions, rather than on the label of one assumption, as done by the three conditions of admissible assumption labellings.

To adhere to the structure of the conditions of admissible assumption labellings, we instead add an additional restriction to the conditions of UNDEC and OUT labelled assumptions, which ensures that an assumption can only be labelled UNDEC or OUT if it is not derivable from the set of IN labelled assumptions using the inference rules. To express this new restriction, we introduce the notion of a set of assumptions *supporting* an assumption.

Definition 14. Let $Asms \subseteq \mathcal{A}$ and $\alpha \in \mathcal{A}$. *Asms supports* α if and only if there exists an argument $Asms' \vdash \alpha$ and $Asms' \subseteq Asms$. Equivalently, we say that α is supported by $Asms$.

The following definition extends Definition 2 to admissible assumption labellings of possibly non-flat ABA frameworks.

Definition 15. Let $LabAsm$ be an assumption labelling. $LabAsm$ is an *admissible assumption labelling* if and only if for each assumption $\alpha \in \mathcal{A}$ it holds that:

- if $LabAsm(\alpha) = \text{IN}$ then for each closed set of assumptions $Asms$ attacking α there exists some $\beta \in Asms$ such that $LabAsm(\beta) = \text{OUT}$;
- if $LabAsm(\alpha) = \text{OUT}$ then there exists a closed set of assumptions $Asms_1$ attacking α such that for all $\beta \in Asms_1$, $LabAsm(\beta) = \text{IN}$, and there exists no set of assumptions $Asms_2$ supporting α such that for all $\gamma \in Asms_2$, $LabAsm(\gamma) = \text{IN}$;
- if $LabAsm(\alpha) = \text{UNDEC}$ then for each closed set of assumptions $Asms_1$ attacking α there exists some $\beta \in Asms_1$ such that $LabAsm(\beta) \neq \text{IN}$, and there exists no set of assumptions $Asms_2$ supporting α such that for all $\gamma \in Asms_2$, $LabAsm(\gamma) = \text{IN}$.

According to the revised definition only $LabAsm_1$ and $LabAsm_2$ of ABA_{14} from Example 26 are admissible assumption labellings. $LabAsm_3$ and $LabAsm_4$ are not admissible assumption labellings since ρviolates the new restriction on UNDEC/OUT labelled assumptions as ρ is supported by $\{\chi\}$ and χ is labelled IN.

Note that we also incorporated the more general notion of defence into Definition 15 by only considering closed sets of assumptions attacking an assumption in question.

Observation 31. *Let LabAsm be an assumption labelling of a flat ABA framework. Then LabAsm is an admissible assumption labelling according to Definition 2 if and only if it is an admissible assumption labelling according to Definition 15.*

The following theorem states that Definition 15 correctly expresses the admissible semantics of possibly non-flat ABA frameworks, i.e. that there is a one-to-one correspondence between admissible assumption extensions and labellings of possibly non-flat ABA frameworks.

Theorem 32.

1. *Let Asms be an admissible assumption extension. Then LabAsm with* $\text{IN}(LabAsm) = Asms$, $\text{OUT}(LabAsm) = Asms^+$ *and* $\text{UNDEC}(LabAsm) = \mathcal{A} \setminus (Asms \cup Asms^+)$ *is an admissible assumption labelling.*
2. *Let LabAsm be an admissible assumption labelling. Then* $Asms = \text{IN}(LabAsm)$ *is an admissible assumption extension with* $Asms^+ = \text{OUT}(LabAsm)$ *and* $\mathcal{A} \setminus (Asms \cup Asms^+) = \text{UNDEC}(LabAsm)$.

6.2. Complete semantics

We first recall the definition of complete assumption extensions for possibly non-flat ABA frameworks.

> A set of assumption $Asms \subseteq \mathcal{A}$ is a *complete assumption extension* if and only if *Asms* is closed, conflict-free, and consists of all assumptions it defends.

For flat ABA frameworks, complete assumption labellings are defined as admissible assumption labellings satisfying an additional condition. Analogously, we define complete assumption labellings of possibly non-flat ABA frameworks.

Definition 16. Let *LabAsm* be an assumption labelling. *LabAsm* is a *complete assumption labelling* if and only if *LabAsm* is an admissible assumption labelling and for each assumption $\alpha \in \mathcal{A}$ it holds that:

- if $LabAsm(\alpha) = \text{UNDEC}$ then there exists a closed set of assumptions $Asms_3$ attacking α such that for all $\delta \in Asms_3$, $LabAsm(\delta) \neq \text{OUT}$.

Analogously to the definition of admissible assumption labellings of possibly non-flat ABA frameworks, the additional condition of complete assumption labellings only takes into account attacking sets of assumptions which are closed. Without this restriction, the definition would yield different assumption labellings.

Observation 33. *Let LabAsm be an assumption labelling of a flat ABA framework. Then LabAsm is a complete assumption labelling according to Definition 3 if and only if it is a complete assumption labelling according to Definition 16.*

As intended, complete assumption labellings and extensions of possibly non-flat ABA frameworks are in one-to-one correspondence.

Theorem 34.

1. *Let Asms be a complete assumption extension. Then LabAsm with* $\textsc{in}(LabAsm) = Asms$*,* $\textsc{out}(LabAsm) = Asms^+$ *and* $\textsc{undec}(LabAsm) = \mathcal{A} \setminus (Asms \cup Asms^+)$ *is a complete assumption labelling.*
2. *Let LabAsm be a complete assumption labelling. Then* $Asms = \textsc{in}(LabAsm)$ *is a complete assumption extension with* $Asms^+ = \textsc{out}(LabAsm)$ *and* $\mathcal{A} \setminus (Asms \cup Asms^+) = \textsc{undec}(LabAsm)$*.*

For flat ABA frameworks we identified two equivalent variations to the definition of complete assumption labellings. One of them used the converse of each condition in the definition of a complete assumption labellings (see Lemma 3). Extending this alternative definition of complete assumption labellings of flat ABA frameworks with an additional condition ensuring that the set of IN labelled assumptions is closed and considering only attacking sets of assumptions which are closed, makes it equivalent to the definition of complete assumption labellings for possibly non-flat ABA frameworks.

Proposition 35. *Let LabAsm be an assumption labelling. The following statements are equivalent:*

1. *LabAsm is a complete assumption labelling.*
2. *LabAsm is such that for each* $\alpha \in \mathcal{A}$ *it holds that:*
   - *if there exists a set of assumptions Asms supporting* α *such that for all* $\beta \in Asms$*,* $LabAsm(\beta) = \textsc{in}$*, then* $LabAsm(\alpha) = \textsc{in}$*;*
   - *if for each closed set of assumptions Asms attacking* α *there exists some* $\beta \in Asms$ *such that* $LabAsm(\beta) = \textsc{out}$*, then* $LabAsm(\alpha) = \textsc{in}$*;*
   - *if there exists a closed set of assumptions Asms attacking* α *such that for all* $\beta \in Asms$*,* $LabAsm(\beta) = \textsc{in}$*, then* $LabAsm(\alpha) = \textsc{out}$*;*
   - *if for each closed set of assumptions* $Asms_1$ *attacking* α *there exists some* $\beta \in Asms_1$ *such that* $LabAsm(\beta) \neq \textsc{in}$*, and there exists a closed set of assumptions* $Asms_2$ *attacking* α *such that for all* $\gamma \in Asms_2$*,* $LabAsm(\gamma) \neq \textsc{out}$*, then* $LabAsm(\alpha) = \textsc{undec}$*.*

Note that reversing the conditions in Definition 16 does not result in an equivalent definition of complete assumption labellings for possibly non-flat ABA frameworks. For example, the assumption labelling $LabAsm = \{(\rho, \textsc{undec}), (\psi, \textsc{in}), (\chi, \textsc{in})\}$ of ABA_{14} (see Example 26) satisfies the converse of each condition in Definition 16: for both ψ and χ the converse of the first condition applies and is satisfied, and for ρ none of the converses of the three conditions applies, so ρ trivially satisfies the converse conditions. However, *LabAsm* is not a complete assumption labelling of ABA_{14} since ABA_{14} has no complete assumption labellings.

The other equivalent definition of complete assumption labellings for flat ABA frameworks we identified was the "if and only if" version of the first and second conditions of a complete assumption labelling for flat ABA frameworks (see Lemma 3). The analogue for complete assumption labellings of possibly non-flat ABA frameworks does however not result in an equivalent definition. That is, an assumption labelling satisfying the the following conditions

- $LabAsm(\alpha) = \text{IN}$ if and only if for each closed set of assumptions $Asms$ attacking α there exists some $\beta \in Asms$ such that $LabAsm(\beta) = \text{OUT}$;
- $LabAsm(\alpha) = \text{OUT}$ if and only if there exists a closed set of assumptions $Asms_1$ attacking α such that for all $\beta \in Asms_1$, $LabAsm(\beta) = \text{IN}$, and there exists no set of assumptions $Asms_2$ supporting α such that for all $\gamma \in Asms_2$, $LabAsm(\gamma) = \text{IN}$;

is not generally a complete assumption labelling of a possibly non-flat ABA framework since for instance $LabAsm = \{(\rho, \text{UNDEC}), (\psi, \text{IN}), (\chi, \text{IN})\}$ of ABA_{14} (see Example 26) satisfies both conditions, but $LabAsm$ is not a complete assumption labelling of ABA_{14}.

6.3. Grounded, preferred, ideal, semi-stable, and stable semantics

Originally, the grounded, preferred, and stable assumption extensions of possibly non-flat ABA frameworks were defined as specific admissible rather than complete assumption extensions. For flat ABA frameworks these two definitions are equivalent, bus as we will show in this section for non-flat ABA frameworks they are not.

We first recall the definitions of grounded, preferred, and stable assumption extensions for possibly non-flat ABA frameworks [4]. A set of assumptions $Asms \subseteq \mathcal{A}$ is

- a *grounded assumption extension* if and only if $Asms$ is the intersection of all complete assumption extensions;
- a *preferred assumption extension* if and only if $Asms$ is a maximal (w.r.t. $\subseteq$) admissible assumption extension;
- a *stable assumption extension* if and only if $Asms$ is closed, conflict-free, and for all $\alpha \in \mathcal{A}$ it holds that if $\alpha \notin Asms$ then $Asms$ attacks α.

Since ideal and semi-stable semantics have only been defined for flat ABA frameworks so far, we will investigate these semantics after dealing with the grounded, preferred, and stable semantics.

6.3.1. Grounded semantics

The following example illustrates that for possibly non-flat ABA frameworks, the minimally complete assumption extensions do not generally coincide with the grounded assumption extensions.

Example 27. Let ABA_{15} be the non-flat ABA framework with:

$\mathcal{L} = \{\rho, \psi, \chi, \omega, x\}$,
$\mathcal{R} = \{x \leftarrow \rho;\ x \leftarrow \psi;\ \chi \leftarrow\}$,
$\mathcal{A} = \{\rho, \psi, \chi, \omega\}$,
$\overline{\rho} = \psi$, $\overline{\psi} = \rho$, $\overline{\chi} = \omega$, $\overline{\omega} = x$.

ABA_{15} has two complete assumption extensions: $Asms_1 = \{\rho, \chi\}$ and $Asms_2 = \{\psi, \chi\}$. $Asms_1$ and $Asms_2$ are both minimally complete, but the grounded assumption extension is $Asms_3 = \{\chi\}$.

In order to express the grounded semantics of possibly non-flat ABA frameworks in terms of assumption labellings, the set of IN labelled assumptions has to be the intersection of the sets of IN labelled assumptions of all complete assumption labellings.

Definition 17. Let $LabAsm$ be an assumption labelling. $LabAsm$ is a *grounded assumption labelling* if and only if for all $\alpha \in \mathcal{A}$ it holds that:

- $LabAsm(\alpha) = \textsc{in}$ if and only if for all complete assumption labellings $LabAsm'$, $LabAsm'(\alpha) = \textsc{in}$;
- $LabAsm(\alpha) = \textsc{out}$ if and only if there exists a closed set of assumptions $Asms$ attacking α such that for all $\beta \in Asms$, $LabAsm(\beta) = \textsc{in}$.

The second condition ensures the one-to-one correspondence between grounded assumption labellings and extensions of possibly non-flat ABA frameworks.

Theorem 36.

1. *Let $Asms$ be a grounded assumption extension. Then $LabAsm$ with $\textsc{in}(LabAsm) = Asms$, $\textsc{out}(LabAsm) = Asms^+$ and $\textsc{undec}(LabAsm) = \mathcal{A} \setminus (Asms \cup Asms^+)$ is a grounded assumption labelling.*
2. *Let $LabAsm$ be a grounded assumption labelling. Then $Asms = \textsc{in}(LabAsm)$ is a grounded assumption extension with $Asms^+ = \textsc{out}(LabAsm)$ and $\mathcal{A} \setminus (Asms \cup Asms^+) = \textsc{undec}(LabAsm)$.*

Based on the correspondence between grounded assumption labellings and extensions of possibly non-flat ABA frameworks and results from [4], we prove that for flat ABA frameworks Definition 17 is equivalent to the definition of grounded assumption labellings for flat ABA frameworks.

Proposition 37. *Let $LabAsm$ be an assumption labelling of a flat ABA framework. Then $LabAsm$ is a grounded assumption labelling according to Definition 4 if and only if it is a grounded assumption labelling according to Definition 17.*

6.3.2. Preferred semantics

The non-flat ABA framework ABA_{14} from Example 26 illustrates that maximally complete assumption extensions do not generally coincide with preferred assumption extensions: ABA_{14} has no complete assumption extensions, but $\{\psi\}$ is its preferred assumption extension. We thus define preferred assumption labellings of possibly non-flat ABA frameworks as admissible, rather than complete, assumption labellings with a maximal set of $\textsc{in}$ labelled assumptions.

Definition 18. Let $LabAsm$ be an assumption labelling. $LabAsm$ is a *preferred assumption labelling* if and only if $LabAsm$ is an admissible assumption labelling and $\textsc{in}(LabAsm)$ is maximal (w.r.t. $\subseteq$) among all admissible assumption labellings.

Since preferred assumption labellings for flat ABA frameworks can be equivalently defined as *admissible* assumption labellings with a maximal set of $\textsc{in}$ labelled assumptions (see Proposition 7) and since for flat ABA frameworks Definition 15 coincides with Definition 2, it follows that for flat ABA frameworks Definition 18 coincides with the definition of preferred assumption labellings of flat ABA frameworks.

Proposition 38. *Let $LabAsm$ be an assumption labelling of a flat ABA framework. Then $LabAsm$ is a preferred assumption labelling according to Definition 4 if and only if it is a preferred assumption labelling of according to Definition 18.*

As desired, preferred assumption labellings correctly express the preferred semantics of possibly non-flat ABA frameworks.

Theorem 39.

1. *Let Asms be a preferred assumption extension. Then LabAsm with* $\textsc{in}(LabAsm) = Asms$, $\textsc{out}(LabAsm) = Asms^+$ *and* $\textsc{undec}(LabAsm) = \mathcal{A} \setminus (Asms \cup Asms^+)$ *is a preferred assumption labelling.*
2. *Let LabAsm be a preferred assumption labelling. Then* $Asms = \textsc{in}(LabAsm)$ *is a preferred assumption extension with* $Asms^+ = \textsc{out}(LabAsm)$ *and* $\mathcal{A} \setminus (Asms \cup Asms^+) = \textsc{undec}(LabAsm)$.

6.3.3. Stable semantics

Even though stable assumption extensions of possibly non-flat ABA frameworks are not defined as specific admissible or complete assumption extensions, it was shown in [4] that stable assumption extensions are always complete assumption extensions. Therefore, we define stable assumption labellings of possibly non-flat ABA frameworks in the same way as for flat ABA frameworks, i.e. as complete assumption labellings which label no assumptions as UNDEC.

Definition 19. Let *LabAsm* be an assumption labelling. *LabAsm* is a *stable assumption labelling* if and only if *LabAsm* is a complete assumption labelling and $\textsc{undec}(LabAsm) = \emptyset$.

From Observation 33 and Definition 19 it follows straightaway that for flat ABA frameworks Definition 19 is equivalent to the definition of stable assumption labellings of flat ABA frameworks.

Observation 40. *Let LabAsm be an assumption labelling of a flat ABA framework. Then LabAsm is a stable assumption labelling according to Definition 4 if and only if it is a stable assumption labelling according to Definition 19.*

Furthermore, there is a one-to-one correspondence between stable assumption labellings and extensions of possibly non-flat ABA frameworks.

Theorem 41.

1. *Let Asms be a stable assumption extension. Then LabAsm with* $\textsc{in}(LabAsm) = Asms$, $\textsc{out}(LabAsm) = Asms^+$ *and* $\textsc{undec}(LabAsm) = \mathcal{A} \setminus (Asms \cup Asms^+)$ *is a stable assumption labelling.*
2. *Let LabAsm be a stable assumption labelling. Then* $Asms = \textsc{in}(LabAsm)$ *is a stable assumption extension with* $Asms^+ = \textsc{out}(LabAsm)$ *and* $\mathcal{A} \setminus (Asms \cup Asms^+) = \textsc{undec}(LabAsm)$.

6.3.4. Ideal semantics

Since the ideal semantics has so far only been defined in the context of flat ABA frameworks, we define ideal assumption extensions of possibly non-flat ABA frameworks, following the original definition for flat ABA frameworks where ideal assumption extensions are defined as specific admissible rather than complete assumption extensions [23].

Definition 20. A set of assumptions $Asms \subseteq \mathcal{A}$ is an *ideal assumption extension* if and only if *Asms* is a maximal (w.r.t. $\subseteq$) admissible assumption extension satisfying that for all preferred assumption extensions $Asms'$, $Asms \subseteq Asms'$.

Just as for preferred assumption extensions, ideal assumption extensions of possibly non-flat ABA frameworks do not generally coincide with maximally complete assumption extensions which are a subset of each preferred assumption extension. We thus define ideal assumption labellings of possibly non-flat ABA frameworks in terms of admissible rather than complete assumption labellings.

Definition 21. Let $LabAsm$ be an assumption labelling. $LabAsm$ is an *ideal assumption labelling* if and only if $LabAsm$ is an admissible assumption labelling and $\textsc{in}(LabAsm)$ is maximal (w.r.t. $\subseteq$) among all admissible assumption labellings satisfying that for all preferred assumption labellings $LabAsm'$, $\textsc{in}(LabAsm) \subseteq \textsc{in}(LabAsm')$.

From Proposition 7 and Observation 31 it follows that for flat ABA frameworks Definition 21 coincides with the definition of ideal assumption labellings of flat ABA frameworks.

Proposition 42. *Let LabAsm be an assumption labelling of a flat ABA framework. Then LabAsm is an ideal assumption labelling according to Definition 4 if and only if it is an ideal assumption labelling according to Definition 21.*

Furthermore, as desired there is a one-to-one correspondence between ideal assumption extensions and labellings of possibly non-flat ABA frameworks.

Theorem 43.

1. *Let Asms be an ideal assumption extension. Then LabAsm with* $\textsc{in}(LabAsm) = Asms$, $\textsc{out}(LabAsm) = Asms^+$ *and* $\textsc{undec}(LabAsm) = \mathcal{A} \setminus (Asms \cup Asms^+)$ *is an ideal assumption labelling.*
2. *Let LabAsm be an ideal assumption labelling. Then* $Asms = \textsc{in}(LabAsm)$ *is an ideal assumption extension with* $Asms^+ = \textsc{out}(LabAsm)$ *and* $\mathcal{A} \setminus (Asms \cup Asms^+) = \textsc{undec}(LabAsm)$.

6.3.5. Semi-stable semantics

Just like the ideal semantics, the semi-stable semantics has so far only been defined for flat ABA frameworks. Semi-stable assumption extensions are originally defined as specific complete assumption extensions, but for flat ABA frameworks they can be equivalently defined as specific admissible assumption extensions. For non-flat ABA frameworks this is not the case.

Example 28. Let ABA_{16} be the non-flat ABA framework with:

$\mathcal{L} = \{\rho, \psi, \chi, \omega, p\}$,
$\mathcal{R} = \{p \leftarrow \rho;\ p \leftarrow \chi;\ p \leftarrow \psi;\ \psi \leftarrow \rho, \chi\}$,
$\mathcal{A} = \{\rho, \psi, \chi, \omega\}$,
$\overline{\rho} = \psi$, $\overline{\psi} = p$, $\overline{\chi} = \psi$, $\overline{\omega} = \chi$.

The only complete assumption extension of ABA_{16} is $Asms_1 = \{\}$, and thus $Asms_1 \cup Asms_1^+$ is maximal among all complete assumption extensions. In contrast, there are three admissible assumption extensions: $Asms_1$, $Asms_2 = \{\rho\}$, and $Asms_3 = \{\chi\}$. Among these, $Asms_3 \cup Asms_3^+$ is maximal.

One of the defining properties of semi-stable assumption extensions of flat ABA frameworks is that they are preferred assumption extensions [24]. To retain this property, we define semi-stable assumption extensions and labellings of possibly non-flat ABA frameworks in terms of admissible rather than complete assumption extensions and labellings.

Definition 22. A set of assumptions $Asms \subseteq \mathcal{A}$ is a *semi-stable assumption extension* if and only if $Asms$ is an admissible assumption extension and for all admissible assumption extensions $Asms'$, $Asms \cup Asms^+ \not\subset Asms' \cup Asms'^+$.

Definition 23. Let $LabAsm$ be an assumption labelling . $LabAsm$ is a *semi-stable assumption labelling* if and only if $LabAsm$ is an admissible assumption labelling and $\text{UNDEC}(LabAsm)$ is minimal (w.r.t. $\subseteq$) among all admissible assumption labellings.

By Proposition 7, for flat ABA frameworks Definition 23 coincides with the definition of semi-stable assumption labellings of flat ABA frameworks.

Proposition 44. *Let LabAsm be an assumption labelling of a flat ABA framework. Then LabAsm is a semi-stable assumption labelling according to Definition 4 if and only if it is a semi-stable assumption labelling according to Definition 23.*

As desired, there is a one-to-one correspondence between semi-stable assumption extensions and labellings of possibly non-flat ABA frameworks.

Theorem 45.

1. *Let Asms be a semi-stable assumption extension. Then LabAsm with* $\text{IN}(LabAsm) = Asms$, $\text{OUT}(LabAsm) = Asms^+$ *and* $\text{UNDEC}(LabAsm) = \mathcal{A} \setminus (Asms \cup Asms^+)$ *is a semi-stable assumption labelling.*
2. *Let LabAsm be a semi-stable assumption labelling. Then* $Asms = \text{IN}(LabAsm)$ *is a semi-stable assumption extension with* $Asms^+ = \text{OUT}(LabAsm)$ *and* $\mathcal{A} \setminus (Asms \cup Asms^+) = \text{UNDEC}(LabAsm)$.

Finally, we prove that semi-stable assumption labellings of possibly non-flat ABA frameworks satisfy the property we desired, namely that they are preferred assumption labellings.

Proposition 46. *Let LabAsm be a semi-stable assumption labelling. Then LabAsm is a preferred assumption labelling.*

7. Conclusion

We studied assumption labellings of flat as well as possibly non-flat ABA frameworks for the admissible, grounded, complete, preferred, ideal, semi-stable, and stable semantics and proved that there is a one-to-one correspondence with the respective assumption extensions. Even though some of the definitions of assumption labellings for possibly non-flat ABA frameworks are (considerably) different from the definitions of assumption labellings for flat ABA frameworks, in particular those for the grounded, preferred, ideal, and semi-stable semantics, we proved that applying the definitions of assumption labellings for possibly non-flat ABA frameworks to flat ABA frameworks yields the correct assumption labellings (as defined for flat ABA frameworks). Furthermore, we showed that assumption labellings of flat ABA frameworks can be equivalently defined based on attacks between all sets of assumptions and attacks between argument-supporting sets of assumptions. Using the latter notion, we demonstrated how argument-supporting sets of assumptions and attacks between them can be naturally represented as a graph, which expresses sufficient information to determine the assumption labellings of a flat ABA framework.

We also investigated the relationship of assumption labellings of flat ABA frameworks and argument labellings of their corresponding AA frameworks and found that grounded, complete, preferred, ideal, and stable assumption and argument labellings are in a one-to-one correspondence, whereas semi-stable assumption labellings and extensions do not generally correspond. Admissible assumption and argument labellings are in a one-to-many correspondence. Based on this finding, we defined a variant of admissible argument labellings, which we call committed admissible argument labellings and which mirrors admissible assumption labellings by requiring that arguments which are attacked by `in` labelled arguments are labelled `out`, rather than `undec`. We proved that admissible assumption labellings and committed admissible argument labellings are in a "more refined" one-to-many correspondence as compared to admissible argument labellings, i.e. an admissible assumption labelling may correspond to fewer committed admissible argument labellings than admissible argument labellings. In contrast to admissible argument labellings, committed admissible argument labellings are furthermore in one-to-one correspondence with admissible argument extensions.

In this paper, we did not focus on any particular instance of ABA. For example, logic programs are special instances of flat ABA frameworks [4], so the results presented here for flat ABA frameworks also apply to the instance of ABA which corresponds to logic programs. Caminada and Schulz [29] show that, conversely, a specific fragment of (flat) ABA frameworks is as an instance of logic programs, where the models of the logic program and the assumption labellings of the ABA framework coincide. Furthermore, Dung [3] as well as Caminada et al. [40] study logic programs as an instance of AA frameworks, proving that the models of the logic program and the argument labellings of the AA framework correspond. In Section 5, we investigated (flat) ABA as a *direct* instance of AA (as for example in [23, 24]) and analysed the (non-)correspondence between assumption and argument labellings in more detail than previously done for assumption and argument extensions [23, 24], without restricting to any fragment or instance of flat ABA.

Since the stable semantics of ABA, both in terms of assumption and argument extensions, corresponds to the stable model semantics for logic programs [4, 22], ABA has proven useful to argumentatively explain the truth value (true or false) of literals with respect to stable models of logic programs [22]. Other logic programming semantics such as 3-valued stable models [41], well-founded models [42], and regular models [43] also include a third truth value (undefined) for literals. Since these 3-valued logic programming models correspond to assumption labellings [28, 29], assumption labellings could be used to argumentatively explain the truth value of literals with respect to 3-valued logic programming models. Furthermore, assumption labellings may prove useful for explaining inconsistencies in logic programs under the stable model semantics since some inconsistency scenarios can be attributed to the "undefined" truth value of literals [44], which corresponds to UNDEC assumptions in ABA frameworks [22, 29].

The ABA graphs introduced here are fundamentally different from previous graphical representations of ABA frameworks, notably the attack relationship graphs in [4] and the argument graphs in [36]. As logic programs are special instances of ABA frameworks, ABA graphs can also be used to represent logic programs [28]. For logic programs, various other graphical representations have been proposed, which can be compared to ABA graphs of ABA frameworks instantiated by logic programs. Dimopoulos and Torres [45] introduce minimal attack graphs of logic programs, which are very similar to ABA graphs when used on ABA frameworks which are instantiated by a logic program. The difference is that ABA graphs may have more nodes since all argument-supporting sets of assumptions form nodes, rather than only minimal ones. Costantini [46] introduces Extended Dependency Graphs to represent a special type of logic programs. The graphs have one node for every rule, holding the rule's head atom. The representation is thus rather different from ABA graphs, even when considering ABA frameworks which are instantiated by logic programs. Graphical representations of logic programs have also been used to compute the semantics of logic programs [47, 48]. In future work, we will study the suitability of ABA graphs for computing assumption labellings of ABA frameworks.

In future work we are planning to investigate the role of distinguishing between rejected and undecided assumptions in dispute derivations [34]. Currently, dispute derivations focus on determining whether an assumption is accepted with respect to some extension. If no dispute derivation can be constructed, the assumption in question is deemed non-accepted. We will explore whether assumption labellings can help to further differentiate between rejected and undecided assumptions in dispute derivations.

Furthermore, ABA frameworks have recently been extended to incorporate preferences [49, 50]. Future work will reveal whether assumption labellings can be directly applied to these extended ABA frameworks. Another recent development regarding ABA frameworks is the definition of semantics for ABA frameworks in terms of argument graphs rather than argument extensions [36]. It will be interesting to investigate if argument or assumption labellings can be directly applied to argument graphs or if a completely new labelling approach is needed.

Argument labellings have inspired efficient algorithms for computing the semantics of an AA framework [51, 52, 53]. Assumption labellings may thus prove useful for algorithms to compute ABA semantics in terms of assumptions, which is another line of future research.

Appendix A. Proofs

Proof of Theorem 1

1. First note that $Asms \cap Asms^+ = \emptyset$ since $Asms$ does not attack itself. Thus each $\alpha \in \mathcal{A}$ is either contained in $\textsc{in}(LabAsm)$, in $\textsc{out}(LabAsm)$, or in $\textsc{undec}(LabAsm)$.
    - Let $LabAsm(\alpha) = \textsc{in}$. Then $\alpha \in Asms$, so $Asms$ defends α, i.e. for all sets of assumptions $Asms_1$ attacking α there exists some $\beta \in Asms_1$ such that $Asms$ attacks β. Thus, $\beta \in Asms^+$ and consequently $LabAsm(\beta) = \textsc{out}$.
    - Let $LabAsm(\alpha) = \textsc{out}$. Then $\alpha \in Asms^+$, so $Asms$ attacks α. Since $Asms = \textsc{in}(LabAsm)$, there exists a set of assumptions $Asms_1$ attacking α such that for all $\beta \in Asms_1$, $LabAsm(\beta) = \textsc{in}$.
    - Let $LabAsm(\alpha) = \textsc{undec}$. Then $\alpha \notin Asms$ and $\alpha \notin Asms^+$, so α is not attacked and not defended by $Asms$. Since α is not attacked by $Asms$, for each set of assumptions $Asms_1$ attacking α there exists some $\beta \in Asms_1$ such that $\beta \notin Asms$, and thus $LabAsm(\beta) \neq \textsc{in}$.
2. We first prove that $\textsc{in}(LabAsm)$ is an admissible assumption extension.
    - $\textsc{in}(LabAsm)$ is conflict-free: Assume $\textsc{in}(LabAsm)$ is not conflict-free. Then $\textsc{in}(LabAsm)$ attacks some $\alpha \in \textsc{in}(LabAsm)$. By Definition 2, for each set of assumptions $Asms_1$ attacking α there exists some $\beta \in Asms_1$ such that $LabAsm(\beta) = \textsc{out}$. Hence, $\textsc{in}(LabAsm)$ contains some β such that $LabAsm(\beta) = \textsc{out}$. Contradiction.

- IN($LabAsm$) defends all $\alpha \in$ IN($LabAsm$): Let $\alpha \in$ IN($LabAsm$). Then by Definition 2, for each set of assumptions $Asms_1$ attacking α there exists some $\beta \in Asms_1$ such that $LabAsm(\beta) =$ OUT. Furthermore, for each such β there exists a set of assumptions $Asms_2$ attacking β such that for all $\gamma \in Asms_2$, $LabAsm(\gamma) =$ IN so $Asms_2 \subseteq$ IN($LabAsm$). Hence, IN($LabAsm$) attacks all sets of assumptions attacking α.

$$Asms^+ = \{\alpha \in \mathcal{A} \mid Asms \textit{ attacks } \alpha\} = \{\alpha \in \mathcal{A} \mid \text{IN}(LabAsm) \textit{ attacks } \alpha\}$$
$$= \{\alpha \in \mathcal{A} \mid \alpha \in \text{OUT}(LabAsm)\} = \text{OUT}(LabAsm)$$
$$\mathcal{A} \setminus (Asms \cup Asms^+) = \{\alpha \in \mathcal{A} \mid \alpha \notin \text{IN}(LabAsm), \alpha \notin \text{OUT}(LabAsm)\}$$
$$= \{\alpha \in \mathcal{A} \mid \alpha \in \text{UNDEC}(LabAsm)\} = \text{UNDEC}(LabAsm)$$

Proof of Lemma 2

Since each set of assumptions attacking α contains some β such that $LabAsm(\beta) =$ OUT, $LabAsm(\alpha) \neq$ OUT. If $LabAsm(\alpha) =$ IN, then $\forall \gamma \in \alpha^\star : LabAsm(\gamma) =$ OUT and therefore $LabAsm' = LabAsm$, so trivially $LabAsm'$ is an admissible assumption labelling. If $LabAsm(\alpha) =$ UNDEC, then $\forall \gamma \in \alpha^\star : LabAsm(\gamma) =$ UNDEC or $LabAsm(\gamma) =$ OUT. Furthermore, $\alpha \notin \alpha^\star$ since each set of assumptions attacking α contains some β such that $LabAsm(\beta) =$ OUT, so if $\alpha \in \alpha^\star$ then $\exists Asms$ attacking α such that $LabAsm(\alpha) =$ OUT (since all $\forall \delta \in Asms : \delta \neq \alpha \rightarrow LabAsm(\delta) =$ IN), which is a contradiction. Therefore, $LabAsm'$ is an assumption labelling.

- Let $LabAsm'(\epsilon) =$ IN. If $LabAsm(\epsilon) =$ IN, then for each set of assumption $Asms_1$ attacking ϵ there exists some $\eta \in Asms_1$ such that $LabAsm(\eta) =$ OUT and therefore $LabAsm'(\eta) =$ OUT. If $LabAsm(\epsilon) \neq$ IN then $\epsilon = \alpha$, so for each set of assumptions $Asms_2$ attacking α there exists some $\beta \in Asms$ such that $LabAsm(\beta) =$ OUT and thus $LabAsm'(\beta) =$ OUT.

- Let $LabAsm'(\epsilon) =$ OUT. If $LabAsm(\epsilon) =$ OUT then there exists a set of assumptions $Asms_1$ attacking ϵ such that for all $\eta \in Asms_1$, $LabAsm(\eta) =$ IN and therefore $LabAsm'(\eta) =$ IN. If $LabAsm(\epsilon) \neq$ OUT then $\epsilon \in \alpha^\star$, so there exists a set of assumptions $Asms_2$ attacking ϵ such that $\forall \delta \in Asms_2$ with $\delta \neq \alpha$ it holds that $LabAsm(\delta) =$ IN and thus $LabAsm'(\delta) =$ IN. Since $LabAsm'(\alpha) =$ IN the set of assumptions $Asms_2$ attacking ϵ is such that for all $\eta \in Asms_3$, $LabAsm'(\eta) =$ IN.

- Let $LabAsm'(\epsilon) =$ UNDEC. Then $LabAsm(\epsilon) =$ UNDEC. Thus, for each set of assumptions $Asms_1$ attacking ϵ there exists some $\eta \in Asms_1$ such that $LabAsm(\eta) \neq$ IN. Since $\epsilon \notin \alpha^\star$, for each such set of assumptions $Asms_1$ attacking ϵ either $\alpha \notin Asms_1$ or there exists some $\kappa \in Asms_1$ such that $\kappa \neq \alpha$ and $LabAsm(\kappa) \neq$ IN. In the first case $\eta \neq \alpha$, so $LabAsm'(\eta) \neq$ IN. In the second case, $LabAsm'(\kappa) \neq$ IN. Thus, for each set of assumptions $Asms_1$ attacking ϵ there exists some $\lambda \in Asms_1$ such that $LabAsm'(\lambda) \neq$ IN.

Proof of Theorem 3

Equivalence of first and second item:

- First item implies second item: Let $LabAsm$ be a complete assumption labelling. Then clearly the "only if" part of both conditions of the second item are satisfied since they are the same as the conditions in Definition 3. To prove that the "if" part of the conditions in the second item holds:

- Let α be an assumption such that for each set of assumptions $Asms$ attacking α there exists some $\beta \in Asms$ such that $LabAsm(\beta) = \text{OUT}$. Then $LabAsm(\alpha) \neq \text{OUT}$ because there exists no set of assumptions $Asms_1$ attacking α such that for all $\beta \in Asms_1$, $LabAsm(\beta) = \text{IN}$. Furthermore, $LabAsm(\alpha) \neq \text{UNDEC}$ because there exists no set of assumptions $Asms_2$ attacking α such that for all $\gamma \in Asms_2$, $LabAsm(\gamma) \neq \text{OUT}$. Hence, $LabAsm(\alpha) = \text{IN}$.
- Let α be an assumption such that there exists a set of assumptions $Asms$ attacking α such that for all $\beta \in Asms$, $LabAsm(\beta) = \text{IN}$. Then $LabAsm(\alpha) \neq \text{IN}$ because not for each set of assumptions $Asms_1$ attacking α there exists some $\beta \in Asms_1$ such that $LabAsm(\beta) = \text{OUT}$. Furthermore, $LabAsm(\alpha) \neq \text{UNDEC}$ because not for each set of assumptions $Asms_2$ attacking α there exists some $\gamma \in Asms_2$ such that $LabAsm(\gamma) \neq \text{IN}$. Hence, $LabAsm(\alpha) = \text{OUT}$.

- Second item implies first item: Let $LabAsm$ be such that the second item holds. We prove that $LabAsm$ is a complete assumption labelling. Clearly the first two conditions of complete assumption labellings are satisfied since they are the same as the "only if" part of the conditions in the second item. To prove the third condition of complete assumption labellings, let $LabAsm(\alpha) = \text{UNDEC}$. From the first condition of the second item we know that not for each set of assumptions $Asms_1$ attacking α there exists some $\beta \in Asms_1$ such that $LabAsm(\beta) = \text{OUT}$, so there exists a set of assumptions $Asms_2$ attacking α such that for all $\gamma \in Asms_2$, $LabAsm(\gamma) \neq \text{OUT}$. From the second condition of the second item we know that there exists no set of assumptions $Asms_3$ attacking α such that for all $\delta \in Asms_3$, $LabAsm(\delta) = \text{IN}$, so for each set of assumptions $Asms_4$ attacking α there exists some $\epsilon \in Asms_4$ such that $LabAsm(\epsilon) \neq \text{IN}$.

Equivalence of second and third item:

- Second item implies third item: Let $LabAsm$ be such that the second item holds. Then clearly the first two conditions of the third item are satisfied since they are the same as the "if" part of the second item. To prove the third condition of the third item, let α be such that for each set of assumptions $Asms_1$ attacking α there exists some $\beta \in Asms_1$ such that $LabAsm(\beta) \neq \text{IN}$, and there exists a set of assumptions $Asms_2$ attacking α such that for all $\gamma \in Asms_2$, $LabAsm(\gamma) \neq \text{OUT}$. Then $LabAsm(\alpha) \neq \text{IN}$ because not for each set of assumptions $Asms_3$ attacking α there exists some $\delta \in Asms_3$ such that $LabAsm(\delta) = \text{OUT}$, and $LabAsm(\alpha) \neq \text{OUT}$ because there exists no set of assumptions $Asms_4$ attacking α such that for all $\epsilon \in Asms_4$, $LabAsm(\epsilon) = \text{IN}$. Hence, $LabAsm(\alpha) = \text{UNDEC}$.

- Third item implies second item: Assume that $LabAsm$ is such that the third item holds. Then clearly the "if" part of both conditions in the second item are satisfied since they the same as the conditions in the third item. To prove that the "only if" parts of the conditions in the second item are satisfied, first note that for every $\alpha \in \mathcal{A}$ exactly one of the "if" parts of the three conditions in the third item is satisfied. Thus, if $LabAsm(\alpha) = \text{IN}$ the "if" part of the second and third condition in the third item are not satisfied. It follows that the "if" part of the first condition is satisfied, so for each set of assumptions $Asms_1$ attacking α there exists some $\beta \in Asms_1$ such that $LabAsm(\beta) = \text{OUT}$. Analogously, if $LabAsm(\alpha) = \text{OUT}$ only the "if" part of the second condition in the third item applies, so there exists a set of assumptions $Asms_2$ attacking α such that for all $\beta \in Asms_2$, $LabAsm(\beta) = \text{IN}$.

Proof of Theorem 4

1. Since $Asms$ is a complete assumption extension it is by definition also an admissible assumption extension. By Theorem 1, $LabAsm$ is an admissible assumption labelling. It remains to prove that the additional condition of complete assumption labellings is satisfied. Let $LabAsm(\alpha) = \textsc{undec}$. Then $\alpha \notin Asms$ and $\alpha \notin Asms^+$, so α is not attacked and not defended by $Asms$. Since α is not defended by $Asms$, there exists a set of assumption $Asms_1$ attacking α such that $Asms_1$ is not attacked by $Asms$. Thus, for all $\gamma \in Asms_1$ it holds that $\gamma \notin Asms^+$. Consequently, $LabAsm(\gamma) \neq \textsc{out}$.
2. Since $LabAsm$ is a complete assumption labelling it is by Definition 3 also an admissible assumption labelling. Thus, by Theorem 1 $Asms$ is an admissible assumption extension with $Asms^+ = \textsc{out}(LabAsm)$ and $\mathcal{A} \setminus (Asms \cup Asms^+) = \textsc{undec}(LabAsm)$. It remains to prove that all assumptions defended by $Asms$ are contained in $Asms$. Let α be defended by $Asms$ and thus by $\textsc{in}(LabAsm)$. Then for each set of assumptions $Asms_1$ attacking α, $\textsc{in}(LabAsm)$ attacks $Asms_1$. Thus, for each such $Asms_1$ there exists some $\beta \in Asms_1$ which is attacked by $\textsc{in}(LabAsm)$, and therefore $LabAsm(\beta) = \textsc{out}$. Since this holds for each $Asms_1$ attacking α, $LabAsm(\alpha) = \textsc{in}$.

Proof of Theorem 5

1. Let $Asms$ be a 1) grounded 2) preferred 3) ideal 4) semi-stable 5) stable assumption extension. By definition $Asms$ is a complete assumption extension. Furthermore, for all complete assumption extensions $Asms'$ it holds that 1) $Asms' \not\subset Asms$ 2) $Asms' \not\supset Asms$ 3) if for all preferred assumption extensions $Asms''$ it holds that $Asms' \subseteq Asms''$ then $Asms' \not\supset Asms$ 4) $Asms' \cup Asms'^+ \not\supset Asms \cup Asms^+$ 5) $Asms \cup Asms^+ = \mathcal{A}$. By Theorem 4 $LabAsm$ is a complete assumption labelling. Furthermore, from the above and Theorem 4, for all complete assumption labellings $LabAsm'$ it holds that 1) $\textsc{in}(LabAsm') \not\subset \textsc{in}(LabAsm)$ 2) $\textsc{in}(LabAsm') \not\supset \textsc{in}(LabAsm)$ 3) if for all preferred assumption labellings $LabAsm''$ it holds that $\textsc{in}(LabAsm') \subseteq \textsc{in}(LabAsm'')$ then $\textsc{in}(LabAsm') \not\supset \textsc{in}(LabAsm)$ 4) $\textsc{in}(LabAsm') \cup \textsc{out}(LabAsm') \not\supset \textsc{in}(LabAsm) \cup \textsc{out}(LabAsm)$, and consequently $\textsc{undec}(LabAsm') \not\subset \textsc{undec}(LabAsm)$ 5) $\textsc{in}(LabAsm) \cup \textsc{out}(LabAsm) = \mathcal{A}$, and consequently $\textsc{undec}(LabAsm) = \emptyset$. Therefore, $LabAsm$ is a 1) grounded 2) preferred 3) ideal 4) semi-stable 5) stable assumption labelling.
2. Let $LabAsm$ be a 1) grounded 2) preferred 3) ideal 4) semi-stable 5) stable assumption labelling. By definition $LabAsm$ is a complete assumption labelling. Furthermore, for all complete assumption labellings $LabAsm'$ it holds that 1) $\textsc{in}(LabAsm') \not\subset \textsc{in}(LabAsm)$ 2) $\textsc{in}(LabAsm') \not\supset \textsc{in}(LabAsm)$ 3) if for all preferred assumption labellings $LabAsm''$ it holds that $\textsc{in}(LabAsm') \subseteq \textsc{in}(LabAsm'')$ then $\textsc{in}(LabAsm') \not\supset \textsc{in}(LabAsm)$ 4) $\textsc{undec}(LabAsm') \not\subset \textsc{undec}(LabAsm)$, or equivalently $\textsc{in}(LabAsm') \cup \textsc{out}(LabAsm') \not\supset \textsc{in}(LabAsm) \cup \textsc{out}(LabAsm)$ 5) $\textsc{undec}(LabAsm) = \emptyset$, or equivalently $\textsc{in}(LabAsm) \cup \textsc{out}(LabAsm) = \mathcal{A}$. By Theorem 4 $Asms = \textsc{in}(LabAsm)$ is a complete assumption extension with $Asms^+ = \textsc{out}(LabAsm)$ and $\mathcal{A} \setminus (Asms \cup Asms^+) = \textsc{undec}(LabAsm)$. Furthermore, from the above and by Theorem 4, for all complete assumption extensions $Asms'$ it holds that 1) $Asms' \not\subset Asms$ 2) $Asms' \not\supset Asms$ 3) if for all preferred assumption extensions $Asms''$ it holds that $Asms' \subseteq Asms''$ then $Asms' \not\supset Asms$ 4) $Asms' \cup Asms'^+ \not\supset Asms \cup Asms^+$ 5) $Asms \cup Asms^+ = \mathcal{A}$. Therefore, $Asms$ is a 1) grounded 2) preferred 3) ideal 4) semi-stable 5) stable assumption extension.

Proof of Proposition 7

- Preferred: Follows from the one-to-one correspondence between complete assumption labellings and extensions (Theorem 4) and between admissible assumption labellings and extensions (Theorem 1) together with Theorem 8 in [24].

- Ideal: Follows from the one-to-one correspondence between complete assumption labellings and extensions (Theorem 4) and between admissible assumption labellings and extensions (Theorem 1) together with Theorem 10 in [24].

- Semi-stable: From left to right: Let $LabAsm$ be a semi-stable assumption labelling, i.e. a complete assumption labelling such that $\text{UNDEC}(LabAsm)$ is minimal among all complete assumption labellings. By definition, $LabAsm$ is an admissible assumption labelling. Assume $\text{UNDEC}(LabAsm)$ is not minimal among all admissible assumptions labellings, i.e. $\exists LabAsm'$ with $\text{UNDEC}(LabAsm') \subset \text{UNDEC}(LabAsm)$ and $LabAsm'$ is an admissible assumption labelling but not a complete assumption labelling. Thus $LabAsm'$ satisfies Definition 2 but not Definition 3, so $\exists \alpha \in \text{UNDEC}(LabAsm')$ such that for all sets of assumptions $Asms$ attacking α there exists some $\beta \in Asms$ such that $LabAsm'(\beta) = \text{OUT}$. By Lemma 2, $LabAsm''$ with $\text{IN}(LabAsm'') = \text{IN}(LabAsm') \cup \{\alpha\}$, $\text{OUT}(LabAsm'') = \text{OUT}(LabAsm') \cup \alpha^{\star}$, and $\text{UNDEC}(LabAsm'') = \text{UNDEC}(LabAsm') \setminus (\{\alpha\} \cup \alpha^{\star})$ is an admissible assumption labelling. Clearly, $\text{UNDEC}(LabAsm'') \subset \text{UNDEC}(LabAsm')$, so $\text{UNDEC}(LabAsm')$ is not minimal among all admissible assumption labellings. Contradiction.
  From right to left: Let $LabAsm$ be an admissible assumption labelling such that $\text{UNDEC}(LabAsm)$ is minimal (w.r.t. $\subseteq$) among all admissible assumption labellings. Assume that $LabAsm$ is not a complete assumption labelling. By the same reasoning as above, $\exists \alpha \in \text{UNDEC}(LabAsm)$ such that for all sets of assumptions $Asms$ attacking α there exists some $\beta \in Asms$ such that $LabAsm(\beta) = \text{OUT}$. It follows that there exists an admissible assumption labelling $LabAsm''$ with $\text{UNDEC}(LabAsm'') \subset \text{UNDEC}(LabAsm)$. Contradiction. Thus, $LabAsm$ is a complete assumption labelling. Furthermore, since for all admissible assumption labellings $LabAsm'$, $\text{UNDEC}(LabAsm') \not\subset \text{UNDEC}(LabAsm)$ and since every complete assumption labelling is an admissible assumption labelling, it follows that for all complete assumption labellings $LabAsm'$, $\text{UNDEC}(LabAsm') \not\subset \text{UNDEC}(LabAsm)$. Thus, $\text{UNDEC}(LabAsm)$ is minimal (w.r.t. $\subseteq$) among all complete assumption labellings.

- Stable: From left to right: Let $LabAsm$ be a complete assumption labelling such that $\text{UNDEC}(LabAsm) = \emptyset$. By definition, $LabAsm$ is admissible.
  From right to left: Let $LabAsm$ be an admissible assumption labelling such that $\text{UNDEC}(LabAsm) = \emptyset$. Then $LabAsm$ is also a complete assumption labelling since the conditions for IN and OUT assumptions are the same for admissible and complete assumption labellings (see Definitions 2 and 3).

Proof of Lemma 8

Follows from Theorem 1 and Theorem 4.2 in [33].

Proof of Proposition 9

- From left to right: Let $LabAsm$ be a complete assumption labelling. By definition, $LabAsm$ is an admissible assumption labelling and by Lemma 8 an admissible assumption labelling with respect to argument-supporting sets. It remains to prove that the additional condition of complete assumption labellings with respect to argument-supporting sets is satisfied. Let $LabAsm(\alpha) = \textsc{undec}$. Then there exists a set of assumptions $Asms$ attacking α such that for all $\beta \in Asms$, $LabAsm(\beta) \neq \textsc{out}$. Thus, there exists an argument $Asms' \vdash \overline{\alpha}$ such that $Asms' \subseteq Asms$. Therefore, $Asms'$ is an argument-supporting set of assumptions attacking α such that for all $\gamma \in Asms'$, $LabAsm(\gamma) \neq \textsc{out}$.

- From right to left: Let $LabAsm$ be a complete assumption labelling with respect to argument-supporting sets. By definition, $LabAsm$ is and admissible assumption labelling with respect to argument-supporting sets and by Lemma 8 an admissible assumption labelling. It remains to prove that the additional condition of complete assumption labellings is satisfied. Let $LabAsm(\alpha) = \textsc{undec}$. Then there exists an argument-supporting set of assumptions $Asms$ attacking α such that for all $\gamma \in Asms$, $LabAsm(\gamma) \neq \textsc{out}$.

Proof of Proposition 11

Let there be an infinite sequence of edges $(\alpha_1, \alpha_2), (\alpha_2, \alpha_3), \ldots$ in $\mathcal{G}_{att}$. Then there exists a set of assumptions $Asms_{\alpha_1}$ attacking α_2 such that $\alpha_1 \in Asms_{\alpha_1}$, a set of assumptions $Asms_{\alpha_2}$ attacking α_3 such that $\alpha_2 \in Asms_{\alpha_2}$, and so on. Thus, in $\mathcal{G}$ there exists an edge from $Asms_{\alpha_1}$ to $\{\alpha_2\}$ as well as to every other set of assumptions containing α_2, in particular an edge to $Asms_{\alpha_2}$. Furthermore, there is an edge from $Asms_{\alpha_2}$ to $\{\alpha_3\}$ as well as every set of assumptions containing α_3, and so on. Thus, there is an infinite sequence of edges $(Asms_{\alpha_1}, Asms_{\alpha_2}), (Asms_{\alpha_2}, Asms_{\alpha_3}), \ldots$ in $\mathcal{G}$.
Example 13 proves that the converse does not hold.

Proof of Lemma 13

Note first that `LabAsm2LabArg` is clearly a function.

- Injective: We prove that no two different assumption labellings $LabAsm_1$ and $LabAsm_2$ are mapped to the same argument labelling by `LabAsm2LabArg`. Let $LabAsm_1 \neq LabAsm_2$. Thus, $\exists \alpha \in \mathcal{A}$ such that $LabAsm_1(\alpha) \neq LabAsm_2(\alpha)$. If $\alpha \in \textsc{in}(LabAsm_1)$ then $\alpha \notin \textsc{in}(LabAsm_2)$, so $\{\alpha\} \vdash \alpha \in \mathtt{in}(\mathtt{LabAsm2LabArg}(LabAsm_1))$ but $\{\alpha\} \vdash \alpha \notin \mathtt{in}(\mathtt{LabAsm2LabArg}(LabAsm_2))$. Analogous results are reached assuming that $\alpha \in \textsc{out}(LabAsm_1)$ and that $\alpha \in \textsc{undec}(LabAsm_1)$. Thus, $\mathtt{LabAsm2LabArg}(LabAsm_1) \neq \mathtt{LabAsm2LabArg}(LabAsm_2)$.

- Not generally surjective: The following ABA framework illustrates that there may be some $LabArg \in \mathscr{L}_{Arg}$ such that there exists no $LabAsm \in \mathscr{L}_{Asm}$ with $LabArg = \mathtt{LabAsm2LabArg}(LabAsm)$: $\mathcal{L} = \{r, \rho\}$, $\mathcal{R} = \{r \leftarrow\}$, $\mathcal{A} = \{\rho\}$, $\overline{\rho} = r$. There are three possible assumption labellings: $LabAsm_1 = \{(\rho, \textsc{in})\}$, $LabAsm_2 = \{(\rho, \textsc{out})\}$, and $LabAsm_3 = \{(\rho, \textsc{undec})\}$. The corresponding AA framework has two arguments: $Ar_{ABA} = \{A_1 : \{\rho\} \vdash \rho, A_2 : \{\} \vdash r\}$. In the translations of all three assumption labellings in terms of `LabAsm2LabArg`, A_2 is labelled `in`. Thus, for instance for the argument labelling $\{(A_1, \mathtt{in}), (A_2, \mathtt{out})\}$ there exists no $LabAsm$ such that $\mathtt{LabAsm2LabArg}(LabAsm) = \{(A_1, \mathtt{in}), (A_2, \mathtt{out})\}$.

Proof of Lemma 14

Note first that `LabArg2LabAsm` is clearly a function.

- Surjective: We prove that for every $LabAsm \in \mathscr{L}_{Asm}$ there exists some $LabArg \in \mathscr{L}_{Arg}$ such that $\texttt{LabArg2LabAsm}(LabArg) = LabAsm$. Let $LabAsm \in \mathscr{L}_{Asm}$. Furthermore, let $LabArg$ be an argument labelling which satisfies that for all $\alpha \in \mathcal{A}$, $\{\alpha\} \vdash \alpha \in \texttt{in}(LabArg)$ if $\alpha \in \textsc{in}(LabAsm)$, $\{\alpha\} \vdash \alpha \in \texttt{out}(LabArg)$ if $\alpha \in \textsc{out}(LabAsm)$, and $\{\alpha\} \vdash \alpha \in \texttt{undec}(LabArg)$ if $\alpha \in \textsc{undec}(LabAsm)$. Then $\texttt{LabArg2LabAsm}(LabArg) = LabAsm$. Clearly $LabArg \in \mathscr{L}_{Arg}$.

- Not generally injective: Consider the ABA framework from the proof of Lemma 13 and the two argument labellings $LabArg_1 = \{(A_1, \texttt{in}), (A_2, \texttt{out})\}$ and $LabArg_2 = \{(A_1, \texttt{in}), (A_2, \texttt{in})\}$. Then $\texttt{LabArg2LabAsm}(LabArg_1) = \texttt{LabArg2LabAsm}(LabArg_2) = \{(\rho, \textsc{in})\}$.

Lemma 47. *Let $Asms \subseteq \mathcal{A}$ and $Args = \{Asms' \vdash s \in Ar_{ABA} \mid Asms' \subseteq Asms\}$.*

- $Args^+ = \{Asms' \vdash s \in Ar_{ABA} \mid \exists \alpha \in Asms' : \alpha \in Asms^+\}$;
- $Ar_{ABA} \setminus (Args \cup Args^+) = \{Asms' \vdash s \in Ar_{ABA} \mid Asms' \nsubseteq Asms, \nexists \alpha \in Asms' : \alpha \in Asms^+\}$.

Proof. We prove both statements:

- $Args^+ = \{Asms' \vdash s \in Ar_{ABA} \mid Args \text{ attacks } Asms' \vdash s\}$
  $= \{Asms' \vdash s \in Ar_{ABA} \mid \exists \alpha \in Asms' : \exists Asms'' \vdash \overline{\alpha} \in Args\}$
  $= \{Asms' \vdash s \in Ar_{ABA} \mid \exists \alpha \in Asms' : \exists Asms'' \vdash \overline{\alpha} \text{ and } Asms'' \subseteq Asms\}$
  $= \{Asms' \vdash s \in Ar_{ABA} \mid \exists \alpha \in Asms' : Asms \text{ attacks } \alpha\}$
  $= \{Asms' \vdash s \in Ar_{ABA} \mid \exists \alpha \in Asms' : \alpha \in Asms^+\}$
- $Ar_{ABA} \setminus (Args \cup Args^+) = \{Asms' \vdash s \in Ar_{ABA} \mid Asms' \vdash s \notin Args, Asms' \vdash s \notin Args^+\} = \{Asms' \vdash s \in Ar_{ABA} \mid Asms' \nsubseteq Asms, \nexists \alpha \in Asms' : \alpha \in Asms^+\}$

Lemma 48. *Let $Args \subseteq Ar_{ABA}$ and let $Asms = \{\alpha \in \mathcal{A} \mid \exists Asms' : \alpha \in Asms' \text{ and } Asms' \vdash s \in Args\}$. Then*

- $Asms^+ = \{\alpha \in \mathcal{A} \mid \{\alpha\} \vdash \alpha \in Args^+\}$;
- $\mathcal{A} \setminus (Asms \cup Asms^+) = \{\alpha \in \mathcal{A} \mid \{\alpha\} \vdash \alpha \notin Args, \{\alpha\} \vdash \alpha \notin Args^+\}$.

Proof. We prove both statements:

- $Asms^+ = \{\alpha \in \mathcal{A} \mid Asms \text{ attacks } \alpha\} = \{\alpha \in \mathcal{A} \mid \exists Asms' \vdash \overline{\alpha} : Asms' \subseteq Asms\}$
  $= \{\alpha \in \mathcal{A} \mid \exists Asms' \vdash \overline{\alpha} \in Args\} = \{\alpha \in \mathcal{A} \mid Args \text{ attacks } \{\alpha\} \vdash \alpha\}$
  $= \{\alpha \in \mathcal{A} \mid \{\alpha\} \vdash \alpha \in Args^+\}$
- $\mathcal{A} \setminus (Asms \cup Asms^+) = \{\alpha \in \mathcal{A} \mid \alpha \notin Asms, \alpha \notin Asms^+\}$
  $= \{\alpha \in \mathcal{A} \mid \nexists Asms' : \alpha \in Asms' \text{ and } Asms' \vdash s \in Args, \{\alpha\} \vdash \alpha \notin Args^+\}$
  $= \{\alpha \in \mathcal{A} \mid \{\alpha\} \vdash \alpha \notin Args, \{\alpha\} \vdash \alpha \notin Args^+\}$

Proof of Theorem 15

- From left to right: Let LabAsm be a complete assumption labelling. Firstly note that for all $\mathit{Asms} \vdash s \in \mathit{Ar}_{ABA}$ exactly one of the three conditions in the definition of $\texttt{LabAsm2LabArg}$ applies, so all $\mathit{Asms} \vdash s$ are in exactly one of $\texttt{in}(\mathit{LabArg})$, $\texttt{out}(\mathit{LabArg})$, or $\texttt{undec}(\mathit{LabArg})$.

  By Theorem 4: $\mathit{Asms} = \textsc{in}(\mathit{LabAsm})$ is a complete assumption extension with $\mathit{Asms}^+ = \textsc{out}(\mathit{LabAsm})$ and $\mathcal{A} \setminus (\mathit{Asms} \cup \mathit{Asms}^+) = \textsc{undec}(\mathit{LabAsm})$.

  By Theorem 6.1 in [24]: $\mathit{Args} = \{\mathit{Asms}' \vdash s \mid \mathit{Asms}' \subseteq \textsc{in}(\mathit{LabAsm})\}$ is a complete argument extension.

  By Lemma 47: $\mathit{Args}^+ = \{\mathit{Asms}' \vdash s \mid \exists \alpha \in \mathit{Asms}' : \alpha \in \textsc{out}(\mathit{LabAsm})\}$ and $\mathit{Ar}_{ABA} \setminus (\mathit{Args} \cup \mathit{Args}^+) = \{\mathit{Asms}' \vdash s \mid \mathit{Asms}' \nsubseteq \textsc{in}(\mathit{LabAsm}), \nexists \alpha \in \mathit{Asms}' : \alpha \in \textsc{out}(\mathit{LabAsm})\} = \{\mathit{Asms}' \vdash s \mid \exists \alpha \in \mathit{Asms}' : \alpha \in \textsc{undec}(\mathit{LabAsm}), \mathit{Asms}' \cap \textsc{out}(\mathit{LabAsm}) = \emptyset\}$.

  By Theorem 10 in [25]: LabArg with $\texttt{in}(\mathit{LabArg}) = \mathit{Args}$, $\texttt{out}(\mathit{LabArg}) = \mathit{Args}^+$, $\texttt{undec}(\mathit{LabArg}) = \mathit{Ar}_{ABA} \setminus (\mathit{Args} \cup \mathit{Args}^+)$ is a complete argument labelling.

- From right to left: Let $\mathit{LabArg} = \texttt{LabAsm2LabArg}(\mathit{LabAsm})$ be a complete argument labelling where LabAsm is an assumption labelling. Since $\texttt{LabAsm2LabArg}$ is injective by Lemma 13, LabAsm is unique.

  By Theorems 9 and 11 in [25]: $\mathit{Args} = \texttt{in}(\mathit{LabArg}) = \{\mathit{Asms}' \vdash s \mid \mathit{Asms}' \subseteq \textsc{in}(\mathit{LabAsm})\}$ is a complete argument extension with $\mathit{Args}^+ = \texttt{out}(\mathit{LabArg}) = \{\mathit{Asms}' \vdash s \mid \exists \alpha \in \mathit{Asms}' : \alpha \in \textsc{out}(\mathit{LabAsm})\}$ and $\mathit{Ar}_{ABA} \setminus (\mathit{Args} \cup \mathit{Args}^+) = \texttt{undec}(\mathit{LabArg}) = \{\mathit{Asms}' \vdash s \mid \exists \alpha \in \mathit{Asms}' : \alpha \in \textsc{undec}(\mathit{LabAsm}), \mathit{Asms}' \cap \textsc{out}(\mathit{LabAsm}) = \emptyset\}$.

  By Theorem 6.1 in [24]: $\mathit{Asms} = \{\alpha \in \mathcal{A} \mid \exists \mathit{Asms}' : \alpha \in \mathit{Asms}' \textit{ and } \mathit{Asms}' \vdash s \in \mathit{Args}\} = \textsc{in}(\mathit{LabAsm})$ is a complete assumption extension.

  By Lemma 48: $\mathit{Asms}^+ = \{\alpha \mid \{\alpha\} \vdash \alpha \in \mathit{Args}^+\} = \{\alpha \mid \alpha \in \textsc{out}(\mathit{LabAsm})\} = \textsc{out}(\mathit{LabAsm})$ and $\mathcal{A} \setminus (\mathit{Asms} \cup \mathit{Asms}^+) = \{\alpha \mid \{\alpha\} \vdash \alpha \notin \mathit{Args}, \{\alpha\} \vdash \alpha \notin \mathit{Args}^+\} = \{\alpha \mid \{\alpha\} \vdash \alpha \in \texttt{undec}(\mathit{LabArg})\} = \{\alpha \mid \alpha \in \textsc{undec}(\mathit{LabAsm})\} = \textsc{undec}(\mathit{LabAsm})$.

  By Theorem 4: LabAsm is a complete assumption labelling.

Proof of Proposition 16

By Theorem 9 in [25], $\mathit{Args} = \texttt{in}(\mathit{LabArg})$ is a complete argument extension.
By Theorem 11 in [25], $\mathit{Args}^+ = \texttt{out}(\mathit{LabArg})$ and $\mathit{Ar}_{ABA} \setminus (\mathit{Args} \cup \mathit{Args}^+) = \texttt{undec}(\mathit{LabArg})$.
By Theorem 6.1 in [24], $\mathit{Asms} = \{\alpha \mid \exists \mathit{Asms}' : \alpha \in \mathit{Asms}', \mathit{Asms}' \vdash s \in \mathit{Args}\}$ is a complete assumption extension.
From Theorem 6.1 and Proposition 1 in [24] it also follows that $\mathit{Args} = \{\mathit{Asms}' \vdash s \mid \mathit{Asms}' \subseteq \mathit{Asms}\}$. By Lemma 47, $\mathit{Args}^+ = \{\mathit{Asms}' \vdash s \mid \exists \alpha \in \mathit{Asms}' : \alpha \in \mathit{Asms}^+\}$, and $\mathit{Ar}_{ABA} \setminus (\mathit{Args} \cup \mathit{Args}^+) = \{\mathit{Asms}' \vdash s \mid \mathit{Asms}' \nsubseteq \mathit{Asms}, \nexists \alpha \in \mathit{Asms}' : \alpha \in \mathit{Asms}^+\}$.
By Theorem 4, LabAsm with $\textsc{in}(\mathit{LabAsm}) = \mathit{Asms}$, $\textsc{out}(\mathit{LabAsm}) = \mathit{Asms}^+$, and $\textsc{undec}(\mathit{LabAsm}) = \mathcal{A} \setminus (\mathit{Asms} \cup \mathit{Asms}^+)$ is a complete assumption labelling.
It follows that, $\mathit{Args} = \{\mathit{Asms}' \vdash s \mid \mathit{Asms}' \subseteq \textsc{in}(\mathit{LabAsm})\} = \texttt{in}(\mathit{LabArg})$. Furthermore, $\mathit{Args}^+ = \{\mathit{Asms}' \vdash s \mid \exists \alpha \in \mathit{Asms}' : \alpha \in \textsc{out}(\mathit{LabAsm})\} = \texttt{out}(\mathit{LabArg})$, and $\mathit{Ar}_{ABA} \setminus (\mathit{Args} \cup \mathit{Args}^+) = \{\mathit{Asms}' \vdash s \mid \mathit{Asms}' \nsubseteq \textsc{in}(\mathit{LabAsm}), \nexists \alpha \in \mathit{Asms}' : \alpha \in \textsc{out}(\mathit{LabAsm})\} = \{\mathit{Asms}' \vdash s \mid \exists \alpha \in \mathit{Asms}' : \alpha \in \textsc{undec}(\mathit{LabAsm}), \mathit{Asms}' \cap \textsc{out}(\mathit{LabAsm}) = \emptyset\} = \texttt{undec}(\mathit{LabArg})$.
Thus, $\texttt{LabAsm2LabArg}(\mathit{LabAsm}) = \mathit{LabArg}$. Since $\texttt{LabAsm2LabArg}$ is injective by Lemma 13, LabAsm is unique.

Proof of Theorem 17

By Theorems 9 and 11 in [25], $Args = \mathtt{in}(LabArg)$ is a complete argument extension with $Args^+ = \mathtt{out}(LabArg)$ and $Ar_{ABA} \setminus (Args \cup Args^+) = \mathtt{undec}(LabArg)$.

By Theorem 6.1 in [24], $Asms = \{\alpha \mid \exists Asms' : \alpha \in Asms' \textit{ and } Asms' \vdash s \in \mathtt{in}(LabArg)\}$ is a complete assumption extension.

By Lemma 48, $Asms^+ = \{\alpha \mid \{\alpha\} \vdash \alpha \in Args^+\} = \{\alpha \mid \{\alpha\} \vdash \alpha \in \mathtt{out}(LabArg)\}$ and $\mathcal{A} \setminus (Asms \cup Asms^+) = \{\alpha \mid \{\alpha\} \vdash \alpha \notin Args, \{\alpha\} \vdash \alpha \notin Args^+\} = \{\alpha \mid \{\alpha\} \vdash \alpha \in \mathtt{undec}(LabArg)\}$.

Since for an argument $Asms' \vdash s \in \mathtt{in}(LabArg)$ it holds that all attackers are labelled $\mathtt{out}$, it follows that $\forall \alpha \in Asms'$: all attackers of $\{\alpha\} \vdash \alpha$ are labelled $\mathtt{out}$, so by the definition of complete argument labellings $\{\alpha\} \vdash \alpha \in \mathtt{in}(LabArg)$. Thus, $Asms = \{\alpha \mid \{\alpha\} \vdash \alpha \in \mathtt{in}(LabArg)\}$.

By Theorem 4, $LabAsm$ with $\textsc{in}(LabAsm) = Asms$, $\textsc{out}(LabAsm) = Asms^+$ and $\textsc{undec}(LabAsm) = \mathcal{A} \setminus (Asms \cup Asms^+)$ is a complete assumption labelling.

Proof of Lemma 18

Let $LabArg = \mathtt{LabAsm2LabArg}(LabAsm)$, so by Theorem 15 $LabArg$ is a complete argument labelling. Now let $LabAsm' = \mathtt{LabArg2LabAsm}(LabArg)$, so $\textsc{in}(LabAsm') = \{\alpha \mid \{\alpha\} \subseteq \textsc{in}(LabAsm)\} = \textsc{in}(LabAsm)$, $\textsc{out}(LabAsm') = \{\alpha \mid \alpha \in \textsc{out}(LabAsm)\} = \textsc{out}(LabAsm)$, $\textsc{undec}(LabAsm') = \{\alpha \mid \alpha \in \textsc{undec}(LabAsm), \{\alpha\} \cap \textsc{out}(LabAsm) = \emptyset\} = \textsc{undec}(LabAsm)$. Thus, $LabAsm = LabAsm'$, so there exists a complete argument labelling $LabArg$ such that $\mathtt{LabArg2LabAsm}(LabArg) = LabAsm$.

Proof of Lemma 19

Let $\mathtt{LabArg2LabAsm}(LabArg_1) = LabAsm_1$ and $\mathtt{LabArg2LabAsm}(LabArg_2) = LabAsm_2$. Assume by contradiction that $LabAsm_1 = LabAsm_2$. Since $LabArg_1 \neq LabArg_2$, $\exists Asms_1 \vdash s_1 \in Ar_{ABA}$ such that $LabArg_1(Asms_1 \vdash s_1) \neq LabArg_2(Asms_1 \vdash s_1)$.

- Let $Asms_1 \vdash s_1 \in \mathtt{in}(LabArg_1)$, so $Asms_1 \vdash s_1 \notin \mathtt{in}(LabArg_2)$. Then there exists some $Asms_2 \vdash \overline{\alpha}$ attacking $Asms_1 \vdash s_1$ where $\alpha \in Asms_1$ and $Asms_2 \vdash \overline{\alpha} \notin \mathtt{out}(LabArg_2)$. However, $Asms_2 \vdash \overline{\alpha} \in \mathtt{out}(LabArg_1)$ since all attackers of $Asms_1 \vdash s_1$ are labelled $\mathtt{out}$ by $LabArg_1$. Thus, $\{\alpha\} \vdash \alpha \notin \mathtt{in}(LabArg_2)$ but $\{\alpha\} \vdash \alpha \in \mathtt{in}(LabArg_1)$, so $\alpha \in \mathtt{in}(LabAsm_1)$ but $\alpha \notin \mathtt{in}(LabAsm_2)$. Contradiction.

- Let $Asms_1 \vdash s_1 \in \mathtt{out}(LabArg_1)$, so $Asms_1 \vdash s_1 \notin \mathtt{out}(LabArg_2)$. Then there exists some $Asms_2 \vdash \overline{\alpha}$ attacking $Asms_1 \vdash s_1$ where $\alpha \in Asms_1$ and $Asms_2 \vdash \overline{\alpha} \in \mathtt{in}(LabArg_1)$. However, $Asms_2 \vdash \overline{\alpha} \notin \mathtt{in}(LabArg_2)$ since no attacker of $Asms_1 \vdash s_1$ is labelled $\mathtt{in}$ by $LabArg_2$. Thus, $\{\alpha\} \vdash \alpha \in \mathtt{out}(LabArg_1)$ but $\{\alpha\} \vdash \alpha \notin \mathtt{out}(LabArg_2)$, so $\mathtt{LabArg2LabAsm}$ that $\alpha \in \textsc{out}(LabAsm_1)$ but $\alpha \notin \textsc{out}(LabAsm_2)$. Contradiction.

- Let $Asms_1 \vdash s_1 \in \mathtt{undec}(LabArg_1)$, so $Asms_1 \vdash s_1 \notin \mathtt{undec}(LabArg_2)$. Then either for all $Asms_2 \vdash \overline{\alpha}$ attacking $Asms_1 \vdash s_1$ where $\alpha \in Asms_1$ it holds that $Asms_2 \vdash \overline{\alpha} \in \mathtt{out}(LabArg_2)$ or there exists some $Asms_3 \vdash \overline{\beta}$ attacking $Asms_1 \vdash s_1$ where $\beta \in Asms_1$ and $Asms_3 \vdash \overline{\beta} \in \mathtt{in}(LabArg_2)$. In the first case for all $\{\alpha\} \vdash \alpha$, $\{\alpha\} \vdash \alpha \in \mathtt{in}(LabArg_2)$ but some $\{\alpha\} \vdash \alpha \in \mathtt{undec}(LabArg_1)$ since there exists an attacker $Asms_2 \vdash \overline{\alpha}$ of $Asms_1 \vdash s_1$ such that $Asms_2 \vdash \overline{\alpha} \notin \mathtt{out}(LabArg_1)$. It follows that $\alpha \in \textsc{undec}(LabAsm_1)$ but $\alpha \notin \textsc{undec}(LabAsm_2)$. Contradiction. In the second case, $\{\beta\} \vdash \beta \in \mathtt{out}(LabArg_2)$ but $\{\beta\} \vdash \beta \notin \mathtt{out}(LabArg_1)$ since no attacker of $Asms_1 \vdash s_1$ is labelled $\mathtt{in}$ by $LabArg_1$. Thus, $\beta \in \textsc{out}(LabAsm_2)$ but $\beta \notin \textsc{out}(LabAsm_1)$. Contradiction.

Proof of Theorem 22
Analogous to the proof of Theorem 15 but using Theorem 5 instead of Theorem 4, Theorem 6.2/6.3/6.4/6.5 in [24] instead of Theorem 6.1 in [24], the analogues of Theorems 10 and 11 in [25] for the grounded/preferred/stable semantics (only informally given in [25]) instead of Theorems 9, 10, and 11 in [25], and Theorem 3.7 in [32] for the ideal semantics instead of Theorems 9, 10, and 11 in [25].

Proof of Theorem 23
Analogous to the proof of Theorem 17 but using Theorem 5 instead of Theorem 4, Theorem 6.2/6.3/6.4/6.5 in [24] instead of Theorem 6.1 in [24], the analogues of Theorems 9 and 11 in [25] for the grounded/preferred/stable semantics (only informally given in [25]) instead of Theorems 9 and 11 in [25], and Theorem 3.7 in [32] for the ideal semantics instead of Theorems 9 and 11 in [25].

Proof of Theorem 24
Analogous to the "from left to right" part of the proof of Theorem 15 but using Theorem 1 instead of Theorem 4, Theorem 2.2 in [23] instead of Theorem 6.1 in [24], and Theorem 21 in [25] instead of Theorem 10 in [25].

Proof of Proposition 25
Analogous to the proof of Lemma 18 but using Theorem 24 instead of Theorem 15.

Proof of Theorem 27

1. First note that $Args \cap Args^+ = \emptyset$ since $Args$ does not attack itself. Thus each $A \in Ar$ is either contained in $\mathtt{in}(LabArg)$, $\mathtt{out}(LabArg)$, or $\mathtt{undec}(LabArg)$, so $LabArg$ is an argument labelling. We prove that $LabArg$ satisfies Definition 13.
   - Let $LabArg(A) = \mathtt{in}$. Then $A \in Args$. Thus, all attackers B of A are attacked by some $C \in Args$, so $B \in Args^+$. Consequently, for each attacker B of A, $LabArg(B) = \mathtt{out}$.
   - Let $LabArg(A) = \mathtt{out}$. Then $A \in Args^+$. Thus, A is attacked by some $B \in Args$, and therefore there exists some B attacking A such that $LabArg(B) = \mathtt{in}$.
   - Let $LabArg(A) = \mathtt{undec}$. Then $A \notin Args^+$. Thus, A is not attacked by any $B \in Args$ and consequently there exists no B attacking A such that $LabArg(B) = \mathtt{in}$.
2. We prove that $\mathtt{in}(LabArg)$ is an admissible argument extension.
   - $\mathtt{in}(LabArg)$ is conflict-free: Assume $\mathtt{in}(LabArg)$ is not conflict-free. Then there exist $A, B \in \mathtt{in}(LabArg)$ such that A attacks B, so B is attacked by an argument which is not labelled $\mathtt{out}$. Contradiction.
   - All arguments in $\mathtt{in}(LabArg)$ are defended by $\mathtt{in}(LabArg)$: Let $A \in \mathtt{in}(LabArg)$. Then for each attacker B of A, $LabArg(B) = \mathtt{out}$ and therefore for each B there exists an attacker C such that $LabArg(C) = \mathtt{in}$. Thus, each attacker of A is attacked by $\mathtt{in}(LabArg)$, i.e. $\mathtt{in}(LabArg)$ defends A.

   $Args^+ = \{A \mid Args \ attacks \ A\} = \{A \mid \mathtt{in}(LabArg) \ attacks \ A\}$
   $= \{A \mid A \in \mathtt{out}(LabArg)\} = \mathtt{out}(LabArg)$
   $Ar \setminus (Args \cup Args^+) = \{A \mid A \notin Args, A \notin Args^+\}$
   $= \{A \mid A \notin \mathtt{in}(LabArg), A \notin \mathtt{out}(LabArg)\} = \{A \mid A \in \mathtt{undec}(LabArg)\} = \mathtt{undec}(LabArg)$

Proof of Theorem 28

Analogous to the proof of Theorem 15, but using Theorem 1 instead of Theorem 4, Theorem 27 instead of Theorems 10 and 11 in [25], and Theorem 2.2 in [23] instead of Theorem 6.1 in [24].

Proof of Theorem 29

Analogous to the proof of Theorem 17 but using Theorem 27 instead of Theorems 9 and 11 in [25], Theorem 2.2 in [23] instead of Theorem 6.1 in [24], and Theorem 1 instead of Theorem 4.

Proof of Proposition 30

Analogous to the proof of Lemma 18 but using Theorem 28 instead of Theorem 15.

Proof of Theorem 32

1. First note that $Asms \cap Asms^+ = \emptyset$ since $Asms$ does not attack itself. Thus each $\alpha \in \mathcal{A}$ is either contained in $\text{IN}(LabAsm)$, in $\text{OUT}(LabAsm)$, or in $\text{UNDEC}(LabAsm)$. We prove that $LabAsm$ satisfies Definition 15.
    - Let $LabAsm(\alpha) = \text{IN}$. Then $\alpha \in Asms$, so $Asms$ defends α, i.e. for all closed sets of assumptions $Asms_1$ attacking α there exists some $\beta \in Asms_1$ such that $Asms$ attacks β. Thus, $\beta \in Asms^+$ and consequently $LabAsm(\beta) = \text{OUT}$. Therefore, for each closed set of assumptions $Asms_1$ attacking α there exists some $\beta \in Asms_1$ such that $LabAsm(\beta) = \text{OUT}$.
    - Let $LabAsm(\alpha) = \text{OUT}$. Then $\alpha \in Asms^+$, so $Asms$ attacks α. Since $Asms = \text{IN}(LabAsm)$ and since $Asms$ is a closed set of assumptions, there exists a closed set of assumptions $Asms_1$ attacking α such that for all $\beta \in Asms_1$, $LabAsm(\beta) = \text{IN}$. Furthermore, since $Asms$ is a closed set of assumptions, for all δ supported by $Asms$ it holds that $\delta \in Asms$. Since $\alpha \in Asms^+$ and since $Asms \cap Asms^+ = \emptyset$, it follows that $\alpha \notin Asms$ and therefore α is not supported by $Asms$. Thus there exists no set of assumptions $Asms_2$ supporting α such that for all $\gamma \in Asms_2$, $LabAsm(\gamma) = \text{IN}$.
    - Let $LabAsm(\alpha) = \text{UNDEC}$. Then $\alpha \notin Asms$ and $\alpha \notin Asms^+$, so α is not attacked and not defended by $Asms$. Since α is not attacked by $Asms$, for each closed set of assumptions $Asms_1$ attacking α there exists some $\beta \in Asms_1$ such that $\beta \notin Asms$, and thus $LabAsm(\beta) \neq \text{IN}$. Furthermore, since $Asms$ is a closed set of assumptions, it follows from the same reasoning as in the previous item that there exists no set of assumptions $Asms_2$ supporting α such that for all $\gamma \in Asms_2$, $LabAsm(\gamma) = \text{IN}$.
2. We first prove that $\text{IN}(LabAsm)$ is an admissible assumption extension.
    - $\text{IN}(LabAsm)$ is closed: Assume $\text{IN}(LabAsm)$ is not closed. Then $\exists \alpha \notin \text{IN}(LabAsm)$ such that $\text{IN}(LabAsm)$ supports α. Thus, $LabAsm(\alpha) = \text{OUT}$ or $LabAsm(\alpha) = \text{UNDEC}$. Contradiction since in either case there exists no set of assumptions $Asms_1$ supporting α such that for all $\gamma \in Asms_1$, $LabAsm(\gamma) = \text{IN}$.
    - $\text{IN}(LabAsm)$ is conflict-free: Assume $\text{IN}(LabAsm)$ is not conflict-free. Then $\text{IN}(LabAsm)$ attacks some $\alpha \in \text{IN}(LabAsm)$. By Definition 15, for each closed set of assumptions $Asms_1$ attacking α there exists some $\beta \in Asms_1$ such that $LabAsm(\beta) = \text{OUT}$. Since $\text{IN}(LabAsm)$ is a closed set of assumptions, there exists some $\beta \in \text{IN}(LabAsm)$ such that $LabAsm(\beta) = \text{OUT}$. Contradiction.

- IN($LabAsm$) defends all $\alpha \in$ IN($LabAsm$): Let $\alpha \in$ IN($LabAsm$). Then by Definition 15, for each closed set of assumptions $Asms_1$ attacking α there exists some $\beta \in Asms_1$ such that $LabAsm(\beta) =$ OUT. Furthermore, for each such β there exists a closed set of assumptions $Asms_2$ attacking β such that for all $\gamma \in Asms_2$, $LabAsm(\gamma) =$ IN so $Asms_2 \subseteq$ IN($LabAsm$). Hence, IN($LabAsm$) attacks all closed sets of assumptions attacking α.

$Asms^+ =$ OUT($LabAsm$) and $\mathcal{A} \setminus (Asms \cup Asms^+) =$ UNDEC($LabAsm$) as in the proof of Theorem 1.

Proof of Theorem 34

1. Since $Asms$ is a complete assumption extension it is by definition also an admissible assumption extension [4]. By Theorem 32, $LabAsm$ is an admissible assumption labelling. It remains to prove that the additional condition of complete assumption labellings is satisfied. Let $LabAsm(\alpha) =$ UNDEC. Then $\alpha \notin Asms$ and $\alpha \notin Asms^+$, so α is not attacked and not defended by $Asms$. Since α is not defended by $Asms$, there exists a closed set of assumption $Asms_1$ attacking α such that $Asms_1$ is not attacked by $Asms$. Thus, for all $\gamma \in Asms_1$ it holds that $\gamma \notin Asms^+$. Consequently, $LabAsm(\gamma) \neq$ OUT.
2. Since $LabAsm$ is a complete assumption labelling it is by Definition 16 also an admissible assumption labelling. Thus, by Theorem 32 $Asms$ is an admissible assumption extension with $Asms^+ =$ OUT($LabAsm$) and $\mathcal{A} \setminus (Asms \cup Asms^+) =$ UNDEC($LabAsm$). It remains to prove that all assumptions defended by $Asms$ are contained in $Asms$. Let α be defended by $Asms$ and thus by IN($LabAsm$). Then for each closed set of assumptions $Asms_1$ attacking α, IN($LabAsm$) attacks $Asms_1$. Thus, for each such $Asms_1$ there exists some $\beta \in Asms_1$ which is attacked by IN($LabAsm$), and therefore $LabAsm(\beta) =$ OUT. Since this holds for each $Asms_1$ attacking α, $LabAsm(\alpha) =$ IN.

Proof of Proposition 35

First item implies second item:

- Let α be such that there exists a set of assumptions $Asms$ supporting α such that for all $\beta \in Asms$, $LabAsm(\beta) =$ IN. If $LabAsm(\alpha) =$ OUT or $LabAsm(\alpha) =$ UNDEC then the second or third, respectively, condition of complete assumption labellings is violated. Thus $LabAsm(\alpha) =$ IN since it satisfies the first condition.

- Let α be such that for each closed set of assumptions $Asms$ attacking α there exists some $\beta \in Asms$ such that $LabAsm(\beta) =$ OUT. If $LabAsm(\alpha) =$ OUT or $LabAsm(\alpha) =$ UNDEC then the second or third, respectively, condition of complete assumption labellings is violated. Thus $LabAsm(\alpha) =$ IN since it satisfies the first condition.

- Let α be such that there exists a closed set of assumptions $Asms$ attacking α such that for all $\beta \in Asms$, $LabAsm(\beta) =$ IN. If $LabAsm(\alpha) =$ IN or $LabAsm(\alpha) =$ UNDEC then the first or third, respectively, condition of complete assumption labellings is violated. Thus $LabAsm(\alpha) =$ OUT since it satisfies the second condition.

- Let α be such that for each closed set of assumptions $Asms_1$ attacking α there exists some $\beta \in Asms_1$ such that $LabAsm(\beta) \neq$ IN, and there exists a closed set of assumptions $Asms_2$ attacking α such that for all $\gamma \in Asms_2$, $LabAsm(\gamma) \neq$ OUT. If

$LabAsm(\alpha) = \text{IN}$ or $LabAsm(\alpha) = \text{OUT}$ then the first or second, respectively, condition of complete assumption labellings is violated. Thus $LabAsm(\alpha) = \text{UNDEC}$ since it satisfies the third condition.

Second item implies first item.

- Let $LabAsm(\alpha) = \text{IN}$. Then for each closed set of assumptions $Asms_1$ attacking α there exists some $\beta \in Asms_1$ such that $LabAsm(\beta) \neq \text{IN}$. Furthermore, it either holds that there exists a closed set of assumptions $Asms_2$ attacking α such that for all $\gamma \in Asms_2$, $LabAsm(\gamma) = \text{IN}$, (contradiction) or that for each closed set of assumptions $Asms_3$ attacking α there exists some $\delta \in Asms_3$ such that $LabAsm(\delta) = \text{OUT}$. Thus, the second part of the or-statement applies.

- Let $LabAsm(\alpha) = \text{OUT}$. Then there exists no set of assumptions $Asms_1$ supporting α such that for all $\beta \in Asms_1$, $LabAsm(\beta) = \text{IN}$. Furthermore, there exists a closed set of assumptions $Asms_2$ attacking α such that for all $\gamma \in Asms_2$, $LabAsm(\gamma) \neq \text{OUT}$. Furthermore, it either holds that there exists a closed set of assumptions $Asms_3$ attacking α such that for all $\delta \in Asms_3$, $LabAsm(\delta) = \text{IN}$, or that for each closed set of assumptions $Asms_4$ attacking α there exists some $\epsilon \in Asms_4$ such that $LabAsm(\epsilon) = \text{OUT}$ (contradiction). Thus, the first part of the or-statement applies.

- Let $LabAsm(\alpha) = \text{UNDEC}$. Then there exists no set of assumptions $Asms_1$ supporting α such that for all $\beta \in Asms_1$, $LabAsm(\beta) = \text{IN}$. Furthermore, there exists a closed set of assumptions $Asms_2$ attacking α such that for all $\gamma \in Asms_2$, $LabAsm(\gamma) \neq \text{OUT}$. Furthermore, for each closed set of assumptions $Asms_3$ attacking α there exists some $\delta \in Asms_3$ such that $LabAsm(\delta) \neq \text{IN}$.

Proof of Theorem 36

1. First note that since $Asms$ is the intersection of all complete assumption labellings, which are all conflict-free, it follows that $Asms \cap Asms^+ = \emptyset$ and thus each $\alpha \in \mathcal{A}$ is either contained in $\text{IN}(LabAsm)$, $\text{OUT}(LabAsm)$, or $\text{UNDEC}(LabAsm)$, so $LabAsm$ is an assumption labelling. Furthermore, note that grounded assumption extensions are always closed, even though this is not explicitly required in their definition. Since the grounded assumption extension is a subset of every complete assumption extension, any assumption α supported by the grounded assumption extension is also supported by each complete assumption extension. Since complete assumption extensions are closed, α is thus in each complete assumption extension and consequently part of the grounded assumption extension. We prove that $LabAsm$ satisfies Definition 17.
   - From left to right: Let $LabAsm(\alpha) = \text{IN}$. Then $\alpha \in Asms$. Therefore, for all complete assumption extensions $Asms'$, $\alpha \in Asms'$. By Theorem 34, for each $Asms'$ it holds that $LabAsm'$ with $\text{IN}(LabAsm') = Asms'$, $\text{OUT}(LabAsm') = Asms'^+$, and $\text{UNDEC}(LabAsm') = \mathcal{A} \backslash (Asms' \cup Asms'^+)$ is a complete assumption labelling and there are no other complete assumption labellings. Thus, for all complete assumption labellings $LabAsm'$, $LabAsm'(\alpha) = \text{IN}$.
     From right to left: Let α be such that for all complete assumption labellings $LabAsm'$, $LabAsm'(\alpha) = \text{IN}$. Then by Theorem 34, for each $LabAsm'$ it holds that $Asms' = \text{IN}(LabAsm')$ is a complete assumption extension and there are no other complete assumption extensions. Thus, for all complete assumption extensions $Asms'$, $\alpha \in Asms'$. Therefore, $\alpha \in Asms$ and thus $LabAsm(\alpha) = \text{IN}$.

- From left to right: Let $\mathit{LabAsm}(\alpha) = \textsc{out}$. Then $\alpha \in \mathit{Asms}^+$. Thus, α is attacked by Asms. Assume Asms is not closed. Then $\exists \beta \notin \mathit{Asms}$ and $\mathit{Asms}_1 \subseteq \mathit{Asms}$ such that $\mathit{Asms}_1 \vdash \beta$. Since for all complete assumption extensions Asms' it holds that $\mathit{Asms} \subseteq \mathit{Asms}'$ and Asms' is closed, it follows that $\beta \in \mathit{Asms}'$. Thus, $\beta \in \mathit{Asms}$, which is a contradiction. Thus, Asms is closed, so α is attacked by by a closed set of assumptions all labelled $\textsc{in}$ by LabAsm.
  From right to left: Let α be such that there exists a closed set of assumptions Asms_1 attacking α such that for all $\beta \in \mathit{Asms}_1$, $\mathit{LabAsm}(\beta) = \textsc{in}$. Then $\mathit{Asms}_1 \subseteq \mathit{Asms}$, so α is attacked by Asms. Therefore, $\alpha \in \mathit{Asms}^+$ and thus $\mathit{LabAsm}(\alpha) = \textsc{out}$.

2. Since LabAsm is a grounded assumption labelling, it holds that for all $\alpha \in \mathcal{A}$: $\mathit{LabAsm}(\alpha) = \textsc{in}$ if and only if for all complete assumption labellings LabAsm', $\mathit{LabAsm}'(\alpha) = \textsc{in}$. By Theorem 34, for each LabAsm' it holds that $\mathit{Asms}' = \textsc{in}(\mathit{LabAsm}')$ with $\mathit{Asms}'^+ = \textsc{out}(\mathit{LabAsm}')$ and $\mathcal{A} \setminus (\mathit{Asms}' \cup \mathit{Asms}'^+) = \textsc{undec}(\mathit{LabAsm}')$ is a complete assumption extension and there are no other complete assumption extensions. Thus, for all $\alpha \in \mathcal{A}$: $\alpha \in \mathit{Asms}$ if and only if for all complete assumption extensions Asms', $\alpha \in \mathit{Asms}'$. Therefore, Asms is the intersection of all complete assumption extensions.
   $\mathit{Asms}^+ = \{\alpha \in \mathcal{A} \mid \mathit{Asms}\ \mathit{attacks}\ \alpha\} = \{\alpha \in \mathcal{A} \mid \textsc{in}(\mathit{LabAsm})\ \mathit{attacks}\ \alpha\}$
   $= \{\alpha \in \mathcal{A} \mid \alpha \in \textsc{out}(\mathit{LabAsm})\} = \textsc{out}(\mathit{LabAsm})$
   $\mathcal{A} \setminus (\mathit{Asms} \cup \mathit{Asms}^+) = \{\alpha \in \mathcal{A} \mid \alpha \notin \textsc{in}(\mathit{LabAsm}), \alpha \notin \textsc{out}(\mathit{LabAsm})\}$
   $= \{\alpha \in \mathcal{A} \mid \alpha \in \textsc{undec}(\mathit{LabAsm})\} = \textsc{undec}(\mathit{LabAsm})$

Proof of Proposition 37

- From right to left: Let LabAsm be a grounded assumption labelling according to Definition 17. By Theorem 36 $\mathit{Asms} = \textsc{in}(\mathit{LabAsm})$ is a grounded assumption extension of possibly non-flat ABA frameworks with $\mathit{Asms}^+ = \textsc{out}(\mathit{LabAsm})$ and $\mathcal{A} \setminus (\mathit{Asms} \cup \mathit{Asms}^+) = \textsc{undec}(\mathit{LabAsm})$. By Theorem 6.2 in [4], for flat ABA frameworks Asms is a minimal (w.r.t. $\subseteq$) complete assumption extension, and thus Asms is a grounded assumption extension as defined for flat ABA frameworks. By Theorem 5, LabAsm is a grounded assumption labelling according to Definition 4.

- From left to right: Let LabAsm be a grounded assumption labelling according to Definition 4. By Theorem 5 $\mathit{Asms} = \textsc{in}(\mathit{LabAsm})$ is a grounded assumption extension as defined for flat ABA frameworks with $\mathit{Asms}^+ = \textsc{out}(\mathit{LabAsm})$ and $\mathcal{A} \setminus (\mathit{Asms} \cup \mathit{Asms}^+) = \textsc{undec}(\mathit{LabAsm})$, i.e. Asms is a minimal (w.r.t. $\subseteq$) complete assumption extension. Let Asms' be the intersection of all complete assumption extensions, i.e. a grounded assumption extension of possibly non-flat ABA frameworks. Since the grounded extension of a flat ABA framework is unique, Asms is unique and so is Asms'. Thus, by Theorem 6.2 in [4] $\mathit{Asms}' = \mathit{Asms}$. Then by Theorem 36 LabAsm is a grounded assumption labelling according to Definition 17.

Proof of Theorem 39

Analogous to the proof of Theorem 5, but using the definition of admissible assumption extensions and labellings of possibly non-flat ABA frameworks instead of complete assumption extensions and labellings of flat ABA frameworks, as well as Theorem 32 instead of Theorem 4.

Proof of Theorem 41

1. By Theorem 5.5 in [4], Asms is a complete assumption extension. By Theorem 34, LabAsm is a complete assumption labelling. Furthermore, since for all $\alpha \in \mathcal{A}$ it holds that if $\alpha \notin \mathit{Asms}$ then Asms attacks α, it follows that $\mathit{Asms} \cup \mathit{Asms}^+ = \mathcal{A}$. Then $\textsc{in}(\mathit{LabAsm}) \cup \textsc{out}(\mathit{LabAsm}) = \mathcal{A}$, so $\textsc{undec}(\mathit{LabAsm}) = \emptyset$. Thus, LabAsm is a stable assumption labelling.
2. By definition LabAsm is a complete assumption labelling. By Theorem 34, Asms is a complete assumption extension. Since $\textsc{undec}(\mathit{LabAsm}) = \emptyset$ it follows that for all $\alpha \in \mathcal{A}$, $\alpha \in \textsc{in}(\mathit{LabAsm})$ of $\alpha \in \textsc{out}(\mathit{LabAsm})$. And thus $\alpha \in \mathit{Asms}$ or $\alpha \in \mathit{Asms}^+$. Thus, if $\alpha \notin \mathit{Asms}$ then Asms attacks α.

Proof of Theorem 43

Analogous to the proof of Theorem 5, but using the definition of admissible assumption extensions and labellings of possibly non-flat ABA frameworks instead of complete assumption extensions of flat ABA frameworks, as well as the definition of preferred assumption extensions and labellings of possibly non-flat ABA frameworks instead of preferred assumption extensions and labellings of flat ABA frameworks, and Theorem 32 instead of Theorem 4.

Proof of Theorem 45

Analogous to the proof of Theorem 5, but using the definition of admissible assumption extensions and labellings of possibly non-flat ABA frameworks instead of complete assumption extensions and labellings of flat ABA frameworks, and Theorem 32 instead of Theorem 4.

Proof of Proposition 46

Since $\textsc{undec}(\mathit{LabAsm})$ is minimal it follows that $\textsc{in}(\mathit{LabAsm}) \cup \textsc{out}(\mathit{LabAsm})$ is maximal among all admissible assumption labellings. Assume by contradiction that there exists an admissible assumption labelling LabAsm' such that $\textsc{in}(\mathit{LabAsm}) \subset \textsc{in}(\mathit{LabAsm}')$. Then for all $\alpha \in \mathcal{A}$ such that $\textsc{in}(\mathit{LabAsm})$ attacks α, $\textsc{in}(\mathit{LabAsm}')$ also attacks α. Thus, $\textsc{out}(\mathit{LabAsm}) \subseteq \textsc{out}(\mathit{LabAsm}')$. It follows that $\textsc{in}(\mathit{LabAsm}) \cup \textsc{out}(\mathit{LabAsm}) \subset \textsc{in}(\mathit{LabAsm}') \cup \textsc{out}(\mathit{LabAsm}')$. Contradiction.

References

[1] T. J. M. Bench-Capon, P. E. Dunne, Argumentation in Artificial Intelligence, Artificial Intelligence 171 (10-15) (2007) 619–641.

[2] I. Rahwan, G. R. Simari, Argumentation in Artificial Intelligence, Springer US, 2009.

[3] P. M. Dung, On the Acceptability of Arguments and its Fundamental Role in Nonmonotonic Reasoning, Logic Programming and n-Person Games, Artificial Intelligence 77 (2) (1995) 321–357.

[4] A. Bondarenko, P. M. Dung, R. A. Kowalski, F. Toni, An Abstract, Argumentation-Theoretic Approach to Default Reasoning, Artificial Intelligence 93 (1-2) (1997) 63–101.

[5] P. Besnard, A. Hunter, A Logic-Based Theory of Deductive Arguments, Artificial Intelligence 128 (1-2) (2001) 203–235.

[6] A. J. García, G. R. Simari, Defeasible Logic Programming: An Argumentative Approach, Theory and Practice of Logic Programming 4 (1-2) (2004) 95–138.

[7] S. Modgil, H. Prakken, A General Account of Argumentation with Preferences, Artificial Intelligence 195 (2013) 361–397.

[8] P. Besnard, A. J. García, A. Hunter, S. Modgil, H. Prakken, G. R. Simari, F. Toni, Introduction to Structured Argumentation, Argument & Computation 5 (1) (2014) 1–4.

[9] P. M. Dung, R. A. Kowalski, F. Toni, Assumption-Based Argumentation, in: G. R. Simari, I. Rahwan (Eds.), Argumentation in Artificial Intelligence, Springer US, 2009, pp. 199–218.

[10] F. Toni, A Tutorial on Assumption-Based Argumentation, Argument & Computation 5 (1) (2014) 89–117.

[11] D. Gaertner, F. Toni, Preferences and Assumption-Based Argumentation for Conflict-Free Normative Agents, in: Revised Selected and Invited Papers of the 4th International Workshop on Argumentation in Multi-Agent Systems (ArgMAS'07), 2007, pp. 94–113.

[12] M. Morge, P. Mancarella, Assumption-Based Argumentation for the Minimal Concession Strategy, in: Revised Selected and Invited Papers of the 6th International Workshop on Argumentation in Multi-Agent Systems (ArgMAS'09), 2009, pp. 114–133.

[13] A. Hussain, F. Toni, Assumption-Based Argumentation for Communicating Agents, in: Papers from the 2009 AAAI Fall Symposium on the Uses of Computational Argumentation, 2009.

[14] X. Fan, F. Toni, A. Hussain, Two-Agent Conflict Resolution with Assumption-Based Argumentation, in: Proceedings of the 3rd International Conference on Computational Models of Argument (COMMA'10), 2010, pp. 231–242.

[15] M. Morge, P. Mancarella, Arguing over Goals for Negotiation: Adopting an Assumption-Based Argumentation Decision Support System, Group Decision and Negotiation 23 (5) (2012) 979–1012.

[16] X. Fan, F. Toni, A General Framework for Sound Assumption-Based Argumentation Dialogues, Artificial Intelligence 216 (2014) 20–54.

[17] X. Fan, F. Toni, Decision Making with Assumption-Based Argumentation, in: Revised Selected Papers of the 2nd International Workshop on Theory and Applications of Formal Argumentation (TAFA' 13), 2013, pp. 127–142.

[18] X. Fan, R. Craven, R. Singer, F. Toni, M. Williams, Assumption-Based Argumentation for Decision-Making with Preferences: A Medical Case Study, in: Proceedings of the 14th International Workshop on Computational Logic in Multi-Agent Systems (CLIMA'13), 2013, pp. 374–390.

[19] X. Fan, F. Toni, A. Mocanu, M. Williams, Dialogical Two-Agent Decision Making with Assumption-Based Argumentation, in: Proceedings of the 13th International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS'14), 2014, pp. 533–540.

[20] F. Toni, Reasoning on the Web with Assumption-Based Argumentation, in: Proceedings of the 8th International Summer School on Reasoning Web, 2012, pp. 370–386.

[21] X. Fan, F. Toni, On Computing Explanations in Argumentation, in: Proceedings of the 29th AAAI Conference on Artificial Intelligence (AAAI'15), 2015, pp. 1496–1502.

[22] C. Schulz, F. Toni, Justifying Answer Sets using Argumentation, Theory and Practice of Logic Programming 16 (01) (2016) 59–110.

[23] P. M. Dung, P. Mancarella, F. Toni, Computing Ideal Sceptical Argumentation, Artificial Intelligence 171 (10-15) (2007) 642–674.

[24] M. Caminada, S. Sá, J. Alcântara, W. Dvořák, On the Difference between Assumption-Based Argumentation and Abstract Argumentation, The IfCoLog Journal of Logics and their Applications 2 (1) (2015) 16–34.

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

[26] P. Baroni, M. Caminada, M. Giacomin, An Introduction to Argumentation Semantics, The Knowledge Engineering Review 26 (04) (2011) 365–410.

[27] C. Schulz, F. Toni, Complete Assumption Labellings, in: Proceedings of the 5th International Conference on Computational Models of Argument (COMMA'14), 2014, pp. 405–412.

[28] C. Schulz, F. Toni, Logic Programming in Assumption-Based Argumentation Revisited - Semantics and Graphical Representation, in: Proceedings of the 29th AAAI Conference on Artificial Intelligence (AAAI'15), 2015, pp. 1569–1575.

[29] M. Caminada, C. Schulz, On the Equivalence between Assumption-Based Argumentation and Logic Programming, in: Proceedings of the 1st International Workshop on Argumentation and Logic Programming (ArgLP'15), 2015.

[30] M. Caminada, Semi-Stable Semantics, in: Proceedings of the 1st International Conference on Computational Models of Argument (COMMA'06), 2006, pp. 121–130.

[31] M. Caminada, On the Issue of Reinstatement in Argumentation, in: Proceedings of the 10th European Conference on Logics in Artificial Intelligence (JELIA'06), 2006, pp. 111–123.

[32] M. Caminada, A Labelling Approach for Ideal and Stage Semantics, Argument & Computation 2 (1) (2011) 1–21.

[33] P. M. Dung, R. A. Kowalski, F. Toni, Dialectic Proof Procedures for Assumption-Based, Admissible Argumentation, Artificial Intelligence 170 (2) (2006) 114–159.

[34] F. Toni, A Generalised Framework for Dispute Derivations in Assumption-Based Argumentation, Artificial Intelligence 195 (2013) 1–43.

[35] J. Heyninck, C. Straßer, Relations between Assumption-Based Approaches in Nonmonotonic Logic and Formal Argumentation, in: Proceedings of the 16th International Workshop on Non-Monotonic Reasoning (NMR'16), 2016.

[36] R. Craven, F. Toni, Argument Graphs and Assumption-Based Argumentation, Artificial Intelligence 233 (2016) 1–59.

[37] P. Baroni, M. Giacomin, On Principle-Based Evaluation of Extension-Based Argumentation Semantics, Artificial Intelligence 171 (10-15) (2007) 675–700.

[38] M. Caminada, Strong Admissibility Revisited, in: Proceedings of the 5th International Conference on Computational Models of Argument (COMMA'14), 2014, pp. 197–208.

[39] R. C. Moore, Semantical Considerations on Nonmonotonic Logic, Artificial Intelligence 25 (1) (1985) 75 – 94.

[40] M. Caminada, S. Sá, J. Alcântara, W. Dvoák, On the Equivalence between Logic Programming Semantics and Argumentation Semantics, International Journal of Approximate Reasoning 58 (2015) 87–111.

[41] T. C. Przymusinski, Stable Semantics for Disjunctive Programs, New Generation Computing 9 (3/4) (1991) 401–424.

[42] A. V. Gelder, K. A. Ross, J. S. Schlipf, The Well-Founded Semantics for General Logic Programs, Journal of the ACM 38 (3) (1991) 620–650.

[43] J.-H. You, L. Y. Yuan, Three-Valued Formalization of Logic Programming: Is It Needed?, in: Proceedings of the 9th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS'90), 1990, pp. 172–182.

[44] C. Schulz, K. Satoh, F. Toni, Characterising and Explaining Inconsistency in Logic Programs, in: Proceedings of the 13th International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR'15), 2015, pp. 467–479.

[45] Y. Dimopoulos, A. Torres, Graph Theoretical Structures in Logic Programs and Default Theories, Theoretical Computer Science 170 (1-2) (1996) 209–244.

[46] S. Costantini, Comparing Different Graph Representations of Logic Programs under the Answer Set Semantics, in: Proceedings of the 1st International Workshop on Answer Set Programming: Towards Efficient and Scalable Knowledge Representation and Reasoning (ASP'01), 2001.

[47] T. Linke, V. Sarsakov, Suitable Graphs for Answer Set Programming, in: Proceedings of the 11th International Conference on Logic for Programming, Artificial Intelligence, and Reasoning (LPAR'04), 2004, pp. 154–168.

[48] K. Konczak, T. Linke, T. Schaub, Graphs and Colorings for Answer Set Programming, Theory and Practice of Logic Programming 6 (1-2) (2006) 61–106.

[49] T. Wakaki, Assumption-Based Argumentation Equipped with Preferences, in: Proceedings of the 17th International Conference on Principles and Practice of Multi-Agent Systems (PRIMA'14), 2014, pp. 116–132.

[50] K. Cyras, F. Toni, ABA+: Assumption-Based Argumentation with Preferences, in: Proceedings of the15th International Conference on Principles of Knowledge Representation and Reasoning (KR'16), 2016, pp. 553–556.

[51] S. Modgil, M. Caminada, Proof Theories and Algorithms for Abstract Argumentation Frameworks, in: G. Simari, I. Rahwan (Eds.), Argumentation in Artificial Intelligence, Springer US, 2009, pp. 105–129.

[52] T. Wakaki, K. Nitta, Computing Argumentation Semantics in Answer Set Programming, in: Revised Selected Papers of the 22st Anual Conference of the Japanese Society for Artificial Intelligence (JSAI'08), 2008, pp. 254–269.

[53] G. Charwat, W. Dvořák, S. A. Gaggl, J. P. Wallner, S. Woltran, Methods for Solving Reasoning Problems in Abstract Argumentation - A Survey, Artificial intelligence 220 (2015) 28–63.