Argument and Computation, 2015
http://dx.doi.org/10.1080/19462166.2014.1002535

Taylor & Francis
Taylor & Francis Group

Senses of 'argument' in instantiated argumentation frameworks†

Adam Wyner[a]*, Trevor Bench-Capon[b], Paul Dunne[b] and Federico Cerutti[a]

[a]*Department of Computing Science, University of Aberdeen, Meston Building, Meston Walk, Aberdeen AB24 3UE, UK;* [b]*Department of Computing Science, University of Liverpool, Liverpool, UK*

(*Received 19 December 2014; accepted 20 December 2014*)

Abstract Argumentation Frameworks (AFs) provide a fruitful basis for exploring issues of defeasible reasoning. Their power largely derives from the abstract nature of the arguments within the framework, where arguments are atomic nodes in an undifferentiated relation of attack. This abstraction conceals different senses of argument, namely a single-step reason to a claim, a series of reasoning steps to a single claim, and reasoning steps for and against a claim. Concrete instantiations encounter difficulties and complexities as a result of conflating these senses. To distinguish them, we provide an approach to instantiating AFs in which the nodes are restricted to literals and rules, encoding the underlying theory directly. Arguments in these senses emerge from this framework as distinctive structures of nodes and paths. As a consequence of the approach, we reduce the effort of computing argumentation extensions, which is in contrast to other approaches. Our framework retains the theoretical and computational benefits of an abstract AF, distinguishes senses of argument, and efficiently computes extensions. Given the mixed intended audience of the paper, the style of presentation is semi-formal.

Keywords: argumentation frameworks; instantiated argumentation; structured argumentation; logic

1. Introduction

Abstract Argumentation Frameworks (AFs) (Dung, 1995) provide a fruitful basis for exploring issues of defeasible reasoning. Their power largely derives from the abstract nature of the arguments within the framework, where arguments are atomic nodes in an undifferentiated relation of attack; such AFs provide a very clean acceptability semantics (Dung, 1995; Dunne & Bench-Capon, 2002).

While abstract approaches facilitate the study of arguments and the relations between them, it is necessary to instantiate arguments to apply the theory.[1] In instantiated argumentation, arguments are constructed from premises together with rules, deriving conclusions.

The semantics of these inference rules, consistent with the literature on logic programming (Gelfond & Lifschitz, 1988), is different from the material implication of classical logic in the sense that they are not 'contrapositive' (e.g. $(\neg q, p \rightarrow q) \nvdash \neg p$). However, to more naturally represent instantiated arguments, we want to stay as close as possible to classical logic rather than logic programming (Lifschitz, Pearce, and Valverde, 2001), which has been shown to be close to intuitionistic logic. Therefore, we explicitly consider the *tertium non datur* principle (i.e. $\neg p \vee p$ is a tautology): this has an interesting and positive collateral outcome, namely the possibility to reason defeasibly (*what-if* scenarios) about uncertain worlds and to adhere to the *ex falso quodlibet* principle. Indeed, although there might be no ultimate proofs in favour or against a consequent (i.e. the antecedent of a rule is denied), we can still consider alternative possible worlds: one where the consequent is true and one where it is not.

*Corresponding author. Email: azwyner@abdn.ac.uk

†This paper was developed from Wyner, Bench-Capon, and Dunne (2009, 2013)

In addition, by exploiting argumentation techniques, we can reason with inconsistent knowledge bases (KBs) and derive consistent subsets of the KB. Such techniques are not novel in the literature (Besnard & Hunter, 2008; Bondarenko, Dung, Kowalski, and Toni, 1997; Brewka & Woltran, 2010; Caminada & Amgoud, 2007; García & Simari, 2004; Prakken, 2010), but for the sake of the discussion we will consider the Argumentation Service Platform with Integrated Components (ASPIC) (Caminada & Amgoud, 2007) only.

Over the course of the paper, we will show that although ASPIC is rather general (in comparison to the proposed approach), there are some drawbacks. First, restricting ASPIC to adhere to the *tertium non datur* principle in order to have automatic what-if scenarios would require additional rules to be explicitly included in the KB. Although this might be desirable, it increments the complexity of the overall framework. The approach proposed here does not require such additional rules.

Second, in order to satisfy the rationality postulates (Caminada & Amgoud, 2007), ASPIC requires a priori choices to be made (e.g. *restricted rebut* and preferred semantics vs. *unrestricted rebut* and grounded semantics). In this work, we derive straightforwardly a multiple-status semantics – basically preferred semantics with an additional requirement – which provides us with rational outcomes.

Finally, before applying argumentation semantics, ASPIC derives a set of arguments which are not atomic entities: each argument can be subargument of a second argument; therefore, an attack against the first reflects as an attack against the second as well. This paradigm of nesting arguments inside other arguments has been proved to be very powerful and flexible, but in this paper we show an easier way to construct atomic *abstract arguments* – in a one-to-one correspondence with propositions and rules – and *attacks* among them (Dung, 1995) so to compute the derived conclusions without the need of nesting rules – under our operative constraints discussed above.

Our argument construction paradigm allows us not only to exploit computational properties of Dung's AFs (1995), but it gives us an insight of the different terminological meanings of the word *argument*: (i) a one-step reason for a claim (*Argument* with capital 'A'); (ii) a chain of reasoning leading towards a claim (*Case*); and (iii) reasons for and against a claim (*Debate*). By adopting the proposed approach for representing a KB, these three senses of the word *argument* have a direct and clear counterpart in the graph: each of them is a subgraph with specific characteristics. Although this has not an immediate effect on the computation of extensions, it can provide a useful linkage between the community of computational models of arguments and more philosophical approaches to argument representation and reasoning in less formal – although not less rigorous – contexts (Hoffmann, 2005; Laronge, 2009). Indeed, legal reasoning widely uses the *tertium non datur* principle – that is, John is either guilty or innocent – thus making it an interesting domain to explore. To this aim, the paper is deliberately written in 'semi-technical' way: we have been rigorous in definitions and properties discussion, but we have made the discussion available even beyond the argumentation in artificial intelligence community. Future work will be more focused on each of the two communities.

The structure of the paper is as follows. In Section 2, we outline the AFs of Dung (1995) and the instantiated argumentation of Caminada and Amgoud (2007), which for the purpose of this work are equivalent. In Section 3, we characterise the types of KB we are working with, then show how a KB is represented in a derived AF. We illustrate the approach with basic examples of the definitions, a simple example of a combination of strict and defeasible rules, and the relationship of extensions to classical logic models. The different senses of *argument* are then characterised in terms of particular structures within the AF as presented in Section 4. In Section 5, we discuss various aspects of the proposal: we compare it to the approach to KB instantiation of Caminada and Amgoud (2007) along with key examples; consider other approaches to

KB instantiation; and outline a range of additional strengths. We conclude in Section 6 with some remarks and future work.

2. Argumentation frameworks

In this section, we briefly outline abstract and instantiated argumentation.

2.1. *Abstract argumentation (Dung, 1995)*

An AF is defined as follows (Dung, 1995).

Definition 1 An AF is a pair $\langle \mathcal{N}, \mathcal{O} \rangle$, where $\mathcal{N}$ is a finite set of *arguments*, $\{p_1, p_2, \ldots, p_n\}$, and $\mathcal{O}$ is an *attack* relation between elements of $\mathcal{N}$. For $\langle p_i, p_j \rangle \in \mathcal{O}$ we say the argument p_i attacks argument p_j. We assume that no argument attacks itself.

The relevant auxiliary definitions are as follows, where $S \subseteq \mathcal{N}$:

Definition 2 We say that $p \in \mathcal{N}$ is *acceptable with respect to* S if for every $q \in \mathcal{N}$ that attacks p, there is some $r \in S$ that attacks q. S is *conflict-free* if no argument in S is attacked by any other argument in S. S is *admissible* if it is conflict-free and every $p \in S$ is acceptable to S. A *preferred extension* is a maximal (w.r.t. $\subseteq$) admissible set. The argument $p \in \mathcal{N}$ is *credulously accepted* if it is in at least one preferred extension, and *sceptically accepted* if it is in *every* preferred extension.

There are a variety of other semantics, for example, *grounded*, *stable*, and others (see Baroni, Caminada, and Giacomin, 2011 for an introduction), but for our purposes, we only consider preferred extensions (Dung, 1995).

As it is our intention to clarify the notion of *argument* itself, we want to use content neutral terminology in the expression of the syntax rather than relying on terminology such as *argument* and *attacks*, which introduce semantic interpretations. Thus, we usually refer to the so-called arguments of AFs defined above as graph-theoretic *nodes* (denoted by $\mathcal{N}$) and their attack relations as *arcs* (denoted by $\mathcal{O}$).

2.2. *Instantiated argumentation (Caminada & Amgoud, 2007)*

In this section, we briefly review the key components of the benchmark argument instantiation method ASPIC (Caminada & Amgoud, 2007), which we later exemplify and compare to our proposal.

In constructing arguments, several functions are introduced: `Prem` is the set of premises of the argument, `Conc` returns the last conclusion of an argument, `Sub` returns all the subarguments of an argument, `DefRules` returns all the defeasible rules used in an argument, and `TopRule` returns the last inference rule used in the argument. Theory Bases $\mathcal{T}$ comprised strict and defeasible implications. Arguments have a deductive form and are constructed recursively from the rules of the Theory Base. To distinguish strict or defeasible rules from the deductive form of arguments, we use *short* arrows, $\rightarrow$ and $\Rightarrow$, for the former and *long* arrows, $\longrightarrow$ and $\Longrightarrow$, for the latter. For brevity, we only provide the clauses for the construction of strict arguments as the clauses for the construction of defeasible arguments are analogous (including among the DefRules the TopRule(A) that is defeasible) (Caminada & Amgoud, 2007).

Definition 3 (Argument) Suppose a *Theory Base*, $\mathcal{T}$, with strict and defeasible rules. An argument A is:

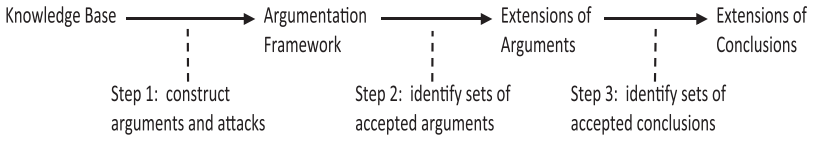

Figure 1. Three steps of argumentation.

$A_1, \ldots, A_n \longrightarrow \psi$ if $A_1, \ldots, A_n$, with $n \geq 0$, are arguments such that there exists a strict rule $\texttt{Conc}(A_1), \ldots, \texttt{Conc}(A_n) \rightarrow \psi$.
$\texttt{Prem}(A) = \texttt{Prem}(A_1) \cup \cdots \cup \texttt{Prem}(A_n)$,
$\texttt{Conc}(A) = \psi$,
$\texttt{Sub}(A) = \texttt{Sub}(A_1) \cup \cdots \cup \texttt{Sub}(A_n) \cup \{A\}$,
$\texttt{DefRules}(A) = \texttt{DefRules}(A_1) \cup \cdots \cup \texttt{DefRules}(A_n)$,
$\texttt{TopRule}(A) = \texttt{Conc}(A_1), \ldots, \texttt{Conc}(A_n) \rightarrow \psi$

Such a system determines the acceptability status of propositions generally in three steps as in Figure 1 (from Caminada & Wu, 2011). Starting with an inconsistent KB comprised of facts and rules, we construct arguments (nodes) and attacks (arcs) from this KB, resulting in an AF (Step 1); evaluate the AF according to a variety of semantics, resulting in extensions (sets) of arguments (Step 2); and extract the conclusions from the arguments, resulting in extensions of conclusions (Step 3). Thus, from a KB that is initially inconsistent (or derives inconsistency), we can nonetheless identify consistent sets of propositions.

In Section 5.1, we provide an example of a KB and arguments as generated using ASPIC.

3. Representing a theory as an AF

In this section, we introduce our approach in two parts (the presentation is a revision of Wyner et al., 2009, 2013). In the first part, we represent a Theory Base $\mathcal{T}$, which represents the KB. Then, we construct an AF from the KB, following Step 1 of Figure 1, where the nodes of an AF are labelled with respect to the literals and inference rules of the Theory Base, while the attack relation is partitioned with respect to the nodes. In the second part, we impose conditions on the assertion of literals with respect to the AF. Following the theoretical presentation, we provide basic examples, carrying out Step 2 of Figure 1 to evaluate an AF according to Definitions 1 and 2. As we see, Step 2 and Step 3 of Figure 1 collapse together.

3.1. *Theory Base $\mathcal{T}$*

Definition 4 A *Theory Base*, $\mathcal{T}$, comprises a pair $(\mathcal{L}, \mathcal{R})$ in which

$$\mathcal{L} = \{x_1, \ldots, x_n\} \cup \{\neg x_1, \ldots, \neg x_n\}$$

is a set of literals over a set of propositional variables $\{x_1, \ldots, x_n\}$. We use y_i to denote an arbitrary literal from $\{x_i, \neg x_i\}$.

We have a set of *proper names* of rules $\{r_1, r_2, \ldots, r_n\}$. Rules are either strict ($r \in \mathcal{R}_{str}$) or defeasible ($r \in \mathcal{R}_{dfs}$), and $\mathcal{R}_{str} \cap \mathcal{R}_{dfs} = \emptyset$. $\mathcal{R} = \mathcal{R}_{str} \cup \mathcal{R}_{dfs}$ where

$$\mathcal{R} = \{r_1, r_2, \ldots, r_n\}$$

in which $r \in \mathcal{R}$ has a body, bd $(r) \subseteq \mathcal{L}$, and a head, hd $(r) \in \mathcal{L}$.

We refer to the literals in bd(r) as *premises* and the literal in hd(r) as the *claim*.

For easy reference to the 'content' of the rule, we assume each rule has an associated *definite description* as follows. For $r \in \mathcal{R}_{str}$, the definite description of r has the form $r : bd(r) \rightarrow hd(r)$, where $hd(r) \in \mathcal{L}$ and $bd(r) \subseteq \mathcal{L}$. Similarly, the definite description for $r \in \mathcal{R}_{dfs}$ has the form $r : bd(r) \Rightarrow hd(r)$. Where a rule has an empty body, $bd(r) = \emptyset$, we have $r :\rightarrow hd(r)$ or $r :\Rightarrow hd(r)$, which are *strict* and *defeasible* assertions, respectively. To refer distinctly to the set of rules with non-empty bodies and those with empty bodies (*assertions*), we have $\mathcal{R} = \text{PRules} \cup \text{ARules}$, where $\text{PRules} = \{r \mid r \in \mathcal{R} \wedge bd(r) \neq \emptyset\}$ and $\text{ARules} = \{r \mid r \in \mathcal{R} \wedge bd(r) = \emptyset\}$.

We constrain a Theory Base, which we refer to as a *Well-formed Theory*, with the following Constraints 1–4. The purpose of the constraints is to reduce the expressivity of the language that appears in the Theory Base, guiding the knowledge engineer to produce a *sound* formalism. The first two constrain the relation of literals and rules; the third constrain reuse of literals between strict and defeasible rules; while the fourth prohibits strictly asserted contradiction.

First, every literal appears in some rule.

Constraint 1 For *Theory Base* $(\mathcal{L}, \mathcal{R})$, if $y \in \mathcal{L}$, then $\exists r \in \mathcal{R}$, $y \in bd(r)$ or $y = hd(r)$.

In addition, every rule has a claim.

Constraint 2 For *Theory Base* $(\mathcal{L}, \mathcal{R})$, if $r \in \mathcal{R}$, then $\exists y \in \mathcal{L}$, $y = hd(r)$.

Furthermore, the relationship between literals of strict and defeasible rules is constrained such that a strict rule is not, in effect, 'contained within' a defeasible rule:

Constraint 3 For *Theory Base* $(\mathcal{L}, \mathcal{R})$, $\forall r \in \mathcal{R}_{str}$, there is no rule, $r' \in \mathcal{R}_{dfs}$ with $hd(r) = hd(r')$ and $bd(r) \subseteq bd(r')$.

Finally, no literal and its negation can both be strictly asserted.

Constraint 4 For *Theory Base* $(\mathcal{L}, \mathcal{R})$, if $r \in \mathcal{R}$, where $r :\rightarrow hd(r)$, then $r' \notin \mathcal{R}$, where $r' :\rightarrow \neg hd(r)$.

Given these constraints, we have the following:

Definition 5 A *Well-formed Theory*, $\mathcal{W}$, is a Theory Base, $\mathcal{T}$, abiding Constraints 1–4.

For the semantics, we assume standard notions of *truth* and *falsity* of literals. For the difference between strict and defeasible rules, we use quantification over circumstances (Lewis, 1975). Semantically, a rule $r \in \mathcal{R}_{str}$ represents the notion that hd(r) 'always' holds if *all* of the literals in bd(r) simultaneously hold; that is, there are no circumstances where *all* of the literals in bd(r) hold and hd(r) does not. With respect to the rule, we say the bd(r) strictly implies the hd(r). Semantically, a rule $r \in \mathcal{R}_{dfs}$ represents the notion that hd(r) 'usually' holds if *all* of the literals in bd(r) simultaneously hold, but there are circumstances where $\neg$hd(r) holds though *all* of the literals in bd(r) simultaneously hold. With respect to the rule, we say the bd(r) defeasibly implies the hd(r). A formal discussion on the semantics for the Theory Base is provided in Appendix.

While the clauses are similar to the *Horn Clauses* of logic programming, the head literal can be in a positive or negative form. We only have classical negation, not negation as failure; we do not allow iterated negation – that is, $\neg\neg x \equiv x$. The rationale for this choice of clauses is that it naturally supports our analysis of *argument* and its various senses as discussed in Section 4.

3.2. *Deriving an AF from a Theory Base*

A core element of our approach is the concept of the AF derived from a Theory Base. The AF uses a set of labels for the nodes in the graph: $\{\mathsf{x}_1, \ldots, \mathsf{x}_n\} \cup \{\neg\mathsf{x}_1, \ldots, \neg\mathsf{x}_n\} \cup \{\mathsf{r}_1, \ldots, \mathsf{r}_n\}$ (or for clarity, the definite description of the rule name). Thus, we can see how elements of a Theory Base, $\mathcal{T}$, correspond to but are distinct from elements of the derived AF, indexing the AF to the $\mathcal{T}$. While in standard approaches to AFs, the graph is described in terms of *arguments*, we wish the syntax of the graph to be semantically unloaded; thus we refer to *nodes*. However, we maintain the standard terminology *attacks* as *arcs* between nodes. We first present the syntax of the derivation, then the semantics. Examples of derived AFs are given in Section 3.3.

3.2.1. *Syntax of derivation*

In an $\text{AF}_\mathcal{T}$, the nodes $\mathcal{N}_\mathcal{T}$ are atomic, for no node has a node within it.

The set of attacks of an $\text{AF}_\mathcal{T}$, $\mathcal{O}_\mathcal{T}$, is the union of three disjoint sets which describe (informally) attacks by a literal node on a literal node; attacks by a literal node on a rule node; and attacks by a rule node on a literal node. We recall that $y_i \in \{x_i, \neg x_i\}$ so that $\neg y_i$ is the complementary literal to y_i.

Definition 6 Let $\mathcal{T} = (\mathcal{L}, \mathcal{R})$ be a Theory Base with

$$\mathcal{L} = \{x_1, \ldots, x_n\} \cup \{\neg x_1, \ldots, \neg x_n\}$$
$$\mathcal{R} = \mathcal{R}_{str} \cup \mathcal{R}_{dfs}$$

The *derived AF* from $\mathcal{T}$, $\text{AF}_\mathcal{T}$, is $\langle \mathcal{N}_\mathcal{T}, \mathcal{O}_\mathcal{T} \rangle$ in which

$$\begin{aligned}\mathcal{N}_\mathcal{T} = &\{\mathsf{x},\ \neg\mathsf{x} : x, \neg x \in \mathcal{L}\}\\ &\cup \{\mathsf{r} : \mathsf{bd(r)} \rightarrow \mathsf{hd(r)} : r \in \mathcal{R}_{str}\}\\ &\cup \{\mathsf{r} : \mathsf{bd(r)} \Rightarrow \mathsf{hd(r)} : r \in \mathcal{R}_{dfs}\}\end{aligned}$$

and

$$\mathcal{O}_\mathcal{T} = \mathcal{O}^{ll}_\mathcal{T} \cup \mathcal{O}^{lr}_\mathcal{T} \cup \mathcal{O}^{rl}_\mathcal{T},$$

where

$$\mathcal{O}^{ll}_\mathcal{T} = \{\langle \mathsf{y_i}, \neg\mathsf{y_i}\rangle,\ \langle \neg\mathsf{y_i}, \mathsf{y_i}\rangle\ :\ 1 \leq i \leq n \textit{ and } \mathsf{y_i}, \neg\mathsf{y_i} \in \mathcal{N}_\mathcal{T}\}$$
$$\begin{aligned}\mathcal{O}^{lr}_\mathcal{T} = &\{\langle \neg\mathsf{y_i}, \mathsf{r_j}\rangle : y_i \in bd(r_j) \textit{ and } \neg\mathsf{y_i}, \mathsf{r_j} \in \mathcal{N}_\mathcal{T}\}\\ &\cup \{\langle \neg\mathsf{y_i}, \mathsf{r_j}\rangle : r_j \in \mathcal{R}_{dfs} \textit{ and } hd(r_j) = y_i \textit{ and } \neg\mathsf{y_i} \in \mathcal{N}_\mathcal{T}\}\end{aligned}$$
$$\mathcal{O}^{rl}_\mathcal{T} = \{\langle \mathsf{r_j}, \neg\mathsf{y_i}\rangle : hd(r_j) = y_i \textit{ and } \neg\mathsf{y_i}, \mathsf{r_j} \in \mathcal{N}_\mathcal{T}\}$$

3.2.2. *Semantics of derivation*

In the following, we provide the semantic notions for an $\text{AF}_\mathcal{T}$:

(1) Each literal y in $\mathcal{L}$ of Theory Base $\mathcal{T}$ corresponds to a node labelled y in $\mathcal{N}$ of the derived AF; $\mathcal{N}$ of the derived AF contains, in addition, the node labelled ¬y. Nodes labelled for literals of opposite polarity are mutually attacking.
(2) Each rule in r in $\mathcal{R}$ of a Theory Base $\mathcal{T}$ corresponds one to one to a node label r in $\mathcal{N}$ of the derived AF. Whereas a rule in $\mathcal{R}$ is true (or false) in the Theory Base, in the

derived AF we say it *has been applied* relative to the admissible set where it appears and otherwise *has not been applied*. In the AF, a rule node is attacked by the nodes which correspond to the negation of the body literals and, in addition, attacks the node which corresponds to the negation of the head literal.

(3) For *strict* rules, if a node which corresponds to the negation of a body literal of a rule is in an admissible set, we say the rule node has not been applied relative to that set. In this case, the node which corresponds to the head literal is only credulously admissible. If all the nodes which correspond to the body literals are in an admissible set, then the rule node has been applied and the node which corresponds to head literal is admissible in that set.

(4) For *defeasible* rules, if a node which corresponds to the negation of a body literal or if the node which corresponds to the negation of the head literal of the rule is in an admissible set, we say the rule node has not been applied relative to that set. In both instances, the node corresponding to the literal attacks the rule node. Even if all nodes which correspond to the body literals of a rule are in an admissible set, the rule node or the node which corresponds to the head literal may not be in that set, for they can be defeated.

Given a derived $AF_{\mathcal{T}}$ according to Definition 6, we evaluate it as a standard AF as in Section 2. Thus, the fundamental semantics of abstract AFs are maintained.

For our purposes and relative to our classical logic context, the set of extensions provided by Dungian AFs must be *filtered*. In our approach, AFs are derived from a Theory Base, and the resulting extensions are *not* homogeneous, for they may contain both literals and rules. More importantly, we must ensure that the extensions also serve to satisfy classical logic properties such as *closure* under strict implication. With these points in mind, we have the following, where Constraint 5 ensures *closure* under strict implication. Note that the constraint does not apply to *defeasible rules*, as we discuss further below.

Constraint 5 Consider a, a set that is a preferred extension in the derived AF $\langle \mathcal{N}_{\mathcal{T}}, \mathcal{O}_{\mathcal{T}} \rangle$; $\mathcal{A}$, the set of preferred extensions a; and $\mathcal{R}_{str} \subseteq \mathcal{N}_{\mathcal{T}}$. For every $r \in \mathcal{R}_{str} \cup \mathcal{R}_{dfs}$ and every $a \in \mathcal{A}$, if $r \in a$ and every $\mathrm{bd}(r) \subseteq a$, then $\mathrm{hd}(r) \in a$.

We can now introduce the concept of *Well-formed Preferred Extension* (WFPE), namely a preferred extension satisfying the above constraint. It is worth mentioning that, in general, a preferred extension (Dung, 1995) is not a WFPE, but a WFPE is always a preferred extension.

Definition 7 A preferred extension of the derived AF $\langle \mathcal{N}_{\mathcal{T}}, \mathcal{O}_{\mathcal{T}} \rangle$ is a *WFPE* iff it satisfies Constraint 5.

The implication is that relative to WFPEs, for $r \in \mathcal{R}_{str}$, the $\mathrm{hd}(r)$ is *sceptically acceptable* relative to the derived AF. On the other hand, for $r' \in \mathcal{R}_{dfs}$, $\mathrm{hd}(r')$ is *credulously acceptable* relative to the derived AF.

To this point, we have Theory Bases, corresponding derived AFs, and constraints on extensions. Fundamental observations of our approach are:

Observation 1 For the literals and the rules which are *true* in every model for the Theory Base $\mathcal{T}$, the corresponding nodes of the WFPEs of the derived AF are *sceptically acceptable*.

Observation 2 For the literals and the rules which are *false* in any model of a Theory Base $\mathcal{T}$, the corresponding nodes of the WFPEs of the derived AF $\langle \mathcal{N}_{\mathcal{T}}, \mathcal{O}_{\mathcal{T}} \rangle$ are not an element of any admissible set.

Both of these follow by the evaluation of a derived framework $\langle \mathcal{N}_{\mathcal{T}}, \mathcal{O}_{\mathcal{T}} \rangle$ relative to a $\mathcal{T}$. Thus, the derived AF is *information-preserving* with respect to the Theory Base. The derived AF is an instantiation of the corresponding Theory Base, and the preferred extensions of the AF correspond to models of the Theory Base. A formal discussion on the semantics correspondence is provided in Section A.3.

3.3. *Examples of the definitions*

We now give some examples of the basic definitions, discuss defeasibility, provide a simple combination of strict and defeasible rules, illustrate reasoning with an assertion in a partial KB, and comment on the connection between the extensions and the classical models. In Section 5, we discuss examples from the literature, which have been discussed as problematic for an ASPIC-type approach, but are unproblematic in the approach presented above.

First, we provide a Theory Base $\mathcal{T}_1$ with just one strict rule, the derived AF, a graphic representation of the derived AF, and then the preferred extensions. Since it is always clear in context where we have literals and rules (in a Theory Base) and where we have labels (in an AF), we use one typographic form without confusion.

Example 1 Let $\mathcal{T}_1$ be the pair with $(\mathcal{L}_1, \mathcal{R}_1)$, where

$$\mathcal{L}_1 = \{x_1, x_2\} \cup \{\neg x_1, \neg x_2\}$$
$$\mathcal{R}_1 = \{r_1\}, \text{where } r_1 \textit{has rule name } r_1 : x_1 \rightarrow x_2$$

The *derived framework* from $\mathcal{T}_1$ is $\langle \mathcal{N}_{\mathcal{T}_1}, \mathcal{O}_{\mathcal{T}_1} \rangle$ in which

$$\mathcal{N}_{\mathcal{T}_1} = \{x_1, x_2\} \cup \{\neg x_1, \neg x_2\} \cup \{r_1\}$$

and in which $\mathcal{O}_{\mathcal{T}_1}$ comprises the union of three disjoint sets:

$$\mathcal{O}^{ll}_{\mathcal{T}_1} = \{\langle x_1, \neg x_1\rangle, \langle \neg x_1, x_1\rangle, \langle x_2, \neg x_2\rangle, \langle \neg x_2, x_2\rangle\}$$
$$\mathcal{O}^{lr}_{\mathcal{T}_1} = \{\langle \neg x_1, r_1\rangle\}$$
$$\mathcal{O}^{rl}_{\mathcal{T}_1} = \{\langle r_1, \neg x_2\rangle\}$$

We graphically represent $\langle \mathcal{N}_{\mathcal{T}_1}, \mathcal{O}_{\mathcal{T}_1} \rangle$ as in Figure 2.
In $\langle \mathcal{N}_{\mathcal{T}_1}, \mathcal{O}_{\mathcal{T}_1} \rangle$, the WFPEs are

$$\{x_1, r_1, x_2\}, \{\neg x_1, x_2\}, \{\neg x_1, \neg x_2\}$$

Each of the nodes is *credulously accepted* and none is *sceptically* accepted. The rule is not *defeated* in the sense that where the premises hold, the conclusion *must* hold. Therefore, if x_1 will be asserted – for example, with an attack against $\neg x_1$, there will be a single WFPE, namely $\{x_1, r_1, x_2\}$. No WFPE contains both x_1 and $\neg x_2$: if x_1 is in the set, then r_1 is in the set; r_1 attacks $\neg x_2$, leaving x_2 in the set; if $\neg x_2$ is in the set, then r_1 must be attacked; r_1 can only be attacked by $\neg x_1$, which also attacks x_1, leaving $\neg x_1$ in the set.

$$x_1 \longleftrightarrow \neg x_1 \longrightarrow r_1 \longrightarrow \neg x_2 \longleftrightarrow x_2$$

Figure 2. AF of $x_1 \rightarrow x_2$.

We observe that the WFPEs with respect to an AF correlate with the models of the Theory Base; in this respect, the derived AF is a means to build models for the Theory Base. This observation applies to strict and defeasible rules alike (Section A.3).

The following is an example of a *defeasible* rule.

Example 2 Let $\mathcal{T}_2$ be the pair with $(\mathcal{L}_2, \mathcal{R}_2)$, where

$$\mathcal{L}_2 = \{x_1, x_2\} \cup \{\neg x_1, \neg x_2\}$$
$$\mathcal{R}_2 = \{r_2\}, \textit{ where } r_2 \textit{ has rule name } r_2 : x_1 \Rightarrow x_2$$

We graphically represent the derived AF $\langle \mathcal{N}_{\mathcal{T}_2}, \mathcal{O}_{\mathcal{T}_2} \rangle$ as in Figure 3.

In $\langle \mathcal{N}_{\mathcal{T}_2}, \mathcal{O}_{\mathcal{T}_2} \rangle$, the WFPEs (which coincide with the preferred extensions of $\langle \mathcal{N}_{\mathcal{T}_2}, \mathcal{O}_{\mathcal{T}_2} \rangle$) are as follows:

$$\{x_1, r_2, x_2\}, \{\neg x_1, x_2\}, \{\neg x_1, \neg x_2\}, \{x_1, \neg_2\}$$

The first three preferred extensions are similar to those for a strict rule. In the last extension, $\neg x_2$ itself attacks the rule node r_2; consequently, either x_1 or $\neg x_1$ are in a preferred extension along with $\neg x_2$. This contrasts with the preferred extension of a derived AF with just strict rules. Indeed, if x_1 is asserted by attacking $\neg x_1$, there will be two WFPEs, namely $\{x_1, r_2, x_2\}$ and $\{x_1, x_2\}$. Therefore, differently from strict rules, a defeasible rule is not 'strong enough' to ensure that x_2 will hold (sceptically) since x_1 holds.

Therefore, to make use of a defeasible rule, one must provide the means to *choose between extensions*, for example, by selecting the extension which maximises the number of applicable defeasible rules, or which uses some notion of priority or entrenchment on the rules. Different ways of making this choice give rise to different varieties of non-monotonic logic (Prakken & Sartor, 1997; Reiter, 1980). Circumscription (McCarthy, 1980) could be used by including additional designated nodes such as $ab(r_1)$ which attack the rule r_1 and attack and are attacked by $notab(r_1)$. We then choose the extension containing the most $notab(r_1)$ nodes. Using DeLP defeaters (García & Simari, 2004), we can specify circumstances where the rule is not applied.

In our third example, we show the interaction of defeasible and strict rules, which was the root of several of the problems identified in Caminada and Amgoud (2007).

Example 3 Suppose $\mathcal{T}_3$ with rules $r_2 : x_1 \Rightarrow x_2$ and $r_3 : x_2 \rightarrow x_3$ which has derived AF $\langle \mathcal{N}_{\mathcal{T}_3}, \mathcal{O}_{\mathcal{T}_3} \rangle$ graphically represented as in Figure 4.

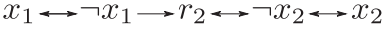

Figure 3. AF of $x_1 \Rightarrow x_2$.

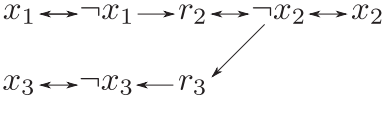

Figure 4. AF derived from $\mathcal{T}_3$ with $x_1 \Rightarrow x_2$ and $x_2 \rightarrow x_3$.

AF $\langle \mathcal{N}_{\mathcal{T}_3}, \mathcal{O}_{\mathcal{T}_3} \rangle$ has the following six preferred extensions:

1. $\{x_1, r_2, x_2, r_3, x_3\}$ 4. $\{\neg x_1, x_2, r_3, x_3\}$
2. $\{x_1, \neg x_2, x_3\}$ 5. $\{\neg x_1, \neg x_2, x_3\}$
3. $\{x_1, \neg x_2, \neg x_3\}$ 6. $\{\neg x_1, \neg x_2, \neg x_3\}$

Given a strict assertion x_1, we would normally choose the preferred extension (1) from (1) to (3), maximising the number of defeasible rules. Thus, normally, we say that x_1 implies x_3. However, we are not obliged to make this choice. In particular, if x_1 and $\neg x_2$ are strictly asserted, r_2 is inapplicable, and x_3 is credulously acceptable ((2) and (3)). Thus, in this AF, a strict assertion of x_1 need not imply that x_3 holds as well. Where the claim of a defeasible rule, for example, x_2 is also a premise of a strict rule, we cannot use it to draw strict inferences about the claim of the strict rule, for example, x_3. Note as well that the defeasible rule is inapplicable where either the claim of the rule ($\neg x_2$) is false ((2), (3), (5), and (6)) or the claim of the strict rule ($\neg x_3$) is false ((3) and (6)). This contrasts with, for example, Reiter (1980), where the defeasible rule is inapplicable only where the claim of the defeasible rule itself is asserted to be false. In the approach here, the falsity of any consequences of the defeasible rule, however remote, will also block the application of the rule.

4. Three senses of argument

In Section 3, we defined AFs for derived theory bases and provided several examples. In this section, we formalise the three senses of argument in this language, introduced in Section 1, which we believe have been conflated in the literature. These senses can be formally articulated in our framework as distinct structures. As discussed in Section 1, the term *argument* is ambiguous (Wyner, Bench-Capon, & Atkinson, 2008), and arguments in a legal setting provide examples. It can mean the reasons for a claim given in one step (an *Argument*); or it can mean a train of reasoning leading towards a claim (a *Case*), that is, a set of linked *Arguments*; or it can be taken as reasons for and against a claim (a *Debate*), that is, a *Case* for the claim and a *Case* against the claim. An additional structure is where the intermediate claims of the *Debate* are also points of dispute, but we will not consider this further here. In the following, we formally define these three senses of *argument* as structures in the AF, starting with *Arguments*, then providing *Cases*, and finally *Debates*. We provide a graphic, examples, and then definitions for the three different kinds of attack: *Rebuttal*, *Undercutting*, and *Undermining*.

We provide a recursive, pointwise definition of a graph which is constructed relative to an AF. Since the sets are constructed relative to a derived AF, we can infer the attack relations which hold among them. The different senses of *argument* are defined as subgraphs. The definition is exercised in Example 4 along with Figure 5.

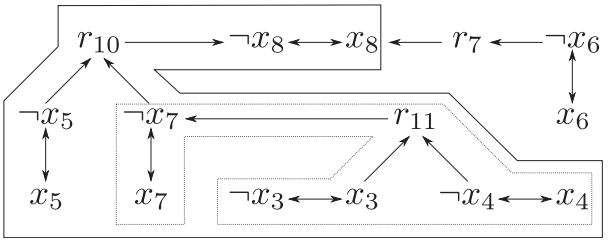

Figure 5. Arguments, cases, and single-point debates.

Definition 8 Suppose there is a derived AF with respect to Theory Base $\mathcal{T} = \langle \mathcal{N}_\mathcal{T}, \mathcal{O}_\mathcal{T} \rangle$, where y and z are arbitrary literals from $\mathcal{N}_\mathcal{T}$, and r and *r*' are arbitrary rules from $\mathcal{O}_\mathcal{T}$. $\mathcal{F}$ abbreviates $\{r : r \textit{ added in } \rho_{2k-1}\}$.

$$\rho_0(y) = \{y, \neg y\}$$

$$\rho_1(y) = \rho_0(y) \cup \bigcup_{\{r:hd(r)=y\}} \{r\}$$

$$\rho_{2k}(y) = \rho_{2k-1}(y) \cup \bigcup_{\{r\in\mathcal{F}\}} \{z, negz : z \in bd(r)\}$$

$$\rho_{2k+1}(y) = \rho_{2k}(y) \cup \bigcup_{\{r\in\mathcal{F}\}} \{r' : z \in hd(r') \cap bd(r)\}$$

$$\rho_{2k+2}(y) = \rho_k(y)$$

$\rho_0(y)$ provides the basis for the construction, which are nodes labelled by literals in an AF that attack one another with respect to the node labelled *y*. At $\rho_1(y)$, we add to the previous set of rules which have *y* as their head, depending on whether we have a strict or a defeasible rule, the rule node attacks and may be attacked by the literal which is the negation of the head. At $\rho_{2k}(y)$, we add the positive and negative literals relative to the body of the rules; each of the negative literals associated with literals of the body of the rule attacks the rule node. At $\rho_{2k+1}(y)$, we link rules: the literals in the body of a rule added at $\rho_1(y)$ serve as the heads of other rules. At $\rho_{2k+2}(y)$, we have iterated the steps $\rho_1(y)$–$\rho_{2k+1}(y)$ until there is no further change. Constructions for negations of literals are similarly defined.

Supposing a derived AF, Arg^{S1} and Arg^{S2} are induced subgraphs of that AF. An *Argument for y*, $Arg^{S1}(y)$, is defined from the elements of $\rho_2(y)$: it is the nodes and their attacks defined at this step relative to the derived AF. A graph defined as $Arg^{S1}(y)$ can *only have one rule in the set of nodes*, namely a rule of the Theory Base with *y* as head (other rules with *y* as head will give rise to distinct arguments for *y* in sense 1). In $Arg^{S1}(y)$, *y* is the *claim* of $Arg^{S1}(y)$ and the literals in the body of the rule are the *premises*. A *Case for y*, $Arg^{S2}(y)$, is defined where $\rho_{k+1}(y) = \rho_k(y)$. $Arg^{S2}(y)$ comprised $Arg^{S1}(y)$ along with graphs of form Arg^{S1} for the literals that are bodies of every rule constructed relative to $Arg^{S1}(y)$. In other words, a Case links together all those graphs of Arguments for a particular *y* where the claim of one rule is the premise of another rule.

Definition 9 Suppose an AF derived from Theory Base $\mathcal{T}$, $\langle \mathcal{N}_\mathcal{T}, \mathcal{O}_\mathcal{T} \rangle$.

We define Arg^{S1}–Arg^{S2} as subgraphs of a derived AF (note that since both Arg^{S1} and Arg^{S2} are *subgraphs* of the AF $\langle \mathcal{N}_\mathcal{T}, \mathcal{O}_\mathcal{T} \rangle$ derived from the Theory Base $\mathcal{T}$ only attacks present in $\mathcal{O}_{TC}$ can be used within them).

$$\textit{An Argument for y is } Arg^{S1}(y) = \langle \mathcal{N}_\mathcal{T}^{S_1 y},\ \mathcal{O}_\mathcal{T}^{S_1 y} \rangle,$$

$$\textit{where } \mathcal{N}_\mathcal{T}^{S_1 y} \subseteq \mathcal{N}_\mathcal{T} \textit{ and } \mathcal{O}_\mathcal{T}^{S_1 y} \subseteq \mathcal{O}_\mathcal{T},$$

$$\forall r, r' \in \mathcal{N}_\mathcal{T}^{S_1 y} r = r',\ \textit{is a subgraph at } \rho_{2k}(y).$$

Furthermore, in an argument $Arg^{S1}(y) = \langle \mathcal{N}_\mathcal{T}^{S_1 y},\ \mathcal{O}_\mathcal{T}^{S_1 y} \rangle$ for *y*, all of the following hold:

(A1) $y \in \mathcal{N}_\mathcal{T}^{S_1 y}$.
(A2) For all $p \in \mathcal{N}_\mathcal{T}^{S_1 y} \setminus \{y\}$ there is a directed path of attacks in $\mathcal{O}^{S_1 y}$ from *p* to *y*.
(A3) There is exactly one $r \in \mathcal{N}_\mathcal{T}^{S_1 y}$ for which $hd(r) = y$ and, for this rule, $bd(r) \subset \mathcal{N}_\mathcal{T}^{S_1 y}$.

Note that (*A*2) ensures $Arg^{S1}(y)$ is 'connected' since no argument within it is isolated from y. Turning to the concept of *Case for y*

$$\begin{aligned}&\textit{A Case for y is } Arg^{S2}(y) = \langle \mathcal{N}_{\mathcal{T}}^{S_2 y},\ \mathcal{O}_{\mathcal{T}}^{S_2 y}\rangle,\\ &\textit{where } \mathcal{N}_{\mathcal{T}}^{S_2 y} \subseteq \mathcal{N}_{\mathcal{T}} \textit{ and } \mathcal{O}_{\mathcal{T}}^{S_2 y} \subseteq \mathcal{O}_{\mathcal{T}} \textit{ is a subgraph}\\ &\textit{at } \rho_{k+1}(y) = \rho_k(y).\end{aligned}$$

Furthermore (*A*1) and (*A*2) must continue to hold while (*A*3) is replaced by the requirement that all rules, r, with $hd(r) = y$ are represented and expanded.

Where we have Arg^{S2}(y) and $Arg^{S2}(\neg y)$, we have a *Single-point Debate about y*, $Arg^{S3}(y)$. The two graphs share only the literals $\{y, \neg y\}$, and no other rules or literals.

Definition 10 Suppose two derived AFs, $Arg^{S2}(\neg y) = \langle \mathcal{N}_{\mathcal{T}}^{S_2 \neg y}, \mathcal{R}_{\mathcal{T}}^{S_2 \neg y}\rangle$ and $Arg^{S2}(y) = \langle \mathcal{N}_{\mathcal{T}}^{S_2 y},\ \mathcal{O}_{\mathcal{T}}^{S_2 y}\rangle$:

$$\begin{aligned}&\textit{A Single-point Debate about y is}\\ &Arg^{S3}(y) = \langle \mathcal{N}_{\mathcal{T}}^{S_2 y} \cup \mathcal{N}_{\mathcal{T}}^{S_2 \neg y}, \mathcal{O}_{\mathcal{T}}^{S_2 y} \cup \mathcal{O}_{\mathcal{T}}^{S_2 \neg y}\rangle,\\ &\textit{where } \mathcal{N}_{\mathcal{T}}^{S_2 \neg y} \cap \mathcal{N}_{\mathcal{T}}^{S_2 y} = \{y, \neg y\}\\ &\textit{and } \mathcal{O}_{\mathcal{T}}^{S_2 \neg y} \cap \mathcal{O}_{\mathcal{T}}^{S_2 y} = \emptyset.\end{aligned}$$

Clearly a debate with subsidiary debates can be constructed to argue pro and con about other literals in the base debate; we start with a Arg_{S2}(y), then add further *Single-point Debates* about some literal in the graph other than y.

Example 4 shows the senses in a derived AF only with strict rules since they restrict the available preferred extensions.

Example 4 Suppose a Theory Base comprised of the rules (and related literals): $r_7 : x_6 \rightarrow \neg x_8$, $r_{10} : x_5, x_7 \rightarrow x_8$, $r_{11} : \neg x_3, x_4 \rightarrow x_7$. Figure 5 graphically represents the various senses of *argument* in an AF derived from this Theory Base.

In Figure 5, we have three subgraphs which represent an Argument; each Argument is derived from the corresponding rule of the Theory Base. For example $Arg^{S1}(\neg x_8)$ (with a grey shade), the argument for $\neg x_8$ is the graph comprised of nodes $\{\neg x_8, x_8, r_7, \neg x_6, x_6\}$ with the relations among them as given; the graph is derived from the rule of the Theory Base which corresponds to $r_7 : x_6 \rightarrow \neg x_8$. The other two rules of the Theory Base are also represented in the graph as subgraphs that represent an Argument. Figure 5 presents two Cases. The Case $Arg^{S2}(x_8)$ (with a solid outline) is derived from the following rules: $r_{10} : x_5, x_7 \rightarrow x_8, r_{11} : \neg x_3, x_4 \rightarrow x_7$. We see how the Arguments in the Case are linked; for instance, the graph of $r_{11} : \neg x_3, x_4 \rightarrow x_7$ (with dotted outline) has as claim x_7, which is the premise of $r_{10} : x_5, x_7 \rightarrow x_8$. The Case $Arg^{S2}(\neg x_8)$ is derived from the following rule (recall that an Argument can also be a Case): $r_7 : x_6 \rightarrow \neg x_8$. The Single-point Debate for x_8, $Arg^{S3}(x_8)$, comprised the Cases $Arg^{S2}(x_8)$ and $Arg^{S2}(\neg x_8)$.

4.1. *Rebuttal, undermining, and undercutting*

Given these structures we can express the various familiar notions of attack on an argument: a *Undermining of an Argument* is an Argument with claim that is the negation of the premise of

another Argument; a *Rebuttal of an Argument* is an Argument with a claim that is the negation of the claim of another Argument; the *Rebuttal of a Case* is similar to the Rebuttal of an Argument; and an *Undercutter of an Argument* is an attacker of the rule node of an argument.

Definition 11 Suppose an AF derived from Theory Base $\mathcal{T}$, $\langle \mathcal{N}_\mathcal{T}, \mathcal{O}_\mathcal{T} \rangle$.

An Underminer of an Argument for y,

$Arg^{S1}(y) = \langle \mathcal{N}_\mathcal{T}^{S_1 y}, \mathcal{O}_\mathcal{T}^{S_1 y} \rangle$, *is* $Arg^{S1}(\neg y_i)$,

where y_i, $r \in \mathcal{N}_\mathcal{T}^{S_1 y}$ *and* $y_i \in bd(r)$.

An Undercutter of a Argument for y,

$Arg^{S1}(y) = \langle \mathcal{N}_\mathcal{T}^{S_1 y}, \mathcal{O}_{TC}^{S_1 y} \rangle$, *is* $Arg^{S1}(\neg y_i)$,

where y_i, $r \in \mathcal{N}_\mathcal{T}^{S_1 y}$ *and* $y_i \in hd(r)$.

A Rebuttal of an Argument for y, $Arg^{S1}(y)$, *is* $Arg^{S1}(\neg y)$.

A Rebuttal of a Case for y, $Arg^{S2}(y)$, *is* $Arg^{S2}(\neg y)$.

Given the definitions of the three senses of 'argument' and various notions of attack, it would be possible to define a more abstract, derivative AF in which we represent structures at levels Arg^{S1} or Arg^{S2} as nodes in that AF and which, given the requisite attack relations defined above, are in a higher level attack relation. However, we leave such proposals and the analysis of them for future research.

4.2. *Assertions*

So far we have only PRules in a Single-point Debate. Usually in a Theory Base there are assertions which further restrict the preferred extensions. In our framework, assertions are ARules. A Case such as $Arg^{S2}(y)$ is a directed tree where y is the root and the literals in the bodies of constituent arguments at the level where $\rho_{k+1}(y) = \rho_k(y)$ are the leaves. If all the leaves are strictly asserted and all the rules are strict rules, then the root is sceptically accepted. However, where even just one of the leaves of $Arg^{S2}(y)$ is defeasibly asserted, then the root of the tree is only credulously accepted. Such a Case is vulnerable to attack. Similarly, where there are defeasible rules, the root is always credulously accepted. Where we have Arg^{S2}(y) and $Arg^{S2}(\neg$y) for and against a claim and the leaves of each Case are strictly asserted, then the resultant Single-point Debate must be adjudicated according to some principle for choosing between preferred extensions. Matters are complex where we have strict and defeasible assertions of literals at *different levels of the tree*. Further investigation would consider how to adopt some notion of *accrual* in AFs to overcome some of these limitations (Modgil & Bench-Capon, 2010; Prakken, 2005). However, these matters are beyond the scope of this paper.

5. Discussion

In this section, we compare our approach to ASPIC, discuss other approaches, and then outline several additional advantages.

5.1. *Comparison to ASPIC with a base-case example*

In this section, we exemplify the differences between our approach and ASPIC. In Section 5.1.1, we show how the *ex falso quodlibet* and the *tertium non datur* are automatically satisfied by our approach, while they require additional rules in ASPIC. Section 5.1.1 provides an example from Caminada and Amgoud (2007), which highlights an issue in the instantiation method of ASPIC (Caminada and Amgoud, 2007) (see Section 2.2) as well as motivates the *Rationality Postulates*. We then show how such problems do not arise in our approach.

5.1.1. *Credulous reasoning*

Let us consider a Theory Base $\mathcal{T}_{cr}$ with the following rules:

$$r_{cr1} :\rightarrow \neg x_1;\ r_{cr2} :\Rightarrow x_1;\ r_{cr3} : x_1 \Rightarrow x_2.$$

ASPIC constructs the following three arguments:

$$A_{cr1} : [\rightarrow \neg x_1];$$
$$A_{cr2} : [\Rightarrow x_1];$$
$$A_{cr3} : [[\Rightarrow x_1] \Rightarrow x_2].$$

Since $\neg x_1$ is strictly asserted, then A_{cr1} defeats both A_{cr2} and A_{cr3}. This results in having $\neg x_1$ as the only 'accepted' conclusion.

Instead, using the approach described in Section 3, the resulting framework is shown in Figure 6.

In particular, the shown framework has two WFPEs, namely $\{r_{cr1}, \neg x_1, x_2\}$, and $\{r_{cr1}, \neg x_1, \neg x_2\}$. $\neg x_1$ is sceptically accepted, but the system acknowledges the existence of x_2 and forces the choice between assuming either x_2 or $\neg x_2$.

To obtain the same extensions using ASPIC, $\mathcal{T}_{cr}$ should be enlarged with the following rules:

$$\Rightarrow x_2;\ \Rightarrow \neg x_2.$$

These rules give rise to two further arguments in favour of assuming either x_2 or $\neg x_2$:

$$A_{cr4} : [\Rightarrow x_2];$$
$$A_{cr5} : [\Rightarrow \neg x_2].$$

Thus, as pointed out in the introduction, ASPIC requires auxiliary rules that our approach does not.

5.1.2. *Caminada and Amgoud Example 1*

Consider a Theory Base with strict and defeasible rules from which we construct arguments according to the definitions of ASPIC (Caminada & Amgoud, 2007) outlined in Section 2.2. We examine Example 5 of Caminada and Amgoud (2007).

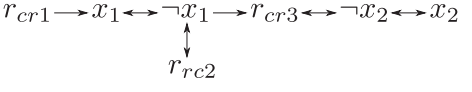

Figure 6. Credulous reasoning.

Example 5 Let $\mathcal{T}_4$ be a Theory Base with the following rules:

$$r_{21} :\rightarrow x_1;\ r_{22} :\rightarrow x_2;\ r_{23} :\rightarrow x_3;\ r_{24} : x_4, x_5 \rightarrow \neg x_3;\ r_{25} : x_1 \Rightarrow x_4;\ r_{26} : x_2 \Rightarrow x_5.$$

We construct the following arguments:

$$A_1 : [[\rightarrow x_1] \Rightarrow x_4];\ A_2 : [[\rightarrow x_2] \Rightarrow x_5];\ A_3 : [\rightarrow x_3];$$
$$A_4 : [\rightarrow x_1];\ A_5 : [\rightarrow x_2];$$
$$A_6 : [[[\rightarrow x_1] \Rightarrow x_4],\ [[\rightarrow x_2] \Rightarrow x_5] \rightarrow \neg x_3].$$

We see clearly that arguments can have subarguments: A_6 has a subargument A_1, A_1 has a subargument A_4, and A_6 has a subargument A_4.

Several additional elements are needed to define *justified conclusions*. An argument is strict if it has no defeasible subargument, otherwise it is defeasible (non-strict). An argument A_i rebuts an argument A_j where the conclusion of some subargument of A_i is the negation of the conclusion of some non-strict subargument of A_j; rebuttal is one way an argument defeats another argument. Admissible argument orderings specify that a strict argument (containing premises that are axioms and rules that are strict) can defeat a defeasible argument, but not vice versa. Moreover, one argument can defeat another argument with respect to subarguments; in effect, defeat of a part is inherited as defeat of a whole. With respect to our example, the undefeated arguments are A_1, A_2, A_3, A_4, and A_5. A_3, which is a strict argument, defeats A_6 but not vice versa since A_6 is a non-strict argument in virtue of having a defeasible subargument. Given the arguments and defeat relation between them, we can provide an AF and the different extensions. The `Output` of an AF, understood as the *justified conclusions* of the AF, is given as the sceptically accepted conclusions of the arguments of the AF.

With respect to the example, Caminada and Amgoud (2007) claim that the justified conclusions are x_1, x_2, x_3, x_4, and x_5 since these are all conclusions of arguments which are not attacked (it appears not to be an example analysed in Prakken, 2010). However, $\neg x_3$ is *not* a justified conclusion, even though it is the conclusion of a strict rule in which all the premises are justified conclusions. This is so since the argument A_6 of which $\neg x_3$ is the conclusion is defeated by but does not defeat A_3 because A_6 has a *subargument* which is a non-strict argument (namely A_1 or A_2), so making A_6 a non-strict argument, while A_3 is a strict argument. Yet, given the antecedents of the strict rule are justified conclusions, it would seem intuitive that the claim of a strict rule should also be a justified conclusion. This, they claim, shows that justified conclusions are not closed under strict rules or could even be inconsistent.

In our view, these notions of argument and defeat are problematic departures from Dung (1995), which has no notion of subargument or of defeat in terms of subarguments. In addition, they give rise to the problems with justified conclusions: what is a strict rule in the Theory Base can appear in the AF as a non-strict argument in virtue of subarguments; what cannot be false in the Theory Base without contradiction is defeated in the AF; thus, what 'ought' to have been a justified conclusion is not. In addition, the notion of justified conclusion leads to some confusion: on the one hand, it only holds for sceptically accepted arguments, which presumably implies that the propositions which constitute them are sceptically accepted; on the other hand, there is no reason to expect that $\neg x_3$ is sceptically accepted, given that it only follows from defeasible antecedents. Clearly the anomaly arises because of the way that arguments can have defeasible subarguments, that the defeat of the whole can be determined by the defeat of a part, and that justified conclusions depend on these notions.

In our approach, the results are straightforward and without anomaly; we do not make use of arguments with subarguments, inheritance of defeasiblity, or problematic notions of justified

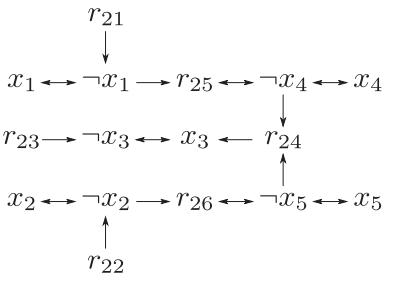

Figure 7. Graph of problem example.

conclusions. We consider a key example from Caminada and Amgoud (2007) as the two other problematic examples cited in Caminada and Amgoud (2007) follow suit. The Theory Base of Example 5 appears as in Figure 7, for which all the preferred extensions for the AF are given. For clarity and discussion, we include undefeated strict and defeasible rules.

1. $\{x_1, r_{21}, r_{25}, x_2, r_{22}, x_3, x_4, r_{25}, \neg x_5\}$
2. $\{x_1, r_{21}, r_{23}, x_2, r_{22}, x_3, \neg x_4, x_5, r_{26}\}$
3. $\{x_1, r_{21}, r_{23}, x_2, r_{22}, x_3, \neg x_4, \neg x_5\}$
4. $\{x_1, r_{21}, r_{23}, r_{25}, x_2, r_{22}, r_{24}, r_{26}, x_4, x_5\}$

Notice that WFPEs (1)–(3) are unproblematic with respect to *consistency* and *closure*. They also satisfy Definition 7, so are the *relevant* extensions to consider. In contrast, extension (4) is problematic in an argumentation theory *without* Definition 7 since the conclusions of strict rules are missing, thus violating *closure*. Yet, (4) does not satisfy Constraint 5: the premises and rule nodes of strict rules are present, but the conclusions are not. With respect to those extensions that satisfy Definition 7, x_1, x_2, x_3 are all sceptically accepted, while x_4 and x_5 are credulously accepted. $\neg x_3$ is not credulously accepted given that x_3 is strictly asserted. Note that every literal which is strictly asserted is sceptically acceptable. Therefore, the rule node r_{24} must be defeated where one or both of $\neg x_4$ and $\neg x_5$ hold. There is, in our view, no reason to expect $\neg x_3$ to hold in any extension since we have no preferred extension in which both x_4 and x_5 are justified conclusions. Given WFPEs, we satisfy the *consistency* rationality postulate; *closure*, which is relevant only if strict rules where all the body literals hold, is not relevant to this problem. Moreover, we can provide machinery to meet Definition 7 in that we can examine the extensions relative to the rules of the AF to determine if Constraint 5 is satisfied. The analysis also corresponds well with model-building for classical logic.

5.2. *Other approaches*

We have considered a widely adopted approach to instantiating Theory Bases in AFs (Caminada & Amgoud, 2007) along with the problems that arise. There are other approaches to instantiating a KB in an AF that may avoid problems with the Rationality Postulates such as Assumption-based (Bondarenko et al., 1997) or Logic-based (Besnard & Hunter, 2008) argumentation. However, these approaches, like the ASPIC approach, follow the three-step structure of Figure 1; we leave further comparison and contrast to future work.

Here we comment briefly on a closely related proposal (Strass, 2013) set in the different approach of *Abstract Dialectical Frameworks* (ADF) (Brewka & Woltran, 2010), which is

presented as a generalisation of Dungian AFs but also as a means to represent instantiated arguments, for example, logic programs (Brewka and Woltran, 2010). Broadly, we may distinguish between approaches based on Dung (1995) that make use of nodes (arguments) and arcs (attacks) alone to determine extensions and those which use *auxiliary conditions to specify extensions with respect to successful attacks* such as preferences (Prakken, 2010) or values (Bench-Capon, 2003). The approach of Brewka and Woltran (2010) is a generalisation of the latter approach: in addition to nodes (which can be statements or literals) and links (generalised from arcs as attacks), there are acceptance conditions, which are functions for each statement from its parents (those nodes in a single link) to {*in*,*out*}. Given this generic approach to acceptance conditions, many complex aspects of argumentation can be accommodated.

Strass (2013) translates Wyner et al. (2009, 2013) into an ADF and applies it to the same problem presented in Figure 7. The approach has a version of Constraint 4, adding *support* along with *attack* and a form of negation on rules (interpreted as *inapplicability*). The system is proven to abide by the Rationality Constraints, though this seems to hinge on the constraint. The main issue, in our view, is that ADF requires the generation of and reasoning with (presumably correct) acceptance conditions in addition to the AF rather than on the graph per se, which was one of the main advantages of the Dungian abstraction.

5.3. *Additional advantages*

Our approach has three additional strengths, which we discuss briefly here, leaving for future work further comparison and contrast.

5.3.1. *The two step*

In Section 2.2, we outlined the argument construction method of ASPIC, which follows the three-step structure of Figure 1 that generates arguments from a KB: arguments are complex 'objects' constructed from the KB, then abstracted over in the AF; once arguments and attacks are determined, we have an AF, from which we determine extensions and consequently the justified conclusions. Yet, this can be construed to *overgenerate* arguments: for instance in Besnard and Hunter (2008), significant effort is devoted to analysis of relations between arguments and the specification of *canonical arguments* (we are not aware of similar analysis in the ASPIC framework). One might regard the arguments of ASPIC or a Logic-based approach as *intermediate concepts* (Wyner, 2008) or as *catalysts* which, having served their purpose in determining justified conclusions, are no longer essential to reasoning.

By contrast, while arguments (in their various senses) can be similarly constructed as in Section 4, this is not necessary in our framework in order to calculate the justified conclusions (our WFPEs); so far as the AF is concerned, the nodes in the graph are atomic, that is, literal or rule nodes, in specified attack relations; WFPEs are determined *only* with respect to such nodes and attacks. In other words, arguments such as in ASPIC or Logic-based argumentation are not essential, and there is no overgeneration of arguments as complex 'objects'. This straightforwardly simplifies analysis as well as the calculation of justified conclusions from the three steps of Figure 1 to the two steps of Figure 8.

5.3.2. *Partial information*

In ASPIC or Logic-based approaches to argumentation, an argument is a rule together with the premises of the rule from which the conclusion of the rule follows. Yet it is unclear whether an argument is well-formed if *no conclusion follows*. This may arise in a partial KB with a rule where only some or none of the premises are asserted. In our approach, this is unproblematic. Suppose

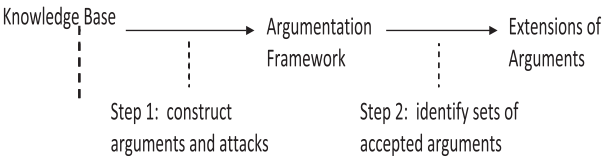

Figure 8. Two steps of argumentation.

a Theory Base with only the following two rules: $\rightarrow x_1$ and $r_4 : x_1, x_3 \rightarrow x_2$. The extensions are

$$\{x_1, x_3, r_4, x_2\}, \{x_1, \neg x_3, x_2\}, \{x_1, \neg x_3, \neg x_2\}$$

In this respect, our approach differs markedly from approaches to instantiated AFs that rely on asserted premises and rules to construct well-formed arguments to provide extensions and then reason about justified conclusions (e.g. Besnard & Hunter, 2008; Prakken, 2010).

5.3.3. *Information dynamics*

Related to the points in Sections 5.3.1 and 5.3.2, ASPIC or Logic-based argumentation require a *recalculation* of the arguments in the AF when the information in the KB dynamically changes by addition or deletion, which is followed by a recalculation of argument extensions and justified conclusions. In our framework, without such intermediate concepts, changes in the KB correlate straightforwardly to changes in the derived AF. There is, moreover, no redundancy of literal or rule nodes or of attacks among them, which may arise in ASPIC or Logic-based argumentation. While recalculation of WFPEs is required, it is clearly a simplification.

6. Concluding remarks and future work

We have presented a method of instantiating a Theory Base which contains strict and defeasible rules in a Dung-style abstract AF, building on and refining Wyner et al. (2009, 2013). We take a two-step approach, where the Theory Base is translated to a corresponding AF, and the conclusions of the Theory Base can be computed as extensions of that AF. This is distinct from the widespread three-step approach, where arguments are generated from the Theory Base, organised, and evaluated in an AF, yielding extensions from which justified conclusions are extracted. Our approach is a rather intuitive way of instantiating Theory Bases in AFs, directly provides justified conclusions as WFPEs, and formally expresses the variety of senses of 'argument', which are structures within the framework. By separating the notion of a node from the ambiguous notion of argument, we have clear criteria for what constitutes a node and attack among nodes in the framework, enabling us to explain our reasoning in terms of arguments of the appropriate granularity, for example, presenting an argument, making a case, and having a debate. In addition to accounting for benchmark problems in argumentation without a priori choices as in ASPIC or acceptability conditions as in ADF, our approach has several attractive features: arguments are not overgenerated; we can determine extensions for partial KBs; and the approach can handle information dynamics.

In future work, we will demonstrate the formal properties of our approach. In addition, we will further compare and contrast approaches to Theory Base instantiation in AFs. An important avenue of exploration and development is to add values, preferences, and weights to the KB, which then appear in the graph. In a different vein, we will explore the potential for improved explanation offered by our distinction between various senses of the term 'argument'.

Acknowledgements

The authors thank the anonymous reviewers for their helpful comments. They also thank the ASPARTIX research group (Egly, Alice Gaggl, and Woltran, 2010) for their useful web application which has been widely used for this research. The authors appreciate the comments from reviewers and conference participants. Errors and misunderstandings rest with the authors.

Conflict of interest disclosure statement

No potential conflict of interest was reported by the authors.

Note

1. In alternative terminology, we may say *to structure arguments*, which we use interchangeably.

References

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

Bench-Capon T.J.M. (2003). Persuasion in practical argument using value-based argumentation frameworks. *Journal of Logic and Computation*, 13, 429–448.

Besnard P., & Hunter A. (2008). *Elements of argumentation*. Cambridge, MA: MIT Press.

Bondarenko A., Dung P.M., Kowalski R.A., & Toni F. (1997). An abstract, argumentation-theoretic approach to default reasoning. *Artificial Intelligence*, 93, 63–101.

Brewka G., & Woltran S. (2010). *Abstract dialectical frameworks*. Proceedings of the twelfth international conference on the principles of knowledge representation and reasoning (KR 2010), Toronto, pp. 102–211.

Caminada M., & Amgoud L. (2007). On the evaluation of argumentation formalisms. *Artificial Intelligence*, 171, 286–310.

Caminada M., & Wu Y. (2011, November). On the limitations of abstract argumentation. In P.D. Causmaecker, J. Maervoet, T. Messelis, K. Verbeeck, & T. Vermeulen (Eds.), *Proceedings of the 23rd Benelux conference on artificial intelligence* (pp. 59–66). Ghent.

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

Dunne P.E., & Bench-Capon T.J.M. (2002). Coherence in finite argument systems. *Artificial Intelligence*, 141, 187–203.

Egly U., Alice Gaggl S., & Woltran S. (2010). Answer-set programming encodings for argumentation frameworks. *Argument & Computation*, 1, 147–177.

García A.J., & Simari G.R. (2004). Defeasible logic programming: An argumentative approach. *Theory and Practice of Logic Programming*, 4, 95–138.

Gebser M., Kaminski R., Kaufmann B., & Schaub T. (2012). *Answer set solving in practice*. Morgan & Claypool.

Gelfond M. (2008). Answers set. In F. van Harmelen, V. Lifschitz, & B. Porter (Eds.), *Handbook of knowledge representation* (Chap. 7, pp. 285–316). Amsterdam: Elsevier.

Gelfond M., & Lifschitz V. (1988). *The stable model semantics for logic programming*. ICLP/SLP, Seattle, WA, pp. 1070–1080.

Gelfond M., & Lifschitz V. (1991). Classical negation in logic programs and disjunctive databases. *New Generation Computing*, 9, 365–385.

Hoffmann M.H. (2005). Logical argument mapping: A method for overcoming cognitive problems of conflict management. *International Journal of Conflict Management*, 16, 304–334.

Laronge J.A. (2009). A generalizable argument structure using defeasible class-inclusion transitivity for evaluating evidentiary probative relevancy in litigation. *Journal of Logic and Computation*, 22, 129–162.

Lewis D. (1975). Adverbs of quantification. In *Formal semantics of natural language* (pp. 178–188). Cambridge: Cambridge University Press.
Lifschitz V., Pearce D., & Valverde A. (2001). Strongly equivalent logic programs. *ACM Transactions on Computational Logic*, 2, 526–541.
McCarthy J. (1980). Circumscription – a form of non-monotonic reasoning. *Artificial Intelligence*, 13, 27–39.
Modgil S., & Bench-Capon T. (2010). *Integrating dialectical and accrual modes of argumentation*, Proceedings of COMMA 2010, Desenzano del Garda, pp. 335–346.
Prakken H. (2005). *A study of accrual of arguments, with applications to evidential reasoning*. Proceedings of ICAIL 2005, Bologna, pp. 85–94.
Prakken H. (2010). An abstract framework for argumentation with structured arguments. *Argument and Computation*, 1, 93–124.
Prakken H., & Sartor G. (1997). Argument-based extended logic programming with defeasible priorities. *Journal of Applied Non-Classical Logics*, 7, 25–75.
Reiter R. (1980). A logic for default reasoning. *Artificial Intelligence*, 13, 81–132.
Strass H. (2013). *Computational Logic in Multi-Agent Systems, 14th InternationalWorkshop, CLIMA XIV, Corunna, Spain, Proceedings*, Lecture Notes in Artificial Intelligence (Vol. 8143, pp. 86–101). Berlin: Springer-Verlag.
Wyner A. (2008). An ontology in OWL for legal case-based reasoning. *Artificial Intelligence and Law*, 16, 361–387.
Wyner A., Bench-Capon T., & Atkinson K. (2008). Three senses of 'argument'. In G. Sartor, P. Casanovas, R. Rubino, & N. Casellas (Eds.), *Computable models of the law: Languages, dialogues, games, ontologies* (pp. 146–161, LNAI 4884). Dordrecht: Springer.
Wyner A., Bench-Capon T., & Dunne P. (2009). *Instantiating knowledge bases in abstract argumentation frameworks*. Proceedings of the uses of computational argumentation. AAAI Fall Symposium, Arlington, VA.
Wyner A., Bench-Capon T.J.M., & Dunne P.E. (2013). *On the instantiation of knowledge bases in abstract argumentation frameworks*. CLIMA, Coruna, pp. 34–50.

Appendix 1. Semantics for the Theory Base

In line with the spirit of the paper, we do not provide too technical or in depth a discussion; and we focus on the finite case only, which is also the most relevant given our goal to model real-world situations. Moreover, we will express the semantics of a Theory Base using an ASP program as it has a very simple syntax, although it is fully formalised.

A.1. *ASP in a nutshell*

In the following, we provide a brief overview of the syntax and the semantics of Answer Set Prolog – a logic programming language based on the answer sets semantics introduced by Gelfond and Lifschitz (1991). In what follows, we will only consider propositional programs (an interested reader is referred to Gebser, Kaminski, Kaufmann, and Schaub, 2012; Gelfond, 2008).

A *propositional logic program* Π is a set of finite rules r over a set of ground atoms. An atom l and its negation $\neg l$ are referred as *literals*. A rule r is of the form:

$$l_0; \ldots; l_k \leftarrow l_{k+1}, \ldots, l_m, \ not\ l_{m+1}, \ldots, not\ l_n, \tag{1}$$

where l_i's are ground atoms. Connective *not* and ; are called, respectively, *negation as failure* and *epistemic disjunction*.

If r is a rule of type (A1) then $head(r) = \{l_0, \ldots, l_k\}$, $body(r) = \{l_{k+1}, \ldots, l_m, \ not\ l_{m+1}, \ldots, not\ l_n\}$, $body^+(r) = \{l_{k+1}, \ldots, l_m\}$, and $body^-(r) = \{l_{m+1}, \ldots, l_n\}$.

A rule r such that $body(r) = \emptyset$ is called a fact and it is written as

$$l_0; \ldots; l_k. \tag{2}$$

A set X of literals is a *model* of a propositional logic program Π if $head(r) \cap X \neq \emptyset$ whenever $body^+(r) \subseteq X$ and $body^-(r) \cap X = \emptyset$ for every $r \in \Pi$. Given this, a set X is a *stable model* of Π if X is some $\subseteq$-minimal model of $\Pi^X = \{head(r) \leftarrow body^+(r) \mid r \in \Pi, body^-(r) \cap X = \emptyset\}$.

A.2. *A principled transformation of a Theory Base into an ASP*

Recalling our previous discussions in Sections 1 and 3.1, there are three main components in a Theory Base, namely:

- the implicit assumption that each literal must be either true or false (tertium non datur);
- defeasible rules, namely, rules that 'usually' (Lewis, 1975) hold;
- strict rules, namely, rules that 'always' (Lewis, 1975) hold.

In order to formalise the above three assumption and concepts, let us consider a generic but fixed Theory Base $\mathcal{T} = (\mathcal{L}, \mathcal{R})$ such that

- $\mathcal{L} = \mathcal{L}^+ \cup \mathcal{L}^-$, where $\mathcal{L}^+ = \{x_1, \ldots, x_n\}$ and $\mathcal{L}^- = \{\neg x \mid x \in \mathcal{L}^+\}$;
- $\mathcal{R} = \mathcal{R}_{str} \cup \mathcal{R}_{dfs}$, where
- $\mathcal{R}_{str} = \{s_1, \ldots, s_n\}$;
- $\mathcal{R}_{dfs} = \{d_1, \ldots, d_m\}$.

Given a Theory Base $\mathcal{T}$, $\Pi_{\mathcal{T}}$ is the *propositional logic program corresponding to* $\mathcal{T}$.

A.2.1. *Tertium non datur*

In order to ensure that each stable model includes all the literals of the Theory Base, let us add this initial fact:

$$\forall l \in \mathcal{L}^+ \quad l; \neg l. \in \Pi_{\mathcal{T}} \tag{3}$$

Moreover, in order to provide an easy comparison with the AF derived from a Theory Base (cf. Section 3.2), let us include in the propositional logic program corresponding to $\mathcal{T}$ literals representing the rules themselves. In particular, if a rule r is in a stable model, this means that r has been 'used' to derive some other literals.

$$\forall r \in \mathcal{R} \quad r; \neg r. \in \Pi_{\mathcal{T}} \tag{4}$$

A.2.2. *Strict rules*

A rule $s \in \mathcal{R}_{str}$ represents the notion that hd(s) 'always' holds if *all* of the literals in $bd(s)$ simultaneously hold. This 'always' has thus a specific semantics: if s has been 'used' to derive some other literals, then all the premises must hold; and if all the premises hold, then the rule s must be 'used' to derive some other literals. This notion is captured by the following three sets of rules:

$$\forall s \in \mathcal{R}_{str} \quad hd(r) \leftarrow s, bd(r). \in \Pi_{\mathcal{T}} \tag{5}$$

$$\forall s \in \mathcal{R}_{str}, \forall b \in bd(s) \quad b \leftarrow s. \in \Pi_{\mathcal{T}} \tag{6}$$

$$\forall s \in \mathcal{R}_{str} \quad s \leftarrow bd(s). \in \Pi_{\mathcal{T}} \tag{7}$$

A.2.3. *Defeasible rules*

A rule $d \in \mathcal{R}_{dfs}$ represents the notion that $hd(d)$ 'usually' holds if *all* of the literals in $bd(d)$ simultaneously hold, but there are circumstances where $\neg hd(d)$ holds though *all* of the literals in $bd(d)$ simultaneously hold.

This is captured by the following three sets of rules:

$$\forall d \in \mathcal{R}_{dfs} \quad hd(d) \leftarrow d, bd(d). \in \Pi_{\mathcal{T}} \tag{8}$$

$$\forall d \in \mathcal{R}_{dfs}, \forall b \in bd(d) \quad b \leftarrow d. \in \Pi_{\mathcal{T}} \tag{9}$$

$$\forall d \in \mathcal{R}_{dfs} \quad d \leftarrow hd(d), bd(d). \in \Pi_{\mathcal{T}} \tag{10}$$

A.3. *Semantics of $\mathcal{T}$ and its relationship to WFPEs*

Given a Theory Base $\mathcal{T}$, its corresponding propositional logic program $\Pi_{\mathcal{T}}$ contains all and only the rules (A3)–(A10).

A set of literals and rules $X \subseteq \mathcal{L} \cup \mathcal{R}$ is a model for $\mathcal{T}$ iff X is a stable model of $\Pi_{\mathcal{T}}$.

We can finally provide a formal correspondence between $\mathcal{T}$ models and WFPEs of derived AF (cf. Definition 7). To that end, let us introduce two operators: the first one removes from a model all the literals representing negative rules. Formally, given X a model for $\mathcal{T} = (\mathcal{L}, \mathcal{R})$, $X_{\downarrow} = \{x \in X \mid \nexists r \in \mathcal{R} \text{ s.t. } x = \neg r\}$.

The second operator adds a literal representing negative rules if such rules are not present in the given set: formally, given $\mathcal{T} = (\mathcal{L}, \mathcal{R})$ and $X \subseteq \mathcal{L} \cup \mathcal{R}$, $X^{\uparrow} = X \cup \{\neg r \mid r \in \mathcal{R} \text{ and } r \notin X\}$.

Finally, given a Theory Base $\mathcal{T}$ and its derived $\text{AF}_{\mathcal{T}}$, if X is a model for $\mathcal{T}$, then $X_{\downarrow}$ is a WFPEs for $\text{AF}_{\mathcal{T}}$. Moreover, if X is a WFPEs for $\text{AF}_{\mathcal{T}}$, then $X^{\uparrow}$ is a model for $\mathcal{T}$.

A.4. *Examples*

To illustrate the relationship between a Theory Base and it corresponding propositional logic program, let us consider again the example discussed in Example 5, viz. $\mathcal{T}_4 = (\mathcal{L}^4, \mathcal{R}^4)$, where

- $\mathcal{L}^4 = \{x_1, x_2, x_3, x_4, x_5\}$;
- $\mathcal{R}^4 = \mathcal{R}^4_{str} \cup \mathcal{R}^4_{def}$, such that
- $\mathcal{R}^4_{str} = \left\{ \begin{array}{l} r_{21} :\rightarrow x_1 \\ r_{22} :\rightarrow x_2 \\ r_{23} :\rightarrow x_3 \\ r_{24} : x_4, x_5 \rightarrow \neg x_3 \end{array} \right\}$
- $\mathcal{R}^4_{def} = \left\{ \begin{array}{l} r_{25} : x_1 \Rightarrow x_4 \\ r_{26} : x_2 \Rightarrow x_5 \end{array} \right\}$

The propositional logic program corresponding to $\mathcal{T}_4$ is

$$
\begin{aligned}
\Pi_{\mathcal{T}_4} = \{ & x1; \neg x1. \\
& x2; \neg x2. \\
& x3; \neg x3. \\
& x4; \neg x4. \\
& x5; \neg x5. \\
& r21; \neg r21. \\
& r22; \neg r22. \\
& r23; \neg r23. \\
& r24; \neg r24. \\
& r25; \neg r25. \\
& r26; \neg r26. \\
& x1 \leftarrow r21. \\
& r21. \\
& x2 \leftarrow r22. \\
& r22.
\end{aligned}
$$

$$
\begin{aligned}
&x3 \leftarrow r23.\\
&r23.\\
&\neg x3 \leftarrow x4, x5, r24.\\
&x4 \leftarrow r24.\\
&x5 \leftarrow r24.\\
&r24 \leftarrow x4, x5.\\
&x4 \leftarrow x1, r25.\\
&x4 \leftarrow r25.\\
&r25 \leftarrow x1, x4.\\
&x5 \leftarrow x2, r26.\\
&x2 \leftarrow r26.\\
&r26 \leftarrow x2, x5.\}
\end{aligned}
$$

The stable models of $\Pi_{\mathcal{T}_4}$ are

$$
\begin{aligned}
S_1 &= \{x_1, x_2, x_3, x_4, \neg x_5, r_{21}, r_{22}, r_{23}, \neg r_{24}, r_{25}, \neg r_{26}\}\\
S_2 &= \{x_1, x_2, x_3, \neg x_4, x_5, r_{21}, r_{22}, r_{23}, \neg r_{24}, \neg r_{25}, r_{26}\}\\
S_3 &= \{x_1, x_2, x_3, \neg x_4, \neg x_5, r_{21}, r_{22}, r_{23}, \neg r_{24}, \neg r_{25}, \neg r_{26}\}
\end{aligned}
$$

It is easy to see that $(S_1)_{\downarrow}$, $(S_2)_{\downarrow}$, and $(S_3)_{\downarrow}$ are exactly the three WPFEs of the AF corresponding to $\mathcal{T}_4$ (cf. Section 5.1.2).