Fundamenta Informaticae XX (2015) 1–15
DOI 10.3233/FI-2012-0000
IOS Press

On the Equivalence between Assumption-Based Argumentation and Logic Programming

Martin Caminada*
Department of Computing Science
University of Aberdeen, UK
martin.caminada@abdn.ac.uk

Claudia Schulz
Department of Computing
Imperial College London, UK
claudia.schulz@imperial.ac.uk

Abstract. In the current paper, we re-examine the relationship between Assumption-Based Argumentation (ABA) and logic programming (LP). For this, we specify a procedure that, given a flat ABA frameworks with unique non-assumption contraries, yields an associated logic program such that the 3-valued stable (resp. well-founded, regular, (2-valued) stable, and ideal) models of the logic program coincide with the complete (resp. grounded, preferred, stable, and ideal) assumption labellings of the ABA framework. Moreover, we show how our results on the translation from ABA to LP can be reapplied for a reverse translation from LP to ABA, and observe that some of the existing results in the literature are in fact special cases of our work. Overall, we show that a frequently used fragment of ABA (flat ABA frameworks with unique non-assumption contraries under complete, grounded, preferred, stable, or ideal semantics) can be seen as a form of logic programming.

1. Introduction

Assumption-Based Argumentation (ABA) [2, 7, 23] has become one of the leading approaches for formal argumentation. On the one hand, it is an instance of Abstract Argumentation under many well-studied semantics [8, 23], on the other it generalizes logic programming, default logic and other non-monotonic reasoning systems [2, 20] and has as such proven useful for explaining [19] as well as visualizing [17] logic programs under certain semantics. In addition to the "normal" notion of acceptability used in many argumentation formalism, ABA is equipped with a dialectical notion of acceptability [22] which has for instance been applied in agent dialogues [11]. ABA has also proven useful in various application domains ranging from medicine [6] over decision making [13] and negotiation [15] to legal reasoning [9].

*Supported by the Engineering and Physical Sciences Research Council (EPSRC, UK), grant ref. EP/J012084/1 (SAsSy project).

Although ABA was originally specified in a very general way [2], some of the more recent work (like [7, 23, 20]) has focused on flat ABA frameworks (meaning that no assumption can occur in the head of a rule) with a unique contrary for every assumption. Here, we will study a fragment of this type of ABA framework, to be referred to as *normal ABA framework*, where in addition the contraries of assumptions are non-assumptions. We will focus on some of the most commonly studied ABA semantics (complete, grounded, preferred, stable, and ideal) which we will refer to as *common ABA semantics*.

One particular question that has been studied in the literature is how ABA relates to logic programming (LP). Usually, this is done in the form of a translation from LP to ABA [2, 20, 19] and showing that ABA is powerful enough to capture LP. In the current paper, we go the other way around. That is, we provide a translation from ABA to LP and show that LP is powerful enough to capture a frequently studied fragment of ABA, i.e. normal ABA frameworks under common ABA semantics.

2. Formal Preliminaries

In the current section, we provide a number of key definitions on Assumption-Based Argumentation (ABA) and Logic Programming (LP).

Definition 2.1. An *Assumption-Based Argumentation (ABA) Framework* is a tuple $\langle \mathcal{L}, \mathcal{R}, \mathcal{A}, ^- \rangle$ where $\mathcal{L}$ is a set of atoms[1], $\mathcal{R}$ is a set of inference rules based on this language, $\mathcal{A} \subseteq \mathcal{L}$ is a set of assumptions, and $^- : \mathcal{A} \to \mathcal{L}$ is a function that maps each assumption $\chi \in \mathcal{A}$ to what is called its *contrary* $\overline{\chi}$.

We say that an ABA framework is *flat* [2] iff assumptions only occur in the body of the inference rules, and not in the head. Furthermore, we notice that each assumption has a unique contrary. Although this deviates from some generalized work on ABA, where an assumption has a set of possible contraries [14, 11, 12], or where a set of sentences (containing at least one assumption) is associated with a set of sentences which together form the contrary [21], in a lot of work on ABA [8, 7, 23, 18] it is common for the authors to restrict themselves to assumptions with unique contraries as originally defined [2]. In addition, we will here often restrict ourselves to a fragment of ABA where the contrary of an assumption cannot be an assumption, but only a non-assumption, i.e. where $^- : \mathcal{A} \to \mathcal{L} \setminus \mathcal{A}$. In [5] it is shown that this does not affect the expressiveness of ABA. In the current paper, we will use the term *normal ABA frameworks* for flat ABA frameworks where assumptions have unique contraries which are non-assumptions.

Definition 2.2. Let $\mathcal{F} = \langle \mathcal{L}, \mathcal{R}, \mathcal{A}, ^- \rangle$ be an ABA framework. An *ABA argument* $Asms \vdash x$ for conclusion $x \in \mathcal{L}$ supported by assumptions $Asms \subseteq \mathcal{A}$ is a finite tree with nodes labelled with sentences in $\mathcal{L}$ or with the special symbol `TRUE`[2], such that:

- the root is labelled with x
- for every node N:

[1] This is to ensure that no formula contains strong negation or disjunction, which do not have a semantic meaning in ABA but do in a logic program, which could cause problems when translating an ABA framework to a logic program.

[2] We assume that the special symbol `TRUE`, just like the special symbols `FALSE` and `UNDEFINED` do not occur in any ABA framework. So `TRUE`, `FALSE`, `UNDEFINED` $\notin \mathcal{L}$.

- if N is a leaf node, then N is labelled with an assumption or with TRUE
- if N is not a leaf node and $z \in \mathcal{L}$ is the label of N, then there exists a rule in $\mathcal{R}$ of the form $z \leftarrow y_1, \ldots, y_n$ and either $n = 0$ and N has just a single child that is labelled with TRUE, or $n > 0$ and N has n children, labelled with $y_1, \ldots, y_n$ respectively
- $Asms$ is the set of all assumptions labelling leaf nodes

Based on the definition of an ABA argument, we proceed to introduce ABA semantics. For this, we apply the notion of assumption labellings [18].

Definition 2.3. Let $\mathcal{F} = \langle \mathcal{L}, \mathcal{R}, \mathcal{A}, ^- \rangle$ be an ABA framework. An *assumption labelling* of $\mathcal{F}$ is a total function $\mathcal{L}ab : \mathcal{A} \rightarrow \{\text{IN}, \text{OUT}, \text{UNDEC}\}$. We denote by $\text{IN}(\mathcal{L}ab)$ the set of all assumptions labelled IN by $\mathcal{L}ab$, and similarly by $\text{OUT}(\mathcal{L}ab)$ and $\text{UNDEC}(\mathcal{L}ab)$ the sets of assumptions labelled OUT and UNDEC, respectively. An assumption labelling $\mathcal{L}ab$ is called a *complete assumption labelling* of $\mathcal{F}$ iff for each $\chi \in \mathcal{A}$ it holds that:

1. if $\mathcal{L}ab(\chi) = \text{IN}$ then for each argument $Asms \vdash \overline{\chi}$ it holds that $Asms \cap \text{OUT}(\mathcal{L}ab) \neq \emptyset$
2. if $\mathcal{L}ab(\chi) = \text{OUT}$ then there exists an argument $Asms \vdash \overline{\chi}$ such that $Asms \subseteq \text{IN}(\mathcal{L}ab)$
3. if $\mathcal{L}ab(\chi) = \text{UNDEC}$ then there exists an argument $Asms \vdash \overline{\chi}$ such that $Asms \cap \text{OUT}(\mathcal{L}ab) = \emptyset$, and for each argument $Asms \vdash \overline{\chi}$ it holds that $Asms \not\subseteq \text{IN}(\mathcal{L}ab)$

We now introduce some new results and definitions which will be needed for the comparison of ABA and LP semantics. Firstly, we observe that the set of IN assumptions of a complete assumption labelling $\mathcal{L}ab_1$ is a subset of or equal to the set of IN assumptions of another complete assumption labelling $\mathcal{L}ab_2$ if and only if the set of OUT assumptions of $\mathcal{L}ab_1$ is a subset of or equal to the set of OUT assumptions of $\mathcal{L}ab_2$.

Lemma 2.4. Let $\mathcal{F} = \langle \mathcal{L}, \mathcal{R}, \mathcal{A}, ^- \rangle$ be an ABA framework, and let $\mathcal{L}ab_1$ and $\mathcal{L}ab_2$ be complete assumption labellings of $\mathcal{F}$. It holds that $\text{IN}(\mathcal{L}ab_1) \subseteq \text{IN}(\mathcal{L}ab_2)$ iff $\text{OUT}(\mathcal{L}ab_1) \subseteq \text{OUT}(\mathcal{L}ab_2)$.

Proof:
"$\Rightarrow$": Assume that $\text{IN}(\mathcal{L}ab_1) \subseteq \text{IN}(\mathcal{L}ab_2)$. Let $\chi \in \text{OUT}(\mathcal{L}ab_1)$. Then, by the definition of a complete assumption labelling (Definition 2.3) there exists an ABA argument $Asms \vdash \overline{\chi}$ with $Asms \subseteq \text{IN}(\mathcal{L}ab_1)$. Since $\text{IN}(\mathcal{L}ab_1) \subseteq \text{IN}(\mathcal{L}ab_2)$ it follows that $Asms \subseteq \text{IN}(\mathcal{L}ab_2)$. So by Lemma 1 in [18], $\chi \in \text{OUT}(\mathcal{L}ab_2)$.
"$\Leftarrow$": Assume that $\text{OUT}(\mathcal{L}ab_1) \subseteq \text{OUT}(\mathcal{L}ab_2)$. Let $\chi \in \text{IN}(\mathcal{L}ab_1)$. Then, by the definition of a complete assumption labelling (Definition 2.3) it holds that each ABA argument $Asms \vdash \overline{\chi}$ has $Asms \cap \text{OUT}(\mathcal{L}ab_1) \neq \emptyset$. Since $\text{OUT}(\mathcal{L}ab_1) \subseteq \text{OUT}(\mathcal{L}ab_2)$ it follows that $Asms \cap \text{OUT}(\mathcal{L}ab_2) \neq \emptyset$. So by Lemma 1 in [18], $\chi \in \text{IN}(\mathcal{L}ab_2)$. □

We now extend the notion of assumption labellings from the complete semantics as introduced in [18] to other well-known ABA semantics, which were previously defined in terms of extensions rather than labellings [2, 23]. Note that there exists a one-to-one correspondence between the assumption labellings and the assumption extensions of an ABA framework. In essence, the set of IN-labelled assumptions of a complete (resp. grounded, preferred, stable, or ideal) assumption labelling constitutes a complete (resp. grounded, preferred, stable, or (maximal) ideal) assumption extension.

Definition 2.5. Let $\mathcal{F} = \langle \mathcal{L}, \mathcal{R}, \mathcal{A}, ^- \rangle$ be an ABA framework. A complete assumption labelling $\mathcal{L}ab$ of $\mathcal{F}$ is called:

1. a *grounded assumption labelling* iff $\text{IN}(\mathcal{L}ab)$ is minimal (w.r.t. $\subseteq$) among all complete assumption labellings of $\mathcal{F}$
2. a *preferred assumption labelling* iff $\text{IN}(\mathcal{L}ab)$ is maximal (w.r.t. $\subseteq$) among all complete assumption labellings of $\mathcal{F}$
3. a *stable assumption labelling* iff $\text{UNDEC}(\mathcal{L}ab) = \emptyset$
4. an *ideal assumption labelling* iff $\text{IN}(\mathcal{L}ab)$ is maximal (w.r.t. $\subseteq$) among all complete assumption labellings of $\mathcal{F}$ with $\text{IN}(\mathcal{L}ab) \subseteq \bigcap\{\text{IN}(\mathcal{L}ab_{pref}) \mid \mathcal{L}ab_{pref}$ is a preferred assumption labelling of $\mathcal{F}\}$

As complete, grounded, preferred, stable, and ideal semantics are well-studied within ABA, we refer to these as the *common ABA semantics*.

Now that we have introduced the preliminaries of Assumption-Based Argumentation, we shift our attention to logic programming. We start with formally introducing the notion of a logic program.

Definition 2.6. A *logic programming rule* is an expression $x \leftarrow y_1, \ldots, y_n, \texttt{not}\ z_1, \ldots, \texttt{not}\ z_m$ ($n \geq 0$, $m \geq 0$) where x, each y_i ($1 \leq i \leq n$) and each z_j ($1 \leq j \leq m$) is an atom, and `not` represents negation as failure. We say that x is the *head* of the rule, and $y_1, \ldots, y_n, \texttt{not}\ z_1, \ldots, \texttt{not}\ z_m$ the body of the rule. Moreover, we say that $y_1, \ldots, y_n$ is the strong part of the body, and $\texttt{not}\ z_1, \ldots, \texttt{not}\ z_m$ is the weak part of the body. We assume the presence of three special atoms `TRUE`, `FALSE` and `UNDEFINED` that can only occur in the strong part of the body. A *NAF-literal* is an expression $\texttt{not}\ w$, where w is an atom. We say a rule is NAF-free iff it does not contain any NAF-literal (that is, iff $m = 0$). A *logic program* P is a finite set of logic programming rules. A logic program is NAF-free iff each of its rules is NAF-free. A logic program is called *normal*[3] iff none of its rules contains the special atoms `TRUE`, `FALSE` or `UNDEFINED`. The Herbrand Base of a logic program P (written as HB_P) is the set of all atoms in P.

In the following, we recall the definitions of LP semantics.

Definition 2.7. A *3-valued interpretation* of a logic program P is a pair $\langle T, F \rangle$ where $T, F \subseteq HB_P$ and $T \cap F = \emptyset$.

When P is a NAF-free logic program (possibly containing `TRUE`, `FALSE` or `UNDEFINED`), we write $\Phi(P)$ for its unique minimal 3-valued model $\langle T_\Phi, F_\Phi \rangle$ in the sense of [16] (with minimal T_Φ and maximal F_Φ). We proceed to define the well-known Gelfond-Lifschitz reduct in the context of a 3-valued interpretation as done in [16].

Definition 2.8. The reduct of a logic program P w.r.t. a 3-valued interpretation $\mathcal{M}od = \langle T, F \rangle$, written as $P^{\mathcal{M}od}$ is obtained by replacing each NAF literal $\texttt{not}\ x$ by `TRUE` if $x \in F$, by `FALSE` if $x \in T$, and by `UNDEFINED` otherwise.

[3] In general the term "normal" is used for logic programs without strong negation, which is the case for the logic programs considered here. The notion of "normal logic program" used here is thus a special case of its general usage.

Since $P^{\mathcal{M}od}$ is a NAF-free program, it has a unique minimal 3-valued model, written as $\Phi(P^{\mathcal{M}od})$. We now recall different logic programming semantics which are based on 3-valued models [16]. Notice that although our definition of well-founded and regular models is slightly different from what is in the literature, equivalence is shown in [4]. We also define a new semantics based on 3-valued models, namely *ideal models*, inspired by the idea of ideal scenarios for logic programs [1]. In fact our ideal models coincide with ideal scenarios [5].

Definition 2.9. Let P be a logic program and $\mathcal{M}od = \langle T, F\rangle$ a 3-valued interpretation of P. We say that $\mathcal{M}od$ is:

- a *3-valued stable model* iff $\Phi(P^{\mathcal{M}od}) = \mathcal{M}od$
- a *well-founded model* iff $\mathcal{M}od$ is a 3-valued stable model where T is minimal (w.r.t. $\subseteq$) among all 3-valued stable models of P
- a *regular model* [25] iff $\mathcal{M}od$ is a 3-valued stable model where T is maximal (w.r.t. $\subseteq$) among all 3-valued stable models of P
- a *(2-valued) stable model* iff $\mathcal{M}od$ is a 3-valued stable model where $T \cup F = HB_P$
- an *ideal model* iff $\mathcal{M}od$ is a 3-valued stable model where T is maximal (w.r.t. $\subseteq$) among all 3-valued stable models of P with $T \subseteq \bigcap\{T_{reg} \mid \langle T_{reg}, F_{reg}\rangle$ is a regular model of $P\}$

We sometimes refer to 3-valued stable, well-founded, regular, (2-valued) stable, and ideal semantics as the *common LP semantics*.

3. Translating ABA Theories to Logic Programs

In order to compare ABA to logic programming, we first introduce a translation from a normal ABA framework to a logic program. The idea is to take the rules of the ABA framework and substitute each assumption by the NAF literal of its contrary. This means that different assumptions might be substituted by the same NAF literal if they have the same contrary.

Definition 3.1. Let $\mathcal{F} = \langle \mathcal{L}, \mathcal{R}, \mathcal{A}, ^- \rangle$ be a normal ABA framework. We define the *associated logic program* $P_{\mathcal{F}}$ as $\{x \leftarrow y_1, \ldots, y_n, \texttt{not } z_1, \ldots, \texttt{not } z_m \mid x \leftarrow y_1, \ldots, y_n, \zeta_1, \ldots, \zeta_m \in \mathcal{R}$ and $\forall i \in \{1 \ldots m\} : \overline{\zeta_i} = z_i\}$, where $HB_{P_{\mathcal{F}}} = \{w \mid w$ or $\texttt{not } w$ occurs in a rule of $P_{\mathcal{F}}\}$.

As we assume that no ABA framework contains the special symbols `TRUE`, `FALSE` or `UNDEFINED`, the associated logic program will not contain any of these symbols, so $P_{\mathcal{F}}$ is a normal logic program. Note also that since $\mathcal{F}$ is a normal ABA framework, i.e. the contrary of assumptions are non-assumptions, $HB_{P_{\mathcal{F}}}$ contains no atoms which are assumptions in $\mathcal{F}$.

Example 3.2. Let $\mathcal{F} = \langle \mathcal{L}, \mathcal{R}, \mathcal{A}, ^- \rangle$ be an ABA framework with $\mathcal{A} = \{\alpha, \beta, \gamma, \delta\}$, $\mathcal{L} = \mathcal{A} \cup \{a, b, c, d, e\}$, $\overline{\alpha} = a$, $\overline{\beta} = b$, $\overline{\gamma} = e$, $\overline{\delta} = a$, and $\mathcal{R} = \{a \leftarrow \beta;\ b \leftarrow \alpha;\ b \leftarrow \delta;\ c \leftarrow;\ d \leftarrow b, c\}$.
The associated logic program $P_{\mathcal{F}}$ is: $\{a \leftarrow \texttt{not } b;\ b \leftarrow \texttt{not } a;\ c \leftarrow;\ d \leftarrow b, c\}$, which comprises one rule less than the ABA framework, with $HB_{P_{\mathcal{F}}} = \{a, b, c, d\}$.

One of the main aims of the current paper is to examine how ABA semantics are related to logic programming semantics. For this, we introduce the functions `Lab2Mod` and `Mod2Lab` to convert between ABA assumption labellings and logic programming models.

To convert an assumption labelling to a 3-valued interpretation, we start by "inverting" the labelling. That is, we construct an interpretation $\langle T', F' \rangle$ where T' contains the contraries of the assumptions that are OUT, whereas F' contains the contraries of the assumptions that are IN. However, since we started with assumptions, this will only yield the status of atoms which are contraries of assumptions. In order to obtain the status of *all* atoms in the logic program (including those that are not the contrary of any assumption in the ABA framework) we perform a simple trick: apply the Gelfond-Lifschitz reduct.

To convert a 3-valued interpretation to an assumption labelling, the idea is again to "invert" the interpretation. The assumptions whose contrary is in F will be labelled IN. The assumptions whose contrary is in T will be labelled OUT. The assumptions whose contrary is in the Herbrand Base, but not in T or F will be labelled UNDEC. The only remaining case is what to do with the assumptions whose contrary is not even in the Herbrand Base. This case occurs if there exists an assumption in the ABA framework which is not part of any inference rule itself and nor is its contrary. As these assumptions cannot have any attackers, they will simply be labelled IN.

Definition 3.3. Let $\mathcal{F} = \langle \mathcal{L}, \mathcal{R}, \mathcal{A}, ^- \rangle$ be a normal ABA framework and let $P_\mathcal{F}$ be the associated logic program. We define a function `Lab2Mod` that, given an assumption labelling $\mathcal{L}ab$ of $\mathcal{F}$, yields the 3-valued interpretation $\Phi(P_\mathcal{F}^{\langle T', F' \rangle})$ where $T' = \{\overline{\chi} \mid \chi \in \text{OUT}(\mathcal{L}ab)\} \cap HB_{P_\mathcal{F}}$ and $F' = \{\overline{\chi} \mid \chi \in \text{IN}(\mathcal{L}ab)\} \cap HB_{P_\mathcal{F}}$. We also define a function `Mod2Lab` that, given a 3-valued interpretation $\langle T, F \rangle$ of $P_\mathcal{F}$, yields an assumption labelling $\mathcal{L}ab$ of $\mathcal{F}$ with $\text{IN}(\mathcal{L}ab) = \{\chi \in \mathcal{A} \mid \overline{\chi} \in F\} \cup \{\chi \in \mathcal{A} \mid \overline{\chi} \notin HB_{P_\mathcal{F}}\}$, $\text{OUT}(\mathcal{L}ab) = \{\chi \in \mathcal{A} \mid \overline{\chi} \in T\}$ and $\text{UNDEC}(\mathcal{L}ab) = \{\chi \in \mathcal{A} \mid \overline{\chi} \in HB_{P_\mathcal{F}} \setminus (T \cup F)\}$

We observe that the functions `Lab2Mod` and `Mod2Lab` provide a one-to-one mapping between the complete assumption labellings of $\mathcal{F}$ and the 3-valued stable models of $P_\mathcal{F}$.

Theorem 3.4. Let $\mathcal{F} = \langle \mathcal{L}, \mathcal{R}, \mathcal{A}, ^- \rangle$ be a normal ABA framework and let $P_\mathcal{F}$ be the associated logic program. It holds that

1. if $\mathcal{L}ab$ is a complete assumption labelling of $\mathcal{F}$ then `Lab2Mod`$(\mathcal{L}ab)$ is a 3-valued stable model of $P_\mathcal{F}$
2. if $\mathcal{M}od$ is a 3-valued stable model of $P_\mathcal{F}$ then `Mod2Lab`$(\mathcal{M}od)$ is a complete assumption labelling of $\mathcal{F}$
3. when restricted to complete assumption labellings and 3-valued stable models, `Lab2Mod` and `Mod2Lab` become bijections which are each other's inverses

Proof:

1. Let $\mathcal{L}ab$ be a complete assumption labelling of $\mathcal{F}$ and let $\langle T, F \rangle$ be `Lab2Mod`$(\mathcal{L}ab)$. That is, $\langle T, F \rangle = \Phi(P_\mathcal{F}^{\langle T', F' \rangle})$ where $T' = \{\overline{\chi} \mid \chi \in \text{OUT}(\mathcal{L}ab)\} \cap HB_{P_\mathcal{F}}$ and $F' = \{\overline{\chi} \mid \chi \in \text{IN}(\mathcal{L}ab)\} \cap HB_{P_\mathcal{F}}$. We first observe that $\langle T', F' \rangle$ is a well-defined 3-valued interpretation of $P_\mathcal{F}$. This is because $T', F' \subseteq HB_{P_\mathcal{F}}$ and $T' \cap F' = \emptyset$, the latter following from the facts that $\text{IN}(\mathcal{L}ab) \cap$

$\text{OUT}(\mathcal{L}ab) = \emptyset$ and that two assumptions that have the same contrary also have the same label in $\mathcal{L}ab$. From the fact that $\langle T', F' \rangle$ is a well-defined 3-valued interpretation of $P_{\mathcal{F}}$, it follows that also $\langle T, F \rangle = \Phi(P_{\mathcal{F}}^{\langle T', F' \rangle})$ is a well-defined 3-valued interpretation of $P_{\mathcal{F}}$.
We also observe that applying the Gelfond-Lifschitz reduct (with $\langle T', F' \rangle$) does not change the status of the NAF-literals (see [5] for details). That is, for every NAF-literal `not` x occurring in some rule of $P_{\mathcal{F}}$: if $x \in T'$ then $x \in T$, if $x \in F'$ then $x \in F$, and if $x \in HB_{P_{\mathcal{F}}} \setminus (T' \cup F')$ then $x \in HB_{P_{\mathcal{F}}} \setminus (T \cup F)$.
So $\langle T', F' \rangle$ and $\langle T, F \rangle$ agree on the NAF-literals of $P_{\mathcal{F}}$. It should be noted that whenever two 3-valued interpretations $\mathcal{M}od_1$ and $\mathcal{M}od_2$ of some logic program P agree on the NAF-literals of P, the respective reducts $P^{\mathcal{M}od_1}$ and $P^{\mathcal{M}od_2}$ are equal (after all, for determining the Gelfond-Lifschitz reduct, only the NAF literals are relevant). Hence, in our particular case, we have that $P_{\mathcal{F}}^{\langle T', F' \rangle} = P_{\mathcal{F}}^{\langle T, F \rangle}$, so also $\Phi(P_{\mathcal{F}}^{\langle T', F' \rangle}) = \Phi(P_{\mathcal{F}}^{\langle T, F \rangle})$. From $\Phi(P_{\mathcal{F}}^{\langle T', F' \rangle}) = \langle T, F \rangle$ it then directly follows that $\langle T, F \rangle = \Phi(P_{\mathcal{F}}^{\langle T, F \rangle})$, so $\langle T, F \rangle$ is a 3-valued stable model of $P_{\mathcal{F}}$. As by definition $\langle T, F \rangle = \Phi(P_{\mathcal{F}}^{\langle T', F' \rangle})$ it directly follows that $\Phi(P_{\mathcal{F}}^{\langle T', F' \rangle})$ is a 3-valued stable model of $P_{\mathcal{F}}$.

2. Let $\mathcal{M}od$ be a 3-valued stable model of $P_{\mathcal{F}}$, $\mathcal{L}ab = \texttt{Mod2Lab}(\mathcal{M}od)$, and $\chi \in \mathcal{A}$. It can be shown [5] that:
if $\chi \in \text{IN}(Lab)$ then for each ABA argument $Asms \vdash \overline{\chi}$ it holds that $Asms \cap \text{OUT}(\mathcal{L}ab)$,
if $\chi \in \text{OUT}(\mathcal{L}ab)$ then there exists an ABA argument $Asms \vdash \overline{\chi}$ with $Asms \subseteq \text{IN}(\mathcal{L}ab)$, and
if $\chi \in \text{UNDEC}(\mathcal{L}ab)$ then there is no ABA argument $Asms \vdash \overline{\chi}$ with $Asms \subseteq \text{IN}(\mathcal{L}ab)$ and there is an ABA argument $Asms \vdash \overline{\chi}$ with $Asms \cap \text{OUT}(\mathcal{L}ab) = \emptyset$.
This means the conditions of Definition 2.3 are satisfied, so $\mathcal{L}ab$ is a complete assumption labelling.

3. It suffices to prove that if $\mathcal{L}ab$ is a complete assumption labelling then $\texttt{Mod2Lab}(\texttt{Lab2Mod}(\mathcal{L}ab)) = \mathcal{L}ab$, and if $\mathcal{M}od$ is is a 3-valued stable model then $\texttt{Lab2Mod}(\texttt{Mod2Lab}(\mathcal{M}od)) = \mathcal{M}od$.
As for the first equivalence that has to be proved, let $\mathcal{L}ab$ be a complete assumption labelling of $\mathcal{F}$, and let $\chi \in \mathcal{A}$. We distinguish four cases.

   (a) $\chi \in \text{IN}(\mathcal{L}ab)$ and $\overline{\chi} \in HB_{P_{\mathcal{F}}}$. Then, by definition of F' (w.r.t. $\texttt{Lab2Mod}(\mathcal{L}ab)$) it follows that $\overline{\chi} \in F'$. As we have observed earlier (in point 1 of the current theorem) it holds that if the contrary of a particular assumption is in F', then it is also in F. Hence, $\overline{\chi} \in F$, with $\langle T, F \rangle = \texttt{Lab2Mod}(\mathcal{L}ab)$. From the definition of `Mod2Lab` it then follows that χ is labelled IN by $\texttt{Mod2Lab}(\texttt{Lab2Mod}(\mathcal{L}ab))$.

   (b) $\chi \in \text{IN}(\mathcal{L}ab)$ and $\overline{\chi} \notin HB_{P_{\mathcal{F}}}$. We first observe that, $\overline{\chi}$ does not occur in T or F of $P_{\mathcal{F}}$. From the definition of `Mod2Lab` it then follows that χ is labelled IN by $\texttt{Mod2Lab}(\texttt{Lab2Mod}(\mathcal{L}ab))$.

   (c) $\chi \in \text{OUT}(\mathcal{L}ab)$. From point 2 or Definition 2.3 it follows that there exists an ABA argument for $\overline{\chi}$ under $\mathcal{F}$. This means that $\overline{\chi}$ is the head of a rule, so $\overline{\chi} \in HB_{P_{\mathcal{F}}}$. It then follows that $\overline{\chi} \in T'$. As we have observed earlier (in point 1 of the current theorem) it holds that if the contrary of a particular assumption is in T' then it is also in T. Hence, $\overline{\chi} \in T$, with $\langle T, F \rangle = \texttt{Lab2Mod}(\mathcal{L}ab)$. From the definition of `Mod2Lab` it then follows that χ is labelled OUT by $\texttt{Mod2Lab}(\texttt{Lab2Mod}(\mathcal{L}ab))$.

   (d) $\chi \in \text{UNDEC}(\mathcal{L}ab)$. From point 3 of Definition 2.3 it follows that there exists an ABA argument for $\overline{\chi}$ under $\mathcal{F}$. This means that $\overline{\chi}$ is the head of a rule, so $\overline{\chi} \in HB_{P_{\mathcal{F}}}$. Fur-

thermore, from the definition of T' and F' it follows that $\overline{\chi} \notin T'$ and $\overline{\chi} \notin F'$. Hence, $\overline{\chi} \in HB_{P_\mathcal{F}} \setminus (T' \cup F')$. As we have observed earlier (in point 1 of the current theorem) it holds that if the contrary of a particular assumption is in $HB_{P_\mathcal{F}} \setminus (T' \cup F')$ then it is also in $HB_{P_\mathcal{F}} \setminus (T \cup F)$. Hence, $\overline{\chi} \in HB_{P_\mathcal{F}} \setminus (T \cup F)$, with $\langle T, F \rangle = \texttt{Lab2Mod}(\mathcal{L}ab)$. From the definition of Mod2Lab it then follows that χ is labelled UNDEC by $\texttt{Mod2Lab}(\texttt{Lab2Mod}(\mathcal{L}ab))$.

So overall, we observe that if χ is labelled IN (respectively OUT or UNDEC) by $\mathcal{L}ab$ then χ is labelled IN (respectively OUT or UNDEC) by $\texttt{Mod2Lab}(\texttt{Lab2Mod}(\mathcal{L}ab))$, so $\texttt{Mod2Lab}(\texttt{Lab2Mod}(\mathcal{L}ab)) \supseteq \mathcal{L}ab$. Furthermore, $\texttt{Mod2Lab}(\texttt{Lab2Mod}(\mathcal{L}ab))$ does not assign any additional labels other than the ones assigned by $\mathcal{L}ab$: It can easily be verified that $\mathcal{L}ab$ and $\texttt{Mod2Lab}(\texttt{Lab2Mod}(\mathcal{L}ab))$ label the same set of assumptions $\mathcal{A}$. Then, since $\text{IN}(\mathcal{L}ab) \cup \text{OUT}(\mathcal{L}ab) \cup \text{UNDEC}(\mathcal{L}ab) = \mathcal{A}$, it follows that $\texttt{Mod2Lab}(\texttt{Lab2Mod}(\mathcal{L}ab)) = \mathcal{L}ab$.
Due to space limitations, we have to refer to [5] for the second equivalence.

□

Example 3.5. Consider again $\mathcal{F}$ and $P_\mathcal{F}$ from Example 3.2. $\mathcal{F}$ has three complete assumption labellings: $\mathcal{L}ab_1 = \{(\alpha, \text{IN}), (\beta, \text{OUT}), (\gamma, \text{IN}), (\delta, \text{IN})\}$, $\mathcal{L}ab_2 = \{(\alpha, \text{OUT}), (\beta, \text{IN}), (\gamma, \text{IN}), (\delta, \text{OUT})\}$, and $\mathcal{L}ab_3 = \{(\alpha, \text{UNDEC}), (\beta, \text{UNDEC}), (\gamma, \text{IN}), (\delta, \text{UNDEC})\}$. $P_\mathcal{F}$ has three 3-valued stable models: $\mathcal{M}od_1 = \langle \{c, b, d\}, \{a\} \rangle$, $\mathcal{M}od_2 = \langle \{c, a\}, \{b, d\} \rangle$, and $\mathcal{M}od_3 = \langle \{c\}, \{\} \rangle$. It is easy to verify the correspondences between complete assumption labellings and 3-valued stable models, e.g. $\mathcal{M}od_1 = \texttt{Lab2Mod}(\mathcal{L}ab_1)$ and $\mathcal{L}ab_1 = \texttt{Mod2Lab}(\mathcal{M}od_1)$.

Theorem 3.4 is important, since in ABA complete semantics is the basis of various other semantics (like grounded, preferred, ideal, and stable), just like in LP 3-valued stable models are the basis of various other semantics (like well-founded, regular, ideal, and (2-valued) stable). For instance, where preferred semantics takes the complete assumption labellings and selects those with maximal IN, regular semantics takes the 3-valued stable models and selects those with maximal T. Hence, to prove equivalence between preferred semantics in ABA and regular semantics in logic programming, we need to show that there's an equivalence (through the functions Lab2Mod and Mod2Lab) between the complete assumption labellings with maximal IN and the 3-valued stable models with maximal T. For this purpose, we first introduce the following lemma on the correspondence of the set of IN assumptions of a complete assumption labelling and T of a 3-valued stable model.

Lemma 3.6. Let $\mathcal{F} = \langle \mathcal{L}, \mathcal{R}, \mathcal{A}, ^- \rangle$ be a normal ABA framework and $P_\mathcal{F}$ the associated logic program. Let $\mathcal{L}ab_1$ and $\mathcal{L}ab_2$ be complete assumption labellings of $\mathcal{F}$, and let $\mathcal{M}od_1 = \langle T_1, F_1 \rangle = \texttt{Lab2Mod}(\mathcal{L}ab_1)$ and $\mathcal{M}od_2 = \langle T_2, F_2 \rangle = \texttt{Lab2Mod}(\mathcal{L}ab_2)$. It holds that $\text{IN}(\mathcal{L}ab_1) \subseteq \text{IN}(\mathcal{L}ab_2)$ iff $T_1 \subseteq T_2$.

Proof:
"⇒": Assume that $\text{IN}(\mathcal{L}ab_1) \subseteq \text{IN}(\mathcal{L}ab_2)$. From $\mathcal{M}od_1 = \texttt{Lab2Mod}(\mathcal{L}ab_1)$ it follows that $\langle T_1, F_1 \rangle = \Phi(P_\mathcal{F}^{\langle T_1', F_1' \rangle})$ with $T_1' = \{\overline{\chi} \mid \chi \in \text{OUT}(\mathcal{L}ab_1)\} \cap HB_{P_\mathcal{F}}$ and $F_1' = \{\overline{\chi} \mid \chi \in \text{IN}(\mathcal{L}ab_1)\} \cap HB_{P_\mathcal{F}}$. From $\mathcal{M}od_2 = \texttt{Lab2Mod}(\mathcal{L}ab_2)$ it follows that $\langle T_2, F_2 \rangle = \Phi(P_\mathcal{F}^{\langle T_2', F_2' \rangle})$ with $T_2' = \{\overline{\chi} \mid \chi \in \text{OUT}(\mathcal{L}ab_2)\} \cap HB_{P_\mathcal{F}}$ and $F_2' = \{\overline{\chi} \mid \chi \in \text{IN}(\mathcal{L}ab_2)\} \cap HB_{P_\mathcal{F}}$. From the fact that $\text{IN}(\mathcal{L}ab_1) \subseteq \text{IN}(\mathcal{L}ab_2)$ it follows (Lemma 2.4) that $\text{OUT}(\mathcal{L}ab_1) \subseteq \text{OUT}(\mathcal{L}ab_2)$, so we obtain that $T_1' \subseteq T_2'$. It then follows that $T_1 \subseteq T_2$ (see [5] for details).

"$\Leftarrow$": Since Lab2Mod and Mod2Lab are each other's inverses (point 3 of Theorem 3.4) it follows that $\mathcal{L}ab_1 = \texttt{Mod2Lab}(\mathcal{M}od_1)$ and $\mathcal{L}ab_2 = \texttt{Mod2Lab}(\mathcal{M}od_2)$. From the definition of Mod2Lab it then follows that $\text{OUT}(\mathcal{L}ab_1) = \{\chi \in \mathcal{A} \mid \overline{\chi} \in T_1\}$ and $\text{OUT}(\mathcal{L}ab_2) = \{\chi \in \mathcal{A} \mid \overline{\chi} \in T_2\}$. From $T_1 \subseteq T_2$ it then follows that $\text{OUT}(\mathcal{L}ab_1) \subseteq \text{OUT}(\mathcal{L}ab_2)$. From Lemma 2.4 it then follows that $\text{IN}(\mathcal{L}ab_1) \subseteq \text{IN}(\mathcal{L}ab_2)$. □

From the fact that for complete assumption labellings and 3-valued stable models Lab2Mod and Mod2Lab are each other's inverses, it follows that Lemma 3.6 can also be applied for two 3-valued stable models $\mathcal{M}od_1$ and $\mathcal{M}od_2$ of $P_\mathcal{F}$ and the associated assumption labellings $\mathcal{L}ab_1 = \texttt{Mod2Lab}(\mathcal{M}od_1)$ and $\mathcal{L}ab_2 = \texttt{Mod2Lab}(\mathcal{M}od_2)$ of $\mathcal{F}$. We will sometimes do so in the proof of the following theorem.

Theorem 3.7. Let $\mathcal{F} = \langle \mathcal{L}, \mathcal{R}, \mathcal{A}, ^- \rangle$ be a normal ABA framework and let $P_\mathcal{F}$ be the associated logic program. It holds that:

1. if $\mathcal{L}ab$ is a grounded assumption labelling of $\mathcal{F}$ then $\texttt{Lab2Mod}(\mathcal{L}ab)$ is a well-founded model of $P_\mathcal{F}$
2. if $\mathcal{M}od$ is a well-founded model of $P_\mathcal{F}$ then $\texttt{Mod2Lab}(\mathcal{M}od)$ is a grounded assumption labelling of $\mathcal{F}$
3. if $\mathcal{L}ab$ is a preferred assumption labelling of $\mathcal{F}$ then $\texttt{Lab2Mod}(\mathcal{L}ab)$ is a regular model of $P_\mathcal{F}$
4. if $\mathcal{M}od$ is a regular model of $P_\mathcal{F}$ then $\texttt{Mod2Lab}(\mathcal{M}od)$ is a preferred assumption labelling of $\mathcal{F}$
5. if $\mathcal{L}ab$ is a stable assumption labelling of $\mathcal{F}$ then $\texttt{Lab2Mod}(\mathcal{L}ab)$ is a (2-valued) stable model of $P_\mathcal{F}$
6. if $\mathcal{M}od$ is a (2-valued) stable model of $P_\mathcal{F}$ then $\texttt{Mod2Lab}(\mathcal{M}od)$ is a stable assumption labelling of $\mathcal{F}$
7. if $\mathcal{L}ab$ is an ideal assumption labelling of $\mathcal{F}$ then $\texttt{Lab2Mod}(\mathcal{L}ab)$ is an ideal model of $P_\mathcal{F}$
8. if $\mathcal{M}od$ is an ideal model of $P_\mathcal{F}$ then $\texttt{Mod2Lab}(\mathcal{M}od)$ is an ideal assumption labelling of $\mathcal{F}$

Proof:

1. Let $\mathcal{L}ab$ be a grounded assumption labelling of $\mathcal{F}$, and let $\mathcal{M}od = \texttt{Lab2Mod}(\mathcal{L}ab)$ with $\mathcal{M}od = \langle T, F \rangle$. Then, from Theorem 3.4 it follows that $\mathcal{M}od$ is a 3-valued stable model of $P_\mathcal{F}$. In order to show that $\mathcal{M}od$ is also a well-founded model of $P_\mathcal{F}$ we have to additionally prove that T is *minimal* among all 3-valued stable models of $P_\mathcal{F}$. Let $\mathcal{M}od^* = \langle T^*, F^* \rangle$ be an arbitrary 3-valued stable model of $P_\mathcal{F}$. We have to prove that if $T^* \subseteq T$ then $T^* = T$. Suppose $T^* \subseteq T$. Then, according to Lemma 3.6, $\text{IN}(\mathcal{L}ab^*) \subseteq \text{IN}(\mathcal{L}ab)$, with $\mathcal{L}ab^* = \texttt{Mod2Lab}(\mathcal{M}od^*)$. From $\mathcal{L}ab$ being a grounded assumption labelling of $\mathcal{F}$, it follows that $\text{IN}(\mathcal{L}ab)$ is minimal among all complete assumption labellings of $\mathcal{F}$. Hence, from $\text{IN}(\mathcal{L}ab^*) \subseteq \text{IN}(\mathcal{L}ab)$ it follows that $\text{IN}(\mathcal{L}ab^*) = \text{IN}(\mathcal{L}ab)$, so $\text{IN}(\mathcal{L}ab^*) \supseteq \text{IN}(\mathcal{L}ab)$. From Lemma 3.6 it then follows that $T^* \supseteq T$, which together with $T^* \subseteq T$ implies that $T^* = T$. Hence, $\mathcal{M}od$ is a well-founded model of $P_\mathcal{F}$.

2. Let $\mathcal{M}od = \langle T, F\rangle$ be a well-founded model of $P_{\mathcal{F}}$, and let $\mathcal{L}ab = \texttt{Mod2Lab}(\mathcal{M}od)$. From Theorem 3.4 it then follows that $\mathcal{L}ab$ is a complete assumption labelling of $\mathcal{F}$. In order to show that $\mathcal{M}od$ is also a grounded assumption labelling of $\mathcal{F}$ we have to additionally prove that $\text{IN}(\mathcal{L}ab)$ is *minimal* among all complete assumption labellings of $\mathcal{F}$. Let $\mathcal{L}ab^*$ be an arbitrary complete assumption labelling of $\mathcal{F}$. We have to prove that if $\text{IN}(\mathcal{L}ab^*) \subseteq \text{IN}(\mathcal{L}ab)$ then $\text{IN}(\mathcal{L}ab^*) = \text{IN}(\mathcal{L}ab)$. Suppose $\text{IN}(\mathcal{L}ab^*) \subseteq \text{IN}(\mathcal{L}ab)$. Then, according to Lemma 3.6, $T^* \subseteq T$, with $\langle T^*, F^*\rangle = \mathcal{M}od^* = \texttt{Lab2Mod}(\mathcal{L}ab^*)$. From $\mathcal{M}od$ being a well-founded model of $P_{\mathcal{F}}$, it follows that T is minimal among all 3-valued stable models of $P_{\mathcal{F}}$. Hence, from $T^* \subseteq T$ it follows that $T^* = T$, so $T^* \supseteq T$. From Lemma 3.6 it then follows that $\text{IN}(\mathcal{L}ab^*) \supseteq \text{IN}(\mathcal{L}ab)$, which together with $\text{IN}(\mathcal{L}ab^*) \subseteq \text{IN}(\mathcal{L}ab)$ implies that $\text{IN}(\mathcal{L}ab^*) = \text{IN}(\mathcal{L}ab)$. Hence, $\mathcal{L}ab$ is a grounded assumption labelling of $\mathcal{F}$.

3. Similar to 1, but assuming $T^* \supseteq T$ to show that T is *maximal*.

4. Similar to 2, but assuming $\text{IN}(\mathcal{L}ab^*) \supseteq \text{IN}(\mathcal{L}ab)$ to show that $\text{IN}(\mathcal{L}ab)$ is *maximal*.

5. Proof by contraposition. Suppose that $\mathcal{M}od = \langle T, F\rangle = \texttt{Lab2Mod}(\mathcal{L}ab)$ is *not* a (2-valued) stable model of $P_{\mathcal{F}}$. In case $\mathcal{M}od$ is not even a 3-valued stable model of $P_{\mathcal{F}}$, Theorem 3.4 implies that $\mathcal{L}ab$ is not a complete assumption labelling of $\mathcal{F}$, so $\mathcal{L}ab$ is also not a stable assumption labelling of $\mathcal{F}$. In the remainder of this proof, we will therefore treat the case that $\mathcal{M}od$ is a 3-valued stable model of $P_{\mathcal{F}}$. From the fact that $\mathcal{M}od$ is not a (2-valued) stable model of $P_{\mathcal{F}}$ it then follows that $T \cup F \neq HB_{P_{\mathcal{F}}}$, so there exists a $x \in HB_{P_{\mathcal{F}}} \setminus (T \cup F)$. It follows that there exists some `not` z in $P_{\mathcal{F}}$ which is used in the "derivation" of x such that $z \in HB_{P_{\mathcal{F}}}$, and $z \notin T'$ and $z \notin F'$ (see [5] for details) where $\mathcal{M}od = \texttt{Lab2Mod}(\mathcal{L}ab) = \Phi(P_{\mathcal{F}}^{\langle T', F'\rangle})$ with $T' = \{\overline{\chi} \mid \chi \in \text{OUT}(\mathcal{L}ab)\} \cap HB_{P_{\mathcal{F}}}$ and $F' = \{\overline{\chi} \mid \chi \in \text{IN}(\mathcal{L}ab)\} \cap HB_{P_{\mathcal{F}}}$. Then there exists $\zeta \in \mathcal{A}$ such that $\overline{\zeta} = z$. It then follows from the definition of `Lab2Mod` that $\zeta \notin \text{OUT}(\mathcal{L}ab)$ and $\zeta \notin \text{IN}(\mathcal{L}ab)$, so $\zeta \in \text{UNDEC}(\mathcal{L}ab)$. This then implies that $\text{UNDEC}(\mathcal{L}ab) \neq \emptyset$, so $\mathcal{L}ab$ is not a stable assumption labelling of $\mathcal{F}$.

6. Proof by contraposition. Suppose that $\mathcal{L}ab = \texttt{Mod2Lab}(\mathcal{M}od)$ is *not* a stable assumption labelling of $\mathcal{F}$. In case $\mathcal{L}ab$ is not even a complete assumption labelling of $\mathcal{F}$, Theorem 3.4 implies that $\mathcal{M}od$ is not a 3-valued stable model of $P_{\mathcal{F}}$, so $\mathcal{M}od$ is also not a (2-valued) stable model of $P_{\mathcal{F}}$. In the remainder of this proof, we will therefore treat the case that $\mathcal{L}ab$ is a complete assumption labelling of $\mathcal{F}$. From the fact that $\mathcal{L}ab$ is not a stable assumption labelling of $\mathcal{F}$ it then follows that $\text{UNDEC}(\mathcal{L}ab) \neq \emptyset$, so there exists an $\chi \in \text{UNDEC}(\mathcal{L}ab)$. That is, $\chi \in \text{UNDEC}(\texttt{Mod2Lab}(\mathcal{M}od))$. From the definition of `Mod2Lab` it then follows that $\overline{\chi} \in HB_{P_{\mathcal{F}}} \setminus (T \cup F)$, so $T \cup F \neq HB_{P_{\mathcal{F}}}$, so $\mathcal{M}od$ is not a (2-valued) stable model of $P_{\mathcal{F}}$.

7. Let $\mathcal{L}ab$ be an ideal assumption labelling of $\mathcal{F}$, i.e. $\text{IN}(\mathcal{L}ab)$ is maximal among all complete assumption labellings of $\mathcal{F}$ with $\text{IN}(\mathcal{L}ab) \subseteq \bigcap\{\text{IN}(\mathcal{L}ab_{pref}) \mid \mathcal{L}ab_{pref}$ is a preferred assumption labelling of $\mathcal{F}\}$. By Theorem 3.4 $\mathcal{M}od = \langle T, F\rangle = \texttt{Lab2Mod}(\mathcal{L}ab)$ is a 3-valued stable model of $P_{\mathcal{F}}$. Since for all preferred assumption labellings $\mathcal{L}ab_{pref}$ of $\mathcal{F}$ it holds that $\text{IN}(\mathcal{L}ab) \subseteq \text{IN}(\mathcal{L}ab_{pref})$, it follows by Lemma 3.6 that $T \subseteq T_{reg}$ for all $\mathcal{M}od_{reg} = \texttt{Lab2Mod}(\mathcal{L}ab_{pref}) = \langle T_{reg}, F_{reg}\rangle$. Furthermore, by Theorem 3.7 (point 3) all $\mathcal{M}od_{reg}$ are regular models of $P_{\mathcal{F}}$. Thus, $\mathcal{M}od$ is a 3-valued stable model of $P_{\mathcal{F}}$ with $T \subseteq \bigcap\{T_{reg} \mid \langle T_{reg}, F_{reg}\rangle$ is a regular model of $P_{\mathcal{F}}\}$. To show that in addition T is *maximal* among all 3-valued stable models of

$P_{\mathcal{F}}$ with $T \subseteq \bigcap\{T_{reg} \mid \langle T_{reg}, F_{reg}\rangle$ is a regular model of $P_{\mathcal{F}}\}$, let $\mathcal{M}od^* = \langle T^*, F^*\rangle$ be a 3-valued stable model of $P_{\mathcal{F}}$ with $T^* \subseteq \bigcap\{T_{reg} \mid \langle T_{reg}, F_{reg}\rangle$ is a regular model of $P_{\mathcal{F}}\}$. We have to prove that if $T^* \supseteq T$ then $T^* = T$. Suppose $T^* \supseteq T$. Since for every regular model $\mathcal{M}od_{reg} = \langle T_{reg}, F_{reg}\rangle$ of $P_{\mathcal{F}}$ it holds that $T^* \subseteq T_{reg}$, it follows from Lemma 3.6 that $\text{IN}(\mathcal{L}ab^*) \subseteq \text{IN}(\mathcal{L}ab_{pref})$ where $\mathcal{L}ab^* = \texttt{Mod2Lab}(\mathcal{M}od^*)$ and $\mathcal{L}ab_{pref} = \texttt{Mod2Lab}(\mathcal{M}od_{reg})$. Furthermore by Theorem 3.7 (point 4), all $\mathcal{L}ab_{pref}$ are preferred assumption labellings of $\mathcal{F}$. Thus, $\text{IN}(\mathcal{L}ab^*) \subseteq \bigcap\{\text{IN}(\mathcal{L}ab_{pref}) \mid \mathcal{L}ab_{pref}$ is a preferred assumption labelling of $\mathcal{F}\}$. But by Lemma 3.6 also $\text{IN}(\mathcal{L}ab^*) \supseteq \text{IN}(\mathcal{L}ab)$. However, since $\text{IN}(\mathcal{L}ab)$ is also *maximal* among all complete assumption labellings of $\mathcal{F}$ with $\text{IN}(\mathcal{L}ab) \subseteq \{\text{IN}(\mathcal{L}ab_{pref}$ is a preferred assumption labelling of $\mathcal{F}\}$, it follows that $\text{IN}(\mathcal{L}ab^*) = \text{IN}(\mathcal{L}ab)$, so trivially $\text{IN}(\mathcal{L}ab^*) \subseteq \text{IN}(\mathcal{L}ab)$. From Lemma 3.5 it then follows that $T^* \subseteq T$, which together with $T* \supseteq T$ implies that $T^* = T$ Hence, $\mathcal{M}od$ is an ideal model of $P_{\mathcal{F}}$.

8. Let $\mathcal{M}od = \langle T, F\rangle$ be an ideal model of $P_{\mathcal{F}}$, i.e. T is maximal among all 3-valued stable models of $P_{\mathcal{F}}$ with $T \subseteq \bigcap\{T_{reg} \mid \langle T_{reg}, F_{reg}\rangle$ is a regular model of $P_{\mathcal{F}}\}$. By Theorem 3.4 $\mathcal{L}ab = \texttt{Mod2Lab}(\mathcal{M}od)$ is a complete assumption labelling of $\mathcal{F}$. Since for all regular models $\mathcal{M}od_{reg} = \langle T_{reg}, F_{reg}\rangle$ of $P_{\mathcal{F}}$ it holds that $T \subseteq T_{reg}$, it follows by Lemma 3.6 that $\text{IN}(\mathcal{L}ab) \subseteq \text{IN}(\mathcal{L}ab_{pref})$ with $\mathcal{L}ab_{pref} = \texttt{Mod2Lab}(\mathcal{M}od_{reg})$. Furthermore, by Theorem 3.7 (point 4) all $\mathcal{L}ab_{pref}$ are preferred assumption labellings of $\mathcal{F}$. Thus, $\mathcal{L}ab$ is a complete assumption labelling of $\mathcal{F}$ with $\text{IN}(\mathcal{L}ab) \subseteq \bigcap\{\text{IN}(\mathcal{L}ab_{pref}) \mid \mathcal{L}ab_{pref}$ is a preferred assumption labelling of $\mathcal{F}\}$. To show that in addition $\text{IN}(\mathcal{L}ab)$ is *maximal* among all complete assumption labellings of $\mathcal{F}$ with $\text{IN}(\mathcal{L}ab) \subseteq \bigcap\{\text{IN}(\mathcal{L}ab_{pref}) \mid \mathcal{L}ab_{pref}$ is a preferred assumption labelling of $\mathcal{F}\}$, let $\mathcal{L}ab^*$ be a complete assumption labelling of $\mathcal{F}$ with $\text{IN}(\mathcal{L}ab^*) \subseteq \bigcap\{\text{IN}(\mathcal{L}ab_{pref}) \mid \mathcal{L}ab_{pref}$ is a preferred assumption labelling of $\mathcal{F}\}$. We have to prove that if $\text{IN}(\mathcal{L}ab^*) \supseteq \text{IN}(\mathcal{L}ab)$ then $\text{IN}(\mathcal{L}ab^*) = \text{IN}(\mathcal{L}ab)$. Suppose $\text{IN}(\mathcal{L}ab^*) \supseteq \text{IN}(\mathcal{L}ab)$. Since for every preferred assumption labelling $\mathcal{L}ab_{pref}$ of $\mathcal{F}$ it holds that $\text{IN}(\mathcal{L}ab^*) \subseteq \text{IN}(\mathcal{L}ab_{pref})$, it follows from Lemma 3.6 that $T^* \subseteq T_{reg}$ where $\mathcal{M}od^* = \langle T^*, F^*\rangle = \texttt{Lab2Mod}(\mathcal{L}ab^*)$ and $\mathcal{M}od_{reg} = \langle T_{reg}, F_{reg}\rangle = \texttt{Lab2Mod}(\mathcal{L}ab_{pref})$. Furthermore by Theorem 3.7 (point 3), all $\mathcal{M}od_{reg}$ are regular models of $P_{\mathcal{F}}$. Thus, $T^* \subseteq \bigcap\{T_{reg} \mid \langle T_{reg}, F_{reg}\rangle$ is a regular model of $P_{\mathcal{F}}\}$. But by Lemma 3.6 also $T^* \supseteq T$. However, since T is also *maximal* among all 3-valued stable models of $P_{\mathcal{F}}$ with $T \subseteq \bigcap\{T_{reg} \mid \langle T_{reg}, F_{reg}\rangle$ is a regular model of $P_{\mathcal{F}}\}$, it follows that $T^* = T$, so trivially $T^* \subseteq T$. From Lemma 3.5 it then follows that $\text{IN}(\mathcal{L}ab^*) \subseteq \text{IN}(\mathcal{L}ab)$, which together with $\text{IN}(\mathcal{L}ab^*) \supseteq \text{IN}(\mathcal{L}ab)$ implies that $\text{IN}(\mathcal{L}ab^*) = \text{IN}(\mathcal{L}ab)$. Hence, $\mathcal{L}ab$ is an ideal assumption labelling of $\mathcal{F}$.

□

4. Translating Logic Programs to ABA Theories

In the previous section, we have studied a translation from normal ABA to LP, and have observed that the various types of labellings of an ABA framework coincide with the various types of models of the associated logic program. In this section, we go the other way around. That is, we examine a translation from LP to ABA.

Definition 4.1. Let P be a normal logic program. We define the *associated ABA framework* $\mathcal{F}_P = \langle \mathcal{L}, \mathcal{R}, \mathcal{A}, ^- \rangle$ with $\mathcal{A} = \{\texttt{not\_w} \mid w \in HB_P\}$, $\mathcal{L} = HB_P \cup \mathcal{A}$, $\mathcal{R} = \{x \leftarrow y_1, \ldots, y_n, \texttt{not\_z}_1, \ldots,$

$\texttt{not\_z}_m \mid x \leftarrow y_1, \ldots, y_n, \texttt{not } z_1, \ldots, \texttt{not } z_m \in P\}$ and $\overline{\texttt{not\_w}} = w$ for every $\texttt{not\_w} \in \mathcal{A}$.

We define LP2ABA to be the function that, given a logic program P, yields the associated ABA framework $\mathcal{F}_P$ (Definition 4.1). Similarly, we define ABA2LP to be the function that, given a normal ABA framework $\mathcal{F}$, yields the associated logic program $P_{\mathcal{F}}$ (Definition 3.1).

Theorem 4.2. Let P be a normal logic program. It holds that $\texttt{ABA2LP}(\texttt{LP2ABA}(P)) = P$.

Proof:
Let $\mathcal{F}_P = \langle \mathcal{L}_P, \mathcal{R}_P, \mathcal{A}_P, ^- \rangle$ be $\texttt{LP2ABA}(P)$.
"$\subseteq$": Let $x \leftarrow y_1, \ldots, y_n, \texttt{not } z_1, \ldots, \texttt{not } z_m$ be a logic programming rule in $\texttt{ABA2LP}(\texttt{LP2ABA}(P))$. Then from the definition of ABA2LP it follows that there exists an ABA rule $x \leftarrow y_1, \ldots, y_n, \zeta_1, \ldots, \zeta_m$ in $\mathcal{R}_P$ with $\zeta_i \in \mathcal{A}_P$ and $\overline{\zeta_i} = z_i$ $(1 \leq i \leq m)$. From the definition of LP2ABA it then follows that $\zeta_i = \texttt{not\_z}_i$ (because $\texttt{not\_z}_i$ is the only assumption in $\mathcal{A}_P$ that has z_i as its contrary $(1 \leq i \leq m)$) and that there exists a rule $x \leftarrow y_1, \ldots, y_n, \texttt{not } z_1, \ldots, \texttt{not } z_m$ in P.
"$\supseteq$": Let $x \leftarrow y_1, \ldots, y_n, \texttt{not } z_1, \ldots, \texttt{not } z_m$ be a logic programming rule in P. Then from the definition of LP2ABA it follows that $\mathcal{R}_P$ contains a rule $x \leftarrow y_1, \ldots, y_n, \texttt{not\_z}_1, \ldots, \texttt{not\_z}_m$ with $\overline{\texttt{not\_z}_i} = z_i$ $(1 \leq i \leq m)$. From the definition of ABA2LP this then implies that $\texttt{ABA2LP}(\texttt{LP2ABA}(P))$ contains a rule $x \leftarrow y_1, \ldots, y_n, \texttt{not } z_1, \ldots, \texttt{not } z_m$. □

Theorem 4.3. Let P be a logic program and let $\mathcal{F}_P = \texttt{LP2ABA}(P)$ be its associated ABA framework. It holds that:

1. if $\mathcal{M}od$ is a 3-valued stable model of P, then $\texttt{Mod2Lab}(\mathcal{M}od)$ is a complete assumption labelling of $\mathcal{F}_P$
2. if $\mathcal{L}ab$ is a complete assumption labelling of $\mathcal{F}_P$, then $\texttt{Lab2Mod}(\mathcal{L}ab)$ is a 3-valued stable model of P

Proof:
[4] Let $P_{\mathcal{F}_P}$ be the associated logic program of $\mathcal{F}_P$, i.e. $P_{\mathcal{F}_P} = \texttt{ABA2LP}(\mathcal{F}_P)$. From $\mathcal{F}_P = \texttt{LP2ABA}(P)$ it then follows that $P_{\mathcal{F}_P} = \texttt{ABA2LP}(\texttt{LP2ABA}(P))$. It then follows from Theorem 4.2 that $P_{\mathcal{F}_P} = P$.

1. Let $\mathcal{M}od$ be a 3-valued stable model of P. As $P = P_{\mathcal{F}_P}$, it directly follows that $\mathcal{M}od$ is a 3-valued stable model of $P_{\mathcal{F}_P}$. From Theorem 3.4 (point 2) it then follows that $\texttt{Mod2Lab}(\mathcal{M}od)$ is a complete assumption labelling of $\mathcal{F}_P$.
2. Let $\mathcal{L}ab$ be a complete assumption labelling of $\mathcal{F}_P$. From Theorem 3.4 (point 1) it then follows that $\texttt{Lab2Mod}(\mathcal{L}ab)$ is a 3-valued stable model of $P_{\mathcal{F}_P}$. From the fact that $P_{\mathcal{F}_P} = P$, it then directly follows that $\texttt{Lab2Mod}(\mathcal{L}ab)$ is also a 3-valued stable model of P.

□

We now extend the correspondence results from Theorem 4.3 to common ABA and LP semantics.

Theorem 4.4. Let P be a logic program and let $\mathcal{F}_P = \texttt{LP2ABA}(P)$ be its associated ABA framework. It holds that:

[4] This was proven in [20]. Here, we use a different proof which will serve as an illustration for the proofs of Theorem 4.4.

1. if $\mathcal{M}od$ is a well-founded model of P, then $\texttt{Mod2Lab}(\mathcal{M}od)$ is a grounded assumption labelling of $\mathcal{F}_P$
2. if $\mathcal{L}ab$ is a grounded assumption labelling of $\mathcal{F}_P$, then $\texttt{Lab2Mod}(\mathcal{L}ab)$ is a well-founded model of P
3. if $\mathcal{M}od$ is a regular model of P, then $\texttt{Mod2Lab}(\mathcal{M}od)$ is a preferred assumption labelling of $\mathcal{F}_P$
4. if $\mathcal{L}ab$ is a preferred assumption labelling of $\mathcal{F}_P$, then $\texttt{Lab2Mod}(\mathcal{L}ab)$ is a regular model of P
5. if $\mathcal{M}od$ is a (2-valued) stable model of P, then $\texttt{Mod2Lab}(\mathcal{M}od)$ is a stable assumption labelling of $\mathcal{F}_P$
6. if $\mathcal{L}ab$ is a stable assumption labelling of $\mathcal{F}_P$, then $\texttt{Lab2Mod}(\mathcal{L}ab)$ is a (2-valued) stable model of P
7. if $\mathcal{M}od$ is an ideal model of P, then $\texttt{Mod2Lab}(\mathcal{M}od)$ is an ideal assumption labelling of $\mathcal{F}_P$
8. if $\mathcal{L}ab$ is an ideal assumption labelling of $\mathcal{F}_P$, then $\texttt{Lab2Mod}(\mathcal{L}ab)$ is an ideal model of P

Proof:
Similar to the proof of Theorem 4.3, but applying Theorem 3.7 instead of Theorem 3.4. □

Theorem 4.3 and Theorem 4.4 point out that our results regarding the translation from ABA to LP, as stated in the previous section, can be reused for the translation from LP to ABA. Hence, with respect to normal logic programs, our work generalizes the work of [20, 2] where only the LP to ABA direction is considered.

If it is possible to reuse the results from the ABA to LP translation for the LP to ABA translation, then is the reverse also possible? That is, we ask ourselves whether it is possible to reuse some of the existing work on the LP to ABA translation (like for instance stated in [20, 2]) to obtain similar results for the ABA to LP translation (like for instance stated in the previous section). The short answer is no, at least not in any obvious way. Our ability to reuse the results from the ABA to LP translation for the LP to ABA translation (and therefore to write the sort of proofs of Theorem 4.3 and Theorem 4.4) critically depends on the fact that $\texttt{ABA2LP}(\texttt{LP2ABA}(P)) = P$ (Theorem 4.2). To be able to reuse the results of the LP to ABA direction for the ABA to LP direction in a similar way as that is done in the current section would require the property that for any ABA framework $\mathcal{F}$, $\texttt{LP2ABA}(\texttt{ABA2LP}(\mathcal{F})) = \mathcal{F}$. However, this property does *not* hold, for the simple reason that when translating from ABA to LP some information gets lost (like the precise set of assumptions, some of which may not occur in any rule) as in essence only its set of rules $\mathcal{R}$ gets translated.

5. Discussion

In the current paper we examined the relation between ABA and LP, and found that a frequently studied fragment of ABA is subsumed by normal logic programming, and vice versa. That is, the kind of outcome that is yielded by a normal ABA framework under common ABA semantics is essentially the same as the outcome yielded by its associated normal logic program under common LP semantics. The only real

difference is that whereas in ABA the outcome is defined in terms of assumptions (which correspond to the NAF-literals in the associated logic program) in logic programming the outcome is defined in terms of *all* the literals in the logic program (NAF as well as non-NAF). However, since the status of the non-NAF literals is determined solely by the status of the NAF-literals (basically, by applying the Gelfond-Lifschitz reduct, as is done by `Lab2Mod`) both approaches are equivalent.

The results of our paper enable researchers to switch freely between ABA syntax and semantics, and LP syntax and semantics. For instance, when applying normal ABA for reasoning about a particular domain, one could equally apply normal LP for the same reasoning. The advantage of doing so is that, as more people are familiar with LP than with ABA, the results can be disseminated to a wider audience. Similarly, the equivalence between ABA and LP allows some of the techniques developed in the context of ABA to be carried over to the context of LP. For instance, the argument-based proof procedures of [8, 22], when used in a normal ABA framework, can be carried over to logic programming in a straightforward way. Of course, this would require the notion of an argument to be defined in the context of a logic program, but this has already been done in [24, 4]. In fact, LP arguments have been applied in some of the detailed proofs of the current paper's technical results [5].

It should be emphasized that our results regarding the equivalence between ABA and LP are restricted to normal ABA and normal LP. For instance, logic programming would struggle to model non-flat ABA frameworks, as this would require NAF-literals to occur in the head of LP rules. Similarly, ABA would struggle to model disjunctive logic programming (where the head of a rule can be a disjunction) as it is not clear how arguments can be constructed in this context.

Another issue is how the ABA-LP equivalence is affected when applying non-common ABA and LP semantics. Take for instance the case of semi-stable semantics for ABA [3, 20] and L-stable semantics for LP [10, 4]. A semi-stable assumption labelling is defined as a complete assumption labelling $\mathcal{L}ab$ where $\text{UNDEC}(\mathcal{L}ab) = \mathcal{A} \setminus (\text{IN}(\mathcal{L}ab) \cup \text{OUT}(\mathcal{L}ab))$ is minimal, whereas an L-stable model is defined as a 3-valued stable model $\langle T, F \rangle$ where $HB_P \setminus (T \cup F)$ is minimal. Although one would expect these semantics to coincide, this turns out not to be the case, although equivalence does hold for very restricted types of ABA frameworks [5]. Overall, we hold the issue of how the results in the current paper can be generalized beyond normal ABA frameworks, normal logic programs and common semantics to be an interesting topic for further research.

References

[1] Alferes, J. J., Dung, P. M., Pereira, L. M.: Scenario Semantics of Extended Logic Programs, *LPNMR'93* (A. Nerode, L. Pereira, Eds.), MIT Press, 1993.

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

[3] Caminada, M. W. A., Sá, S., Alcântara, J., Dvořák, W.: On the Difference between Assumption-Based Argumentation and Abstract Argumentation, *IFCoLog J. of Logics and their Applications*, 2015, In print.

[4] Caminada, M. W. A., Sá, S., Alcântara, J., Dvořák, W.: On the Equivalence between Logic Programming Semantics and Argumentation Semantics, *Int. J. of Approximate Reasoning*, **58**, 2015, 87–111.

[5] Caminada, M. W. A., Schulz, C.: *On the Equivalence between Assumption-Based Argumentation and Logic Programming*, Technical report, University of Aberdeen, 2015.

[6] Craven, R., Toni, F., Cadar, C., Hadad, A., Williams, M.: Efficient Argumentation for Medical Decision-Making, *KR'12* (G. Brewka, T. Eiter, S. A. McIlraith, Eds.), AAAI Press, 2012.

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

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

[9] Dung, P. M., Thang, P. M.: Modular Argumentation for Modelling Legal Doctrines in Common Law of Contract, *Artificial Intelligence and Law*, **17**(3), 2009, 167–182.

[10] Eiter, T., Leone, N., Saccà, D.: On the Partial Semantics for Disjunctive Deductive Databases, *Annals of Mathematics and Artificial Intelligence*, **19**(1-2), 1997, 59–96.

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

[12] Fan, X., Toni, F.: On Computing Explanations in Argumentation, *AAAI'15* (B. Bonet, S. Koenig, Eds.), AAAI Press, 2015.

[13] Fan, X., Toni, F., Mocanu, A., Williams, M.: Dialogical Two-Agent Decision Making with Assumption-Based Argumentation, *AAMAS'14* (A. L. C. Bazzan, M. N. Huhns, A. Lomuscio, P. Scerri, Eds.), IFAA-MAS/ACM, 2014.

[14] Gaertner, D., Toni, F.: Computing Arguments and Attacks in Assumption-Based Argumentation, *IEEE Intelligent Systems*, **22**(6), 2007, 24–33.

[15] Morge, M., Mancarella, P.: Assumption-Based Argumentation for the Minimal Concession Strategy, in: *Argumentation in Multi-Agent Systems* (P. McBurney, I. Rahwan, S. Parsons, N. Maudet, Eds.), Springer Berlin Heidelberg, 2010, 114–133.

[16] Przymusinski, T. C.: The Well-Founded Semantics Coincides with the Three-Valued Stable Semantics, *Fundamenta Informaticae*, **13**(4), 1990, 445–463.

[17] Schulz, C.: Graphical Representation of Assumption-Based Argumentation, *AAAI'15* (B. Bonet, S. Koenig, Eds.), AAAI Press, 2015.

[18] Schulz, C., Toni, F.: Complete Assumption Labellings, *COMMA'14* (S. Parsons, N. Oren, C. Reed, F. Cerutti, Eds.), IOS Press, 2014.

[19] Schulz, C., Toni, F.: Justifying Answer Sets using Argumentation, *Theory and Practice of Logic Programming*, **FirstView**, 2015, 1–52.

[20] Schulz, C., Toni, F.: Logic Programming in Assumption-Based Argumentation Revisited - Semantics and Graphical Representation, *AAAI'15* (B. Bonet, S. Koenig, Eds.), AAAI Press, 2015.

[21] Toni, F.: Assumption-Based Argumentation for Closed and Consistent Defeasible Reasoning, *JSAI'07* (K. Satoh, A. Inokuchi, K. Nagao, T. Kawamura, Eds.), Springer Berlin Heidelberg, 2007.

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

[23] Toni, F.: A Tutorial on Assumption-Based Argumentation, *Argument & Computation*, **5**, 2014, 89–117.

[24] Wu, Y., Caminada, M. W. A., Gabbay, D. M.: Complete Extensions in Argumentation Coincide with 3-Valued Stable Models in Logic Programming, *Studia Logica*, **93**(2-3), 2009, 383–403.

[25] You, J. H., Yuan, L. Y.: Three-Valued Formalization of Logic Programming: Is It Needed?, *PODS'90*, ACM, 1990.