On the Equivalence between Logic Programs and Bipolar Argumentation Frameworks

JOÃO ALCÂNTARA and RENAN CORDEIRO*, Federal University of Ceará, Brazil

Abstract Argumentation Frameworks (*AAF*s) and Normal Logic Programs (*NLP*s) are closely related formalisms for which many equivalences have already been elicited. In this paper, we extend this line of research by considering Bipolar Argumentation Frameworks (*BAF*s), in which arguments have an explicit support relation, independent of the attack relation. We provide direct translations from *BAF*s to *NLP*s (and vice versa) in a one-to-one correspondence between several argumentation and 3-valued logic programming semantics. This includes the equivalence involving L-stable semantics. Besides, we deepen the connection between *NLP*s and *BAF*s by finding subsets of them for which the proposed translations are each other's inverse up to isomorphism.

JAIR Associate Editor: Davide Grossi

JAIR Reference Format:
João Alcântara and Renan Cordeiro. 2025. On the Equivalence between Logic Programs and Bipolar Argumentation Frameworks. *Journal of Artificial Intelligence Research* 84, Article 17 (October 2025), 43 pages. DOI: 10.1613/jair.1.18086

1 Introduction

Formal argumentation and logic programming are two highly successful paradigms in Artificial Intelligence (AI). They offer robust reasoning, decision-making, and problem-solving frameworks in complex domains. In his seminal work, Dung (1995) introduced Abstract Argumentation Frameworks (*AAF*s), defining an *AAF* as a set of arguments and an attack relation between them. In *AAF*s, an argument represents a claim whose acceptability is determined by specific criteria (referred to as semantics), depending on how arguments interact. Each argument is treated as an abstract entity without any internal structure, meaning semantics can only consider the interactions between arguments. On the other hand, Normal Logic Programs (*NLP*s) take another perspective and represent knowledge and rules in a structured, logical form, in which atoms represent claims whose evaluation determines whether they can be inferred from these rules.

Despite expressing information from distinct perspectives, argumentation and logic programming are deeply interconnected. Their connection is a very relevant line of research, as it allows one to view logic programs as argumentation frameworks and vice versa, enabling the most suitable formalism to be employed for a specific application. Besides, algorithms, techniques and semantics developed for a formalism can often be adapted to the other. Since then, many works have expanded equivalence results for other semantics and variants of argumentation and logic programming formalisms. For *AAF*s and *NLP*s, there is a correspondence between complete and partial stable semantics (Caminada, Sá, et al. 2015; Wu et al. 2009), grounded and well-founded semantics (Caminada, Sá, et al. 2015; Dung 1995), preferred and regular semantics (Caminada, Sá, et al. 2015), stable (argumentation) and stable (logic programming) semantics (Caminada, Sá, et al. 2015; Dung 1995), but no correspondence was found between semi-stable and L-stable semantics (Caminada, Sá, et al. 2015).

*Corresponding Author.

Authors' Contact Information: João Alcântara, ORCID: 0000-0002-4297-2970, jnando@dc.ufc.br; Renan Cordeiro, ORCID: 0000-0001-9502-3413, renandcsc@alu.ufc.br, Federal University of Ceará, Fortaleza, Ceará, Brazil.

DOI: 10.1613/jair.1.18086

In this paper, we continue this line of research by instead considering bipolar argumentation frameworks (*BAF*s) (Amgoud et al. 2008), in which there is a support relation between arguments: a positive interaction independent of the attack relation. There is no consensus on how exactly supports and attacks should interact with each other, and, as a result, many semantics for *BAF*s have been proposed (e.g., Alcântara and Cordeiro 2024; Cayrol and Lagasquie-Schiex 2013; Gabbay 2016; Nouioua and Risch 2011), each with its own interpretation of the meaning of support.

Below, we outline existing links between bipolar argumentation and logic programming (though not for all semantics or not in both directions):

- According to Čyras et al. (2017), *BAF*s can be translated into Assumption-Based Argumentation (*ABA*) frameworks for four interpretations of support (Cayrol and Lagasquie-Schiex 2013; Gabbay 2016; Nouioua and Risch 2011), whilst preserving the admissible, preferred and stable semantics. As there is a semantic correspondence between *ABA* frameworks and *NLP*s in both directions (Caminada and Schulz 2017), we can try to link *BAF*s and *NLP*s for the admissible, preferred and stable semantics in one direction: any *BAF* can be translated into an *ABA* framework, which in turn corresponds to some *NLP*. However, the inverse direction from *NLP*s to *BAF*s and the correspondence between the semi-stable and L-stable semantics remain unclear, as Čyras et al. (2017) study only the direction from *BAF*s to *ABA* frameworks and consider only the admissible, preferred, and stable semantics.
- Rather than composing transformations, Nouioua and Boutouhami (2023) present a direct link between *BAF*s and *NLP*s by providing a one-to-one semantic correspondence from *BAF*s to *NLP*s, and vice versa, between the main acceptability semantics and logic programming semantics, with a notable exception: in general, the L-stable semantics of an *NLP* does not correspond to the semi-stable semantics of its corresponding *BAF*.
- In the context of recursive argumentation, where attacks/supports can target attacks/supports, Alfano et al. (2024) map recursive *BAF*s (Rec-*BAF*) to *NLP*s, with complete extensions corresponding to partial stable models. They also consider non-recursive *BAF*s, but do not study in neither case the semi-stable/L-stable semantics nor the inverse direction: as any *AAF* is a (recursive) *BAF*, Caminada, Sá, et al.'s (2015) translation from *NLP*s to *AAF*s is already sufficient for semantic correspondence from *NLP*s to (recursive) *BAF*s when not considering the semi-stable/L-stable semantics.

In summary, the semi-stable/L-stable semantics remain a primary gap in the connections between *NLP* semantics and these semantics for *BAF*. Additionally, these works have not explored whether it is possible to find a link between *BAF*s and *NLP*s such that the translation from *BAF*s to *NLP*s acts as an inverse of the translation from *NLP*s to *BAF*s (when ignoring names of arguments/atoms). This motivates our focus on the following questions:

- Can *NLP*s be translated to *BAF*s with a one-to-one semantic correspondence (including one semantics for *BAF* corresponding to L-stable)?
- To what extent are the translations from *NLP*s to *BAF*s and from *BAF*s to *NLP*s the inverse of each other?

Among these semantics for *BAF*s, we choose the β-semantics introduced by Alcântara and Cordeiro (2024). A remarkable characteristic of them is the dual criteria for acceptance and rejection of arguments. An argument A is accepted by a set of arguments $\mathcal{S}$ if at least a direct or indirect supporter of A (notice A is a supporter of itself) is not attacked by any argument in $\mathcal{S}$; on the other hand, A is rejected by $\mathcal{S}$ if every direct or indirect supporter of A is attacked by at least one argument in $\mathcal{S}$. Then the authors generalise to *BAF*s basic notions defined for *AAF*s, such as conflict-freeness, acceptable arguments, and admissible sets of arguments. These new definitions lead naturally to these new β-semantics (namely β-admissible, β-complete, β-grounded, β-preferred, β-stable, and β-semi-stable), generalising the corresponding semantics from Dung's framework to *BAF*s.

Taking profit from this dual perspective, we establish in this work a one-to-one correspondence between β-semantics and various logic programming semantics, including the challenging L-stable semantics. We recall

that for traditional attack-only *AAF*s, the translation proposed by Caminada, Sá, et al. (2015) from *NLP*s to *AAF*s does not guarantee semantic correspondence between (argumentation) semi-stable and (logic programming) L-stable semantics. To prove this semantic equivalence for *BAF*s, we provide direct translations between *BAF*s and *NLP*s (and their semantics) in both directions. We also show that the relationship between *NLP*s and *BAF*s is demonstrably more robust than between *NLP*s and *AAF*s, extending beyond semantics to encompass structural aspects. For example, we find classes of *BAF*s and *NLP*s, respectively Redundancy-Free *BAF*s of Support cliques ($\mathfrak{S}$-*RFBAF*s) and Reduced Atomic Logic Programs (*RALP*s), for which these transformations act as inverses (up to isomorphism) of each other, meaning that each transformation undoes the other if we ignore the names of arguments and atoms. This elicits a notion of structural (or syntactic) equivalence, as no relevant information is lost by translating a formalism into the other. In addition, we find that the class of *NLP*s that are structurally equivalent to *BAF*s is as expressive as the set of all *NLP*s, meaning that any *NLP* is semantically equivalent to (i.e., as expressive as) the corresponding *NLP* of some *BAF*.

As a collateral effect of this semantic and structural equivalence, our results reveal that the β-semantics interpretation of support aligns with *NLP* semantics, offering new insights for knowledge representation and reasoning with *BAF*s.

Several works also relate more expressive argumentation formalisms to logic programming. For instance, Abstract Dialectical Frameworks (*ADF*s) (Alcântara, Sá, et al. 2019; Brewka and Woltran 2010; Strass 2013), Assumption-Based Argumentation (*ABA*) (Caminada and Schulz 2017), Claim-augmented Argumentation Frameworks (*CAF*s) (Rocha and Cozman 2022; Toni et al. 2022), Frameworks with Sets of attacking arguments (*SETAF*s) (Alcântara, Cordeiro, and Sá 2024; Toni et al. 2022) and many others (Alfano et al. 2020) are connected to logic programming. Many of these approaches lack some result that we correspondingly establish between *BAF*s and *NLP*s: the semi-stable semantics of an *ABA* framework and the L-stable semantics of its corresponding *NLP* do not coincide according to Caminada and Schulz's (2017) translation of *ABA* frameworks into *NLP*s, and the approaches from Brewka and Woltran (2010); Strass (2013) fail to express in *ADF*s the partial stable semantics of *NLP*s. Although this is enabled by Alcântara, Sá, et al.'s (2019) approach, it lacks a structural correspondence, as the translations from *ADF*s to *NLP*s, and vice versa, are not each other's inverse, not even up to isomorphism. Alcântara, Cordeiro, and Sá (2024) find a one-to-one correspondence involving *SETAF*s and *NLP*s for the main semantics, including the L-stable semantics. Similarly to our work, there is also a structural equivalence between these formalisms for a specific class of *NLP*s in which the proposed translations are each other's inverse. However, the structural correspondence we find between *BAF*s and *NLP*s is slightly weaker than that between *SETAF*s and *NLP*s, as our translations act as inverses with respect to an isomorphism relation instead of the equality relation, i.e., the names of arguments and atoms are not preserved when translating from *BAF*s to *NLP*s or vice versa.

The work most closely related to ours is that of Rocha and Cozman (2022), where the authors relate Bipolar Claim-augmented Argumentation Frameworks (*BCAF*s) to *NLP*s. They establish a one-to-one correspondence between *BCAF*s and *NLP*s in both directions, while preserving the main semantics, including the L-stable semantics. The main distinction between *BCAF*s and *BAF*s (and hence between their work and ours) lies in the additional structure of *BCAF*s: each argument in a *BCAF* is associated with an explicit claim (or conclusion). In contrast, arguments in a *BAF* are completely abstract, with the β-semantics being dependent exclusively on the attack and support relation between arguments, treated as objects without any internal structure. In this sense, to the best of our knowledge, ours is the first work to provide translations from *BAF*s to *NLP*s, and vice versa, preserving the semantics based on partial stable models, including the L-stable semantics. Unlike Rocha and Cozman's (2022) work, we also find classes of *NLP*s and *BAF*s for which the formalisms are structurally equivalent.

This paper extends our previous work (Cordeiro and Alcântara 2025), in which we first revealed the semantic equivalence between *BAF*s and *NLP*s. In particular, we now explore this equivalence also from a syntactic perspective, by showing that the proposed translations from *BAF*s to *NLP*s and from *NLP*s to *BAF*s are inverses of each other (up to isomorphism) for some specific class of *BAF*s and *NLP*s: respectively, Redundancy-Free

*BAF*s of Support cliques ($\mathfrak{S}$-*RFBAF*) and Reduced Atomic Logic Programs (*RALP*s). In addition, we compare the expressiveness between *RALP*s and *NLP*s and find that they are equally expressive under the semantics considered in this paper. Furthermore, we include full proofs in Appendix A for both the new results and those whose proofs were omitted in the conference paper.

This work is organised as follows: in Section 2, we briefly review the foundations of Bipolar Argumentation Frameworks (*BAF*s) and their β-semantics, and of Normal Logic Programs (*NLP*s) and their 3-valued semantics; following in Section 3, we provide a translation from *NLP*s to *BAF*s and between their semantics; the inverse direction from *BAF*s to *NLP*s is discussed in Section 4; we introduce in Section 5 the classes of Reduced Atomic Logic Programs (*RALP*s) and Redundancy-Free *BAF*s of Support cliques ($\mathfrak{S}$-*RFBAF*) for which the proposed translations act as each other's inverse up to isomorphism; we relate *NLP*s to *RALP*s in Section 6 and show they are equally expressive for the semantics considered in this paper; we collect our conclusions in Section 7.

2 Preliminaries

This section reviews the core formalisms underlying our work: Bipolar Argumentation Frameworks (*BAF*s), Normal Logic Programs (*NLP*s), and their associated semantics.

2.1 Bipolar Argumentation Framework (BAF)

Dung's (1995) Abstract Argumentation Frameworks (*AAF*s) were extended by Amgoud et al. (2008) to incorporate an explicit notion of support between arguments, independent of the attack relation. The resulting framework, called *BAF*, is defined next:

Definition 1 (*BAF*, Amgoud et al. 2008)**.** A Bipolar Argumentation Framework (*BAF*) is a tuple $(\mathcal{A}, Att, Sup)$, in which $\mathcal{A}$ is a set of arguments, $Att \subseteq \mathcal{A} \times \mathcal{A}$ is the attack relation and $Sup \subseteq \mathcal{A} \times \mathcal{A}$ is the support relation. For an argument $A \in \mathcal{A}$, we define $Sup(A) = \{B \in \mathcal{A} \mid (B, A) \in Sup\}$ as the set of direct supporters of A. Alternatively, we can depict a *BAF* as a direct graph, where the nodes represent the arguments, the dashed directed edges represent the notion of support, and the solid directed edges represent the notion of attack.

In *AAF*s, the interaction between arguments is specified only by the attack relation. In *BAF*s, the novelty is that arguments can also support other arguments. This means that *BAF*s $(\mathcal{A}, Att, Sup)$ with $Sup = \emptyset$ amount to (standard Dung) *AAF*s.

We are interested not only in the direct supporters of an argument but also in the indirect ones. They are indistinctly called the supporters of an argument:

Definition 2 (Supporters, Alcântara and Cordeiro 2024)**.** Let $\mathcal{B} = (\mathcal{A}, Att, Sup)$ be a *BAF* and $A \in \mathcal{A}$ an argument. We define the supporters of A recursively as follows:

- A is a supporter of A in $\mathcal{B}$;
- If A' is a supporter of A in $\mathcal{B}$ and $(B, A') \in Sup$, then B is a supporter of A in $\mathcal{B}$.

By $\mathfrak{Sup}(A) = \{A' \in \mathcal{A} \mid A' \textit{ is a supporter of } A \textit{ in } \mathcal{B}\}$, we denote the set of all supporters of A in $\mathcal{B}$. Additionally, we denote the relation of being a supporter by $\mathfrak{Sup} = \{(A, B) \mid A \in \mathfrak{Sup}(B)\}$.

Semantics for *BAF*s can be equivalently characterised in terms of extensions or labellings (Alcântara and Cordeiro 2024). We will focus on labelling-based semantics as it allows a more direct comparison to semantics for logic programs.

Definition 3 (Labelling)**.** Let $\mathcal{B} = (\mathcal{A}, Att, Sup)$ be a *BAF*. A labelling (of $\mathcal{B}$) is a total function $\mathcal{L} : \mathcal{A} \to \{\mathtt{in}, \mathtt{out}, \mathtt{undec}\}$.

When $\mathcal{L}$ is a labelling, we write $\mathtt{in}(\mathcal{L})$ to denote the set $\{A \mid \mathcal{L}(A) = \mathtt{in}\}$, $\mathtt{out}(\mathcal{L})$ for $\{A \mid \mathcal{L}(A) = \mathtt{out}\}$ and $\mathtt{undec}(\mathcal{L})$ for $\{A \mid \mathcal{L}(A) = \mathtt{undec}\}$. As a labelling defines a partition among arguments, when convenient

we write $\mathcal{L}$ as the triple $(\texttt{in}(\mathcal{L}), \texttt{out}(\mathcal{L}), \texttt{undec}(\mathcal{L}))$. Intuitively, the label in indicates acceptance, the label out indicates rejection and the label undec indicates that the argument is undecided, i.e., neither accepted nor rejected.

There is no consensus on how exactly supports and attacks should interact with each other and, as a result, many semantics for *BAF*s have been proposed (e.g., Alcântara and Cordeiro 2024; Cayrol and Lagasquie-Schiex 2013; Gabbay 2016; Nouioua and Risch 2011), each with its own interpretation of the meaning of support. We employ the β-semantics from Alcântara and Cordeiro (2024) to ensure that arguments supporting each other share the same acceptability degree, although many alternative semantics are available. This approach helps guarantee the correspondence with the L-stable semantics. The β-semantics are described as follows:

Definition 4 (Labelling-based β-semantics for *BAF*s, Alcântara and Cordeiro 2024)**.** Let $\mathcal{B} = (\mathcal{A}, Att, Sup)$ be a *BAF*. A labelling $\mathcal{L}$ is a β-complete labelling of $\mathcal{B}$ if for any $A \in \mathcal{A}$,

- $\mathcal{L}(A) = \texttt{in}$ if and only if there exists $A' \in \mathfrak{Sup}(A)$ such that for every $B \in Att(A')$, it holds $\mathcal{L}(B) = \texttt{out}$.
- $\mathcal{L}(A) = \texttt{out}$ if and only if for every $A' \in \mathfrak{Sup}(A)$, there exists $B \in Att(A')$ such that $\mathcal{L}(B) = \texttt{in}$.

Additionally, we say a β-complete labelling $\mathcal{L}$ of $\mathcal{B}$ is

- β-grounded if $\texttt{in}(\mathcal{L})$ is minimal (w. r. t. $\subseteq$) among the β-complete labellings of $\mathcal{B}$.
- β-preferred if $\texttt{in}(\mathcal{L})$ is maximal (w. r. t. $\subseteq$) among the β-complete labellings of $\mathcal{B}$.
- β-stable if $\texttt{undec}(\mathcal{L}) = \emptyset$.
- β-semi-stable if $\texttt{undec}(\mathcal{L})$ is minimal (w. r. t. $\subseteq$) among the β-complete labellings of $\mathcal{B}$.

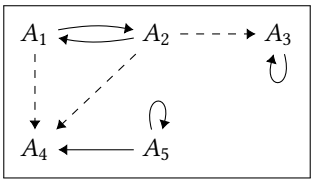


Fig. 1. *BAF*.

The β-complete, β-grounded, β-preferred, β-stable, and β-semi-stable labellings for *BAF*s are generalisations, respectively, of the complete, grounded, preferred, stable, and semi-stable labellings defined for *AAF*s. Labelling-based β-semantics for *BAF*s also preserve remarkable properties of the corresponding labelling-based semantics for *AAF*s: for every *BAF* there is always a uniquely defined β-complete labelling; every β-stable labelling is a β-preferred labelling; and if there exist β-stable labellings, they coincide with β-semi-stable and β-preferred labellings.

Example 1. Recall that a labelling $\mathcal{L}$ can be represented by the triple $(\texttt{in}(\mathcal{L}), \texttt{out}(\mathcal{L}), \texttt{undec}(\mathcal{L}))$. Considering the *BAF* in Fig. 1, we obtain the following labelling-based semantics:

- β-complete labellings: $(\emptyset, \emptyset, \mathcal{A})$,
  $(\{A_1, A_4\}, \{A_2\}, \{A_3, A_5\})$ and
  $(\{A_2, A_3, A_4\}, \{A_1\}, \{A_5\})$.
- β-grounded labelling: $(\emptyset, \emptyset, \mathcal{A})$.
- β-preferred labellings: $(\{A_1, A_4\}, \{A_2\}, \{A_3, A_5\})$ and
  $(\{A_2, A_3, A_4\}, \{A_1\}, \{A_5\})$.
- β-stable labellings: None.

- β-semi-stable labelling: $(\{A_2, A_3, A_4\}, \{A_1\}, \{A_5\})$.

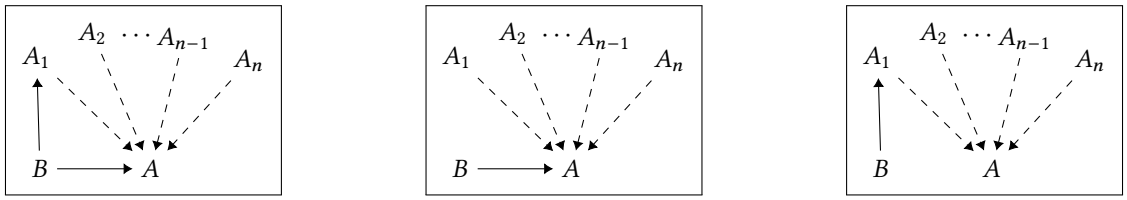
A notable property of β-complete labellings is

Proposition 1. (Alcântara and Cordeiro 2024) Let $\mathcal{B} = (\mathcal{A}, Att, Sup)$ be a *BAF* and $\mathcal{L}$ be a β-complete labelling of $\mathcal{B}$. Then for any $A \in \mathcal{A}$,

- $\mathcal{L}(A) = \texttt{in}$ iff $\mathcal{L}(A') = \texttt{in}$ for every $A' \in \mathcal{A}$ supported by A ($A \in \mathfrak{Sup}(A')$);
- $\mathcal{L}(A) = \texttt{out}$ iff $\mathcal{L}(A') = \texttt{out}$ for every $A' \in \mathcal{A}$ supporting A ($A' \in \mathfrak{Sup}(A)$).

Proposition 29 shows the dual nature of the support relation when determining whether an argument is `in` or `out` in a β-complete labelling. It also shows that β-complete labellings follow a deductive interpretation of support (if an argument A is `in`, every argument supported by A is also `in`).

Another fundamental aspect to discuss is the intuitive meaning of the support relation. In the sequel, we will exploit the three scenarios depicted in Fig. 2. We intend to motivate that depending on the kind of interaction between supports and attacks, the support relation can play a distinct role.



(a) $\mathcal{B}_1$ (b) $\mathcal{B}_2$ (c) $\mathcal{B}_3$

Fig. 2. Intuitive meaning of the support relation.

In the three *BAF*s of Fig. 2, assume that arguments $A_1, A_2, \ldots, A_{n-1}$ and A_n are interpreted as a deductive support for A. In $\mathcal{B}_1$ (Figure 2a), argument B attacks both A and A_1; in $\mathcal{B}_2$ (Figure 2b), argument B attacks only A; in $\mathcal{B}_3$ (Fig. 2c), argument B attacks only A_1. Let us now exploit these three scenarios with concrete examples and possible interpretations of the support relation:

2.1.1 Scenario I. One possibility for the first scenario (Fig. 2a) is that the rejection of A_1 does not contribute to the acceptance of any $A_i \in \{A_2, \ldots, A_n\}$. When B attacks A_1, it attacks a pillar sustaining A and brings evidence against the acceptance of A; in other words, it also attacks A. Such an attack can be interpreted as a secondary attack on A without necessarily rejecting it, as other arguments may support A. This is the scenario of Example 2, adapted from Cohen et al. (2014):

Example 2. Consider that two soccer teams, Liverpool and Manchester United, are in the final race to win the Premier League. Suppose that Liverpool wins the Premiere League ($\mathcal{LPL}$) if it wins its last match ($\mathcal{LW}$) or Manchester United does not win its own ($\mathcal{MUNW}$). Note that Liverpool and Manchester United are not playing against each other; thus, their matches' results are independent. Suppose now that Liverpool loses its last match ($\mathcal{LL}$) and Manchester United does not win its own. Then, according to the interpretation given to support in the β-complete semantics, we have that both $\mathcal{LW}$ and $\mathcal{MUNW}$ support $\mathcal{LPL}$, as accepting $\mathcal{LW}$ (or $\mathcal{MUNW}$) implies accepting $\mathcal{LPL}$. Furthermore, $\mathcal{LL}$ attacks both $\mathcal{LW}$ and $\mathcal{LPL}$, as $\mathcal{LL}$ brings evidence against them (see *BAF* $\mathcal{B}_4$ in Fig. 3).

$\mathcal{LL}$ is enough to reject $\mathcal{LW}$ according to the β-complete semantics. However, $\mathcal{LL}$ does not suffice to reject $\mathcal{LPL}$, because $\mathcal{LPL}$ is supported by another piece of evidence ($\mathcal{MUNW}$). In this case, although $\mathcal{LL}$ attacks $\mathcal{LPL}$, it is reasonable to accept $\mathcal{LL}$, $\mathcal{LPL}$ and $\mathcal{MUNW}$.

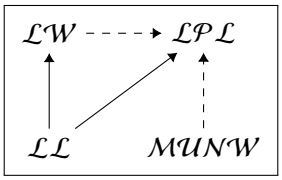


Fig. 3. $\mathcal{B}_4$

2.1.2 Scenario II. In the second scenario (Fig. 2b), we can interpret the support given by $A_1, A_2, \ldots, A_{n-1}, A_n$ to A as an exception to the rule that B is normally not A. This is the case of Example 3:

Example 3. Consider the following arguments:

- A: Tweety cannot fly;
- B: Tweety is a bird (and in general, birds can fly);
- C: Tweety is a penguin.

There is a mutual attack between A and B as each argument brings evidence against the other. Additionally, accepting C implies accepting A and B; so C supports both of them (see *BAF* $\mathcal{B}_5$ in Fig. 4).

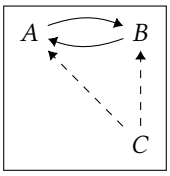


Fig. 4. $\mathcal{B}_5$

Due to the support from C, neither does accepting A lead to rejecting B, nor does accepting B lead to rejecting A (despite the mutual attack between A and B). Note that C represents an exception to the general rule that birds usually fly (and if something cannot fly, then it usually is not a bird). Thus, in the unique β-complete labelling of $\mathcal{B}_5$, A, B and C have the label `in`.

2.1.3 Scenario III. Fig. 2c depicts a case where although B attacks A_1 (a supporter of A), we cannot say that B brings evidence against A (possibly because the rejection of A_1 contributes to the acceptance of some $A_i \in \{A_2, \ldots, A_n\}$). We illustrate this scenario with the following example: suppose that Liverpool will finish in second place in the Premier League ($\mathcal{L}2\mathcal{P}\mathcal{L}$) if $\mathcal{MWCL}$ or $\mathcal{MLCW}$, where $\mathcal{MWCL}$ is the argument Manchester United wins its last match and Chelsea does not win its last match, and $\mathcal{MLCW}$ is the argument Manchester United does not win its last match and Chelsea wins its last match. Assume that Manchester United loses its last match ($\mathcal{ML}$). We have that $\mathcal{MWCL}$ and $\mathcal{MLCW}$ attack each other, and also support $\mathcal{L}2\mathcal{P}\mathcal{L}$. In addition, $\mathcal{ML}$ attacks $\mathcal{MWCL}$. However, $\mathcal{ML}$ does not bring evidence against $\mathcal{L}2\mathcal{P}\mathcal{L}$, as $\mathcal{ML}$ also contributes favourably with $\mathcal{MLCW}$, which is a supporter of $\mathcal{L}2\mathcal{P}\mathcal{L}$. Because of this dubious behaviour, $\mathcal{ML}$ attacks $\mathcal{MWCL}$, but it does not attack $\mathcal{L}2\mathcal{P}\mathcal{L}$. The resulting *BAF* is displayed in Fig. 5.

2.2 Logic Programs and Semantics

Now, we take a look at Propositional Normal Logic Programs. To delve into their definition and semantics, we will follow the presentation outlined by Caminada, Sá, et al. (2015), which draws from the foundation laid out by Przymusinski (1990).

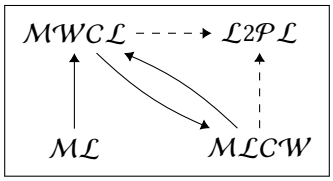


Fig. 5. A new scenario for the Premier League example.

Definition 5. A *rule* r is an expression

$$r : c \leftarrow a_1, \ldots, a_m, \texttt{not}\ b_1, \ldots, \texttt{not}\ b_n, \tag{1}$$

where $(m, n \geq 0)$; c, each a_i $(1 \leq i \leq m)$ and each b_j $(1 \leq j \leq n)$ are atoms, and not represents negation as failure. A literal is either an atom a (positive literal) or a negated atom not a (negative literal). Given a rule r as above, c is called the *head* of r, which we denote as $head(r)$, and $body(r) = \{a_1, \ldots, a_m, \texttt{not}\ b_1, \ldots, \texttt{not}\ b_n\}$ is called the *body* of r. Further, we divide $body(r)$ in two sets $body^+(r) = \{a_1, \ldots, a_m\}$ and $body^-(r) = \{\texttt{not}\ b_1, \ldots, \texttt{not}\ b_n\}$. A *fact* is a rule where $m = n = 0$. A Normal Logic Program (*NLP*) or simply a *logic program* P is a finite set of rules. If every $r \in P$ has $body^-(r) = \emptyset$, P is a positive program. The *Herbrand Base* of P is the set HB_P of all atoms appearing in P.

A wide range of *NLP* semantics are based on the 3-valued interpretations of programs (Przymusinski 1990):

Definition 6 (3-Valued Herbrand Interpretation, Przymusinski 1990)**.** A 3-valued Herbrand Interpretation $\mathcal{I}$ (or simply interpretation) of an *NLP* P is a total function $\mathcal{I} : HB_P \rightarrow \{\mathbf{t}, \mathbf{f}, \mathbf{u}\}$.

Given an interpretation $\mathcal{I}$ of an *NLP* P, we write $\mathbf{t}(\mathcal{I})$ to denote the set $\{a \in HB_P \mid \mathcal{I}(a) = \mathbf{t}\}$, $\mathbf{f}(\mathcal{I})$ for $\{a \in HB_P \mid \mathcal{I}(a) = \mathbf{f}\}$ and $\mathbf{u}(\mathcal{I})$ for $\{a \in HB_P \mid \mathcal{I}(a) = \mathbf{u}\}$. As an interpretation defines a partition among atoms in HB_P, when convenient we write $\mathcal{I}$ as the triple $(\mathbf{t}(\mathcal{I}), \mathbf{f}(\mathcal{I}), \mathbf{u}(\mathcal{I}))$. Intuitively, the value $\mathbf{t}$ indicates the atom is true in $\mathcal{I}$, the value $\mathbf{f}$ indicates the atom is false in $\mathcal{I}$ and the value $\mathbf{u}$ indicates the atom is undefined in $\mathcal{I}$, i.e., neither true nor false.

Now we will consider the main semantics for *NLP*s. Let $\mathcal{I}$ be a 3-valued Herbrand interpretation of an *NLP* P; the reduct of P with respect to $\mathcal{I}$ (written as $P/\mathcal{I}$) is the *NLP* constructed using the following steps:

(1) Remove any $a \leftarrow a_1, \ldots, a_m, \texttt{not}\ b_1, \ldots, \texttt{not}\ b_n \in P$ such that $\mathcal{I}(b_j) = \mathbf{t}$ for some j $(1 \leq j \leq n)$;
(2) Afterwards, remove any occurrence of not b_j from P such that $\mathcal{I}(b_j) = \mathbf{f}$.
(3) Then, replace any occurrence of not b_j left by a special atom $\mathbf{u}$ $(\mathbf{u} \notin HB_P)$.

In the above procedure, $\mathbf{u}$ is assumed to be an atom not in HB_P which is undefined in all interpretations of P (a constant). Note that $P/\mathcal{I}$ is a positive program since all negative literals have been removed. As a consequence, $P/\mathcal{I}$ has a unique least 3-valued model (Przymusinski 1990) denoted as $\Psi_P(\mathcal{I})$[1] with minimal $\mathbf{t}(\Psi_P(\mathcal{I}))$ and maximal $\mathbf{f}(\Psi_P(\mathcal{I}))$ (w.r.t. set inclusion) such that for every $a \in HB_P$,

- $a \in \mathbf{t}(\Psi_P(\mathcal{I}))$ if there is a rule $r' \in P/\mathcal{I}$ with $head(r') = a$ and $body^+(r') \subseteq \mathbf{t}(\Psi_P(\mathcal{I}))$;
- $a \in \mathbf{f}(\Psi_P(\mathcal{I}))$ if every rule $r' \in P/\mathcal{I}$ with $head(r') = a$ has $body^+(r') \cap \mathbf{f}(\Psi_P(\mathcal{I})) \neq \emptyset$;

We can now describe the logic programming semantics studied in this paper:

Definition 7. Let P be an *NLP* and $\mathcal{I}$ be an interpretation:

- $\mathcal{I}$ is a partial stable model of P iff $\Psi_P(\mathcal{I}) = \mathcal{I}$ (Przymusinski 1990).

[1]The above definition consists of a least fix-point of the immediate consequence operator Ψ defined by Przymusinski (1990), which is guaranteed to exist and be unique for positive programs.

- $\mathcal{I}$ is a well-founded model of P iff $\mathcal{I}$ is a partial stable model of P where there is no partial stable model $\mathcal{I}'$ of P such that $\mathbf{t}(\mathcal{I}') \subset \mathbf{t}(\mathcal{I})$, i.e., $\mathbf{t}(\mathcal{I})$ is minimal (w.r.t. set inclusion) among all partial stable models of P (Przymusinski 1990).
- $\mathcal{I}$ is a regular model of P iff $\mathcal{I}$ is a partial stable model of P where there is no partial stable model $\mathcal{I}'$ of P such that $\mathbf{t}(\mathcal{I}) \subset \mathbf{t}(\mathcal{I}')$, i.e., $\mathbf{t}(\mathcal{I})$ is maximal (w.r.t. set inclusion) among all partial stable models of P (Sacca 1997).
- $\mathcal{I}$ is a (2-valued) stable model of P iff $\mathcal{I}$ is a partial stable model of P where $\mathbf{u}(\mathcal{I}) = \emptyset$ (Przymusinski 1990).
- $\mathcal{I}$ is an L-stable model of P iff $\mathcal{I}$ is a partial stable model of P where there is no partial stable model $\mathcal{I}'$ of P such that $\mathbf{u}(\mathcal{I}') \subset \mathbf{u}(\mathcal{I})$, i.e., $\mathbf{u}(\mathcal{I})$ is minimal (w.r.t. set inclusion) among all partial stable models of P (Sacca 1997).

Although some of these definitions are not standard in logic programming literature, their equivalence is proved by Caminada, Sá, et al. (2015). This format helps us to relate *NLP* and *BAF* semantics due to the structural similarities between Definition 7 and Definitions 3 and 4. We illustrate these semantics in the following example:

Example 4. Consider the following logic program P:

$$\begin{array}{llll} r_0: & d \leftarrow \texttt{not}\ d & r_1: & c \leftarrow \texttt{not}\ c, \texttt{not}\ d \\ r_2: & a \leftarrow \texttt{not}\ b & r_3: & b \leftarrow \texttt{not}\ a \\ r_4: & c \leftarrow \texttt{not}\ c, \texttt{not}\ a & r_5: & e \leftarrow \texttt{not}\ e, \texttt{not}\ b. \end{array}$$

Recall that interpretation $\mathcal{I}$ can be represented by the triple $(\mathbf{t}(\mathcal{I}), \mathbf{f}(\mathcal{I}), \mathbf{u}(\mathcal{I}))$. This program has

- Partial Stable Models: $\mathcal{M}_1 = (\emptyset, \emptyset, HB_P)$, $\mathcal{M}_2 = (\{a\}, \{b\}, \{c, d, e\})$ and $\mathcal{M}_3 = (\{b\}, \{a, e\}, \{c, d\})$;
- Well-founded model: $\mathcal{M}_1 = (\emptyset, \emptyset, HB_P)$;
- Regular models: $\mathcal{M}_2 = (\{a\}, \{b\}, \{c, d, e\})$ and $\mathcal{M}_3 = (\{b\}, \{a, e\}, \{c, d\})$;
- Stable models: none;
- L-stable model: $\mathcal{M}_3 = (\{b\}, \{a, e\}, \{c, d\})$.

3 From NLP to BAF

In this section, we revisit the three-step process of establishing the argumentation framework used by Caminada, Sá, et al. (2015) to translate a *NLP* into an *AAF*. This method is based on the approach introduced by Wu et al. (2009) and shares similarities with the procedures used in ASPIC (Caminada and Amgoud 2005; Caminada and Amgoud 2007) and logic-based argumentation (Gorogiannis and Hunter 2011). Its first step involves taking an *NLP* and constructing its associated *AAF*. Then we apply *AAF* semantics in the second step, followed by an analysis of the implications of these semantics at the level of conclusions (step 3). The novelty of our approach is that we include a support relation between arguments with the same conclusion in the first step. We now detail this process.

3.1 BAF Construction

We will present a translation from *NLP* to *BAF* sufficiently robust to establish a correspondence between partial stable models and β-complete labellings, well-founded models and β-grounded labellings, regular models and β-preferred labellings, stable models and β-stable labellings, L-stable models and β-semi-stable labellings. Taking an *NLP* P, we can start to construct arguments recursively as follows:

Definition 8. (Caminada, Sá, et al. 2015) Let P be an *NLP*.

- If $c \leftarrow \texttt{not}\ b_1, \ldots, \texttt{not}\ b_m$ is a rule in P, it is also an argument (say A) with
  - $\texttt{Conc}(A) = c$,
  - $\texttt{Rules}(A) = \{c \leftarrow \texttt{not}\ b_1, \ldots, \texttt{not}\ b_m\}$,

  - $\texttt{Vul}(A) = \{b_1, \ldots, b_m\}$, and
  - $\texttt{Sub}(A) = \{A\}$.
- If $c \leftarrow a_1, \ldots, a_n, \texttt{not}\ b_1, \ldots, \texttt{not}\ b_m$ is a rule (say r) in P and for each a_i ($1 \leq i \leq n$) there exists an argument A_i with $\texttt{Conc}(A_i) = a_i$ and $c \leftarrow a_1, \ldots, a_n, \texttt{not}\ b_1, \ldots, \texttt{not}\ b_m$ is not contained in $\texttt{Rules}(A_i)$, then r derives the argument $c \leftarrow (A_1), \ldots, (A_n), \texttt{not}\ b_1, \ldots, \texttt{not}\ b_m$ (say A) with
  - $\texttt{Conc}(A) = c$,
  - $\texttt{Rules}(A) = \texttt{Rules}(A_1) \cup \ldots \cup \texttt{Rules}(A_n) \cup \{c \leftarrow a_1, \ldots, a_n, \texttt{not}\ b_1, \ldots, \texttt{not}\ b_m\}$,
  - $\texttt{Vul}(A) = \texttt{Vul}(A_1) \cup \ldots \cup \texttt{Vul}(A_n) \cup \{b_1, \ldots, b_m\}$, and
  - $\texttt{Sub}(A) = \{A\} \cup \texttt{Sub}(A_1) \cup \ldots \cup \texttt{Sub}(A_n)$.

By $\mathcal{A}_P$ we mean the set of all arguments constructed from P as above. Each argument A represents a derivation in P for the atom $\texttt{Conc}(A)$. We explain the intuition behind some of the definitions above:

- $\texttt{Rules}(A)$ is the set of all rules used for deriving $\texttt{Conc}(A)$. For instance, for a program with three rules $r_1 = c \leftarrow a, \texttt{not}\ b$ and $r_2 = a \leftarrow \texttt{not}\ a$ and $r_3 = a \leftarrow \texttt{not}\ c$, we find an argument C_1 with $\texttt{Conc}(C_1) = c$ and $\texttt{Rules}(C_1) = \{r_1, r_2\}$, and also an argument C_2 with $\texttt{Conc}(C_2) = c$ and $\texttt{Rules}(C_2) = \{r_1, r_3\}$. The latter used r_3 instead of r_2 for deriving c. The main purpose of $\texttt{Rules}(\cdot)$ is keeping track of which rules were already used, so that no argument can use a same rule twice for deriving some conclusion. For example, for a program with two rules $r'_1 = c \leftarrow a, \texttt{not}\ b$ and $r'_2 = a \leftarrow a, \texttt{not}\ d$, we cannot build an argument A with $\texttt{Conc}(A) = a$, as using the rule $a \leftarrow a, \texttt{not}\ d$ requires the use of some *other* rule for a.
- $\texttt{Vul}(A)$ is the set of vulnerabilities of the argument/derivation A. In any partial stable model $\mathcal{M}$ of P, a rule r is effectively ignored if there is some $v \in \mathbf{t}(\mathcal{M})$ such that $v \in body^-(r)$. Thus, a vulnerability of A is an atom that appears in the negative body of some rule in the derivation. Vulnerabilities of an argument A will be used to determine the attackers of A.
- $\texttt{Sub}(A)$ is the set of subarguments/subderivations used in an argument/derivation A. For each positive body atom a in a rule deriving argument C with $\texttt{Conc}(C) = c$, we require some argument A with $\texttt{Conc}(A) = a$ to be a subargument of C. Stated differently, if viewing a derivation A as a tree, the (proper) subarguments of A would be the (proper) subtrees of A. While the notion of subarguments is not used for defining corresponding *BAF*s, it is a useful concept when proving some theorems or talking about derivations.

Now we will clarify the connection between the existence of arguments and the existence of a derivation in a reduct.

Lemma 2. (Alcântara, Cordeiro, and Sá 2024) Let P be an *NLP*, $\mathcal{I}$ an interpretation and $\Psi_P(\mathcal{I})$ the least 3-valued model of $P/\mathcal{I}$. It holds

(i) $c \in \mathbf{t}(\Psi_P(\mathcal{I}))$ iff there exists an argument A constructed from P such that $\texttt{Conc}(A) = c$ and $\texttt{Vul}(A) \subseteq \mathbf{f}(\mathcal{I})$.
(ii) $c \in \mathbf{f}(\Psi_P(\mathcal{I}))$ iff for each argument A constructed from P such that $\texttt{Conc}(A) = c$, we have $\texttt{Vul}(A) \cap \mathbf{t}(\mathcal{I}) \neq \emptyset$.

Lemma 2 ensures that arguments are closely related to derivations in a reduct. An atom c is true in the least 3-valued model of $P/\mathcal{I}$ iff we can construct an argument with conclusion c and whose vulnerabilities are false according to $\mathcal{I}$; otherwise, c is false in the least 3-valued model of $P/\mathcal{I}$ iff for every argument whose conclusion is c, at least one of its vulnerabilities is true in $\mathcal{I}$. The next result is a direct consequence of Lemma 2:

Corollary 3. (Alcântara, Cordeiro, and Sá 2024) Let P be an *NLP* and $c \in HB_P$.

- Assume $\mathcal{I} = (\emptyset, HB_P, \emptyset)$. We have $c \in \mathbf{t}(\Psi_P(\mathcal{I}))$ iff there exists an argument A constructed from P such that $\texttt{Conc}(A) = c$.
- There is no argument A constructed from P such that $\texttt{Conc}(A) = c$ iff $c \in \mathbf{f}(\Psi_P(\mathcal{I}))$ for every interpretation $\mathcal{I}$.

The reduct of P with respect to $(\emptyset, HB_P, \emptyset)$ gives us all the possible derivations of P, and from these derivations, we can construct all the arguments associated with P. On the other hand, the atoms that are lost in the translation, i.e., the atoms not associated with arguments are simply those that are false in the least 3-valued model of every possible reduct of P. Besides establishing connections between arguments and derivations in a reduct, Lemma 2 also plays a central role in the proof of Theorems 6 and 8.

Next, we exemplify how an argument can be seen as a tree-like structure representing a possible derivation of an atom from the rules of a program.

Example 5. Consider the *NLP* P below with rules $\{r_1, \ldots, r_8\}$:

$$\begin{array}{lll} r_1: \ a & r_2: \ b \leftarrow a & r_3: \ c \leftarrow \mathtt{not}\ c \\ r_4: \ d \leftarrow b, \mathtt{not}\ a & r_5: \ d \leftarrow \mathtt{not}\ c & r_6: \ e \leftarrow b, c, \mathtt{not}\ e \\ r_7: \ c \leftarrow f, \mathtt{not}\ g & r_8: \ f \leftarrow c, g. & \end{array}$$

According to Definition 8, we can construct the following arguments from P:

$$\begin{array}{lll} A_1: \ a & A_2: \ b \leftarrow (A_1) & A_3: \ c \leftarrow \mathtt{not}\ c \\ A_4: \ d \leftarrow (A_2), \mathtt{not}\ a & A_5: \ d \leftarrow \mathtt{not}\ c & A_6: \ e \leftarrow (A_2), (A_3), \mathtt{not}\ e. \end{array}$$

In the next table, we give the conclusions and vulnerabilities of each argument:

	A_1	A_2	A_3	A_4	A_5	A_6
$\mathtt{Conc}(.)$	a	b	c	d	d	e
$\mathtt{Vul}(.)$	$\emptyset$	$\emptyset$	$\{c\}$	$\{a\}$	$\{c\}$	$\{c, e\}$

Alternatively, we can depict arguments as possible derivations as in Fig. 6.

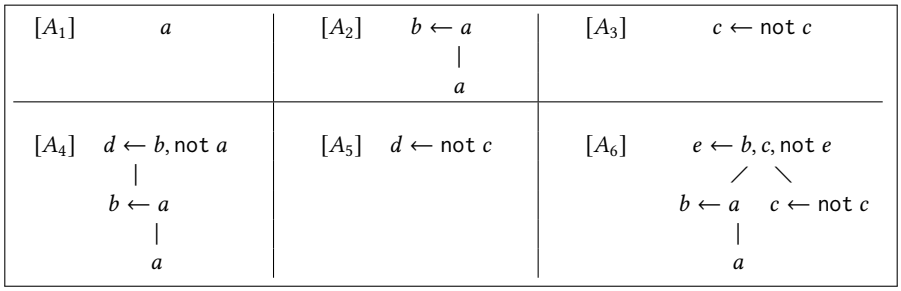


Fig. 6. Arguments constructed from P.

The vulnerabilities of an argument A are associated with the negative literals found in the derivation of A. If $\mathtt{not}\ a$ is one of them, we know that a is one of its vulnerabilities. This means that if a is derived, then $\mathtt{Conc}(A)$ cannot be obtained via this derivation represented by A. However, it can still be obtained via other derivations/arguments. For instance, in the program P of Example 5, the derivation of a suffices to prevent the derivation of d via argument A_4 (for that reason, $a \in \mathtt{Vul}(A_4)$), but we still can derive d via A_5. Notice also that there are no arguments with conclusions f and g. From Corollary 3, we know that it is not possible to derive them from P as they are false in the least three-valued model of each reduct of P.

Now we are entitled to determine both the attack and support relations:

Definition 9. Let P be an *NLP* and let A and B be arguments in the sense of Definition 8. We define the attack relation Att_P and the support relation Sup_P as follows:

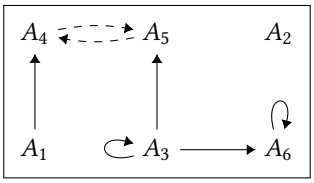


Fig. 7. The corresponding *BAF* $\mathcal{B}_P = (\mathcal{A}_P, Att_P, Sup_P)$.

- $(A, B) \in Att_P$ iff $\mathtt{Conc}(A) \in \mathtt{Vul}(B)$.
- $(A, B) \in Sup_P$ iff $A \neq B$ and $\mathtt{Conc}(A) = \mathtt{Conc}(B)$.

Our definition of the set Att_P of attacks obtained from an *NLP* P is exactly as it has been characterised by Caminada, Sá, et al. (2015). The novelty here is the definition of Sup_P, which in our proposal is determined by arguments with the same conclusion (the purpose of the condition $A \neq B$ is to avoid incorporating the redundant case of an argument supporting itself). For the arguments of Example 5, A_1 attacks A_4, A_3 attacks itself, A_5 and A_6; A_6 attacks itself. We also have A_4 and A_5 supporting each other. Using the thus-defined concepts of arguments, attacks and supports, we can define the corresponding Bipolar Argumentation Framework of a particular *NLP*:

Definition 10 (Corresponding *BAF*)**.** Let P be an *NLP*. We define its corresponding *BAF* as $\mathcal{B}_P = (\mathcal{A}_P, Att_P, Sup_P)$, where $\mathcal{A}_P$ is the set of arguments in the sense of Definition 8, and Att_P and Sup_P are respectively the attack and support relations in the sense of Definition 9.

As an example, the corresponding *BAF* $\mathcal{B}_P = (\mathcal{A}_P, Att_P, Sup_P)$ of the *NLP* of Example 5 is depicted in Figure 7.

3.2 Equivalence Results

In the sequel, we show the equivalence between the semantics of an *NLP* P and their counterpart for the corresponding *BAF* $\mathcal{B}_P$. The equivalence results are established by identifying connections between the fundamental concepts that underlie the semantics of *NLP*s and *BAF*s. For this purpose, we introduce two functions: $\mathcal{L}2\mathcal{I}_P$, which maps each labelling to an interpretation, and $\mathcal{I}2\mathcal{L}_P$, which maps each interpretation to a labelling. We then investigate the conditions under which these functions act as inverses of each other and employ these results to prove the equivalence between the semantics. Notably, these functions permit us to treat interpretations and labellings interchangeably.

Definition 11 ($\mathcal{L}2\mathcal{I}_P$ and $\mathcal{I}2\mathcal{L}_P$ Functions)**.** Let P be an *NLP*, $\mathcal{B}_P = (\mathcal{A}_P, Att_P, Sup_P)$ be its corresponding *BAF*, $\mathcal{I}nt$ be the set of all the 3-valued interpretations of P and $\mathcal{L}ab$ be the set of all labellings of $\mathcal{B}_p$. We introduce a function $\mathcal{L}2\mathcal{I}_P : \mathcal{L}ab \to \mathcal{I}nt$ such that $\mathcal{L}2\mathcal{I}_P(\mathcal{L}) = (T, F, U)$, in which

- $T = \{c \in HB_P \mid \mathcal{L}(A) = \mathtt{in}$ *for some* $A \in \mathcal{A}_P$ *such that* $\mathtt{Conc}(A) = c\}$;
- $F = \{c \in HB_P \mid \mathcal{L}(A) = \mathtt{out}$ *for every* $A \in \mathcal{A}_P$ *such that* $\mathtt{Conc}(A) = c\}$;
- $U = \{c \in HB_P \mid c \notin T \cup F\}$.

We introduce a function $\mathcal{I}2\mathcal{L}_P : \mathcal{I}nt \to \mathcal{L}ab$ such that for any $\mathcal{I} \in \mathcal{I}nt$ and any $A \in \mathcal{A}_P$,

- $\mathcal{I}2\mathcal{L}_P(\mathcal{I})(A) = \mathtt{in}$ if $\mathcal{I}(\mathtt{Conc}(A)) = \mathbf{t}$;
- $\mathcal{I}2\mathcal{L}_P(\mathcal{I})(A) = \mathtt{out}$ if $\mathcal{I}(\mathtt{Conc}(A)) = \mathbf{f}$;
- $\mathcal{I}2\mathcal{L}_P(\mathcal{I})(A) = \mathtt{undec}$ if $\mathcal{I}(\mathtt{Conc}(A)) = \mathbf{u}$.

The direction from interpretations to labellings is straightforward: the label `in`, `out` or `undec` assigned to an argument A follows respectively from the truth-value $\mathbf{t}$, $\mathbf{f}$ or $\mathbf{u}$ of its conclusion $\mathtt{Conc}(A)$. From labellings to interpretations, the connection is clear for atoms $c \in HB_P$ with some argument A with $\mathtt{Conc}(A) = c$. In this

case, c is interpreted as true if some argument A with $\mathtt{Conc}(A) = c$ is accepted; c is interpreted as false if every argument A with $\mathtt{Conc}(A) = c$ is rejected; otherwise, c is undecided. For any atom c with no corresponding argument, c is interpreted as false. In general, $\mathcal{I}2\mathcal{L}_P(\mathcal{L}2\mathcal{I}_P(\mathcal{L}))$ is not equal to $\mathcal{L}$. For instance, considering the labelling $\mathcal{L} = (\emptyset, \{A_4\}, \{A_1, A_2, A_3, A_5, A_6\})$ and the *BAF* $\mathcal{B}_P$ of Figure 7, we have that $\mathcal{I}2\mathcal{L}_P(\mathcal{L}2\mathcal{I}_P(\mathcal{L})) = (\emptyset, \emptyset, \{A_1, A_2, A_3, A_4, A_5, A_6\})$. Such an inequality will always occur when two arguments with the same conclusion receive distinct labels. In our example, A_4 and A_5 have the same conclusion d, but $\mathcal{L}(A_4) = \mathtt{out}$ and $\mathcal{L}(A_5) = \mathtt{undec}$. It holds $\mathcal{L}2\mathcal{I}_P(\mathcal{L})(d) = \mathbf{u}$, and consequently, $\mathcal{I}2\mathcal{L}_P(\mathcal{L}2\mathcal{I}_P(\mathcal{L}))(A_4) = \mathcal{I}2\mathcal{L}_P(\mathcal{L}2\mathcal{I}_P(\mathcal{L}))(A_5) = \mathtt{undec}$, i.e., $\mathcal{I}2\mathcal{L}_P(\mathcal{L}2\mathcal{I}_P(\mathcal{L}))$ is not equal to $\mathcal{L}$. However, if $\mathcal{L}$ is a β-complete labelling of $\mathcal{B}_P$, arguments with the same conclusion will always have the same label:

Theorem 4. Let P be an *NLP* and $\mathcal{B}_P = (\mathcal{A}_P, Att_P, Sup_P)$ be its corresponding *BAF*. If $\mathcal{L}$ is a β-complete labelling of $\mathcal{B}_P$ and $\mathtt{Conc}(A) = \mathtt{Conc}(B)$, then $\mathcal{L}(A) = \mathcal{L}(B)$.

Thus for β-complete labellings, we have that $\mathcal{I}2\mathcal{L}_P(\mathcal{L}2\mathcal{I}_P(\mathcal{L})) = \mathcal{L}$:

Theorem 5. Let P be an *NLP* and $\mathcal{B}_P = (\mathcal{A}_P, Att_P, Sup_P)$ be its corresponding *BAF*. For any β-complete labelling $\mathcal{L}$ of $\mathcal{B}_P$, it holds $\mathcal{I}2\mathcal{L}_P(\mathcal{L}2\mathcal{I}_P(\mathcal{L})) = \mathcal{L}$.

In general, $\mathcal{L}2\mathcal{I}_P(\mathcal{I}2\mathcal{L}_P(\mathcal{I}))$ is not equal to $\mathcal{I}$, because of those atoms c occurring in an *NLP* P, but not in $\mathcal{A}_P$. However, when $\mathcal{M}$ is a partial stable model, $\mathcal{L}2\mathcal{I}_P(\mathcal{I}2\mathcal{L}_P(\mathcal{M})) = \mathcal{M}$:

Theorem 6. Let P be an *NLP*, $\mathcal{B}_P = (\mathcal{A}_P, Att_P)$ be its corresponding *BAF* and $\mathcal{M}$ be a partial stable model of P. It holds that $\mathcal{L}2\mathcal{I}_P(\mathcal{I}2\mathcal{L}_P(\mathcal{M})) = \mathcal{M}$.

This means that when restricted to β-complete labellings and partial stable models, $\mathcal{L}2\mathcal{I}_P$ and $\mathcal{I}2\mathcal{L}_P$ are each other's inverse. The function $\mathcal{L}2\mathcal{I}_P$ when applied to β-complete labellings can be characterised as follows:

Proposition 7. Let P be an *NLP* and $\mathcal{B}_P = (\mathcal{A}_P, Att_P, Sup_P)$ be its corresponding *BAF*. If $\mathcal{L}$ is a β-complete labelling of $\mathcal{B}_P$, then

$$\mathcal{L}2\mathcal{I}_P(\mathcal{L})(c) = \begin{cases} \mathbf{t} & \exists A \in \mathcal{A}_P \textit{ such that } \mathtt{Conc}(A) = c \textit{ and } \mathcal{L}(A) = \mathtt{in} \\ \mathbf{u} & \exists A \in \mathcal{A}_P \textit{ such that } \mathtt{Conc}(A) = c \textit{ and } \mathcal{L}(A) = \mathtt{undec} \\ \mathbf{f} & \textit{otherwise.} \end{cases}$$

In β-complete labellings, different arguments with the same conclusion support each other and share the same acceptability degree. Hence, the acceptability of an atom c follows directly from the acceptability of arguments with conclusion c. If those arguments are accepted (resp. rejected, undecided), c is interpreted as true (resp. false, undecided). If there are no arguments with conclusion c, then c is interpreted as false.

From Lemma 2, and Theorems 5 and 6, we can obtain the following result:

Theorem 8. Let P be an *NLP* and $\mathcal{B}_P = (\mathcal{A}_P, Att_P)$ be its corresponding *BAF*. It holds

- $\mathcal{L}$ is a β-complete labelling of $\mathcal{B}_P$ iff $\mathcal{L}2\mathcal{I}_P(\mathcal{L})$ is a partial stable model of P.
- $\mathcal{M}$ is a partial stable model of P iff $\mathcal{I}2\mathcal{L}_P(\mathcal{M})$ is a β-complete labelling of $\mathcal{B}_P$.

Theorem 8 is one of the main results of this paper. It plays a central role in ensuring the equivalence between the semantics for *NLP*s and their counterpart for *BAF*s:

Theorem 9. Let P be an *NLP* and $\mathcal{B}_P = (\mathcal{A}_P, Att_P, Sup_P)$ be its corresponding *BAF*. It holds

(1) $\mathcal{L}$ is a β-grounded labelling of $\mathcal{B}_P$ iff $\mathcal{L}2\mathcal{I}_P(\mathcal{L})$ is a well-founded model of P.
(2) $\mathcal{L}$ is a β-preferred labelling of $\mathcal{B}_P$ iff $\mathcal{L}2\mathcal{I}_P(\mathcal{L})$ is a regular model of P.
(3) $\mathcal{L}$ is a β-stable labelling of $\mathcal{B}_P$ iff $\mathcal{L}2\mathcal{I}_P(\mathcal{L})$ is a stable model of P.
(4) $\mathcal{L}$ is a β-semi-stable labelling of $\mathcal{B}_P$ iff $\mathcal{L}2\mathcal{I}_P(\mathcal{L})$ is an L-stable model of P.

The following result is a direct consequence of Theorems 6 and 9:

Corollary 10. Let P be an *NLP* and $\mathcal{B}_P = (\mathcal{A}_P, Att_P, Sup_P)$ be its corresponding *BAF*. It holds

(1) $\mathcal{M}$ is a well-founded model of P iff $\mathcal{I}2\mathcal{L}_P(\mathcal{M})$ is a β-grounded labelling of $\mathcal{B}_P$.
(2) $\mathcal{M}$ is a regular model of P iff $\mathcal{I}2\mathcal{L}_P(\mathcal{M})$ is a β-preferred labelling of $\mathcal{B}_P$.
(3) $\mathcal{M}$ is a stable model of P iff $\mathcal{I}2\mathcal{L}_P(\mathcal{M})$ is a β-stable labelling of $\mathcal{B}_P$.
(4) $\mathcal{M}$ is an L-stable model of P iff $\mathcal{I}2\mathcal{L}_P(\mathcal{M})$ is a β-semi-stable labelling of $\mathcal{B}_P$.

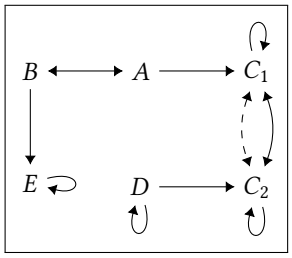
Next, we consider the *NLP* exploited by Caminada, Sá, et al. (2015) as a counterexample to show that in general, L-stable models and semi-stable labellings do not coincide with each other in their translation from *NLP*s to *AAF*s:

Example 6. Let P be the *NLP* and $\mathcal{B}_P$ be the corresponding *BAF* depicted in Figure 8, with conclusions and vulnerabilities shown below:

	A	B	C_1	C_2	D	E
Conc(.)	a	b	c	c	d	e
Vul(.)	$\{a\}$	$\{b\}$	$\{a, c\}$	$\{c, d\}$	$\{d\}$	$\{b, e\}$

$c \leftarrow \texttt{not}\ c, \texttt{not}\ d$ $\quad$ $d \leftarrow \texttt{not}\ d$
$c \leftarrow \texttt{not}\ c, \texttt{not}\ a$ $\quad$ $b \leftarrow \texttt{not}\ a$
$e \leftarrow \texttt{not}\ e, \texttt{not}\ b$ $\quad$ $a \leftarrow \texttt{not}\ b$

(a) P



(b) $\mathcal{B}_P$

Fig. 8. *NLP* P and its associated *BAF* $\mathcal{B}_P$.

Note that in $\mathcal{B}_P$ there is both a mutual attack and a mutual support between arguments C_1 and C_2. As expected from Theorems 8 and 9, we obtain in Table 1 the equivalence between partial stable models and β-complete labellings, well-founded models and β-grounded labellings, regular models and β-preferred labellings, stable models and β-stable labellings, L-stable models and β-semi-stable labellings. We emphasise the coincidence involving L-stable models in Table 1 as it does not occur in Caminada, Sá, et al.'s (2015) approach. In that reference, the associated *AAF* has two semi-stable labels in contrast to the unique L-stable model $\mathcal{M}_3$ of P.

Note that if the support relation was ignored in the framework of Figure 7 as was done by Caminada, Sá, et al. (2015), the β-complete labellings of $\mathcal{B}_P$ would be $\mathcal{L}_1$, $\mathcal{L}_2'$ and $\mathcal{L}_3$, where

$$\mathcal{L}_2' = (\{A\}, \{B, C_1\}, \{C_2, D, E\}),$$

and, as $\mathcal{L}_2'$ and $\mathcal{L}_3$ would be the β-semi-stable labellings, the correspondence with L-stable models would not sustain anymore.

Table 1. Semantics for P and $\mathcal{B}_P$.

Partial Stable Models	Complete Labellings
$\mathcal{M}_1 = (\emptyset, \emptyset, HB_P)$	$\mathcal{L}_1 = (\emptyset, \emptyset, \{A, B, C_1, C_2, D, E\})$
$\mathcal{M}_2 = (\{a\}, \{b\}, \{c, d, e\})$	$\mathcal{L}_2 = (\{A\}, \{B\}, \{C_1, C_2, D, E\})$
$\mathcal{M}_3 = (\{b\}, \{a, e\}, \{c, d\})$	$\mathcal{L}_3 = (\{B\}, \{A, E\}, \{C_1, C_2, D\})$
Well-Founded Model	Grounded Labelling
$\mathcal{M}_1 = (\emptyset, \emptyset, HB_P)$	$\mathcal{L}_1 = (\emptyset, \emptyset, \{A, B, C_1, C_2, D, E\})$
Regular Models	Preferred Labellings
$\mathcal{M}_2 = (\{a\}, \{b\}, \{c, d, e\})$	$\mathcal{L}_2 = (\{A\}, \{B\}, \{C_1, C_2, D, E\})$
$\mathcal{M}_3 = (\{b\}, \{a, e\}, \{c, d\})$	$\mathcal{L}_3 = (\{B\}, \{A, E\}, \{C_1, C_2, D\})$
Stable Models	Stable Labellings
None	None
L-stable Models	Semi-stable Labellings
$\mathcal{M}_3 = (\{b\}, \{a, e\}, \{c, d\})$	$\mathcal{L}_3 = (\{B\}, \{A, E\}, \{C_1, C_2, D\})$

4 From BAF to NLP

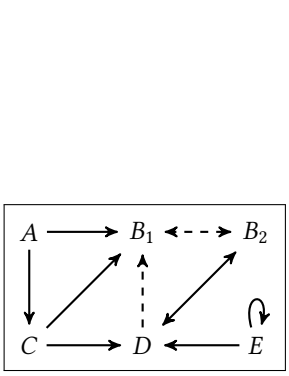
Now we will provide a translation for the opposite direction, i.e., from *BAF*s to *NLP*s. As in the previous section, this translation guarantees the equivalence between the semantics for *NLP*s and their counterpart for *BAF*s. As we are interested in syntactic correspondences, this translation should also be the inverse of the one proposed in Section 3, at least when ignoring argument/atom names.

Before proceeding to the actual translation, we illustrate why a simple translation based on that of Caminada, Sá, et al. (2015) is not sufficient for establishing syntactic correspondences. In Caminada, Sá, et al.'s (2015) translation, each argument A originates a rule r with $head(r) = A$, $body^+(r) = \emptyset$, and $body^-(r) = \{\texttt{not}\ B \mid B \in Att(A)\}$. For instance, we obtain the rule $A \leftarrow \texttt{not}\ B, \texttt{not}\ C$ from an argument A attacked by (only) B and C. In the following example, we adapt this idea to the context of *BAF*s:

Example 7. Let $\mathcal{B} = (\mathcal{A}, Att, Sup)$ be the *BAF* of Figure 9a. The program P in Figure 9b is very similar to that obtained from Caminada, Sá, et al. (2015): the difference is that for each support from A to B we add the rule $B \leftarrow A$. In particular, for mutual supporting arguments A and B, we add both $A \leftarrow B$ and $B \leftarrow A$.

While it is capable of preserving semantics, this simple translation does not suffice for finding syntactic correspondences. The reason is twofold:

(1) The translation from *NLP*s to *BAF*s (Section 3) encodes support between arguments if they derive from rules with a same conclusion, while P does not. For instance, B_1 and B_2 mutually support each other, yet they derive rules $B_1 \leftarrow B_2$ and $B_2 \leftarrow B_1$ (with distinct conclusions) for encoding their support interaction.
(2) The translation from *NLP*s to *BAF*s (Section 3), when applied to P from Figure 9b, can generate an argument X from $B_2 \leftarrow \texttt{not}\ D$ and also an argument Y from the rules $B_2 \leftarrow B_1$ and $B_1 \leftarrow \texttt{not}\ A, \texttt{not}\ C$. Both X and Y have conclusion B_2, which means they both attack the argument Z derived from the rule $D \leftarrow \texttt{not}\ B_2, \texttt{not}\ C, \texttt{not}\ E$. Intuitively, X and Y represent ways of deriving the atom B_2, and Z represents the only way to derive the atom D. Also notice that atoms B_2 and D come from the *BAF* $\mathcal{B}$ (Figure 9a). In $\mathcal{B}$, argument D has 3 attackers: B_2, C, and E. However, we are able to find more than 3 attackers for argument Z: one argument with conclusion C deriving from $C \leftarrow \texttt{not}\ A$, one argument with conclusion E deriving from $E \leftarrow \texttt{not}\ E$, and at least two arguments with conclusion B_2 (namely, X and Y). Clearly, there is no argument in $\mathcal{B}$ with more than 3 attackers. This mismatch shows why this approach fails to preserve the structure of the original *BAF*.



(a) $\mathcal{B}$

A

$B_1 \leftarrow \texttt{not}\ A, \texttt{not}\ C$

$B_2 \leftarrow \texttt{not}\ D$

$C \leftarrow \texttt{not}\ A$

$D \leftarrow \texttt{not}\ B_2, \texttt{not}\ C, \texttt{not}\ E$

$E \leftarrow \texttt{not}\ E$

$B_1 \leftarrow D$

$B_1 \leftarrow B_2$

$B_2 \leftarrow B_1$

(b) *NLP P*

Fig. 9. (a) *BAF* $\mathcal{B}$ and (b) *NLP P*.

Inspired by the problems aforementioned, we define corresponding *BAF*s in a more suitable way for establishing syntactic correspondences. Most of the added complexity serves to avoid redundancies, which in turn allows preserving structure.

Definition 12 (Corresponding *NLP*)**.** Let $\mathcal{B} = (\mathcal{A}, Att, Sup)$ be a *BAF*. For any $A, B \in \mathcal{A}$, define the rule $r_{A,B}$ such that $head(r_{A,B}) = \mathfrak{Sup}(A)$, $body^+(r_{A,B}) = \emptyset$ and $body^-(r_{A,B}) = \{\texttt{not}\ \mathfrak{Sup}(X) \mid X \in Att(B)\}$. The corresponding *NLP* $P_{\mathcal{B}}$ of $\mathcal{B}$ is the program with the following set of rules:

$$P_{\mathcal{B}} = \{r_{A,B} \mid A \in \mathcal{A}, B \in \mathfrak{Sup}(A)\}.$$

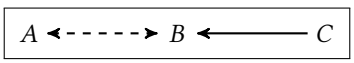
Each argument is encoded in the corresponding *NLP* by its set of supporters and the Herbrand Base of $P_{\mathcal{B}}$ is $HB_{P_{\mathcal{B}}} = \{\mathfrak{Sup}(A) \mid A \in \mathcal{A}\}$. Intuitively, with respect to any partial stable model of $P_{\mathcal{B}}$, we know $\mathfrak{Sup}(A)$ is true iff there exists $B \in \mathfrak{Sup}(A)$ such that $\mathfrak{Sup}(X)$ is false for every $X \in Att(B)$. This property follows directly from Definition 12, that was conceived to reflect the concept of acceptability in the β-complete semantics, where an argument $A \in \mathcal{A}$ is accepted iff there exists $B \in \mathfrak{Sup}(A)$ such that X is rejected for every $X \in Att(B)$.

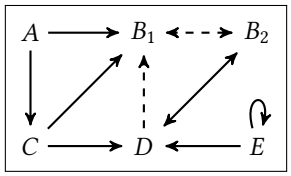
Example 8. Let $\mathcal{B} = (\mathcal{A}, Att, Sup)$ be the *BAF* of Figure 10a, and $P_{\mathcal{B}}$ be its corresponding *NLP* shown in Figure 10b.

Observe that distinct arguments A, A' are associated to the same atom if $\mathfrak{Sup}(A) = \mathfrak{Sup}(A')$. For instance, the corresponding *NLP* of Example 10b encodes both arguments B_1 and B_2 as the atom $\mathfrak{Sup}(B_1) = \mathfrak{Sup}(B_2) = \{B_1, B_2, D\}$. From the supporters of B_1, we obtain the rules $r_{B_1,B_1} = \{B_1, B_2, D\} \leftarrow \texttt{not}\ \{A\}, \texttt{not}\ \{C\}$; $r_{B_1,B_2} = \{B_1, B_2, D\} \leftarrow \texttt{not}\ \{D\}$; and $r_{B_1,D} = \{B_1, B_2, D\} \leftarrow \texttt{not}\ \{B_1, B_2, D\}, \texttt{not}\ \{C\}, \texttt{not}\ \{E\}$. The intuition is that B_1 is accepted iff either (i) A and C are rejected or (ii) D is rejected or (iii) B_1, B_2, C, D and E are rejected. Condition (iii) seems contradictory, but it is similar to what occurs when an argument attacks itself (e.g., argument E is accepted iff E is rejected).

As done in the previous section, we want to establish a bijection between labellings and interpretations, such that β-complete labellings correspond to partial stable models. However, we can have more labellings than interpretations.

Example 9. Let $\mathcal{B}'$ be the *BAF* depicted below:





(a) $\mathcal{B}$

$$
\begin{aligned}
\{A\} & \\
\{B_1, B_2, D\} &\leftarrow \texttt{not}\ \{A\}, \texttt{not}\ \{C\} \\
\{B_1, B_2, D\} &\leftarrow \texttt{not}\ \{D\} \\
\{B_1, B_2, D\} &\leftarrow \texttt{not}\ \{B_1, B_2, D\}, \texttt{not}\ \{C\}, \texttt{not}\ \{E\} \\
\{C\} &\leftarrow \texttt{not}\ \{A\} \\
\{D\} &\leftarrow \texttt{not}\ \{B_1, B_2, D\}, \texttt{not}\ \{C\}, \texttt{not}\ \{E\} \\
\{E\} &\leftarrow \texttt{not}\ \{E\}
\end{aligned}
$$

(b) $P_{\mathcal{B}}$

Fig. 10. (a) *BAF* $\mathcal{B}$ and (b) its corresponding *NLP* $P_{\mathcal{B}}$.

The corresponding *NLP* $P_{\mathcal{B}'}$ is

$$
\begin{aligned}
\{A, B\} &\leftarrow \texttt{not}\ \{C\} \\
\{A, B\} & \\
\{C\}. &
\end{aligned}
$$

There are $3^{|\mathcal{A}|} = 27$ labellings of $\mathcal{B}'$ and $3^{|HB_{P_{\mathcal{B}'}}|} = 9$ interpretations of $P_{\mathcal{B}'}$. Hence, there is no bijection between labellings of $\mathcal{B}'$ and interpretations of $P_{\mathcal{B}'}$. Notice that a labelling may assign different labels to A and B, whereas an interpretation assigns only one truth value to the atom $\{A, B\}$.

Although there is no bijection between labellings and interpretations for the example above, there is always a bijection if we restrict our translations to labellings respecting $\mathfrak{Sup}$, defined next.

Definition 13. Let $\mathcal{B} = (\mathcal{A}, Att, Sup)$ be a *BAF*. A labelling $\mathcal{L}$ of $\mathcal{B}$ respects $\mathfrak{Sup}$ iff for every $A, B \in \mathcal{A}$ with $\mathfrak{Sup}(A) = \mathfrak{Sup}(B)$ it holds $\mathcal{L}(A) = \mathcal{L}(B)$.

For the *BAF* $\mathcal{B}'$ of Example 9, there are 9 labellings respecting $\mathfrak{Sup}$, since a labelling that respects $\mathfrak{Sup}$ assigns the same label to A and B. The representation of possibly many arguments by the same atom is motivated by the fact that every β-complete labelling satisfies the following property:

Proposition 11. If $\mathcal{L}$ is a β-complete labelling of $\mathcal{B} = (\mathcal{A}, Att, Sup)$, then $\mathcal{L}$ respects $\mathfrak{Sup}$.

By restricting labellings to those that respect $\mathfrak{Sup}$, the relationship between labellings and interpretations is straightforward:

Definition 14 ($\mathcal{L}2\mathcal{I}_{\mathcal{B}}$ and $\mathcal{I}2\mathcal{L}_{\mathcal{B}}$ functions)**.** Let $\mathcal{B} = (\mathcal{A}, Att, Sup)$ be a *BAF* and $P_{\mathcal{B}}$ be its corresponding *NLP*. Denote the set of all labellings of $\mathcal{B}$ respecting $\mathfrak{Sup}$ as $\mathcal{L}ab^*$ and the set of all interpretations of $P_{\mathcal{B}}$ as $\mathcal{I}nt$. We introduce the functions

- $\mathcal{L}2\mathcal{I}_{\mathcal{B}} : \mathcal{L}ab^* \to \mathcal{I}nt$, in which for every $\mathcal{L} \in \mathcal{L}ab^*$,

$$\begin{aligned} \mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L})(\mathfrak{Sup}(A)) &= \mathbf{t} && \text{if } \mathcal{L}(A) = \mathtt{in} \\ \mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L})(\mathfrak{Sup}(A)) &= \mathbf{f} && \text{if } \mathcal{L}(A) = \mathtt{out} \\ \mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L})(\mathfrak{Sup}(A)) &= \mathbf{u} && \text{if } \mathcal{L}(A) = \mathtt{undec} \end{aligned}$$

- $\mathcal{I}2\mathcal{L}_{\mathcal{B}} : \mathcal{I}nt \to \mathcal{L}ab^*$, in which for every $\mathcal{M} \in \mathcal{I}nt$,

$$\begin{aligned} \mathcal{I}2\mathcal{L}_{\mathcal{B}}(\mathcal{M})(A) &= \mathtt{in} && \text{if } \mathcal{M}(\mathfrak{Sup}(A)) = \mathbf{t} \\ \mathcal{I}2\mathcal{L}_{\mathcal{B}}(\mathcal{M})(A) &= \mathtt{out} && \text{if } \mathcal{M}(\mathfrak{Sup}(A)) = \mathbf{f} \\ \mathcal{I}2\mathcal{L}_{\mathcal{B}}(\mathcal{M})(A) &= \mathtt{undec} && \text{if } \mathcal{M}(\mathfrak{Sup}(A)) = \mathbf{u}. \end{aligned}$$

Note how the functions are well-defined only for labellings respecting $\mathfrak{Sup}$. Semantics-wise, this is a minor restriction, as every labelling under the β-complete, β-grounded, β-preferred, β-stable or β-semi-stable semantics respects $\mathfrak{Sup}$.

Unlike $\mathcal{L}2\mathcal{I}_P$ and $\mathcal{I}2\mathcal{L}_P$, the functions $\mathcal{L}2\mathcal{I}_{\mathcal{B}}$ and $\mathcal{I}2\mathcal{L}_{\mathcal{B}}$ are the inverse of each other in the general case:

Theorem 12. Let $\mathcal{B} = (\mathcal{A}, Att, Sup)$ be a *BAF* and $P_{\mathcal{B}}$ be its corresponding *NLP*.

- For any labelling $\mathcal{L}$ of $\mathcal{B}$ respecting $\mathfrak{Sup}$, it holds $\mathcal{I}2\mathcal{L}_{\mathcal{B}}(\mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L})) = \mathcal{L}$.
- For any interpretation $\mathcal{I}$ of $P_{\mathcal{B}}$, it holds $\mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{I}2\mathcal{L}_{\mathcal{B}}(\mathcal{I})) = \mathcal{I}$.

Note that the assumption that $\mathcal{L}$ must respect $\mathfrak{Sup}$ is already given by $\mathcal{L}2\mathcal{I}_{\mathcal{B}}$'s domain. The next two lemmas are essential for the correspondence between β-complete labellings of $\mathcal{B}$ and partial stable models of $P_{\mathcal{B}}$.

Lemma 13. Let $\mathcal{B} = (\mathcal{A}, Att, Sup)$ be a *BAF* and $P_{\mathcal{B}}$ be its corresponding *NLP*. Let $\mathcal{L}$ be a labelling of $\mathcal{B}$ respecting $\mathfrak{Sup}$ and $\mathcal{M}$ be an interpretation of $P_{\mathcal{B}}$. If $\mathcal{L} = \mathcal{I}2\mathcal{L}_{\mathcal{B}}(\mathcal{M})$ or $\mathcal{M} = \mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L})$, then for any $A \in \mathcal{A}$,

- There exists $B \in \mathfrak{Sup}(A)$ such that $Att(B) \subseteq \mathtt{out}(\mathcal{L})$ iff there exists $r \in P_{\mathcal{B}}$ with $head(r) = \mathfrak{Sup}(A)$ such that $\{\mathfrak{Sup}(X) \mid \mathtt{not}\ \mathfrak{Sup}(X) \in body^-(r)\} \subseteq \mathbf{f}(\mathcal{M})$.
- For every $B \in \mathfrak{Sup}(A)$ it holds $Att(B) \cap \mathtt{in}(\mathcal{L}) \neq \emptyset$ iff for every rule $r \in P_{\mathcal{B}}$ with $head(r) = \mathfrak{Sup}(A)$ it holds $\{\mathfrak{Sup}(X) \mid \mathtt{not}\ \mathfrak{Sup}(X) \in body^-(r)\} \cap \mathbf{t}(\mathcal{M}) \neq \emptyset$.

From Lemma 13, we obtain a similar result to Theorem 8:

Theorem 14. Let $\mathcal{B} = (\mathcal{A}, Att, Sup)$ be a *BAF*, $P_{\mathcal{B}}$ be its corresponding *NLP*, $\mathcal{L}$ be a labelling of $\mathcal{B}$ respecting $\mathfrak{Sup}$ and $\mathcal{M}$ be an interpretation of $P_{\mathcal{B}}$. It holds

- $\mathcal{L}$ is a β-complete labelling of $\mathcal{B}$ iff $\mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L})$ is a partial stable model of $P_{\mathcal{B}}$.
- $\mathcal{M}$ is a partial stable model of $P_{\mathcal{B}}$ iff $\mathcal{I}2\mathcal{L}_{\mathcal{B}}(\mathcal{M})$ is a β-complete labelling of $\mathcal{B}$.

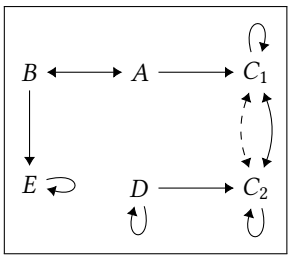
From Theorem 14, we can ensure the equivalence between the semantics for *BAF*s and their counterpart for *NLP*s:

Theorem 15. Let $\mathcal{B}$ be a *BAF* and $P_{\mathcal{B}}$ be its corresponding *NLP*. For any labelling $\mathcal{L}$ of $\mathcal{B}$ respecting $\mathfrak{Sup}$, it holds

1. $\mathcal{L}$ is a β-grounded labelling of $\mathcal{B}$ iff $\mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L})$ is a well-founded model of $P_{\mathcal{B}}$.
2. $\mathcal{L}$ is a β-preferred labelling of $\mathcal{B}$ iff $\mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L})$ is a regular model of $P_{\mathcal{B}}$.
3. $\mathcal{L}$ is a β-stable labelling of $\mathcal{B}$ iff $\mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L})$ is a stable model of $P_{\mathcal{B}}$.
4. $\mathcal{L}$ is a β-semi-stable labelling of $\mathcal{B}$ iff $\mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L})$ is an L-stable model of $P_{\mathcal{B}}$.

The following result is a direct consequence of Theorems 12 and 15:

Corollary 16. Let $\mathcal{B}$ be a *BAF* and $P_{\mathcal{B}}$ be its corresponding *NLP*. It holds



(a) *BAF* from Example 6.

$$\{C_1, C_2\} \leftarrow \texttt{not}\ \{C_1, C_2\}, \texttt{not}\ \{D\}$$
$$\{C_1, C_2\} \leftarrow \texttt{not}\ \{C_1, C_2\}, \texttt{not}\ \{A\}$$
$$\{E\} \leftarrow \texttt{not}\ \{E\}, \texttt{not}\ \{B\}$$
$$\{D\} \leftarrow \texttt{not}\ \{D\}$$
$$\{B\} \leftarrow \texttt{not}\ \{A\}$$
$$\{A\} \leftarrow \texttt{not}\ \{B\}$$

(b) Its corresponding *NLP*.

Fig. 11. The corresponding *NLP* of the *BAF* from Example 6.

Table 2. Semantics for the *BAF* and corresponding *NLP* from Figure 11.

β-Complete Labellings	Partial Stable Models
$\mathcal{L}_1 = (\emptyset, \emptyset, \{A, B, C_1, C_2, D, E\})$	$\mathcal{M}_1 = (\emptyset, \emptyset, \{\{A\}, \{B\}, \{C_1, C_2\}, \{D\}, \{E\}\})$
$\mathcal{L}_2 = (\{A\}, \{B\}, \{C_1, C_2, D, E\})$	$\mathcal{M}_2 = (\{\{A\}\}, \{\{B\}\}, \{\{C_1, C_2\}, \{D\}, \{E\}\})$
$\mathcal{L}_3 = (\{B\}, \{A, E\}, \{C_1, C_2, D\})$	$\mathcal{M}_3 = (\{\{B\}\}, \{\{A\}, \{E\}\}, \{\{C_1, C_2\}, \{D\}\})$
β-Grounded Labelling	Well-Founded Model
$\mathcal{L}_1 = (\emptyset, \emptyset, \{A, B, C_1, C_2, D, E\})$	$\mathcal{M}_1 = (\emptyset, \emptyset, \{\{A\}, \{B\}, \{C_1, C_2\}, \{D\}, \{E\}\})$
β-Preferred Labellings	Regular Models
$\mathcal{L}_2 = (\{A\}, \{B\}, \{C_1, C_2, D, E\})$	$\mathcal{M}_2 = (\{\{A\}\}, \{\{B\}\}, \{\{C_1, C_2\}, \{D\}, \{E\}\})$
$\mathcal{L}_3 = (\{B\}, \{A, E\}, \{C_1, C_2, D\})$	$\mathcal{M}_3 = (\{\{B\}\}, \{\{A\}, \{E\}\}, \{\{C_1, C_2\}, \{D\}\})$
β-Stable Labellings	Stable Models
None	None
β-Semi-stable Labellings	L-Stable Models
$\mathcal{L}_3 = (\{B\}, \{A, E\}, \{C_1, C_2, D\})$	$\mathcal{M}_3 = (\{\{B\}\}, \{\{A\}, \{E\}\}, \{\{C_1, C_2\}, \{D\}\})$

1. $\mathcal{M}$ is a well-founded model of $P_{\mathcal{B}}$ iff $\mathcal{I}2\mathcal{L}_{\mathcal{B}}(\mathcal{M})$ is a β-grounded labelling of $\mathcal{B}$.
2. $\mathcal{M}$ is a regular model of $P_{\mathcal{B}}$ iff $\mathcal{I}2\mathcal{L}_{\mathcal{B}}(\mathcal{M})$ is a β-preferred labelling of $\mathcal{B}$.
3. $\mathcal{M}$ is a stable model of $P_{\mathcal{B}}$ iff $\mathcal{I}2\mathcal{L}_{\mathcal{B}}(\mathcal{M})$ is a β-stable labelling of $\mathcal{B}$.
4. $\mathcal{M}$ is a semi-stable model of $P_{\mathcal{B}}$ iff $\mathcal{I}2\mathcal{L}_{\mathcal{B}}(\mathcal{M})$ is a β-semi-stable labelling of $\mathcal{B}$.

From the *BAF* of Example 6 and its corresponding *NLP*, both depicted in Figure 11, we obtain the expected equivalence results related to their semantics (see Table 2). Observe that from the *NLP* P of Example 6 (Figure 8a), we constructed the corresponding *BAF* $\mathcal{B}_P$ (Figures 8b and 11a) and then its corresponding *NLP* $P_{\mathcal{B}_P}$ (Figure 11b). It is no coincidence that P and $P_{\mathcal{B}_P}$ are isomorphic, i.e., each can be obtained by renaming the atoms from the other. In the next section, we will find a class of *BAF*s and *NLP*s such that the translation from an *NLP* to a *BAF* (Definition 10) behaves as the inverse (up to isomorphism) of the translation from a *BAF* to an *NLP* (Definition 12).

5 On the Relation between BAF and NLP

In preceding sections, we have shown how to translate *BAF*s to *NLP*s, and vice versa, whilst preserving their corresponding semantics. Now we check under which conditions these translations are the inverses (up to isomorphism) of each other, showing that *BAF*s and *NLP*s are essentially the same formalism under these conditions. We say *BAF*s $\mathcal{B} = (\mathcal{A}, Att, Sup)$ and $\mathcal{B}' = (\mathcal{A}', Att', Sup')$ are isomorphic if there exists a bijection $f : \mathcal{A} \to \mathcal{A}'$ such that

- $\{f(A) \mid A \in \mathcal{A}\} = \mathcal{A}'$,
- $\{(f(A), f(B)) \mid (A, B) \in Att\} = Att'$, and
- $\{(f(A), f(B)) \mid (A, B) \in Sup\} = Sup'$.

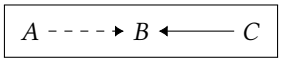
Similarly, *NLP*s P and P' are isomorphic when there exists a bijection $f : HB_P \to HB_{P'}$ such that $\{f(c) \leftarrow f(a_1), \cdots, f(a_m), \texttt{not}\ f(b_1), \cdots, \texttt{not}\ f(b_n) \mid c \leftarrow a_1, \cdots, a_m, \texttt{not}\ b_1, \cdots, \texttt{not}\ b_n \in P\} = P'$.

From a *BAF* $\mathcal{B}$, we obtain its corresponding *NLP* $P_{\mathcal{B}}$ via Definition 12; from $P_{\mathcal{B}}$, we obtain its corresponding *BAF* $\mathcal{B}_{P_{\mathcal{B}}}$ via Definition 10. By following the other direction, from an *NLP* P, we obtain its corresponding *BAF* $\mathcal{B}_P$, and from $\mathcal{B}_P$, its corresponding *NLP* $P_{\mathcal{B}_P}$. In this section, we investigate for which class of *BAF*s we have $\mathcal{B}$ is isomorphic to $\mathcal{B}_{P_{\mathcal{B}}}$, and for which class of *NLP*s the programs P and $P_{\mathcal{B}_P}$ are isomorphic.

5.1 On the Isomorphism between $\mathcal{B}$ and $\mathcal{B}_{P_{\mathcal{B}}}$

Initially, we show that, in general, $\mathcal{B}$ is not isomorphic to $\mathcal{B}_{P_{\mathcal{B}}}$. This result is expected, as by Definition 10, every corresponding *BAF* of an *NLP* has a symmetric support relation.

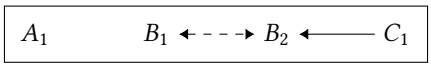
Example 10. Let $\mathcal{B} = (\mathcal{A}, Att, Sup)$ be the following *BAF*:



Its corresponding *NLP* $P_{\mathcal{B}}$ is

$$\begin{array}{ll} \{A\} & \{C\} \\ \{A, B\} & \{A, B\} \leftarrow \texttt{not}\ \{C\}. \end{array}$$

Its corresponding *BAF* $\mathcal{B}_{P_{\mathcal{B}}}$ is shown below, where arguments A_1, B_1, B_2, C_1 are derived respectively from the rules $\{A\}$, $\{A, B\}$, $\{A, B\} \leftarrow \texttt{not}\ \{C\}$ and $\{C\}$:



Notice that the support relation of $\mathcal{B}_{P_{\mathcal{B}}}$ is symmetric, whereas the support relation of $\mathcal{B}$ is not. Clearly, they are not isomorphic. However, symmetry is not the decisive factor for determining whether $\mathcal{B}$ and $\mathcal{B}_{P_{\mathcal{B}}}$ are isomorphic, i.e., they can be non-isomorphic even if $\mathcal{B}$ has a symmetric support relation. This occurs because the support relation of a *BAF* $\mathcal{B}_P$ that corresponds to some *NLP* P is very specific, as distinct arguments support each other in $\mathcal{B}_P$ exactly when they derive from rules in P sharing the same conclusion. An important property of every *BAF* that corresponds to some *NLP* is defined next:

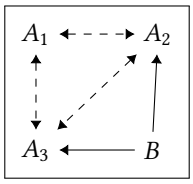
Definition 15 ($\mathfrak{S}$-*BAF*)**.** Let $\mathcal{B} = (\mathcal{A}, Att, Sup)$ be a *BAF*. We say $\mathcal{B}$ is a *BAF* of support cliques ($\mathfrak{S}$-*BAF*) iff *Sup* is irreflexive, symmetric and for any $(A, B), (B, C) \in Sup$ with $A \neq C$, it holds $(A, C) \in Sup$.

The name “support cliques" comes from the fact that we could have equivalently defined it as a *BAF* for which *Sup* is irreflexive, symmetric and $\mathcal{A}$ can be partitioned into maximal cliques of the graph $(\mathcal{A}, Sup)$, where a maximal clique is a subset $S \subseteq \mathcal{A}$ closed for *Sup* and such that any distinct $A, B \in S$ are adjacent (by *Sup*).

Lemma 17. For any *NLP* P, the corresponding *BAF* $\mathcal{B}_P$ is a $\mathfrak{S}$-*BAF*.

There are *BAF*s of support cliques that can only correspond to “redundant" *NLP*s (as we shall define later).

Example 11. Consider the following $\mathfrak{S}$-*BAF*:



Suppose there exists an *NLP* P such that $body^+(r) = \emptyset$ for every rule $r \in P$ and whose corresponding *BAF* is the one of this example. As $(A_2, A_3) \in Sup$, we obtain $A_2 \neq A_3$ and $\mathtt{Conc}(A_2) = \mathtt{Conc}(A_3)$. Let $r_2, r_3 \in P$ be the rules that derived A_2 and A_3, respectively. As $Att(A_2) = Att(A_3)$, we obtain $body(r_2) = body(r_3)$. As $\mathtt{Conc}(A_2) = \mathtt{Conc}(A_3)$, it follows $head(r_2) = head(r_3)$. Therefore, $r_2 = r_3$, which means that $A_2 = A_3$, as only one argument can be derived from a rule r with $body^+(r) = \emptyset$. As $A_2 \neq A_3$, we found an absurd. Hence, we have shown that any *NLP* that corresponds to this *BAF* has some positive body atom (i.e., $body^+(r) \neq \emptyset$ for some rule r), which will be analysed in Subsection 5.2 as a kind of redundancy.

That occurred because arguments A_2 and A_3 share the same set of supporters and attackers. Recall that if arguments A, B satisfy $\mathfrak{Sup}(A) = \mathfrak{Sup}(B)$, then A is labelled the same as B for any β-complete labelling. Therefore, attackers of an argument A represent a possible condition to accept any argument with the same supporters as those of A. More specifically, the acceptance condition is that every attacker of A is rejected. Arguments with identical sets of supporters and attackers are, in that sense, redundant as defined next:

Definition 16 (*RFBAF*)**.** Let $\mathcal{B} = (\mathcal{A}, Att, Sup)$ be a *BAF*. Arguments $A, B \in \mathcal{A}$ are redundant iff $A \neq B$ and $\mathfrak{Sup}(A) = \mathfrak{Sup}(B)$ and $Att(A) = Att(B)$. When a *BAF* $\mathcal{B}$ contains redundant arguments, we say $\mathcal{B}$ is redundant. Otherwise, $\mathcal{B}$ is a redundancy-free *BAF* (*RFBAF*).

When a *RFBAF* is also a $\mathfrak{S}$-*BAF*, we call it a redundancy-free $\mathfrak{S}$-*BAF* ($\mathfrak{S}$-*RFBAF*). As this redundancy arises from the support relation, no *AAF* is redundant.

Proposition 18. Let $\mathcal{B} = (\mathcal{A}, Att, Sup)$ be a *BAF*. If $Sup = \emptyset$, then $\mathcal{B}$ is a *RFBAF*.

Moreover, this concept of redundancy aligns with the proposed translation from *BAF*s to *NLP*s, as redundant arguments derive the same set of rules. For instance, arguments A_2 and A_3 in the *BAF* of Example 11 have the same set of supporters $\{A_1, A_2, A_3\}$ and set of attackers $\{B\}$, and derive the same rule $\{A_1, A_2, A_3\} \leftarrow \mathtt{not}\ \{B\}$. Notice the loss of information: two arguments derive one unique rule, and from this one rule we cannot recover the original two arguments. Now we prove that the isomorphism holds precisely for the class of $\mathfrak{S}$-*RFBAF*s:

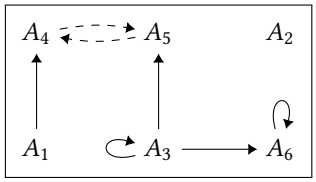
Theorem 19. Let $\mathcal{B}$ be a *BAF*, $P_{\mathcal{B}}$ be its corresponding *NLP*, and $\mathcal{B}_{P_{\mathcal{B}}}$ be the corresponding *BAF* of $P_{\mathcal{B}}$. It holds $\mathcal{B}$ and $\mathcal{B}_{P_{\mathcal{B}}}$ are isomorphic iff $\mathcal{B}$ is a $\mathfrak{S}$-*RFBAF*.

Example 12. Let $\mathcal{B}$ be the *BAF* of Example 5, depicted again in Figure 12 for convenience, next to its corresponding logic program $P_{\mathcal{B}}$ and the corresponding *BAF* $\mathcal{B}_{P_{\mathcal{B}}}$ of $P_{\mathcal{B}}$. As expected, $\mathcal{B}$ and $\mathcal{B}_{P_{\mathcal{B}}}$ are isomorphic.

5.2 On the Isomorphism between P and $P_{\mathcal{B}_P}$

Now we investigate under which conditions P and $P_{\mathcal{B}_P}$ are isomorphic. The next example shows that some *NLP*s correspond to the same *BAF* (up to isomorphism).

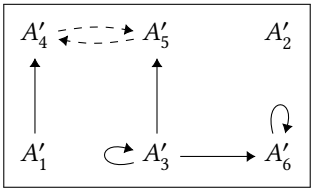
Example 13. Let P be the *NLP* with rules $\{r_1 = a, r_2 = b\}$, P' be the *NLP* with rules $\{r_1' = a, r_2' = b \leftarrow \mathtt{not}\ c\}$, and P'' be the *NLP* with rules $\{r_1'' = a, r_2'' = b, r_3'' = d \leftarrow c\}$.



(a) $\mathcal{B}$

$$\{A_1\} \qquad \{A_2\}$$
$$\{A_4, A_5\} \leftarrow \texttt{not}\ \{A_1\}$$
$$\{A_4, A_5\} \leftarrow \texttt{not}\ \{A_3\}$$
$$\{A_3\} \leftarrow \texttt{not}\ \{A_3\}$$
$$\{A_6\} \leftarrow \texttt{not}\ \{A_3\}, \texttt{not}\ \{A_6\}$$

(b) $P_{\mathcal{B}}$



(c) $\mathcal{B}_{P_{\mathcal{B}}}$

Fig. 12. *BAF* $\mathcal{B}$, its corresponding logic program $P_{\mathcal{B}}$, and the corresponding *BAF* $\mathcal{B}_{P_{\mathcal{B}}}$ of $P_{\mathcal{B}}$. Arguments $A'_1, A'_2, A'_3, A'_4, A'_5, A'_6$ are respectively derived from the rules $\{A_1\}$, $\{A_2\}$, $\{A_3\} \leftarrow \texttt{not}\ \{A_3\}$, $\{A_4, A_5\} \leftarrow \texttt{not}\ \{A_1\}$, $\{A_4, A_5\} \leftarrow \texttt{not}\ \{A_3\}$ and $\{A_6\} \leftarrow \texttt{not}\ \{A_3\}, \texttt{not}\ \{A_6\}$.

Notice that these *NLP*s are almost the same: P is almost equal to P', except for the occurrence of $\texttt{not}\ c$ in the body of r'_2; and P is almost equal to P'', except for the rule $r''_3 \in P''$.

In the corresponding *BAF* of an *NLP* P, there is no argument with conclusion c such that $c \in HB_P - \{head(r) \mid r \in P\}$. Then, according to Definition 10, the literal $\texttt{not}\ c$ is ignored when $\texttt{not}\ c \in body^-(r)$; and the whole rule r is ignored when $c \in body^+(r)$. It follows that some rules are effectively the same, as they derive the same arguments in the corresponding *BAF*. For instance, $\texttt{not}\ c$ is ignored in $r'_2 = b \leftarrow \texttt{not}\ c$, making r'_2 effectively the same as $r_2 = b$; and $r''_3 = d \leftarrow c$ is ignored in P'', making P'' effectively the same as P.

As such, the corresponding *BAF*s of each of these *NLP*s are all isomorphic to the *BAF* $(\{A_0, A_1\}, \emptyset, \emptyset)$, with only two arguments and no attacks/supports. Clearly, the corresponding *NLP* of $(\{A_0, A_1\}, \emptyset, \emptyset)$ cannot be isomorphic to all of P, P' and P''. In fact, it is isomorphic to P which has no ignored atoms or rules, as every atom appears in the head of some rule.

The hint given by the example above is that the isomorphism only holds for *NLP*s such that every atom appears in the head of some rule.

Lemma 20. Let P be an *NLP*, $\mathcal{B}_P$ be its corresponding *BAF* and $P_{\mathcal{B}_P}$ be the corresponding logic program of $\mathcal{B}_P$. If $HB_P \neq \{head(r) \mid r \in P\}$, then P and $P_{\mathcal{B}_P}$ are not isomorphic.

There is another scenario in which the isomorphism does not hold. Note that, by Definition 12, the corresponding logic program P' of some *BAF* $\mathcal{B}$ has only rules r with empty $body^+(r)$. In particular, if P has a rule r with nonempty $body^+(r)$, $P_{\mathcal{B}_P}$ cannot be isomorphic to P. This is the same kind of redundancy as that mentioned in Subsection 5.1.

Lemma 21. Let P be an *NLP*, $\mathcal{B}_P$ be its corresponding *BAF* and $P_{\mathcal{B}_P}$ be the corresponding logic program of $\mathcal{B}_P$. If there is some $r \in P$ such that $body^+(r) \neq \emptyset$, then P and $P_{\mathcal{B}_P}$ are not isomorphic.

From the two conditions discussed above, we define the class of reduced atomic logic programs (*RALP*s):

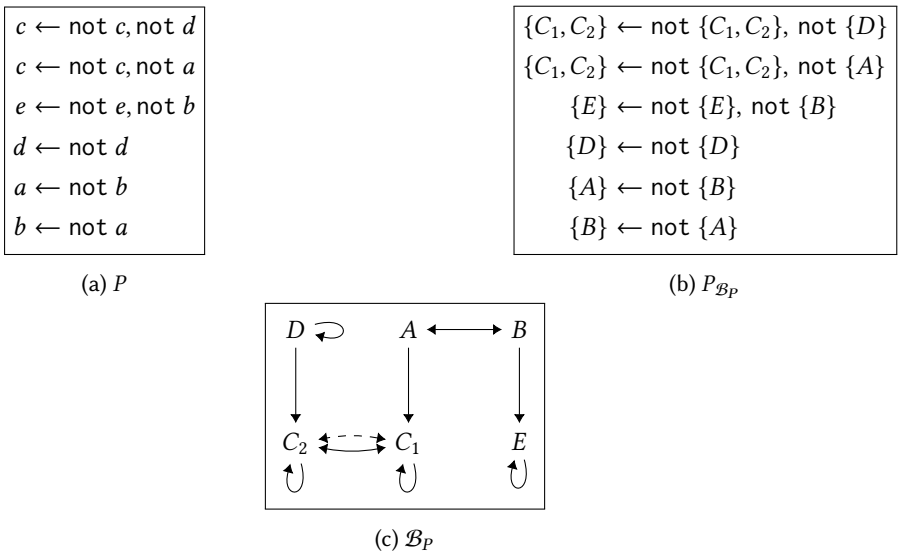
Definition 17 (*RALP*)**.** We say an *NLP* P is a Reduced Atomic Logic Program (*RALP*) if for any rule $r \in P$, it holds $body^+(r) = \emptyset$, and $HB_P = \{head(r) \mid r \in P\}$.

We find that *RALP*s are precisely the *NLP*s for which the isomorphism between P and $P_{\mathcal{B}_P}$ holds.

Theorem 22. Let P be an *NLP*, $\mathcal{B}_P$ be its corresponding *BAF* and $P_{\mathcal{B}_P}$ be the corresponding *NLP* of $\mathcal{B}_P$. It holds P and $P_{\mathcal{B}_P}$ are isomorphic iff P is an *RALP*.

Example 14. Recall the *NLP* P and its corresponding *BAF* $\mathcal{B}_P$ of Example 6. They are depicted again in Figure 13 for convenience, together with the corresponding *NLP* of $\mathcal{B}_P$. Notice the isomorphism between P and $P_{\mathcal{B}_P}$.

The corresponding *NLP* $P_{\mathcal{B}_P}$ encodes arguments that have identical sets of supporters (e.g., C_1, C_2) as only one atom (e.g., $\{C_1, C_2\}$). This is essential for the isomorphism between P and $P_{\mathcal{B}_P}$, as $\mathcal{B}_P$ encodes each atom of P as possibly many arguments sharing the same set of supporters (e.g., from atom c we obtain arguments C_1 and C_2). The isomorphism comes from the collapse of C_1 and C_2 into their set of supporters $\{C_1, C_2\}$.



(a) P (b) $P_{\mathcal{B}_P}$

(c) $\mathcal{B}_P$

Fig. 13. *NLP* P, its corresponding *BAF* $\mathcal{B}_P$, and the corresponding logic program $P_{\mathcal{B}_P}$ of $\mathcal{B}_P$. Arguments A, B, C_1, C_2, D, E are respectively derived from the rules $(a \leftarrow \texttt{not } b)$, $(b \leftarrow \texttt{not } a)$, $(c \leftarrow \texttt{not } c, \texttt{not } a)$, $(c \leftarrow \texttt{not } c, \texttt{not } d)$, $(d \leftarrow \texttt{not } d)$ and $(e \leftarrow \texttt{not } e, \texttt{not } b)$.

6 On the Relation between NLP and RALP

In Section 5, we established that redundancy-free *BAF*s of support cliques ($\mathfrak{S}$-*RFBAF*s) and reduced atomic logic programs (*RALP*s) are essentially the same formalism. Here, we investigate the connection between *NLP*s and *RALP*s, highlighting the link between the corresponding *BAF*s of *NLP*s. Initially, we show that *RALP*s are as expressive as *NLP*s when considering the semantics for *NLP*s exploited in this paper. Our idea is to transform any *NLP* P into a unique *RALP* P^* by resorting to the transformations employed in the characterisation of the *RALP* $P_{\mathcal{B}_P}$ (see Sections 3 and 4). These transformations have already been shown to preserve the semantics under consideration. Additionally, by analysing corresponding *BAF*s, we find many links between *NLP*s and *RALP*s, such as

(1) for any *NLP* P such that $\mathcal{B}_P$ is an *RFBAF*, there is an *RALP* P^* such that $\mathcal{B}_P$ and $\mathcal{B}_{P^*}$ are isomorphic (Theorem 26);
(2) *RALP*s P and P' are isomorphic iff $\mathcal{B}_P$ and $\mathcal{B}_{P'}$ are isomorphic (Theorem 27);
(3) for any *NLP*s P and P' such that their corresponding *BAF*s are *RFBAF*s, it holds that the corresponding *RALP*s of P and P' are isomorphic iff $\mathcal{B}_P$ and $\mathcal{B}_{P'}$ are isomorphic (Theorem 28).

The assumption that the corresponding *BAF* is an *RFBAF* in the results above is a consequence of Definition 8 allowing the construction of redundant arguments. In the end of this section, we also explain how corresponding *BAF*s could have been alternatively defined, by first transforming an *NLP* into an *RALP* before applying Definition 8 to the resulting *RALP*. As every corresponding *BAF* of an *RALP* is an *RFBAF*, this assumption would no longer be required for Theorems 26 and 28.

We proceed by defining formally expressiveness in terms of signatures of the semantics (Dunne et al. 2015):

Definition 18 (Expressiveness)**.** Let $\mathcal{P}$ be a class of *NLP*s. The signature $\Sigma^{\mathcal{P}}_{PSM}$ of the partial stable models associated with $\mathcal{P}$ is defined as

$$\Sigma^{\mathcal{P}}_{PSM} = \{\sigma(P) \mid P \in \mathcal{P}\},$$

where $\sigma(P) = \{\mathcal{I} \mid \mathcal{I}$ is a partial stable model of $P\}$ is the set of all partial stable models of P.

Given two classes $\mathcal{P}_1$ and $\mathcal{P}_2$ of *NLP*s, we say that $\mathcal{P}_1$ and $\mathcal{P}_2$ have the same expressiveness for the partial stable models semantics if $\Sigma^{\mathcal{P}_1}_{PSM} = \Sigma^{\mathcal{P}_2}_{PSM}$.

In other words, $\mathcal{P}_1$ and $\mathcal{P}_2$ have the same expressiveness if

- For every $P_1 \in \mathcal{P}_1$, there exists $P_2 \in \mathcal{P}_2$ such that P_1 and P_2 have the same set of partial stable models.
- For every $P_2 \in \mathcal{P}_2$, there exists $P_1 \in \mathcal{P}_1$ such that P_1 and P_2 have the same set of partial stable models.

Similarly, we can define when $\mathcal{P}_1$ and $\mathcal{P}_2$ have the same expressiveness for the well-founded, regular, stable and L-stable semantics.

As the class of *RALP*s is contained in the class of all *NLP*s, to show that these classes have the same expressiveness for these semantics, it suffices to prove that for every *NLP*, there exists an *RALP* with the same set of partial stable models. We define the corresponding *RALP* of an *NLP* P inspired by the characterisation of $P_{\mathcal{B}_P}$ (Sections 3 and 4):

Definition 19 (Corresponding *RALP*)**.** For any *NLP* P, construct the *NLP* P' inductively as follows:

- If $r \in P$ and $body^+(r) = \emptyset$, then $r \in P'$ and $\texttt{Rules}(r) = \{r\}$;
- If there is a rule $r_0 = a \leftarrow a_1, \ldots, a_n, \texttt{not}\ b_1, \ldots, \texttt{not}\ b_m \in P$ (with $n > 0$) and for each a_i, $1 \leq i \leq n$, there is some rule $r_i \in P'$ such that $head(r_i^*) = a_i$ and $r_0 \notin \texttt{Rules}(r_i)$, then $r = a \leftarrow \texttt{not}\ b'_1, \ldots, \texttt{not}\ b'_q \in P'$ with $\{b'_1, \ldots, b'_q\} = \bigcup_{i=0}^{n} body^-(r_i)$ and $\texttt{Rules}(r) = \{r_0\} \cup \bigcup_{i=1}^{n} \texttt{Rules}(r_i)$.

The corresponding *RALP* P^* of P is the result of removing from the body of the rules in P' occurrences of $\texttt{not}\ x$ for every $x \in HB_P - \{head(r) \mid r \in P'\}$.

Example 15. Let P be the *NLP* with the corresponding *RALP* P^* depicted in Figure 14. Atom e is not the head of any rule in P^*, as the rule $e \leftarrow c, f \in P$ contains a positive body atom f which is not the head of any rule in P. That behaviour mirrors that there is no argument in $\mathcal{B}_P$ with conclusion f (and hence no arguments with conclusion e). Moreover, rule $b \leftarrow a, \texttt{not}\ d$ cannot use $a \leftarrow b, \texttt{not}\ a$ as a subrule, otherwise the latter would be used more than once for deriving a rule in P^*. In addition, $\texttt{not}\ e$ does not occur in P^*, as e is not the head of any rule in P^*. This mimics what occurs in constructing arguments of $\mathcal{B}_P$.

The transformations from *NLP* P to *BAF* $\mathcal{B}_P$ (Section 3) and from *BAF* $\mathcal{B}_P$ to *NLP* $P_{\mathcal{B}_P}$ (Section 4) are the main motivations for defining the corresponding *RALP* P^* of P. Observe how the *RALP*s P^* from Figure 14b and $P_{\mathcal{B}_P}$ from Figure 15b are isomorphic. We show that this is always the case:

Theorem 23. Let P be an *NLP* and P^* be its corresponding *RALP*. It holds P^* and $P_{\mathcal{B}_P}$ are isomorphic.

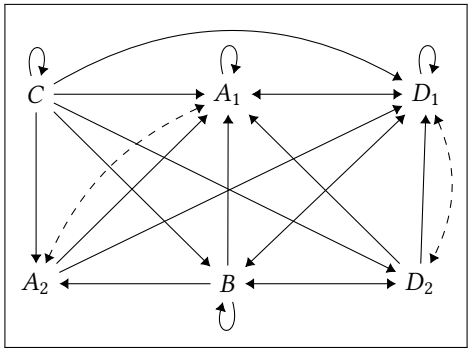
$a \leftarrow b, \texttt{not } a$
$a \leftarrow c, \texttt{not } b$
$b \leftarrow a, \texttt{not } d$
$c \leftarrow \texttt{not } c$
$d \leftarrow a, \texttt{not } e$
$e \leftarrow c, f$

(a) P

$a \leftarrow \texttt{not } a, \texttt{not } b, \texttt{not } c, \texttt{not } d$
$a \leftarrow \texttt{not } b, \texttt{not } c$
$b \leftarrow \texttt{not } b, \texttt{not } c, \texttt{not } d$
$c \leftarrow \texttt{not } c$
$d \leftarrow \texttt{not } a, \texttt{not } b, \texttt{not } c, \texttt{not } d$
$d \leftarrow \texttt{not } b, \texttt{not } c$

(b) P^*

Fig. 14. (a) *NLP* P and (b) its corresponding *RALP* P^*.



(a) $\mathcal{B}_P$

$\{A_1, A_2\} \leftarrow \texttt{not } \{A_1, A_2\}, \texttt{not } \{B\}, \texttt{not } \{C\}, \texttt{not } \{D_1, D_2\}$
$\{A_1, A_2\} \leftarrow \texttt{not } \{B\}, \texttt{not } \{C\}$
$\{B\} \leftarrow \texttt{not } \{B\}, \texttt{not } \{C\}, \texttt{not } \{D_1, D_2\}$
$\{C\} \leftarrow \texttt{not } \{C\}$
$\{D_1, D_2\} \leftarrow \texttt{not } \{A_1, A_2\}, \texttt{not } \{B\}, \texttt{not } \{C\}, \texttt{not } \{D_1, D_2\}$
$\{D_1, D_2\} \leftarrow \texttt{not } \{B\}, \texttt{not } \{C\}$

(b) $P_{\mathcal{B}_P}$

Fig. 15. (a) corresponding *BAF* $\mathcal{B}_P$ of the *NLP* P from Figure 14a, and (b) its corresponding *NLP* $P_{\mathcal{B}_P}$. Arguments A_1, A_2, B, C, D_1, D_2 are respectively $(a \leftarrow (B), \texttt{not } a)$, $(a \leftarrow (C), \texttt{not } b)$, $(b \leftarrow (A_2), \texttt{not } d)$, $(c \leftarrow \texttt{not } c)$, $(d \leftarrow (A_1), \texttt{not } e)$, $(d \leftarrow (A_2), \texttt{not } e)$.

Consequently, the transformation from *NLP* to *RALP* preserves the semantics considered in this paper, as P and P^* share the same partial stable models.

Corollary 24. Let P be an *NLP* with corresponding *RALP* P^*. It holds $\mathcal{M}$ is a partial stable, well-founded, regular, stable and L-stable model of P iff $\mathcal{M}$ is respectively a partial stable, well-founded, regular, stable and L-stable model of P^*.

Given that each *NLP* can be associated with an *RALP* preserving the semantics above, it follows that *NLP*s and *RALP*s have the same expressiveness for those semantics:

Theorem 25. *NLP*s and *RALP*s have the same expressiveness for partial stable, well-founded, regular, stable and L-stable semantics.

Another important result we wish to obtain is the isomorphism between $\mathcal{B}_P$ and $\mathcal{B}_{P^*}$, the corresponding *BAF*s of P and P^*, respectively. This does not hold in general, as shown in the next example:

Example 16. Let P be the *NLP*

$$x \leftarrow$$
$$a \leftarrow \texttt{not}\ x$$
$$a \leftarrow a, \texttt{not}\ x$$

with corresponding *RALP* P^*

$$x \leftarrow$$
$$a \leftarrow \texttt{not}\ x.$$

By Definition 8, from P we can construct 3 arguments (namely, $X_1 = x \leftarrow$, $A_1 = a \leftarrow \texttt{not}\ x$, and $A_2 = a \leftarrow (A_1), \texttt{not}\ x$), whereas from P^* we can only construct 2 arguments ($X_1' = x \leftarrow$ and $A_1' = a \leftarrow \texttt{not}\ x$).

The problem lies in the presence of two distinct arguments (A_1, A_2) with identical conclusion a and vulnerabilities $\{x\}$. As a consequence, $\mathcal{B}_P$ is not an *RFBAF*. We can avoid this by treating each argument A as the pair $(\texttt{Conc}(A), \texttt{Vul}(A))$. Both A_1 and A_2 would instead be the pair $(a, \{x\})$, coinciding with the only argument constructed from P^*. Using this alternative definition of corresponding *BAF*, the corresponding *BAF* of any *NLP* P would be an $\mathfrak{S}$-*RFBAF* (an *RFBAF*, in particular), and we would obtain an equality between the corresponding *BAF*s of P and P^*. In order to avoid introducing another corresponding *BAF* definition, we will instead introduce the assumption that the corresponding *BAF* does not have distinct arguments with identical conclusions and vulnerabilities. Under this assumption, $\mathcal{B}_P$ and $\mathcal{B}_{P^*}$ are isomorphic.

Theorem 26. Let P be an *NLP* and P^* be its corresponding *RALP*. If $\mathcal{B}_P$ is an *RFBAF*, then $\mathcal{B}_P$ and $\mathcal{B}_{P^*}$ are isomorphic.

In addition, as the translations from $\mathfrak{S}$-*RFBAF*s to *RALP*s and, conversely, from *RALP*s to $\mathfrak{S}$-*RFBAF*s are each other's inverse up to isomorphism (Theorems 19 and 22), we obtain that two non-isomorphic $\mathfrak{S}$-*RFBAF*s will always be associated with two non-isomorphic *RALP*s.

Theorem 27. Let P and P' be *RALP*s. It holds P and P' are isomorphic iff $\mathcal{B}_P$ and $\mathcal{B}_{P'}$ are isomorphic.

As the class of *RALP*s is a subset of the class of *NLP*s, many *NLP*s have the same corresponding *BAF*. In particular, *NLP*s with redundancy-free corresponding *BAF*s and with the same corresponding *RALP* have isomorphic corresponding *BAF*s. Next, we prove a more general version of this result:

Theorem 28. Let P_1 and P_2 be *NLP*s with corresponding *RALP*s P_1^* and P_2^*, respectively. If $\mathcal{B}_{P_1}$ and $\mathcal{B}_{P_2}$ are *RFBAF*s, it holds P_1^* and P_2^* are isomorphic iff $\mathcal{B}_{P_1}$ and $\mathcal{B}_{P_2}$ are isomorphic.

The results we have discussed so far also suggest an alternative way to find the corresponding *BAF* of an *NLP* P: instead of resorting directly to Definition 8 to construct arguments, we can first obtain the corresponding *RALP* P^* of P. Then, we apply Definition 8 to this *RALP* to obtain the arguments and Definition 9 for the attack and support relations. Notably, since P^* is an *RALP*, Definition 8 becomes considerably simpler, requiring only its first item to characterise the arguments. Additionally, every argument A of the corresponding *BAF* $\mathcal{B}_P$ in this particular scenario satisfies $|\texttt{Rules}(A)| = 1$ and $\texttt{Sub}(A) = \{A\}$, i.e., argument A can be distinguished from

another argument solely by comparing their conclusions and vulnerabilities, as no argument shares the same set of rules or subarguments. This means that the corresponding *BAF*, when employing this strategy, is always an *RFBAF*, allowing this assumption to be removed from Theorems 26 and 28.

Supported by the findings presented in the current section, we can argue that the class of *NLP*s and *RALP*s are as expressive as each other, and are deeply linked to the corresponding *BAF*s.

7 Conclusions and Future Works

In this paper, we extend the work of Caminada, Sá, et al. (2015) by considering argumentation frameworks with support under the β-semantics from Alcântara and Cordeiro (2024). The guiding principle of this semantics is that (i) if A is accepted, any argument supported by A is also accepted; and (ii) if A is rejected, any argument supporting A is also rejected. We use these properties for establishing correspondences with semantics for *NLP*. We observe that β-complete, β-grounded, β-preferred, β-stable and β-semi-stable semantics of Bipolar Argumentation Frameworks (*BAF*s) respectively coincide with the partial stable, well-founded, regular, stable and L-stable semantics of Normal Logic Programs (*NLP*s).

The proposed translations from *NLP*s to *BAF*s (Section 3), and vice versa (Section 4), show that the inclusion of support in argumentation allows the connection between β-semi-stable and L-stable semantics, whereas Caminada, Sá, et al.'s (2015) translation of *AAF*s into *NLP*s does not guarantee a correspondence between semi-stable and L-stable semantics for attack-only argumentation frameworks. Moreover, although an *NLP* can be translated into an *AAF*, recovering the original *NLP* from the corresponding *AAF* is generally not possible. In contradistinction, our approach reveals a structural equivalence between some class of *BAF*s and some class of *NLP*s, respectively Redundancy-Free *BAF*s of Support cliques ($\mathfrak{S}$-*RFBAF*s) and Reduced Atomic Logic Programs (*RALP*s) (discussed in Section 5), which enables the recovery of an *NLP* with the same attack and support relations as the original *NLP*, except for the arguments' names:

- From the corresponding *BAF* $\mathcal{B}_P$ of an *RALP* P, we can obtain the corresponding *NLP* $P_{\mathcal{B}_P}$, which is isomorphic to P.
- From the corresponding *NLP* $P_{\mathcal{B}}$ of a $\mathfrak{S}$-*RFBAF* $\mathcal{B}$, we can obtain the corresponding *BAF* $\mathcal{B}_{P_{\mathcal{B}}}$, which is isomorphic to $\mathcal{B}$.

Hence, the relationship between *NLP*s and *BAF*s is demonstrably more robust than that between *NLP*s and *AAF*s, extending beyond semantics to encompass structural aspects.

In Section 6, we explain how to translate *NLP*s into *RALP*s and prove that both classes have the same expressiveness under the semantics studied in this paper. This also suggests an alternative way of finding the corresponding *BAF* of an *NLP* P: instead of constructing arguments directly from P, we can first obtain the corresponding *RALP* P^* of P and then obtain arguments from P^*. Since P^* is an *RALP*, the construction presented in Definition 8 becomes considerably simpler, requiring only its first item to characterise the arguments.

In summary, $\mathfrak{S}$-*RFBAF*s and *RALP*s (which are as expressive as *NLP*s) are essentially the same formalism, each encoding knowledge from its own perspective: argumentation emphasises the dynamics of arguments and their interactions, whereas *NLP*s focus on deriving atoms from rules. We provide explicit translations between these formalisms and their semantics, allowing for their interchangeable use and enhancing our understanding of their interrelations.

Regarding the significance and potential impact of our results, we highlight that pursuing this research direction yields insights into which forms of non-monotonic reasoning can and cannot be represented by formal argumentation. In particular, by enlightening these connections between *BAF*s and *NLP*s, many approaches, semantics, and techniques naturally developed for one may be applied to the other, and vice versa. In addition, the representation of *NLP*s as *BAF*s offers an intuitive visualisation of logic programs. Since *NLP*s are closely related to various other formalisms, some of them could benefit from insights derived from argumentation with

support. It also allows connecting *BAF*s to other argumentation formalisms: as there is a translation from *BAF*s (β-complete labellings) to *NLP*s (partial stable models), and from *NLP*s (partial stable models) to *ADF*s (Alcântara, Sá, et al. 2019) (complete labellings), one can indirectly obtain a translation from *BAF*s (β-complete labellings) to *ADF*s (complete labellings). Similar results follow between *BAF*s and *SETAF*s from the connections between *BAF*s and *NLP*s (this paper), and between *NLP*s and *SETAF*s (Alcântara, Cordeiro, and Sá 2024; Toni et al. 2022).

Future works include exploring the connection of *NLP*s with other extensions of *AAF*s and under other semantics. Given that *RALP*s are as expressive as *NLP*s, a possible avenue for investigation is to compare the expressiveness of *BAF*s with the more compact class of $\mathfrak{S}$-*RFBAF*s. Another ramification arising from this work is to identify whether (future) alternative translations for *BAF*s could also preserve the semantics and structures of *NLP*s, and, if so, to compare them to our approach based on the β-semantics.

Acknowledgments

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001 and Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq).

References

J. Alcântara and R. Cordeiro. 2024. "Bipolar Argumentation Frameworks with a dual relation between defeat and defence." *Journal of Logic and Computation*, 35, 2, exae006. eprint: https://academic.oup.com/logcom/article-pdf/35/2/exae006/56912108/exae006.pdf. doi:10.1093/logcom/exae006.

J. Alcântara, R. Cordeiro, and S. Sá. 2024. "On the Equivalence between Logic Programming and SETAF." *Theory and Practice of Logic Programming*, 24, 6, 1208–1236. doi:10.1017/S1471068424000188.

J. Alcântara, S. Sá, and J. Acosta-Guadarrama. 2019. "On the equivalence between abstract dialectical frameworks and logic programs." *Theory and Practice of Logic Programming*, 19, 5-6, 941–956.

G. Alfano, S. Greco, F. Parisi, and I. Trubitsyna. 2024. "Cyclic Supports in Recursive Bipolar Argumentation Frameworks: Semantics and LP Mapping." *Theory and Practice of Logic Programming*, 24, 4, 921–941.

G. Alfano, S. Greco, F. Parisi, and I. Trubitsyna. 2020. "On the semantics of abstract argumentation frameworks: A logic programming approach." *Theory and Practice of Logic Programming*, 20, 5, 703–718.

L. Amgoud, C. Cayrol, M.-C. Lagasquie-Schiex, and P. Livet. 2008. "On bipolarity in argumentation frameworks." *International Journal of Intelligent Systems*, 23, 10, 1062–1093.

G. Brewka and S. Woltran. 2010. "Abstract dialectical frameworks." In: *Twelfth International Conf. on the Principles of Knowledge Representation and Reasoning*. AAAI Press, 102–111.

M. Caminada and L. Amgoud. 2005. "An Axiomatic Account of Formal Argumentation." In: *AAAI*. Vol. 6, 608–613.

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

M. Caminada, S. Sá, J. Alcântara, and W. Dvořák. 2015. "On the equivalence between logic programming semantics and argumentation semantics." *International Journal of Approximate Reasoning*, 58, 87–111.

M. Caminada and C. Schulz. 2017. "On the equivalence between assumption-based argumentation and logic programming." *Journal of Artificial Intelligence Research*, 60, 779–825.

C. Cayrol and M.-C. Lagasquie-Schiex. 2013. "Bipolarity in argumentation graphs: Towards a better understanding." *International Journal of Approximate Reasoning*, 54, 7, 876–899.

A. Cohen, S. Gottifredi, A. J. García, and G. R. Simari. 2014. "A survey of different approaches to support in argumentation systems." *The Knowledge Engineering Review*, 29, 5, 513–550.

R. Cordeiro and J. Alcântara. 2025. "On the Equivalence Between Logic Programs and Bipolar Argumentation Frameworks." In: *Intelligent Systems*. Ed. by A. Paes and F. A. N. Verri. Springer Nature Switzerland, Cham, 238–253.

K. Čyras, C. Schulz, and F. Toni. 2017. "Capturing bipolar argumentation in non-flat assumption-based argumentation." In: *International Conference on Principles and Practice of Multi-Agent Systems*. Springer, 386–402.

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

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

D. M. Gabbay. 2016. "Logical foundations for bipolar and tripolar argumentation networks: preliminary results." *Journal of Logic and Computation*, 26, 1, 247–292.

N. Gorogiannis and A. Hunter. 2011. "Instantiating abstract argumentation with classical logic arguments: Postulates and properties." *Artificial Intelligence*, 175, 9-10, 1479–1497.
F. Nouioua and S. Boutouhami. 2023. "Argumentation frameworks with necessities and their relationship with logic programs." *Argument & Computation*, 14, 1, 17–58.
F. Nouioua and V. Risch. 2011. "Argumentation frameworks with necessities." In: *Scalable Uncertainty Management: 5th International Conference, SUM 2011, Dayton, OH, USA, October 10-13, 2011. Proceedings 5*. Springer, 163–176.
T. Przymusinski. 1990. "The Well-Founded Semantics Coincides with the Three-Valued Stable Semantics." *Fundamenta Informaticae*, 13, 4, 445–463.
V. H. N. Rocha and F. G. Cozman. 2022. "Bipolar Argumentation Frameworks with Explicit Conclusions: Connecting Argumentation and Logic Programming." In: *NMR*, 49–60.
D. Sacca. 1997. "The expressive powers of stable models for bound and unbound DATALOG queries." *Journal of Computer and System Sciences*, 54, 3, 441–464.
H. Strass. 2013. "Approximating operators and semantics for abstract dialectical frameworks." *Artificial Intelligence*, 205, 39–70.
F. Toni et al.. 2022. "Just a matter of perspective." *Computational Models of Argument: Proceedings of COMMA 2022*, 353, 212.
Y. Wu, M. Caminada, and D. M. Gabbay. 2009. "Complete extensions in argumentation coincide with 3-valued stable models in logic programming." *Studia logica*, 93, 2-3, 383.

A Proofs of Theorems

A.1 Theorems and Proofs from Section 3

Proposition 29. (Alcântara and Cordeiro 2024) Let $\mathcal{B} = (\mathcal{A}, Att, Sup)$ be a *BAF* and $\mathcal{L}$ be a β-complete labelling of $\mathcal{B}$. Then for any $A \in \mathcal{A}$,

- $\mathcal{L}(A) = \texttt{in}$ iff $\mathcal{L}(A') = \texttt{in}$ for some $A' \in \mathfrak{Sup}(A)$ iff $\mathcal{L}(A'') = \texttt{in}$ for every $A'' \in \mathcal{A}$ such that $A \in \mathfrak{Sup}(A'')$;
- $\mathcal{L}(A) = \texttt{out}$ iff $\mathcal{L}(A') = \texttt{out}$ for every $A' \in \mathfrak{Sup}(A)$ iff $\mathcal{L}(A'') = \texttt{out}$ for some $A'' \in \mathcal{A}$ such that $A \in \mathfrak{Sup}(A'')$.

Theorem 4. Let P be an *NLP* and $\mathcal{B}_P = (\mathcal{A}_P, Att_P, Sup_P)$ be its corresponding *BAF*. If $\mathcal{L}$ is a β-complete labelling of $\mathcal{B}_P$ and $\texttt{Conc}(A) = \texttt{Conc}(B)$, then $\mathcal{L}(A) = \mathcal{L}(B)$.

Proof. Note $\mathfrak{Sup}(A) = \{A' \in \mathcal{A}_P \mid \texttt{Conc}(A') = \texttt{Conc}(A)\} = \{A' \in \mathcal{A}_P \mid \texttt{Conc}(A') = \texttt{Conc}(B)\} = \mathfrak{Sup}(B)$. It holds

- $\mathcal{L}(A) = \texttt{in}$ iff (Proposition 29) $\mathcal{L}(A') = \texttt{in}$ for some $A' \in \mathfrak{Sup}(A)$ iff $\mathcal{L}(A') = \texttt{in}$ for some $A' \in \mathfrak{Sup}(B)$ iff (Proposition 29) $\mathcal{L}(B) = \texttt{in}$;
- $\mathcal{L}(A) = \texttt{out}$ iff (Proposition 29) $\mathcal{L}(A') = \texttt{out}$ for every $A' \in \mathfrak{Sup}(A)$ iff $\mathcal{L}(A') = \texttt{out}$ for every $A' \in \mathfrak{Sup}(B)$ iff (Proposition 29) $\mathcal{L}(B) = \texttt{out}$.

□

Theorem 5. Let P be an *NLP* and $\mathcal{B}_P = (\mathcal{A}_P, Att_P, Sup_P)$ be its corresponding *BAF*. For any β-complete labelling $\mathcal{L}$ of $\mathcal{B}_P$, it holds $\mathcal{I}2\mathcal{L}_P(\mathcal{L}2\mathcal{I}_P(\mathcal{L})) = \mathcal{L}$.

Proof. Let $A \in \mathcal{A}_P$ and $\mathcal{L}2\mathcal{I}_P(\mathcal{L}) = (T, F, U)$; there are three possibilities:

- $\mathcal{L}(A) = \texttt{in} \Rightarrow \texttt{Conc}(A) \in T \Rightarrow \mathcal{I}2\mathcal{L}_P(\mathcal{L}2\mathcal{I}_P(\mathcal{L}))(A) = \texttt{in}$.
- $\mathcal{L}(A) = \texttt{out} \Rightarrow \mathcal{L}(A') = \texttt{out}$ for every $A' \in \mathcal{A}_P$ such that $\texttt{Conc}(A') = \texttt{Conc}(A)$ (Theorem 4) $\Rightarrow \texttt{Conc}(A) \in F \Rightarrow \mathcal{I}2\mathcal{L}_P(\mathcal{L}2\mathcal{I}_P(\mathcal{L}))(A) = \texttt{out}$.
- $\mathcal{L}(A) = \texttt{undec} \Rightarrow \mathcal{L}(A') = \texttt{undec}$ for every $A' \in \mathcal{A}_P$ such that $\texttt{Conc}(A') = \texttt{Conc}(A)$ (Theorem 4) $\Rightarrow \texttt{Conc}(A) \in U \Rightarrow \mathcal{I}2\mathcal{L}_P(\mathcal{L}2\mathcal{I}_P(\mathcal{L}))(A) = \texttt{undec}$.

□

Theorem 6. Let P be an *NLP*, $\mathcal{B}_P = (\mathcal{A}_P, Att_P)$ be its corresponding *BAF* and $\mathcal{M}$ be a partial stable model of P. It holds that $\mathcal{L}2\mathcal{I}_P(\mathcal{I}2\mathcal{L}_P(\mathcal{M})) = \mathcal{M}$.

Proof. Let $\mathcal{M} = (T, F, U)$ be a partial stable model of P, $\mathcal{L}2\mathcal{I}_P(\mathcal{I}2\mathcal{L}_P(\mathcal{M})) = (T', F', U')$ and $c \in HB_P$. It suffices to prove the following results:

- $c \in T$ iff $c \in T'$.
  - Assume $c \in T$. As $\Psi_P(\mathcal{M}) = \mathcal{M}$, by Lemma 2, there exists an argument $A \in \mathcal{A}_P$ with $\mathtt{Conc}(A) = c$ such that $\mathtt{Vul}(A) \subseteq F$. As $\mathtt{Conc}(A) = c \in T$, it follows $\mathcal{I}2\mathcal{L}_P(\mathcal{M})(A) = \mathtt{in}$. Consequently, $\mathcal{L}2\mathcal{I}_P(\mathcal{I}2\mathcal{L}_P(\mathcal{M}))(c) = \mathtt{in}$, i.e., $c \in T'$.
  - Assume $c \in T'$, i.e., $\mathcal{L}2\mathcal{I}_P(\mathcal{I}2\mathcal{L}_P(\mathcal{M}))(c) = \mathtt{in}$. Then, there exists an argument $A \in \mathcal{A}_p$ with $\mathtt{Conc}(A) = c$ and $\mathcal{I}2\mathcal{L}_P(\mathcal{M})(A) = \mathtt{in}$. It follows $\mathtt{Conc}(A) = c \in T$.
- $c \in F$ iff $c \in F'$.
  - Assume $c \notin F'$. Then there exists an argument $A \in \mathcal{A}_P$ with $\mathtt{Conc}(A) = c$ and $\mathcal{I}2\mathcal{L}_P(\mathcal{M})(A) \neq \mathtt{out}$. It follows $\mathtt{Conc}(A) = c \notin F$.
  - Assume $c \notin F$. As $\Psi_P(\mathcal{M}) = \mathcal{M}$, by Lemma 2, there exists an argument $A \in \mathcal{A}_P$ with $\mathtt{Conc}(A) = c$ such that $\mathtt{Vul}(A) \cap T = \emptyset$. As $\mathtt{Conc}(A) = c \notin F$, it follows $\mathcal{I}2\mathcal{L}_P(\mathcal{M})(A) \neq \mathtt{out}$. Then, $\mathtt{Conc}(A) = c \notin F'$.

□

Proposition 7. Let P be an *NLP* and $\mathcal{B}_P = (\mathcal{A}_P, Att_P, Sup_P)$ be its corresponding *BAF*. If $\mathcal{L}$ is a β-complete labelling of $\mathcal{B}_P$, then

$$\mathcal{L}2\mathcal{I}_P(\mathcal{L})(c) = \begin{cases} \mathbf{t} & \exists A \in \mathcal{A}_P \textit{ such that } \mathtt{Conc}(A) = c \textit{ and } \mathcal{L}(A) = \mathtt{in} \\ \mathbf{u} & \exists A \in \mathcal{A}_P \textit{ such that } \mathtt{Conc}(A) = c \textit{ and } \mathcal{L}(A) = \mathtt{undec} \\ \mathbf{f} & \textit{otherwise.} \end{cases}$$

Proof. Let $c \in HB_P$. It holds

- If there is no $A \in \mathcal{A}_P$ such that $\mathtt{Conc}(A) = c$, it follows $\mathcal{L}2\mathcal{I}_P(\mathcal{L})(c) = \mathbf{f}$.
- If there exists $A \in \mathcal{A}_P$ such that $\mathtt{Conc}(A) = c$, it follows $\mathcal{L}(A) = \mathcal{L}(A')$ whenever $\mathtt{Conc}(A) = \mathtt{Conc}(A')$ (Theorem 4). Thus

$$\mathcal{L}2\mathcal{I}_P(\mathcal{L})(\mathtt{Conc}(A)) = \begin{cases} \mathbf{t} & \textit{if } \mathcal{L}(A) = \mathtt{in} \\ \mathbf{f} & \textit{if } \mathcal{L}(A) = \mathtt{out} \\ \mathbf{u} & \textit{if } \mathcal{L}(A) = \mathtt{undec}. \end{cases}$$

□

Theorem 8. Let P be an *NLP* and $\mathcal{B}_P = (\mathcal{A}_P, Att_P)$ be its corresponding *BAF*. It holds

- $\mathcal{L}$ is a β-complete labelling of $\mathcal{B}_P$ iff $\mathcal{L}2\mathcal{I}_P(\mathcal{L})$ is a partial stable model of P.
- $\mathcal{M}$ is a partial stable model of P iff $\mathcal{I}2\mathcal{L}_P(\mathcal{M})$ is a β-complete labelling of $\mathcal{B}_P$.

Proof.

(1) If $\mathcal{L}$ is a β-complete labelling of $\mathcal{B}_P$, then $\mathcal{L}2\mathcal{I}_P(\mathcal{L})$ is a partial stable model of P:
Let $\mathcal{M} = \mathcal{L}2\mathcal{I}_P(\mathcal{L}) = (T, F, U)$. We will show $\mathcal{M}$ is a partial stable model of P, i.e., $\Psi_P(\mathcal{M}) = (T', F', U') = (T, F, U)$:
   - $c \in T$ iff there exists $A \in \mathcal{A}_P$ such that $\mathtt{Conc}(A) = c$ and $\mathcal{L}(A) = \mathtt{in}$ iff there exists $A \in \mathcal{A}_P$ such that $\mathtt{Conc}(A) = c$ and there exists $A' \in \mathfrak{Sup}(A)$ such that for every $B \in Att(A')$, it holds $\mathcal{L}(B) = \mathtt{out}$ iff there exists $A \in \mathcal{A}_P$ such that $\mathtt{Conc}(A) = c$ and there exists $A' \in \mathcal{A}_P$ such that $\mathtt{Conc}(A') = c$ and for every $B \in Att(A')$, it holds $\mathcal{L}(B) = \mathtt{out}$ iff there exists $A \in \mathcal{A}_P$ such that $\mathtt{Conc}(A) = c$ and there exists $A' \in \mathcal{A}_P$ such that $\mathtt{Conc}(A') = c$ and $\mathtt{Vul}(A') \subseteq F$ iff there exists $A \in \mathcal{A}_P$ such that $\mathtt{Conc}(A) = c$ and $\mathtt{Vul}(A) \subseteq F$ iff (Lemma 2) $c \in T'$.
   - $c \notin F$ iff there exists $A \in \mathcal{A}_P$ such that $\mathtt{Conc}(A) = c$ and $\mathcal{L}(A) \neq \mathtt{out}$ iff there exists $A \in \mathcal{A}_P$ such that $\mathtt{Conc}(A) = c$ and there exists $A' \in \mathfrak{Sup}(A)$ such that for every $B \in Att(A')$, it holds $\mathcal{L}(B) \neq \mathtt{in}$ iff there exists $A \in \mathcal{A}_P$ such that $\mathtt{Conc}(A) = c$ and there exists $A' \in \mathcal{A}_P$ such that $\mathtt{Conc}(A') = c$ and for

every $B \in Att(A')$, it holds $\mathcal{L}(B) \neq \texttt{in}$ iff there exists $A \in \mathcal{A}_P$ such that $\texttt{Conc}(A) = c$ and there exists $A' \in \mathcal{A}_P$ such that $\texttt{Conc}(A') = c$ and $\texttt{Vul}(A') \cap T = \emptyset$ iff there exists $A \in \mathcal{A}_P$ such that $\texttt{Conc}(A) = c$ and $\texttt{Vul}(A) \cap T = \emptyset$ iff (Lemma 2) $c \notin F'$.

(2) If $\mathcal{M}$ is a partial stable model of P, then $\mathcal{I}2\mathcal{L}_P(\mathcal{M})$ is a β-complete labelling of $\mathcal{B}_P$:
Let $\mathcal{M} = (T, F, U)$ be a partial stable model of P. Then $\Psi_P(\mathcal{M}) = (T, F, U)$. We will prove $\mathcal{L} = \mathcal{I}2\mathcal{L}_P(\mathcal{M})$ is a β-complete labelling of $\mathcal{B}_P$. Let A be an argument in $\mathcal{A}_P$.
- $\mathcal{L}(A) = \texttt{in}$ iff $\texttt{Conc}(A) \in T$ iff (Lemma 2) there exists $A' \in \mathcal{A}_P$ with $\texttt{Conc}(A') = \texttt{Conc}(A)$ and $\texttt{Vul}(A') \subseteq F$ iff there exists $A' \in \mathfrak{Sup}(A)$ such that for every $B \in Att(A')$, it holds $\mathcal{L}(B) = \texttt{out}$.
- $\mathcal{L}(A) \neq \texttt{out}$ iff $\texttt{Conc}(A) \notin F$ iff (Lemma 2) there exists $A' \in \mathcal{A}_P$ with $\texttt{Conc}(A') = \texttt{Conc}(A)$ and $\texttt{Vul}(A') \cap T = \emptyset$ iff there exists $A' \in \mathfrak{Sup}(A)$ such that for every $B \in Att(A')$, it holds $\mathcal{L}(B) \neq \texttt{in}$.

(3) If $\mathcal{L}2\mathcal{I}_P(\mathcal{L})$ is a partial stable model of P, then $\mathcal{L}$ is a β-complete labelling of $\mathcal{B}_P$:
It holds that $\mathcal{L}2\mathcal{I}_P(\mathcal{L})$ is a partial stable model of $P \Rightarrow$ according to item 2 above, $\mathcal{I}2\mathcal{L}_P(\mathcal{L}2\mathcal{I}_P(\mathcal{L}))$ is a β-complete labelling of $\mathcal{B}_P \Rightarrow$ (via Theorem 5) $\mathcal{L}$ is a β-complete labelling of $\mathcal{B}_P$.

(4) If $\mathcal{I}2\mathcal{L}_P(\mathcal{M})$ is a β-complete labelling of $\mathcal{B}_P$, then $\mathcal{M}$ is a partial stable model of P:
It holds that $\mathcal{I}2\mathcal{L}_P(\mathcal{M})$ is a β-complete labelling of $\mathcal{B}_P \Rightarrow$ according to item 1 above, $\mathcal{L}2\mathcal{I}_P(\mathcal{I}2\mathcal{L}_P(\mathcal{M}))$ is a partial stable model of $P \Rightarrow$ (via Theorem 6) $\mathcal{M}$ is a partial stable model of P.

□

Lemma 30. Let P be an *NLP*, $\mathcal{B}_P = (\mathcal{A}_P, Att_P, Sup_P)$ be its associated *BAF*. Let $\mathcal{L}_1$ and $\mathcal{L}_2$ be labellings of $\mathcal{B}_P$ respecting $\mathfrak{Sup}$, and $\mathcal{L}2\mathcal{I}_P(\mathcal{L}_1) = (T_1, F_1, U_1)$ and $\mathcal{L}2\mathcal{I}_P(\mathcal{L}_2) = (T_2, F_2, U_2)$. It holds

(1) $\texttt{in}(\mathcal{L}_1) \subseteq \texttt{in}(\mathcal{L}_2)$ iff $T_1 \subseteq T_2$;
(2) $\texttt{in}(\mathcal{L}_1) = \texttt{in}(\mathcal{L}_2)$ iff $T_1 = T_2$;
(3) $\texttt{in}(\mathcal{L}_1) \subset \texttt{in}(\mathcal{L}_2)$ iff $T_1 \subset T_2$.

Proof.

(1) ($\Longrightarrow$) Suppose $\texttt{in}(\mathcal{L}_1) \subseteq \texttt{in}(\mathcal{L}_2)$. If $c \in T_1$, by Definition 11, there exists $A \in \mathcal{A}_P$ such that $\texttt{Conc}(A) = c$ and $\mathcal{L}_1(A) = \texttt{in}$. From our initial assumption, it follows $\mathcal{L}_2(A) = \texttt{in}$. So, by Definition 11, $c \in T_2$.
($\Longleftarrow$) Suppose $T_1 \subseteq T_2$. If $\mathcal{L}_1(A) = \texttt{in}$, by Definition 11, $\texttt{Conc}(A) \in T_1$. From our initial assumption, it follows $\texttt{Conc}(A) \in T_2$. So, by Definition 11, there is some argument $A' \in \mathcal{A}_P$ with $\texttt{Conc}(A') = \texttt{Conc}(A)$ and $\mathcal{L}_2(A') = \texttt{in}$. As $\mathcal{L}_2$ respects $\mathfrak{Sup}$, it follows $\mathcal{L}_2(A) = \texttt{in}$.
(2) It follows directly from point 1.
(3) It follows directly from points 1 and 2.

□

Lemma 31. Let P be an *NLP*, $\mathcal{B}_P = (\mathcal{A}_P, Att_P, Sup_P)$ be its associated *BAF*. Let $\mathcal{L}_1$ and $\mathcal{L}_2$ be labellings of $\mathcal{B}_P$ respecting $\mathfrak{Sup}$, and $\mathcal{L}2\mathcal{I}_P(\mathcal{L}_1) = (T_1, F_1, U_1)$ and $\mathcal{L}2\mathcal{I}_P(\mathcal{L}_2) = (T_2, F_2, U_2)$. It holds

(1) $\texttt{undec}(\mathcal{L}_1) \subseteq \texttt{undec}(\mathcal{L}_2)$ iff $U_1 \subseteq U_2$;
(2) $\texttt{undec}(\mathcal{L}_1) = \texttt{undec}(\mathcal{L}_2)$ iff $U_1 = U_2$;
(3) $\texttt{undec}(\mathcal{L}_1) \subset \texttt{undec}(\mathcal{L}_2)$ iff $U_1 \subset U_2$.

Proof.

(1) ($\Longrightarrow$) Suppose $\texttt{undec}(\mathcal{L}_1) \subseteq \texttt{undec}(\mathcal{L}_2)$. If $c \in U_1$, by Definition 11, there exists an argument $A \in \mathcal{A}_P$ with $\texttt{Conc}(A) = c$ and $\mathcal{L}_1(A) = \texttt{undec}$, and for every argument $A \in \mathcal{A}_P$ with $\texttt{Conc}(A) = c$ it holds $\mathcal{L}_1(A) \neq \texttt{in}$. By our initial assumption, there exists an argument $A \in \mathcal{A}_P$ with $\texttt{Conc}(A) = c$ and $\mathcal{L}_2(A) = \texttt{undec}$. As $\mathcal{L}_2$ respects $\mathfrak{Sup}$, for every argument $A' \in \mathcal{A}_P$ with $\texttt{Conc}(A') = c$ it holds $\mathcal{L}_2(A') = \texttt{undec}$. So, by Definition 11, $c \in U_2$.

($\Longleftarrow$) Suppose $U_1 \subseteq U_2$. Assume $\mathcal{L}_1(A) = \mathtt{undec}$. As $\mathcal{L}_1$ respects $\mathfrak{Sup}$, for every argument $A' \in \mathcal{A}_P$ with $\mathtt{Conc}(A') = \mathtt{Conc}(A)$ it holds $\mathcal{L}_1(A') = \mathtt{undec}$. By Definition 11, it follows $\mathtt{Conc}(A) \in U_1$. From our initial assumption, it follows $\mathtt{Conc}(A) \in U_2$. As $\mathcal{L}_2$ respects $\mathfrak{Sup}$, for every argument $A' \in \mathcal{A}_P$ with $\mathtt{Conc}(A') = \mathtt{Conc}(A)$ it holds $\mathcal{L}_2(A') = \mathtt{undec}$. In particular, $\mathcal{L}_2(A) = \mathtt{undec}$.

(2) It follows directly from point 1.

(3) It follows directly from points 1 and 2.

□

Theorem 9. Let P be an *NLP* and $\mathcal{B}_P = (\mathcal{A}_P, Att_P, Sup_P)$ be its corresponding *BAF*. It holds

(1) $\mathcal{L}$ is a β-grounded labelling of $\mathcal{B}_P$ iff $\mathcal{L}2\mathcal{I}_P(\mathcal{L})$ is a well-founded model of P.

(2) $\mathcal{L}$ is a β-preferred labelling of $\mathcal{B}_P$ iff $\mathcal{L}2\mathcal{I}_P(\mathcal{L})$ is a regular model of P.

(3) $\mathcal{L}$ is a β-stable labelling of $\mathcal{B}_P$ iff $\mathcal{L}2\mathcal{I}_P(\mathcal{L})$ is a stable model of P.

(4) $\mathcal{L}$ is a β-semi-stable labelling of $\mathcal{B}_P$ iff $\mathcal{L}2\mathcal{I}_P(\mathcal{L})$ is an L-stable model of P.

Proof. Let $\mathcal{L}$ be a labelling of $\mathcal{B}_P$ and $\mathcal{L}2\mathcal{I}_P(\mathcal{L}) = (T, F, U)$. The proof is straightforward:

(1) $\mathcal{L}$ is a β-grounded labelling of $\mathcal{B}_P$ iff $\mathcal{L}$ is a β-complete labelling of $\mathcal{B}_P$, and $\mathtt{in}(\mathcal{L})$ is minimal (w.r.t. set inclusion) among all β-complete labellings of $\mathcal{B}_P$ iff (Theorem 8, Proposition 11 and Lemma 30) $\mathcal{L}2\mathcal{I}_P(\mathcal{L})$ is a partial stable model of P, and there is no partial stable model $\mathcal{M}' = (T', F', U')$ of P such that $T' \subset T$ iff $\mathcal{L}2\mathcal{I}_P(\mathcal{L})$ is a well-founded model of P;

(2) $\mathcal{L}$ is a β-preferred labelling of $\mathcal{B}_P$ iff $\mathcal{L}$ is a β-complete labelling of $\mathcal{B}_P$, and $\mathtt{in}(\mathcal{L})$ is maximal (w.r.t. set inclusion) among all β-complete labellings of $\mathcal{B}_P$ iff (Theorem 8, Proposition 11 and Lemma 30) $\mathcal{L}2\mathcal{I}_P(\mathcal{L})$ is a partial stable model of P, and there is no partial stable model $\mathcal{M}' = (T', F', U')$ of P such that $T \subset T'$ iff $\mathcal{L}2\mathcal{I}_P(\mathcal{L})$ is a regular model of P;

(3) $\mathcal{L}$ is a β-stable labelling of $\mathcal{B}_P$ iff $\mathcal{L}$ is a β-complete labelling of $\mathcal{B}_P$ and $\mathtt{undec}(\mathcal{L}) = \emptyset$ iff (Theorem 8) $\mathcal{L}2\mathcal{I}_P(\mathcal{L})$ is a partial stable model such that $U = \emptyset$ iff $\mathcal{L}2\mathcal{I}_P(\mathcal{L})$ is a stable model of P;

(4) $\mathcal{L}$ is a β-semi-stable labelling of $\mathcal{B}_P$ iff $\mathcal{L}$ is a β-complete labelling of $\mathcal{B}_P$, and $\mathtt{undec}(\mathcal{L})$ is minimal (w.r.t. set inclusion) among all β-complete labellings of $\mathcal{B}_P$ iff (Theorem 8, Proposition 11 and Lemma 31) $\mathcal{L}2\mathcal{I}_P(\mathcal{L})$ is a partial stable model of P, and there is no partial stable model $\mathcal{M}' = (T', F', U')$ of P such that $U' \subset U$ iff $\mathcal{L}2\mathcal{I}_P(\mathcal{L})$ is an L-stable model of P.

□

Corollary 10. Let P be an *NLP* and $\mathcal{B}_P = (\mathcal{A}_P, Att_P, Sup_P)$ be its corresponding *BAF*. It holds

(1) $\mathcal{M}$ is a well-founded model of P iff $\mathcal{I}2\mathcal{L}_P(\mathcal{M})$ is a β-grounded labelling of $\mathcal{B}_P$.

(2) $\mathcal{M}$ is a regular model of P iff $\mathcal{I}2\mathcal{L}_P(\mathcal{M})$ is a β-preferred labelling of $\mathcal{B}_P$.

(3) $\mathcal{M}$ is a stable model of P iff $\mathcal{I}2\mathcal{L}_P(\mathcal{M})$ is a β-stable labelling of $\mathcal{B}_P$.

(4) $\mathcal{M}$ is an L-stable model of P iff $\mathcal{I}2\mathcal{L}_P(\mathcal{M})$ is a β-semi-stable labelling of $\mathcal{B}_P$.

Proof. These results come from Theorems 6 and 9. □

A.2 Theorems and Proofs from Section 4

Proposition 11. If $\mathcal{L}$ is a β-complete labelling of $\mathcal{B} = (\mathcal{A}, Att, Sup)$, then $\mathcal{L}$ respects $\mathfrak{Sup}$.

Proof. Let $A, A' \in \mathcal{A}$ such that $\mathfrak{Sup}(A) = \mathfrak{Sup}(A')$.

- $\mathcal{L}(A) = \mathtt{in} \Longleftrightarrow \exists B \in \mathfrak{Sup}(A) : Att(B) \subseteq \mathtt{out}(\mathcal{L}) \Longleftrightarrow \exists B \in \mathfrak{Sup}(A') : Att(B) \subseteq \mathtt{out}(\mathcal{L}) \Longleftrightarrow \mathcal{L}(A') = \mathtt{in}$.
- $\mathcal{L}(A) = \mathtt{out} \Longleftrightarrow \forall B \in \mathfrak{Sup}(A) : Att(B) \cap \mathtt{in}(\mathcal{L}) \neq \emptyset \Longleftrightarrow \forall B \in \mathfrak{Sup}(A') : Att(B) \cap \mathtt{in}(\mathcal{L}) \neq \emptyset \Longleftrightarrow \mathcal{L}(A') = \mathtt{out}$.

□

Theorem 12. Let $\mathcal{B} = (\mathcal{A}, Att, Sup)$ be a *BAF* and $P_{\mathcal{B}}$ be its corresponding *NLP*.

- For any labelling $\mathcal{L}$ of $\mathcal{B}$ respecting $\mathfrak{Sup}$, it holds $\mathcal{I}2\mathcal{L}_{\mathcal{B}}(\mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L})) = \mathcal{L}$.
- For any interpretation $\mathcal{I}$ of $P_{\mathcal{B}}$, it holds $\mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{I}2\mathcal{L}_{\mathcal{B}}(\mathcal{I})) = \mathcal{I}$.

Proof. Both results are immediate:

- Let $\mathcal{L}$ be a labelling of $\mathcal{B}$ respecting $\mathfrak{Sup}$.
  - $\mathcal{L}(A) = \mathtt{in} \Rightarrow \mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L})(\mathfrak{Sup}(A)) = \mathbf{t} \Rightarrow \mathcal{I}2\mathcal{L}_{\mathcal{B}}(\mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L}))(A) = \mathtt{in}$.
  - $\mathcal{L}(A) = \mathtt{out} \Rightarrow \mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L})(\mathfrak{Sup}(A)) = \mathbf{f} \Rightarrow \mathcal{I}2\mathcal{L}_{\mathcal{B}}(\mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L}))(A) = \mathtt{out}$.
  - $\mathcal{L}(A) = \mathtt{undec} \Rightarrow \mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L})(\mathfrak{Sup}(A)) = \mathbf{u} \Rightarrow \mathcal{I}2\mathcal{L}_{\mathcal{B}}(\mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L}))(A) = \mathtt{undec}$.
- Let $\mathcal{I}$ be an interpretation of $P_{\mathcal{B}}$.
  - $\mathcal{I}(\mathfrak{Sup}(A)) = \mathbf{t} \Rightarrow \mathcal{I}2\mathcal{L}_{\mathcal{B}}(\mathcal{I})(A) = \mathtt{in} \Rightarrow \mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{I}2\mathcal{L}_{\mathcal{B}}(\mathcal{I}))(\mathfrak{Sup}(A)) = \mathbf{t}$.
  - $\mathcal{I}(\mathfrak{Sup}(A)) = \mathbf{f} \Rightarrow \mathcal{I}2\mathcal{L}_{\mathcal{B}}(\mathcal{I})(A) = \mathtt{out} \Rightarrow \mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{I}2\mathcal{L}_{\mathcal{B}}(\mathcal{I}))(\mathfrak{Sup}(A)) = \mathbf{f}$.
  - $\mathcal{I}(\mathfrak{Sup}(A)) = \mathbf{u} \Rightarrow \mathcal{I}2\mathcal{L}_{\mathcal{B}}(\mathcal{I})(A) = \mathtt{undec} \Rightarrow \mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{I}2\mathcal{L}_{\mathcal{B}}(\mathcal{I}))(\mathfrak{Sup}(A)) = \mathbf{u}$.

□

Lemma 13. Let $\mathcal{B} = (\mathcal{A}, Att, Sup)$ be a *BAF* and $P_{\mathcal{B}}$ be its corresponding *NLP*. Let $\mathcal{L}$ be a labelling of $\mathcal{B}$ respecting $\mathfrak{Sup}$ and $\mathcal{M}$ be an interpretation of $P_{\mathcal{B}}$. If $\mathcal{L} = \mathcal{I}2\mathcal{L}_{\mathcal{B}}(\mathcal{M})$ or $\mathcal{M} = \mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L})$, then for any $A \in \mathcal{A}$,

- There exists $B \in \mathfrak{Sup}(A)$ such that $Att(B) \subseteq \mathtt{out}(\mathcal{L})$ iff there exists $r \in P_{\mathcal{B}}$ with $head(r) = \mathfrak{Sup}(A)$ such that $\{\mathfrak{Sup}(X) \mid \mathtt{not}\ \mathfrak{Sup}(X) \in body^-(r)\} \subseteq \mathbf{f}(\mathcal{M})$.
- For every $B \in \mathfrak{Sup}(A)$ it holds $Att(B) \cap \mathtt{in}(\mathcal{L}) \neq \emptyset$ iff for every rule $r \in P_{\mathcal{B}}$ with $head(r) = \mathfrak{Sup}(A)$ it holds $\{\mathfrak{Sup}(X) \mid \mathtt{not}\ \mathfrak{Sup}(X) \in body^-(r)\} \cap \mathbf{t}(\mathcal{M}) \neq \emptyset$.

Proof. It follows straightforwardly from Definitions 12 and 14, as $\mathbf{t}(\mathcal{M}) = \{\mathfrak{Sup}(X) \mid X \in \mathtt{in}(\mathcal{L})\}$, $\mathbf{f}(\mathcal{M}) = \{\mathfrak{Sup}(X) \mid X \in \mathtt{out}(\mathcal{L})\}$ and $body^-(r_{A,B}) = \{\mathtt{not}\ \mathfrak{Sup}(X) \mid X \in Att(B)\}$. □

Theorem 14. Let $\mathcal{B} = (\mathcal{A}, Att, Sup)$ be a *BAF*, $P_{\mathcal{B}}$ be its corresponding *NLP*, $\mathcal{L}$ be a labelling of $\mathcal{B}$ respecting $\mathfrak{Sup}$ and $\mathcal{M}$ be an interpretation of $P_{\mathcal{B}}$. It holds

- $\mathcal{L}$ is a β-complete labelling of $\mathcal{B}$ iff $\mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L})$ is a partial stable model of $P_{\mathcal{B}}$.
- $\mathcal{M}$ is a partial stable model of $P_{\mathcal{B}}$ iff $\mathcal{I}2\mathcal{L}_{\mathcal{B}}(\mathcal{M})$ is a β-complete labelling of $\mathcal{B}$.

Proof.

- Let $\mathcal{I} = \mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L})$ be an interpretation of $P_{\mathcal{B}}$. Recall that the least model of $P_{\mathcal{B}}/\mathcal{I}$ is denoted as $\psi_{P_{\mathcal{B}}}(\mathcal{I})$. ($\Longrightarrow$) Assume $\mathcal{L}$ is a β-complete labelling of $\mathcal{B}$. We will prove that $\mathcal{I}$ is a partial stable model of $P_{\mathcal{B}}$ by showing $\mathbf{t}(\mathcal{I}) = \mathbf{t}(\psi_{P_{\mathcal{B}}}(\mathcal{I}))$ and $\mathbf{f}(\mathcal{I}) = \mathbf{f}(\psi_{P_{\mathcal{B}}}(\mathcal{I}))$. Recall that $body^+(r) = \emptyset$ for every rule r in $P_{\mathcal{B}}$. Hence, any rule r in $P_{\mathcal{B}}/\mathcal{I}$ satisfies $body^+(r) = \emptyset$ or $body^+(r) = \{\mathbf{u}\}$.

$$
\begin{aligned}
&\mathfrak{Sup}(A) \in \mathbf{t}(\mathcal{I}) \\
&\iff \mathcal{L}(A) = \mathtt{in} \\
&\iff \exists B \in \mathfrak{Sup}(A) : Att(B) \subseteq \mathtt{out}(\mathcal{L}) \\
&\overset{\text{Lemma 13}}{\iff} \exists r \in P_{\mathcal{B}} : head(r) = \mathfrak{Sup}(A) \wedge \\
&\qquad \{\mathfrak{Sup}(X) \mid \mathtt{not}\ \mathfrak{Sup}(X) \in body^-(r)\} \subseteq \mathbf{f}(\mathcal{I}) \\
&\iff \exists r \in P_{\mathcal{B}}/\mathcal{I} : head(r) = \mathfrak{Sup}(A) \wedge body^+(r) = \emptyset \\
&\iff \exists r \in P_{\mathcal{B}}/\mathcal{I} : head(r) = \mathfrak{Sup}(A) \wedge body^+(r) \subseteq \mathbf{t}(\psi_{P_{\mathcal{B}}}(\mathcal{I})) \\
&\iff \mathfrak{Sup}(A) \in \mathbf{t}(\psi_{P_{\mathcal{B}}}(\mathcal{I})).
\end{aligned}
$$

$$
\begin{aligned}
&\mathfrak{Sup}(A) \in \mathbf{f}(\mathcal{I}) \\
&\iff \mathcal{L}(A) = \mathtt{out} \\
&\iff \forall B \in \mathfrak{Sup}(A) : Att(B) \cap \mathtt{in}(\mathcal{L}) \neq \emptyset \\
&\overset{\text{Lemma 13}}{\iff} \forall r \in P_{\mathcal{B}} : head(r) = \mathfrak{Sup}(A) \rightarrow \\
&\qquad \{\mathfrak{Sup}(X) \mid \mathtt{not}\ \mathfrak{Sup}(X) \in body^-(r)\} \cap \mathbf{t}(\mathcal{I}) \neq \emptyset \\
&\iff \neg\exists r \in P_{\mathcal{B}}/\mathcal{I} : head(r) = \mathfrak{Sup}(A) \\
&\iff \forall r \in P_{\mathcal{B}}/\mathcal{I} : head(r) = \mathfrak{Sup}(A) \rightarrow body^+(r) \cap \mathbf{f}(\psi_{P_{\mathcal{B}}}(\mathcal{I})) \neq \emptyset \\
&\iff \mathfrak{Sup}(A) \in \mathbf{f}(\psi_{P_{\mathcal{B}}}(\mathcal{I})).
\end{aligned}
$$

($\Longleftarrow$) Assume $\mathcal{I}$ is a partial stable model of $P_{\mathcal{B}}$, i.e., $\psi_{P_{\mathcal{B}}}(\mathcal{I}) = \mathcal{I}$. We will prove that $\mathcal{L}$ is a β-complete labelling of $\mathcal{B}$. Let $A \in \mathcal{A}$.

$$
\begin{aligned}
&\mathcal{L}(A) = \mathtt{in} \\
&\iff \mathfrak{Sup}(A) \in \mathbf{t}(\mathcal{I}) \\
&\iff \mathfrak{Sup}(A) \in \mathbf{t}(\psi_{P_{\mathcal{B}}}(\mathcal{I})) \\
&\iff \exists r \in P_{\mathcal{B}}/\mathcal{I} : head(r) = \mathfrak{Sup}(A) \wedge body^+(r) \subseteq \mathbf{t}(\psi_{P_{\mathcal{B}}}(\mathcal{I})) \\
&\iff \exists r \in P_{\mathcal{B}}/\mathcal{I} : head(r) = \mathfrak{Sup}(A) \wedge body^+(r) = \emptyset \\
&\iff \exists r \in P_{\mathcal{B}} : head(r) = \mathfrak{Sup}(A) \wedge \\
&\qquad \{\mathfrak{Sup}(X) \mid \mathtt{not}\ \mathfrak{Sup}(X) \in body^-(r)\} \subseteq \mathbf{f}(\mathcal{I}) \\
&\overset{\text{Lemma 13}}{\iff} \exists B \in \mathfrak{Sup}(A) : Att(B) \subseteq \mathtt{out}(\mathcal{L}).
\end{aligned}
$$

$$
\begin{aligned}
&\mathcal{L}(A) = \mathtt{out} \\
&\iff \mathfrak{Sup}(A) \in \mathbf{f}(\mathcal{I}) \\
&\iff \mathfrak{Sup}(A) \in \mathbf{f}(\psi_{P_{\mathcal{B}}}(\mathcal{I})) \\
&\iff \forall r \in P_{\mathcal{B}}/\mathcal{I} : head(r) = \mathfrak{Sup}(A) \rightarrow body^+(r) \cap \mathbf{f}(\psi_{P_{\mathcal{B}}}(\mathcal{I})) \neq \emptyset \\
&\iff \neg\exists r \in P_{\mathcal{B}}/\mathcal{I} : head(r) = \mathfrak{Sup}(A) \\
&\iff \forall r \in P_{\mathcal{B}} : head(r) = \mathfrak{Sup}(A) \rightarrow \\
&\qquad \{\mathfrak{Sup}(X) \mid \mathtt{not}\ \mathfrak{Sup}(X) \in body^-(r)\} \cap \mathbf{t}(\mathcal{I}) \neq \emptyset \\
&\overset{\text{Lemma 13}}{\iff} \forall B \in \mathfrak{Sup}(A) : Att(B) \cap \mathtt{in}(\mathcal{L}) \neq \emptyset.
\end{aligned}
$$

- Let $\mathcal{L}' = \mathcal{I}2\mathcal{L}_{\mathcal{B}}(\mathcal{M})$ be a labelling of $\mathcal{B}$. Recall that the least model of $P_{\mathcal{B}}/\mathcal{M}$ is denoted as $\psi_{P_{\mathcal{B}}}(\mathcal{M})$, and that $body^+(r) = \emptyset$ for every rule r in $P_{\mathcal{B}}$. Hence, any rule r in $P_{\mathcal{B}}/\mathcal{M}$ satisfies $body^+(r) = \emptyset$ or $body^+(r) = \{\mathbf{u}\}$. ($\Longrightarrow$) Assume $\mathcal{M}$ is a partial stable model of $P_{\mathcal{B}}$, i.e., $\psi_{P_{\mathcal{B}}}(\mathcal{M}) = \mathcal{M}$. We will show that $\mathcal{L}'$ is a β-complete labelling of $\mathcal{B}$. Recall that $\mathcal{L}'$ respects $\mathfrak{Sup}$ by $\mathcal{I}2\mathcal{L}_{\mathcal{B}}$'s definition. Let $A \in \mathcal{A}$.

$$
\begin{aligned}
&\mathcal{L}'(A) = \texttt{in} \\
&\iff \mathfrak{Sup}(A) \in \mathbf{t}(\mathcal{M}) \\
&\iff \mathfrak{Sup}(A) \in \mathbf{t}(\psi_{P_{\mathcal{B}}}(\mathcal{M})) \\
&\iff \exists r \in P_{\mathcal{B}}/\mathcal{M} : head(r) = \mathfrak{Sup}(A) \wedge body^+(r) \subseteq \mathbf{t}(\psi_{P_{\mathcal{B}}}(\mathcal{M})) \\
&\iff \exists r \in P_{\mathcal{B}}/\mathcal{M} : head(r) = \mathfrak{Sup}(A) \wedge body^+(r) = \emptyset \\
&\iff \exists r \in P_{\mathcal{B}} : head(r) = \mathfrak{Sup}(A) \wedge \\
&\qquad \{\mathfrak{Sup}(X) \mid \texttt{not}\ \mathfrak{Sup}(X) \in body^-(r)\} \subseteq \mathbf{f}(\mathcal{M}) \\
&\overset{\text{Lemma 13}}{\iff} \exists B \in \mathfrak{Sup}(A) : Att(B) \subseteq \texttt{out}(\mathcal{L}').
\end{aligned}
$$

$$
\begin{aligned}
&\mathcal{L}'(A) = \texttt{out} \\
&\iff \mathfrak{Sup}(A) \in \mathbf{f}(\mathcal{M}) \\
&\iff \mathfrak{Sup}(A) \in \mathbf{f}(\psi_{P_{\mathcal{B}}}(\mathcal{M})) \\
&\iff \forall r \in P_{\mathcal{B}}/\mathcal{M} : head(r) = \mathfrak{Sup}(A) \rightarrow body^+(r) \cap \mathbf{f}(\psi_{P_{\mathcal{B}}}(\mathcal{M})) \neq \emptyset \\
&\iff \neg\exists r \in P_{\mathcal{B}}/\mathcal{M} : head(r) = \mathfrak{Sup}(A) \\
&\iff \forall r \in P_{\mathcal{B}} : head(r) = \mathfrak{Sup}(A) \rightarrow \\
&\qquad \{\mathfrak{Sup}(X) \mid \texttt{not}\ \mathfrak{Sup}(X) \in body^-(r)\} \cap \mathbf{t}(\mathcal{M}) \neq \emptyset \\
&\overset{\text{Lemma 13}}{\iff} \forall B \in \mathfrak{Sup}(A) : Att(B) \cap \texttt{in}(\mathcal{L}') \neq \emptyset.
\end{aligned}
$$

($\Longleftarrow$) Assume $\mathcal{L}'$ is a β-complete labelling. We will prove that $\mathcal{M}$ is a partial stable model of $P_{\mathcal{B}}$ by showing that $\mathbf{t}(\mathcal{M}) = \mathbf{t}(\psi_{P_{\mathcal{B}}}(\mathcal{M}))$ and $\mathbf{f}(\mathcal{M}) = \mathbf{f}(\psi_{P_{\mathcal{B}}}(\mathcal{M}))$.

$$
\begin{aligned}
&\mathfrak{Sup}(A) \in \mathbf{t}(\mathcal{M}) \\
&\iff \mathcal{L}'(A) = \texttt{in} \\
&\iff \exists B \in \mathfrak{Sup}(A) : Att(B) \subseteq \texttt{out}(\mathcal{L}') \\
&\overset{\text{Lemma 13}}{\iff} \exists r \in P_{\mathcal{B}} : head(r) = \mathfrak{Sup}(A) \wedge \\
&\qquad \{\mathfrak{Sup}(X) \mid \texttt{not}\ \mathfrak{Sup}(X) \in body^-(r)\} \subseteq \mathbf{f}(\mathcal{M}) \\
&\iff \exists r \in P_{\mathcal{B}}/\mathcal{M} : head(r) = \mathfrak{Sup}(A) \wedge body^+(r) = \emptyset \\
&\iff \exists r \in P_{\mathcal{B}}/\mathcal{M} : head(r) = \mathfrak{Sup}(A) \wedge body^+(r) \subseteq \mathbf{t}(\psi_{P_{\mathcal{B}}}(\mathcal{M})) \\
&\iff \mathfrak{Sup}(A) \in \mathbf{t}(\psi_{P_{\mathcal{B}}}(\mathcal{M})).
\end{aligned}
$$

$$
\begin{aligned}
&\mathfrak{Sup}(A) \in \mathbf{f}(\mathcal{M}) \\
&\iff \mathcal{L}'(A) = \texttt{out} \\
&\iff \forall B \in \mathfrak{Sup}(A) : Att(B) \cap \texttt{in}(\mathcal{L}') \neq \emptyset \\
&\overset{\text{Lemma 13}}{\iff} \forall r \in P_{\mathcal{B}} : head(r) = \mathfrak{Sup}(A) \rightarrow \\
&\qquad \{\mathfrak{Sup}(X) \mid \texttt{not}\ \mathfrak{Sup}(X) \in body^-(r)\} \cap \mathbf{t}(\mathcal{M}) \neq \emptyset \\
&\iff \neg\exists r \in P_{\mathcal{B}}/\mathcal{M} : head(r) = \mathfrak{Sup}(A) \\
&\iff \forall r \in P_{\mathcal{B}}/\mathcal{M} : head(r) = \mathfrak{Sup}(A) \rightarrow body^+(r) \cap \mathbf{f}(\psi_{P_{\mathcal{B}}}(\mathcal{M})) \neq \emptyset \\
&\iff \mathfrak{Sup}(A) \in \mathbf{f}(\psi_{P_{\mathcal{B}}}(\mathcal{M})).
\end{aligned}
$$

□

Lemma 32. Let $\mathcal{B}$ be a *BAF* and $P_{\mathcal{B}}$ be its corresponding *NLP*. Let $\mathcal{L}_1$ and $\mathcal{L}_2$ be labellings of $\mathcal{B}$ respecting $\mathfrak{Sup}$. It holds

1. $\mathtt{in}(\mathcal{L}_1) \subseteq \mathtt{in}(\mathcal{L}_2)$ iff $\mathbf{t}(\mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L}_1)) \subseteq \mathbf{t}(\mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L}_2))$;
2. $\mathtt{in}(\mathcal{L}_1) = \mathtt{in}(\mathcal{L}_2)$ iff $\mathbf{t}(\mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L}_1)) = \mathbf{t}(\mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L}_2))$;
3. $\mathtt{in}(\mathcal{L}_1) \subset \mathtt{in}(\mathcal{L}_2)$ iff $\mathbf{t}(\mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L}_1)) \subset \mathbf{t}(\mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L}_2))$.

Proof.

1. ($\Rightarrow$) Suppose $\mathtt{in}(\mathcal{L}_1) \subseteq \mathtt{in}(\mathcal{L}_2)$. If $\mathfrak{Sup}(A) \in \mathbf{t}(\mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L}_1))$, then by Definition 14, $A \in \mathtt{in}(\mathcal{L}_1)$. From our initial assumption, it follows $A \in \mathtt{in}(\mathcal{L}_2)$. So, by Definition 14, $\mathfrak{Sup}(A) \in \mathbf{t}(\mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L}_2))$.
($\Leftarrow$) Suppose $\mathbf{t}(\mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L}_1)) \subseteq \mathbf{t}(\mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L}_2))$. If $A \in \mathtt{in}(\mathcal{L}_1)$, then by Definition 14, $\mathfrak{Sup}(A) \in \mathbf{t}(\mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L}_1))$. From our initial assumption, we obtain $\mathfrak{Sup}(A) \in \mathbf{t}(\mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L}_2))$. So, by Definition 14, $A \in \mathtt{in}(\mathcal{L}_2)$.
2. It follows directly from point 1.
3. It follows directly from points 1 and 2.

□

Lemma 33. Let $\mathcal{B}$ be a *BAF* and $P_{\mathcal{B}}$ be its corresponding *NLP*. Let $\mathcal{L}_1$ and $\mathcal{L}_2$ be labellings of $\mathcal{B}$ respecting $\mathfrak{Sup}$. It holds

1. $\mathtt{undec}(\mathcal{L}_1) \subseteq \mathtt{undec}(\mathcal{L}_2)$ iff $\mathbf{u}(\mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L}_1)) \subseteq \mathbf{u}(\mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L}_2))$;
2. $\mathtt{undec}(\mathcal{L}_1) = \mathtt{undec}(\mathcal{L}_2)$ iff $\mathbf{u}(\mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L}_1)) = \mathbf{u}(\mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L}_2))$;
3. $\mathtt{undec}(\mathcal{L}_1) \subset \mathtt{undec}(\mathcal{L}_2)$ iff $\mathbf{u}(\mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L}_1)) \subset \mathbf{u}(\mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L}_2))$.

Proof.

1. ($\Rightarrow$) Suppose $\mathtt{undec}(\mathcal{L}_1) \subseteq \mathtt{undec}(\mathcal{L}_2)$. If $\mathfrak{Sup}(A) \in \mathbf{u}(\mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L}_1))$, then by Definition 14, $A \in \mathtt{undec}(\mathcal{L}_1)$. From our initial assumption, it follows $A \in \mathtt{undec}(\mathcal{L}_2)$. So, by Definition 14, $\mathfrak{Sup}(A) \in \mathbf{u}(\mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L}_2))$.
($\Leftarrow$) Suppose $\mathbf{u}(\mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L}_1)) \subseteq \mathbf{u}(\mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L}_2))$. If $A \in \mathtt{undec}(\mathcal{L}_1)$, then by Definition 14, $\mathfrak{Sup}(A) \in \mathbf{u}(\mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L}_1))$. From our initial assumption, it follows $\mathfrak{Sup}(A) \in \mathbf{u}(\mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L}_2))$. So, by Definition 14, $A \in \mathtt{undec}(\mathcal{L}_2)$.
2. It follows directly from point 1.
3. It follows directly from points 1 and 2.

□

Theorem 15. Let $\mathcal{B}$ be a *BAF* and $P_{\mathcal{B}}$ be its corresponding *NLP*. For any labelling $\mathcal{L}$ of $\mathcal{B}$ respecting $\mathfrak{Sup}$, it holds

1. $\mathcal{L}$ is a β-grounded labelling of $\mathcal{B}$ iff $\mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L})$ is a well-founded model of $P_{\mathcal{B}}$.
2. $\mathcal{L}$ is a β-preferred labelling of $\mathcal{B}$ iff $\mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L})$ is a regular model of $P_{\mathcal{B}}$.
3. $\mathcal{L}$ is a β-stable labelling of $\mathcal{B}$ iff $\mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L})$ is a stable model of $P_{\mathcal{B}}$.
4. $\mathcal{L}$ is a β-semi-stable labelling of $\mathcal{B}$ iff $\mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L})$ is an L-stable model of $P_{\mathcal{B}}$.

Proof. Let $\mathcal{L}$ be a labelling of $\mathcal{B}$ respecting $\mathfrak{Sup}$. The proof is straightforward:

1. $\mathcal{L}$ is a β-grounded labelling of $\mathcal{B}$ iff $\mathcal{L}$ is a β-complete labelling of $\mathcal{B}$ and $\mathtt{in}(\mathcal{L})$ is minimal (w.r.t. set inclusion) among all β-complete labellings of $\mathcal{B}$ iff (Theorem 14, Proposition 11 and Lemma 32) $\mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L})$ is a partial stable model of $P_{\mathcal{B}}$ and there is no partial stable model $\mathcal{M}$ of $P_{\mathcal{B}}$ such that $\mathbf{t}(\mathcal{M}) \subset \mathbf{t}(\mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L}))$ iff $\mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L})$ is a well-founded model of $P_{\mathcal{B}}$;

2. $\mathcal{L}$ is a β-preferred labelling of $\mathcal{B}$ iff $\mathcal{L}$ is a β-complete labelling of $\mathcal{B}$ and $\texttt{in}(\mathcal{L})$ is maximal (w.r.t. set inclusion) among all β-complete labellings of $\mathcal{B}$ iff (Theorem 14, Proposition 11 and Lemma 32) $\mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L})$ is a partial stable model of $P_{\mathcal{B}}$ and there is no partial stable model $\mathcal{M}$ of $P_{\mathcal{B}}$ such that $\mathbf{t}(\mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L})) \subset \mathbf{t}(\mathcal{M})$ iff $\mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L})$ is a regular model of $P_{\mathcal{B}}$;
3. $\mathcal{L}$ is a β-stable labelling of $\mathcal{B}$ iff $\mathcal{L}$ is a β-complete labelling of $\mathcal{B}$ such that $\texttt{undec}(\mathcal{L}) = \emptyset$ iff (Theorem 14) $\mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L})$ is a partial stable model of $P_{\mathcal{B}}$ such that $\mathbf{u}(\mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L})) = \emptyset$ iff $\mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L})$ is a stable model of $P_{\mathcal{B}}$;
4. $\mathcal{L}$ is a β-semi-stable labelling of $\mathcal{B}$ iff $\mathcal{L}$ is a β-complete labelling of $\mathcal{B}$ and $\texttt{undec}(\mathcal{L})$ is minimal (w.r.t. set inclusion) among all β-complete labellings of $\mathcal{B}$ iff (Theorem 14, Proposition 11 and Lemma 33) $\mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L})$ is a partial stable model of $P_{\mathcal{B}}$ and there is no partial stable model $\mathcal{M}$ of $P_{\mathcal{B}}$ such that $\mathbf{u}(\mathcal{M}) \subset \mathbf{u}(\mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L}))$ iff $\mathcal{L}2\mathcal{I}_{\mathcal{B}}(\mathcal{L})$ is an L-stable model of $P_{\mathcal{B}}$.

□

Corollary 16. Let $\mathcal{B}$ be a *BAF* and $P_{\mathcal{B}}$ be its corresponding *NLP*. It holds

1. $\mathcal{M}$ is a well-founded model of $P_{\mathcal{B}}$ iff $\mathcal{I}2\mathcal{L}_{\mathcal{B}}(\mathcal{M})$ is a β-grounded labelling of $\mathcal{B}$.
2. $\mathcal{M}$ is a regular model of $P_{\mathcal{B}}$ iff $\mathcal{I}2\mathcal{L}_{\mathcal{B}}(\mathcal{M})$ is a β-preferred labelling of $\mathcal{B}$.
3. $\mathcal{M}$ is a stable model of $P_{\mathcal{B}}$ iff $\mathcal{I}2\mathcal{L}_{\mathcal{B}}(\mathcal{M})$ is a β-stable labelling of $\mathcal{B}$.
4. $\mathcal{M}$ is a semi-stable model of $P_{\mathcal{B}}$ iff $\mathcal{I}2\mathcal{L}_{\mathcal{B}}(\mathcal{M})$ is a β-semi-stable labelling of $\mathcal{B}$.

Proof. These results come from Theorems 12 and 15. □

A.3 Theorems and Proofs from Section 5

Lemma 17. For any *NLP* P, the corresponding *BAF* $\mathcal{B}_P$ is a $\mathfrak{S}$-*BAF*.

Proof. Denote $\mathcal{B}_P = (\mathcal{A}_P, Att_P, Sup_P)$. From Definition 10, Sup_P is irreflexive and symmetric, as $(A, B) \in Sup_P$ iff $A \neq B$ and $\texttt{Conc}(A) = \texttt{Conc}(B)$. Now let $(A, B), (B, C) \in Sup_P$ with $A \neq C$. We conclude $\texttt{Conc}(A) = \texttt{Conc}(B) = \texttt{Conc}(C)$. As $A \neq C$ and $\texttt{Conc}(A) = \texttt{Conc}(C)$, it holds $(A, C) \in Sup_P$. We conclude that $\mathcal{B}_P$ is a $\mathfrak{S}$-*BAF*. □

Proposition 18. Let $\mathcal{B} = (\mathcal{A}, Att, Sup)$ be a *BAF*. If $Sup = \emptyset$, then $\mathcal{B}$ is a *RFBAF*.

Proof. Assume $Sup = \emptyset$. For any $A \in \mathcal{A}$, $\mathfrak{Sup}(A) = \{A\}$. Then, for every $A, B \in \mathcal{A}$, $\mathfrak{Sup}(A) = \mathfrak{Sup}(B)$ iff $A = B$. As every argument has a distinct set of supporters, there are no redundant arguments and $\mathcal{B}$ is a *RFBAF*. □

Lemma 34. Let $\mathcal{B} = (\mathcal{A}, Att, Sup)$ be a $\mathfrak{S}$-*BAF*. For every $A, B \in \mathcal{A}$, it holds $\mathfrak{Sup}(A) = \mathfrak{Sup}(B)$ iff $B \in \mathfrak{Sup}(A)$.

Proof.

($\Longrightarrow$) Assume $\mathfrak{Sup}(A) = \mathfrak{Sup}(B)$. We conclude $B \in \mathfrak{Sup}(B) = \mathfrak{Sup}(A)$.

($\Longleftarrow$) Assume $B \in \mathfrak{Sup}(A)$. By the symmetry of Sup, $A \in \mathfrak{Sup}(B)$. Let $X \in \mathfrak{Sup}(A)$. As $A \in \mathfrak{Sup}(B)$, it holds $X \in \mathfrak{Sup}(B)$. Therefore, $\mathfrak{Sup}(A) \subseteq \mathfrak{Sup}(B)$. Similarly, let $Y \in \mathfrak{Sup}(B)$. As $B \in \mathfrak{Sup}(A)$, it holds $Y \in \mathfrak{Sup}(A)$. Therefore, $\mathfrak{Sup}(B) \subseteq \mathfrak{Sup}(A)$. We conclude $\mathfrak{Sup}(A) = \mathfrak{Sup}(B)$.

□

Lemma 35. Let $\mathcal{B} = (\mathcal{A}, Att, Sup)$ be a $\mathfrak{S}$-*BAF*. For every $A, B \in \mathcal{A}$, it holds $(A, B) \in Sup$ iff $A \neq B$ and $\mathfrak{Sup}(A) = \mathfrak{Sup}(B)$.

Proof.

($\Longrightarrow$) Assume $(A, B) \in Sup$. From Lemma 34, $\mathfrak{Sup}(A) = \mathfrak{Sup}(B)$ and, as Sup is irreflexive, $A \neq B$.

($\Longleftarrow$) Assume $A \neq B$ and $\mathfrak{Sup}(A) = \mathfrak{Sup}(B)$. From Lemma 34, $A \in \mathfrak{Sup}(B)$. As $A \neq B$, there exists a sequence of distinct arguments $(A_0 = A, A_1, \ldots, A_n = B)$ such that $(A_i, A_{i+1}) \in Sup$ for $0 \leq i < n$ with $n > 0$. As $\mathcal{B}$ is a $\mathfrak{S}$-*BAF*, $(A, A_i) \in Sup$ for every $0 < i \leq n$, since $(X, Y), (Y, Z) \in Sup$ with $X \neq Z$ implies $(X, Z) \in Sup$ for any $X, Y, Z \in \mathcal{A}$. In particular, $(A, B) \in Sup$.

□

Theorem 19. Let $\mathcal{B}$ be a *BAF*, $P_{\mathcal{B}}$ be its corresponding *NLP*, and $\mathcal{B}_{P_{\mathcal{B}}}$ be the corresponding *BAF* of $P_{\mathcal{B}}$. It holds $\mathcal{B}$ and $\mathcal{B}_{P_{\mathcal{B}}}$ are isomorphic iff $\mathcal{B}$ is a $\mathfrak{S}$-*RFBAF*.

Proof. Let $\mathcal{B} = (\mathcal{A}, Att, Sup)$ and $\mathcal{B}' = \mathcal{B}_{P_{\mathcal{B}}} = (\mathcal{A}', Att', Sup')$.

($\Longrightarrow$) Assume $\mathcal{B}$ is not a $\mathfrak{S}$-*RFBAF*. We will prove that $\mathcal{B}$ and $\mathcal{B}_{P_{\mathcal{B}}}$ are not isomorphic. Either $\mathcal{B}$ is redundant or is not of support cliques. From Lemma 17, $\mathcal{B}_{P_{\mathcal{B}}}$ is a $\mathfrak{S}$-*BAF*. If $\mathcal{B}$ is not of support cliques, it is not isomorphic to $\mathcal{B}_{P_{\mathcal{B}}}$. Hence, we may consider $\mathcal{B}$ being redundant and of support cliques. There exist redundant arguments $R_0, R_1 \in \mathcal{A}$. Then, $R_0 \neq R_1$ and $\mathfrak{Sup}(R_0) = \mathfrak{Sup}(R_1)$ and $Att(R_0) = Att(R_1)$. We obtain $|\{Att(B) \mid B \in \mathfrak{Sup}(R_0)\}| < |\mathfrak{Sup}(R_0)|$. Intuitively, R_0 and R_1 derive an identical rule, because they have identical sets of supporters and attackers. Note that

$$
\begin{aligned}
|\mathcal{A}'| &= |P_{\mathcal{B}}| \\
&= |\{r_{A,B} \mid A \in \mathcal{A}, B \in \mathfrak{Sup}(A)\}| \\
&= \sum_{X \in \{\mathfrak{Sup}(A) \mid A \in \mathcal{A}\}} |\{Att(B) \mid B \in X\}| \\
&= |\{Att(B) \mid B \in \mathfrak{Sup}(R_0)\}| + \sum_{\substack{X \in \{\mathfrak{Sup}(A) \mid A \in \mathcal{A}\} \\ X \neq \mathfrak{Sup}(R_0)}} |\{Att(B) \mid B \in X\}| \\
&< |\mathfrak{Sup}(R_0)| + \sum_{\substack{X \in \{\mathfrak{Sup}(A) \mid A \in \mathcal{A}\} \\ X \neq \mathfrak{Sup}(R_0)}} |\{Att(B) \mid B \in X\}| \\
&\leq |\mathfrak{Sup}(R_0)| + \sum_{\substack{X \in \{\mathfrak{Sup}(A) \mid A \in \mathcal{A}\} \\ X \neq \mathfrak{Sup}(R_0)}} |X| \\
&= \sum_{X \in \{\mathfrak{Sup}(A) \mid A \in \mathcal{A}\}} |X| \\
&= |\mathcal{A}|.
\end{aligned}
$$

The equality $\sum_{X \in \{\mathfrak{Sup}(A) \mid A \in \mathcal{A}\}} |X| = |\mathcal{A}|$ comes from $\mathcal{B}$ being of support cliques. As $|\mathcal{A}'| < |\mathcal{A}|$, $\mathcal{B}$ and $\mathcal{B}_{P_{\mathcal{B}}}$ are not isomorphic.

($\Longleftarrow$) Assume $\mathcal{B}$ is a $\mathfrak{S}$-*RFBAF*. From each argument $A \in \mathcal{A}$ and $B \in \mathfrak{Sup}(A)$, we obtain in $P_{\mathcal{B}}$ a rule $r_{A,B}$, which is an argument of $\mathcal{B}_{P_{\mathcal{B}}}$. Define the function $f : \mathcal{A} \to \mathcal{A}'$ such that $f(A) = r_{A,A}$.
First, we show that f is injective. Let $A, B \in \mathcal{A}$. $f(A) = f(B) \Longrightarrow r_{A,A} = r_{B,B} \Longrightarrow \mathfrak{Sup}(A) = \mathfrak{Sup}(B) \wedge Att(A) = Att(B) \Longrightarrow A = B$. The last step comes from $\mathcal{B}$ not having redundant arguments.
Now we prove that f is surjective. Let $Y \in \mathcal{A}'$. This argument is also a rule in $P_{\mathcal{B}}$, say $r_{A,B}$ for some $A, B \in \mathcal{A}$ such that $B \in \mathfrak{Sup}(A)$. As $\mathcal{B}$ is a $\mathfrak{S}$-*BAF*, from Lemma 34 we obtain $\mathfrak{Sup}(A) = \mathfrak{Sup}(B)$. Then, $f(B) = r_{B,B} = r_{A,B} = Y$.

Therefore, f is bijective. It remains to show that it is an isomorphism.

$$\begin{aligned}(f(A), f(B)) \in Att' &\iff (r_{A,A}, r_{B,B}) \in Att' \\ &\iff \mathtt{Conc}(r_{A,A}) \in \mathtt{Vul}(r_{B,B}) \\ &\iff \mathfrak{Sup}(A) \in \{\mathfrak{Sup}(X) \mid X \in Att(B)\} \\ &\iff (A, B) \in Att.\end{aligned}$$

$$\begin{aligned}(f(A), f(B)) \in Sup' &\iff (r_{A,A}, r_{B,B}) \in Sup' \\ &\iff r_{A,A} \neq r_{B,B} \wedge \mathtt{Conc}(r_{A,A}) = \mathtt{Conc}(r_{B,B}) \\ &\iff A \neq B \wedge \mathfrak{Sup}(A) = \mathfrak{Sup}(B) \\ &\overset{\text{Lemma 35}}{\iff} (A, B) \in Sup.\end{aligned}$$

□

Lemma 20. Let P be an *NLP*, $\mathcal{B}_P$ be its corresponding *BAF* and $P_{\mathcal{B}_P}$ be the corresponding logic program of $\mathcal{B}_P$. If $HB_P \neq \{head(r) \mid r \in P\}$, then P and $P_{\mathcal{B}_P}$ are not isomorphic.

Proof. Let $\mathfrak{Sup}(A) \in HB_{P_{\mathcal{B}_P}}$ be any atom of $P_{\mathcal{B}_P}$, where A is an argument in $\mathcal{B}_P$. By Definition 12, $r_{A,A} \in P_{\mathcal{B}_P}$ and $head(r_{A,A}) = \mathfrak{Sup}(A)$. Hence, every atom of $P_{\mathcal{B}_P}$ is in the head of some rule. As this property does not hold for P, it follows that P and $P_{\mathcal{B}_P}$ are not isomorphic. □

Lemma 21. Let P be an *NLP*, $\mathcal{B}_P$ be its corresponding *BAF* and $P_{\mathcal{B}_P}$ be the corresponding logic program of $\mathcal{B}_P$. If there is some $r \in P$ such that $body^+(r) \neq \emptyset$, then P and $P_{\mathcal{B}_P}$ are not isomorphic.

Proof. By Definition 10, for every rule $r \in P_{\mathcal{B}_P}$ it holds $body^+(r) = \emptyset$. As this property does not hold for P, it follows that P and $P_{\mathcal{B}_P}$ are not isomorphic. □

Theorem 22. Let P be an *NLP*, $\mathcal{B}_P$ be its corresponding *BAF* and $P_{\mathcal{B}_P}$ be the corresponding *NLP* of $\mathcal{B}_P$. It holds P and $P_{\mathcal{B}_P}$ are isomorphic iff P is an *RALP*.

Proof.

($\Longrightarrow$) Assume $HB_P \neq \{head(r) \mid r \in P\}$ or $body^+(r) \neq \emptyset$ for some $r \in P$. By Lemmas 20 and 21, P and $P_{\mathcal{B}_P}$ are not isomorphic.

($\Longleftarrow$) Assume $HB_P = \{head(r) \mid r \in P\}$ and $body^+(r) = \emptyset$ for every $r \in P$. Denote $\mathcal{B}_P = (\mathcal{A}_P, Att_P, Sup_P)$. Define $f : HB_P \to HB_{P_{\mathcal{B}_P}}$ such that $f(c) = \{C \in \mathcal{A}_P \mid \mathtt{Conc}(C) = c\}$.
First, we show that f is well-defined, i.e., $f(c) \in HB_{P_{\mathcal{B}_P}}$ for any $c \in HB_P$. Let $c \in HB_P$, the set $f(c) = \{C \in \mathcal{A}_P \mid \mathtt{Conc}(C) = c\}$ consists of all arguments with conclusion c. By Definition 10, $(C, C') \in Sup_P$ iff $C \neq C'$ and $\mathtt{Conc}(C) = \mathtt{Conc}(C')$. Therefore, $f(c) = \mathfrak{Sup}(C)$ for any $C \in \mathcal{A}_P$ with $\mathtt{Conc}(C) = c$, and it follows $f(c) \in HB_{P_{\mathcal{B}_P}} = \{\mathfrak{Sup}(A) \mid A \in \mathcal{A}_P\}$.
Now we prove that f is injective. Assume $f(a) = f(b)$. Then $\{A \in \mathcal{A}_P \mid \mathtt{Conc}(A) = a\} = \{B \in \mathcal{A}_P \mid \mathtt{Conc}(B) = b\}$, and we conclude $a = b$.
Now we prove that f is surjective. Let $Y \in HB_{P_{\mathcal{B}_P}}$. Denote $Y = \mathfrak{Sup}(A)$ for some $A \in \mathcal{A}_P$. As $A \in \mathcal{A}_P$, $\mathtt{Conc}(A) \in HB_P$. Observe that $f(\mathtt{Conc}(A)) = \{C \in \mathcal{A}_P \mid \mathtt{Conc}(C) = \mathtt{Conc}(A)\} = \mathfrak{Sup}(A) = Y$.
We have shown that f is a bijection. As $body^+(r) = \emptyset$ for every $r \in P$, then, by Definition 10, $r \in P$ iff $r \in \mathcal{A}_P$.

$$
\begin{aligned}
& f(c) \leftarrow \texttt{not}\ f(b_1), \cdots, \texttt{not}\ f(b_m) \in P_{\mathcal{B}_P} \\
\iff & \exists C \in \mathcal{A}_P, \exists B \in \mathfrak{Sup}(C): \\
& \quad r_{C,B} = f(c) \leftarrow \texttt{not}\ f(b_1), \cdots, \texttt{not}\ f(b_m) \in P_{\mathcal{B}_P} \\
\iff & \exists B \in \mathcal{A}_P : r_{B,B} = f(c) \leftarrow \texttt{not}\ f(b_1), \cdots, \texttt{not}\ f(b_m) \in P_{\mathcal{B}_P} \\
\iff & \exists B \in \mathcal{A}_P : \mathfrak{Sup}(B) \leftarrow \texttt{not}\ \mathfrak{Sup}(B_1), \cdots, \texttt{not}\ \mathfrak{Sup}(B_m) \in P_{\mathcal{B}_P} \\
& \quad \text{where } \texttt{Conc}(B) = c, Att_P(B) = \{B_1, \cdots, B_m\} \\
& \quad \text{and } \texttt{Conc}(B_i) = b_i \text{ for } 1 \leq i \leq m \\
\iff & \exists B \in \mathcal{A}_P : \texttt{Conc}(B) = c \wedge Att_P(B) = \{B_1, \cdots, B_m\} \\
& \quad \text{with } \texttt{Conc}(B_i) = b_i \text{ for } 1 \leq i \leq m \\
\iff & c \leftarrow \texttt{not}\ b_1, \cdots, \texttt{not}\ b_m \in P.
\end{aligned}
$$

□

A.4 Theorems and Proofs from Section 6

Lemma 36. Let P be an *NLP*, $\mathcal{B}_P$ its corresponding *BAF* and P^* the corresponding *RALP* of P. For every argument A in $\mathcal{B}_P$, there exists a rule $r \in P^*$ such that $head(r) = \texttt{Conc}(A)$ and $body(r) = \{\texttt{not}\ x \mid x \in \texttt{Vul}(A)\}$.

Proof. We will prove by structural induction that for every argument A in $\mathcal{B}_P$, there exists a rule $r \in P^*$ such that $head(r) = \texttt{Conc}(A)$ and $body(r) = \{\texttt{not}\ x \mid x \in \texttt{Vul}(A)\}$. Let A be an argument in $\mathcal{B}_P$ and assume that for every proper subargument $A' \in \texttt{Sub}(A) - \{A\}$, there exists a rule $r' \in P^*$ such that $head(r') = \texttt{Conc}(A')$ and $body(r') = \{\texttt{not}\ x \mid x \in \texttt{Vul}(A')\}$.

- (Base) Assume A has no proper subarguments, i.e., $\texttt{Sub}(A) = \{A\}$. Then, $A \in P^*$ and $head(A) = \texttt{Conc}(A)$ and $body(A) = \{\texttt{not}\ x \mid x \in \texttt{Vul}(A)\}$.
- (Inductive step) Assume A has some proper subargument. Then, $A = a \leftarrow (A_1), \ldots, (A_n), \texttt{not}\ b_1, \ldots, \texttt{not}\ b_m$ $(n > 0)$ derives from a rule $r_0 = a \leftarrow a_1, \ldots, a_n, \texttt{not}\ b_1, \ldots, \texttt{not}\ b_m$ in P with $\texttt{Conc}(A_i) = a_i$ for $1 \leq i \leq n$. By the induction hypothesis, for each A_i, $1 \leq i \leq n$, there exists $r_i \in P^*$ with $head(r_i) = \texttt{Conc}(A_i)$ and $body(r_i) = \{\texttt{not}\ x \mid x \in \texttt{Vul}(A_i)\}$. By Definition 8, $r_0 \notin \texttt{Rules}(A_i)$ for $1 \leq i \leq n$. By Definition 19, there exists $r \in P^*$ with $head(r) = head(r_0) = a$ and

$$
\begin{aligned}
body(r) &= \bigcup_{i=0}^{n} body^-(r_i) \\
&= \{\texttt{not}\ b_1, \ldots, \texttt{not}\ b_m\} \cup \bigcup_{i=1}^{n} body^-(r_i) \\
&= \{\texttt{not}\ b_1, \ldots, \texttt{not}\ b_m\} \cup \bigcup_{i=1}^{n} \{\texttt{not}\ x \mid x \in \texttt{Vul}(A_i)\} \\
&= \{\texttt{not}\ x \mid x \in \texttt{Vul}(A)\}.
\end{aligned}
$$

□

Lemma 37. Let P be an *NLP*, $\mathcal{B}_P$ its corresponding *BAF* and P^* the corresponding *RALP* of P. For every rule $r \in P^*$, there exists an argument A in $\mathcal{B}_P$ such that $head(r) = \texttt{Conc}(A)$ and $body(r) = \{\texttt{not}\ x \mid x \in \texttt{Vul}(A)\}$.

Proof. We will prove by structural induction that for every rule $r \in P^*$, there exists an argument A in $\mathcal{B}_P$ such that $head(r) = \texttt{Conc}(A)$ and $body(r) = \{\texttt{not}\ x \mid x \in \texttt{Vul}(A)\}$. Let r be a rule in P^* deriving from $r_0 \in P$

and assume that for every proper subrule $r' \in \texttt{Rules}(r) - \{r_0\}$, there exists an argument A' in $\mathcal{B}_P$ such that $head(r') = \texttt{Conc}(A')$ and $body(r') = \{\texttt{not}\ x \mid x \in \texttt{Vul}(A')\}$.

- (Base) Assume r has no proper subrules, i.e., $\texttt{Rules}(r) = \{r_0\}$. Then, $r \in P$ and r is also an argument in $\mathcal{B}_P$ with $head(r) = \texttt{Conc}(r)$ and $body(r) = \{\texttt{not}\ x \mid x \in \texttt{Vul}(r)\}$.
- (Inductive step) Assume r has some proper subrule. Denote $r_0 = a \leftarrow a_1, \ldots, a_n, \texttt{not}\ b_1, \ldots, \texttt{not}\ b_m$ $(n > 0)$. By Definition 19, for every a_i, $1 \leq i \leq n$, there exists $r_i \in P^*$ with $head(r_i) = a_i$ and $r_0 \notin \texttt{Rules}(r_i)$. By the induction hypothesis, for each r_i, $1 \leq i \leq n$, there exists an argument A_i in $\mathcal{B}_P$ with $\texttt{Conc}(A_i) = a_i$ and $body(r_i) = \{\texttt{not}\ x \mid x \in \texttt{Vul}(A_i)\}$. By Definition 8, $\mathcal{B}_P$ has an argument $A = a \leftarrow (A_1), \ldots, (A_n), \texttt{not}\ b_1, \ldots, \texttt{not}\ b_m$ deriving from r_0 with $\texttt{Conc}(A) = a = head(r)$ and

$$\begin{aligned} body(r) &= \bigcup_{i=0}^{n} body^-(r_i) \\ &= \{\texttt{not}\ b_1, \ldots, \texttt{not}\ b_m\} \cup \bigcup_{i=1}^{n} body^-(r_i) \\ &= \{\texttt{not}\ b_1, \ldots, \texttt{not}\ b_m\} \cup \bigcup_{i=1}^{n} \{\texttt{not}\ x \mid x \in \texttt{Vul}(A_i)\} \\ &= \{\texttt{not}\ x \mid x \in \texttt{Vul}(A)\}. \end{aligned}$$

□

Theorem 23. Let P be an *NLP* and P^* be its corresponding *RALP*. It holds P^* and $P_{\mathcal{B}_P}$ are isomorphic.

Proof. Recall that arguments in $\mathcal{B}_P$ are constructed from rules in P (Section 3), and atoms in the rules of $P_{\mathcal{B}_P}$ are sets of supporters of arguments in $\mathcal{B}_P$ (Section 4). Each atom s in $HB_{P_{\mathcal{B}_P}}$ is a subset of arguments in $\mathcal{B}_P$ with the same conclusion. Thus, we can define the function $f : HB_{P_{\mathcal{B}_P}} \rightarrow HB_{P^*}$ such that $f(s) = c$, where $c = \texttt{Conc}(A)$ for every $A \in s$. As c is the conclusion of some argument A in $\mathcal{B}_P$, then, by Lemma 36, there exists a rule $r \in P^*$ with $head(r) = \texttt{Conc}(A) = c$ and therefore $c \in HB_{P^*}$. Hence, f is well-defined.

We will prove that f is injective. Assume $f(s) = f(s') = c$ for $s, s' \in HB_{P_{\mathcal{B}_P}}$. Then, $s = \mathfrak{Sup}(A)$ for some argument A in $\mathcal{B}_P$ with $\texttt{Conc}(A) = c$. Similarly, $s' = \mathfrak{Sup}(A')$ for some argument A' in $\mathcal{B}_P$ with $\texttt{Conc}(A') = c$. As $\texttt{Conc}(A) = \texttt{Conc}(A')$, it follows that $s = \mathfrak{Sup}(A) = \mathfrak{Sup}(A') = s'$.

We will prove that f is surjective. Let $a \in HB_{P^*}$. As P^* is an *RALP*, a is the head of some rule $r \in P^*$. By Lemma 37, $\mathcal{B}_P$ has some argument A with $\texttt{Conc}(A) = a$. Observe that $\mathfrak{Sup}(A) \in HB_{P_{\mathcal{B}_P}}$ and $f(\mathfrak{Sup}(A)) = a$.

We finish this proof by showing that $f(s_0) \leftarrow \texttt{not}\ f(s_1), \ldots, \texttt{not}\ f(s_m) \in P^*$ iff $s_0 \leftarrow \texttt{not}\ s_1, \ldots, \texttt{not}\ s_m \in P_{\mathcal{B}_P}$ for every $s_0, s_1, \ldots, s_m \in HB_{P_{\mathcal{B}_P}}$.

($\Longrightarrow$) Let $r = f(\mathfrak{Sup}(B_0)) \leftarrow \texttt{not}\ f(\mathfrak{Sup}(B_1)), \ldots, \texttt{not}\ f(\mathfrak{Sup}(B_m))$ be a rule in P^* (it always follows this format by the proof that f is surjective) for arguments $B_0, B_1, \ldots, B_m$ in $\mathcal{B}_P$ where $\texttt{Conc}(B_i) = b_i = f(\mathfrak{Sup}(B_i))$ for $0 \leq i \leq m$. By applying Lemma 37 to r, it follows that $\mathcal{B}_P$ has an argument A with $\texttt{Conc}(A) = head(r) = f(\mathfrak{Sup}(B_0)) = b_0$ and $\texttt{Vul}(A) = \{x \mid \texttt{not}\ x \in body(r)\} = \{f(\mathfrak{Sup}(B_1)), \ldots, f(\mathfrak{Sup}(B_m))\} = \{b_1, \ldots, b_m\}$. Note that $\{\mathfrak{Sup}(X) \mid X \in Att(A)\} = \{\mathfrak{Sup}(B_1), \ldots, \mathfrak{Sup}(B_m)\}$. By Definition 12, $r_{A,A} = \mathfrak{Sup}(A) \leftarrow \texttt{not}\ \mathfrak{Sup}(B_1), \ldots, \texttt{not}\ \mathfrak{Sup}(B_m)$ is a rule in $P_{\mathcal{B}_P}$. As $\texttt{Conc}(A) = \texttt{Conc}(B_0)$, then $\mathfrak{Sup}(A) = \mathfrak{Sup}(B_0)$ and $\mathfrak{Sup}(B_0) \leftarrow \texttt{not}\ \mathfrak{Sup}(B_1), \ldots, \texttt{not}\ \mathfrak{Sup}(B_m) \in P_{\mathcal{B}_P}$.

($\Longleftarrow$) Let $r = s_0 \leftarrow \texttt{not}\ s_1, \ldots, \texttt{not}\ s_m$ be a rule in $P_{\mathcal{B}_P}$. By Definition 12, $r = r_{B_0,B_0}$ for some argument B_0 in $\mathcal{B}_P$, i.e., we can write $r = \mathfrak{Sup}(B_0) \leftarrow \texttt{not}\ \mathfrak{Sup}(B_1), \ldots, \texttt{not}\ \mathfrak{Sup}(B_m)$ for arguments $B_0, B_1, \ldots, B_m$ in $\mathcal{B}_P$ where $\texttt{Conc}(B_i) = b_i$ for $0 \leq i \leq m$ and $\texttt{Vul}(B_0) = \{b_1, \ldots, b_m\}$. By applying Lemma 36 to B_0, there is $r_0 \in P^*$ such that $head(r_0) = \texttt{Conc}(B_0) = b_0$ and $body(r_0) = \{\texttt{not}\ x \mid x \in \texttt{Vul}(B_0)\} = \{\texttt{not}\ b_1, \ldots, \texttt{not}\ b_m\}$. Then,

$$\begin{aligned} r_0 &= b_0 \leftarrow \texttt{not}\ b_1, \ldots, \texttt{not}\ b_m \\ &= f(s_0) \leftarrow \texttt{not}\ f(s_1), \ldots, \texttt{not}\ f(s_1) \in P^*. \end{aligned}$$

□

Corollary 24. Let P be an *NLP* with corresponding *RALP* P^*. It holds $\mathcal{M}$ is a partial stable, well-founded, regular, stable and L-stable model of P iff $\mathcal{M}$ is respectively a partial stable, well-founded, regular, stable and L-stable model of P^*.

Proof. It follows from the isomorphism between $P_{\mathcal{B}_P}$ and P^* (Theorem 23), as transforming an *NLP* into a *BAF* (Section 3) and a *BAF* into an *NLP* (Section 4) preserves the partial stable, well-founded, regular, stable and L-stable semantics. Each atom a in P is represented by the atom $\mathfrak{Sup}(A)$ in $P_{\mathcal{B}_P}$ for some argument A in $\mathcal{B}_P$ with $\texttt{Conc}(A) = a$. The isomorphism $f : HB_{P_{\mathcal{B}_P}} \rightarrow HB_{P^*}$ between $P_{\mathcal{B}_P}$ and P^* is such that $f(\mathfrak{Sup}(A)) = a$ (see proof of Theorem 23). Hence, we obtain the coincidence that $\mathcal{M}$ is a partial stable, well-founded, regular, stable and L-stable model of P iff $\mathcal{M}$ is also respectively a partial stable, well-founded, regular, stable and L-stable model of P^*. □

Theorem 25. *NLP*s and *RALP*s have the same expressiveness for partial stable, well-founded, regular, stable and L-stable semantics.

Proof. We have

- For any *NLP* P, there exists an *RALP* P^* such that $\mathcal{M}$ is a partial stable, well-founded, regular, stable, L-stable model of P iff $\mathcal{M}$ is respectively a partial stable, well-founded, regular, stable, L-stable model of P^* (Corollary 24).
- Obviously, any *RALP* is an *NLP*.

Hence, *NLP*s and *RALP*s have the same expressiveness for partial stable, well-founded, regular, stable and L-stable semantics. □

Theorem 26. Let P be an *NLP* and P^* be its corresponding *RALP*. If $\mathcal{B}_P$ is an *RFBAF*, then $\mathcal{B}_P$ and $\mathcal{B}_{P^*}$ are isomorphic.

Proof. By Theorem 23, $P_{\mathcal{B}_P}$ and P^* are isomorphic. Then, $\mathcal{B}_{P_{\mathcal{B}_P}}$ and $\mathcal{B}_{P^*}$ are isomorphic. As $\mathcal{B}_P$ is an *RFBAF*, it is a $\mathfrak{S}$-*RFBAF*. By Theorem 19, $\mathcal{B}_{P_{\mathcal{B}_P}}$ and $\mathcal{B}_P$ are isomorphic. Hence, $\mathcal{B}_P$ and $\mathcal{B}_{P^*}$ are isomorphic. □

Theorem 27. Let P and P' be *RALP*s. It holds P and P' are isomorphic iff $\mathcal{B}_P$ and $\mathcal{B}_{P'}$ are isomorphic.

Proof. ($\Longrightarrow$) Trivial. ($\Longleftarrow$) Assume $\mathcal{B}_P$ and $\mathcal{B}_{P'}$ are isomorphic. Then, $P_{\mathcal{B}_P}$ and $P_{\mathcal{B}_{P'}}$ are isomorphic. As P and P' are *RALP*s, it follows, by Theorem 22, that $P_{\mathcal{B}_P}$ is isomorphic to P, and that $P_{\mathcal{B}_{P'}}$ is isomorphic to P'. Hence, P and P' are isomorphic. □

Theorem 28. Let P_1 and P_2 be *NLP*s with corresponding *RALP*s P_1^* and P_2^*, respectively. If $\mathcal{B}_{P_1}$ and $\mathcal{B}_{P_2}$ are *RFBAF*s, it holds P_1^* and P_2^* are isomorphic iff $\mathcal{B}_{P_1}$ and $\mathcal{B}_{P_2}$ are isomorphic.

Proof. Let $\cong$ be the isomorphism equivalence relation. By Theorem 23, $P_1^* \cong P_{\mathcal{B}_{P_1}}$ and $P_2^* \cong P_{\mathcal{B}_{P_2}}$. Then, $\mathcal{B}_{P_1^*} \cong \mathcal{B}_{P_{\mathcal{B}_{P_1}}}$ and $\mathcal{B}_{P_2^*} \cong \mathcal{B}_{P_{\mathcal{B}_{P_2}}}$. As $\mathcal{B}_{P_1}$ and $\mathcal{B}_{P_2}$ are *RFBAF*s, they are $\mathfrak{S}$-*RFBAF*s. By Theorem 19, $\mathcal{B}_{P_{\mathcal{B}_{P_1}}} \cong \mathcal{B}_{P_1}$ and $\mathcal{B}_{P_{\mathcal{B}_{P_2}}} \cong \mathcal{B}_{P_2}$.

- ($\Longrightarrow$) Assume $P_1^* \cong P_2^*$. Then, $\mathcal{B}_{P_1^*} \cong \mathcal{B}_{P_2^*}$ and $\mathcal{B}_{P_{\mathcal{B}_{P_1}}} \cong \mathcal{B}_{P_{\mathcal{B}_{P_2}}}$. Thus, $\mathcal{B}_{P_1} \cong \mathcal{B}_{P_2}$.
- ($\Longleftarrow$) Assume $\mathcal{B}_{P_1} \cong \mathcal{B}_{P_2}$. Then, $P_{\mathcal{B}_{P_1}} \cong P_{\mathcal{B}_{P_2}}$ and $P_1^* \cong P_2^*$.

□

Received 9 January 2025; accepted 10 October 2025