Handling Support Cycles in the Logical Encoding of Argumentation Frameworks with Higher-order Attacks and Evidential Supports: An Improvement

Marie-Christine Lagasquie-Schiex

Handling Support Cycles in the Logical Encoding of Argumentation Frameworks with Higher-order Attacks and Evidential Supports: An Improvement

M-Christine Lagasquie-Schiex

Université de Toulouse, IRIT,
118 route de Narbonne, 31062 Toulouse, France
{lagasq}@irit.fr

Tech. Report
IRIT/RR- -2021- -04- -FR

Avril 2021

Abstract

We propose an improvement of our work published in [25]. This work concerned a logical encoding of argumentation frameworks with higher-order interactions (*i.e.* attacks/supports whose targets are arguments or other attacks/supports) with an evidential meaning for supports (such frameworks are called REBAF). Then this encoding has been used for giving a characterization of REBAF semantics.

We show in the current paper that the encoding proposed in [25] has some weaknesses when support cycles exist in the REBAF. Our aim is to solve these weaknesses and so to propose a new characterisation that allows to take into account support cycles whatever is the type of these support cycles.

Contents

1 Introduction **1**

2 Background on argumentation frameworks **3**
2.1 The Standard Abstract Framework 3
2.2 A Framework with Higher-Order Evidential Supports and Attacks . . 4

3 Background on the Logical Description of a REBAF given in [25] **8**
3.1 Vocabulary 8
3.2 Logical theory for describing REBAF 9
3.3 Logical Formalization of REBAF semantics 12
3.3.1 Conflict-freeness 12
3.3.2 Self-supporting 12
3.3.3 Defence 13
3.3.4 Reinstatement 14
3.3.5 Stability 14
3.4 Characterizing Semantics of a REBAF 16
3.5 The case of support cycles 19

4 REBAF with support cycles: analysis of [25] proposition **20**
4.1 Support cycles in a REBAF: Basic definitions and examples 20
4.2 Counterexamples of Proposition 6.2 in [25] 22

5 Support cycles in a REBAF: a new proposition **23**

6 Conclusion **28**

A Proofs **32**

1 Introduction

Formal argumentation has become an essential paradigm in Artificial Intelligence, *e.g.* for reasoning from incomplete and/or contradictory information or for modelling the interactions between agents [1]. Formal abstract frameworks have greatly eased the modelling and study of argumentation. The original Dung's argumentation framework (AF) [2] consists of a collection of *arguments* interacting with each other through a relation reflecting conflicts between them, called *attack*, and enables to determine *acceptable* sets of arguments called *extensions*.

AF have been extended along different lines, *e.g.* by enriching them with positive interactions between arguments (usually expressed by a support relation), or higher-order interactions (*i.e.* interactions whose targets are other interactions).

Positive interactions between arguments. They have been first introduced in [3, 4]. In [5], the support relation is left general so that the bipolar framework keeps a high level of abstraction. The associated semantics are based on the combination of the attack relation with the support relation which results in new complex attack relations. However, there is no single interpretation of the support, and a number of researchers proposed specialized variants of the support relation (deductive support [6], necessary support [7, 8], evidential support [9, 10]). Each specialization can be associated with an appropriate modelling using an appropriate complex attack. These proposals have been developed quite independently, based on different intuitions and with different formalizations. [11] presents a comparative study in order to restate these proposals in a common setting, the bipolar argumentation framework (see also [12] for another survey).

Higher-order interactions. The idea of encompassing attacks to attacks in abstract argumentation frameworks has been first considered in [13] in the context of an extended framework handling argument strengths and their propagation. Then, higher-order attacks have been considered for representing preferences between arguments (second-order attacks in [14]), or for modelling situations where an attack might be defeated by an argument, without contesting the acceptability of the source of the attack [15]. Attacks to attacks and supports have been first considered in [16] with higher level networks, then in [17]; and more generally, [18] proposes an Attack-Support Argumentation Framework which allows for nested attacks and supports, *i.e.* attacks and supports whose targets can be other attacks or supports, at any level.

Here are examples of higher-order interactions in the legal field. The first example considers only higher-order attacks (this example is borrowed from [19]).

Example 1 *The lawyer says that the defendant did not have intention to kill the victim (argument b). The prosecutor says that the defendant threw a sharp knife towards the victim (argument a). So, there is an attack from a to b. And the intention to kill should be inferred. Then the lawyer says that the defendant was in a habit of throwing the knife at his wife's foot once drunk. This latter argument (argument c) is better considered attacking the attack from a to b, than argument a itself. Now the prosecutor's argumentation seems no longer sufficient for proving the intention to kill.* □

The second example is a variant of the first one and considers higher-order attacks and evidential supports.

Example 2 *The prosecutor says that the defendant had intention to kill the victim (argument b). A witness says that she saw the defendant throwing a sharp knife towards the victim (argument a). Argument a can be considered as a support for argument b. The lawyer argues back that the defendant was in a habit of throwing the knife at his wife's foot once drunk. This latter argument (argument c) is better considered attacking the support from a to b, than argument a or b themselves. Once again, the prosecutor's argumentation seems no longer sufficient for proving the intention to kill.* □

We follow here an evidential understanding of the support relation [9] that allows to distinguish between two different kinds of arguments: *prima-facie* and *standard arguments*. *Prima-facie* arguments were already present in [4] as those that are justified whenever they are not defeated. On the other hand, *standard arguments* are not directly assumed to be justified and must inherit support from prima-facie arguments through a chain of supports. For instance, in Example 2, arguments a and c are considered as prima-facie arguments while b is regarded as a standard argument. Hence, while a and c can be accepted as in Dung's argumentation, b must inherit support from a: this holds if c is not accepted, but does not otherwise. Indeed, in the latter, the support from a to b is defeated by c.

A natural idea that has proven useful to define semantics for these extended frameworks, known as "flattening technique", consists in turning the original extended framework into an AF, by introducing meta-arguments and a new simple (first-order) attack relation involving these meta-arguments [15, 18, 20], or by reducing higher-order attacks to first-order joint attacks [21]. More recently, alternative acceptability semantics have been defined in a direct way for argumentation frameworks with higher-order attacks [22] or for higher-order attacks and supports (necessary supports: [23], evidential supports: [24]). The idea is to specify the conditions under which the arguments (resp. the interactions) are considered as accepted directly on the extended framework, without translating the original framework into an AF.

Morever, in [25], a logical encoding of argumentation frameworks with higher-order attacks and evidential supports (REBAF) has been proposed. This encoding is able to take into account REBAF without support cycles. And a first proposition has been presented in order to also handle the case of REBAF with support cycles. Nevertheless we show in the current paper that this proposition has some weaknesses and a new proposition is given here in order to solve them.

The paper is organized as follows: the necessary background about argumentation frameworks is given in Section 2; the logical encoding for frameworks with higher-order attacks and evidential supports (REBAF) is recalled in Section 3; an analysis of the case of REBAF with support cycles is presented in Section 4 and the new proposition that can handle supports cycles is given in Section 5; Section 6 concludes the paper. The proofs are given in Appendix A.

2 Background on argumentation frameworks

Note that the text (definitions, propositions and examples) of this section is extracted from [25].

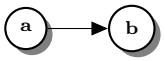
2.1 The Standard Abstract Framework

The standard case handles only one kind of interaction: attacks between arguments.

Definition 1 **[26]** *A* Dung's argumentation framework (AF) *is a tuple* $\mathbf{AF} = \langle \mathbf{A}, \mathbf{R} \rangle$, *where* $\mathbf{A}$ *is a finite and non-empty set of* arguments *and* $\mathbf{R} \subseteq \mathbf{A} \times \mathbf{A}$ *is a binary* attack relation *on the arguments, with* $(a, b) \in \mathbf{R}$ *indicates that* a *attacks* b.

A graphical representation can be used for an AF.

Example 3 *An attack* $(a, b) \in \mathbf{R}$ *is represented by two nodes* a, b *(in a circle) and a simple edge from* a *to* b*:*



□

We recall the definitions[1] of some well-known extension-based semantics. Such a semantics specifies the requirements that a set of arguments should satisfy. The basic requirements are the following ones:

- An extension can "stand together". This corresponds to the conflict-freeness principle.
- An extension can "stand on its own", namely is able to counter all the attacks it receives. This corresponds to the defence principle.
- Reinstatement is a kind of dual principle. An attacked argument which is defended by an extension is reinstated by the extension and should belong to it.
- Stability: an argument that does not belong to an extension must be attacked by this extension.

Definition 2 **[26]** *Let* $\mathbf{AF} = \langle \mathbf{A}, \mathbf{R} \rangle$ *and* $S \subseteq \mathbf{A}$.

- S *is* conflict-free *iff* $(a, b) \notin \mathbf{R}$ *for all* $a, b \in S$.
- $a \in \mathbf{A}$ *is* acceptable *w.r.t.* S *(or equivalently* S defends a*) iff for each* $b \in \mathbf{A}$ *with* $(b, a) \in \mathbf{R}$*, there is* $c \in S$ *with* $(c, b) \in \mathbf{R}$.
- *The* characteristic function $\mathcal{F}$ *of* $\mathbf{AF}$ *is defined by:* $\mathcal{F}(S) = \{a \in \mathbf{A}$ *such that* a *is acceptable w.r.t.* $S\}$.
- S *is* admissible *iff* S *is conflict-free and* $S \subseteq \mathcal{F}(S)$.

[1] Where "iff" (resp. "w.r.t.") stands for "if and only if" (resp. "with respect to").

- S *is* a complete extension *of* $\mathbf{AF}$ *iff it is conflict-free and a fixed point of* $\mathcal{F}$.
- S *is* the grounded extension *of* $\mathbf{AF}$ *iff it is the minimal (w.r.t.* $\subseteq$*) fixed point*[2] *of* $\mathcal{F}$.
- S *is* a preferred extension *of* $\mathbf{AF}$ *iff it is a maximal (w.r.t.* $\subseteq$*) complete extension.*
- S *is* a stable extension *of* $\mathbf{AF}$ *iff it is conflict-free and for each* $a \notin S$*, there is* $b \in S$ *with* $(b, a) \in \mathbf{R}$.

Note that the complete (resp. grounded, preferred, stable) semantics satisfies the conflict-freeness, defence and reinstatement principles.

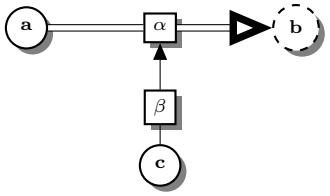
2.2 A Framework with Higher-Order Evidential Supports and Attacks

In this section, we recall the extension of [22] proposed in [24] for handling recursive attacks and evidence-based supports.

Definition 3 [24] *An* evidence-based recursive argumentation framework (REBAF) *is a sextuple* $\langle \mathbf{A}, \mathbf{R}_a, \mathbf{R}_e, \mathbf{s}, \mathbf{t}, \mathbf{P} \rangle$ *where* $\mathbf{A}$*,* $\mathbf{R}_a$ *and* $\mathbf{R}_e$ *are three (possible infinite) pairwise disjunct sets respectively representing arguments, attacks and supports names, and where* $\mathbf{P} \subseteq \mathbf{A} \cup \mathbf{R}_a \cup \mathbf{R}_e$ *is a set representing the* prima-facie *elements that do not need to be supported. Functions* $\mathbf{s} : (\mathbf{R}_a \cup \mathbf{R}_e) \longrightarrow 2^{\mathbf{A}} \setminus \varnothing$ *and* $\mathbf{t} : (\mathbf{R}_a \cup \mathbf{R}_e) \longrightarrow (\mathbf{A} \cup \mathbf{R}_a \cup \mathbf{R}_e)$ *respectively map each attack and support to its source and its target.*

Note that the source of attacks and supports is a set of arguments, the set $\mathbf{P}$ may contain several prima-facie elements (arguments, attacks and supports) and no constraint on the prima-facie elements is assumed (they can be attacked or supported).

Example 2 (cont'd): The argumentation framework corresponding to the second example given in the introduction can be represented as follows (a solid border denotes prima-facie elements while a dashed border denotes standard elements; supports are represented by double edges):



□

Semantics of REBAF are defined in [24] using the extension of the notion of structure introduced in [22]. The idea is to characterize which arguments are regarded as "acceptable", and which attacks and supports are regarded as "valid", with respect to some structure.

Consider a given framework $\mathbf{REBAF} = \langle \mathbf{A}, \mathbf{R}_a, \mathbf{R}_e, \mathbf{s}, \mathbf{t}, \mathbf{P} \rangle$.

[2] It can be proved that the minimal fixed point of $\mathcal{F}$ is conflict-free.

Definition 4 **[24]** *A triple $U = (S, \Gamma, \Delta)$ is said to be a* structure *of* REBAF *iff it satisfies: $S \subseteq \mathbf{A}$, $\Gamma \subseteq \mathbf{R}_a$ and $\Delta \subseteq \mathbf{R}_e$.*

Intuitively, the set S represents the set of "acceptable" arguments w.r.t. the structure U, while Γ and Δ respectively represent the set of "valid attacks" and "valid supports" w.r.t. U. Any attack[3] $\alpha \in \overline{\Gamma}$ is understood as "non-valid" and, in this sense, it cannot defeat the element that it is targeting. Similarly, any support $\beta \in \overline{\Delta}$ is understood as "non-valid" and it cannot support the element that it is targeting.

The following definitions are extensions of the corresponding ones defined in [22] in order to take into account the evidential supports.

Definition 5 **[24]** *Given a structure $U = (S, \Gamma, \Delta)$,*

- *The sets of defeated elements w.r.t. U are:*

$$Def_X(U) \stackrel{\text{def}}{=} \{x \in X | \exists \alpha \in \Gamma, \mathbf{s}(\alpha) \subseteq S \text{ and } \mathbf{t}(\alpha) = x\}$$

with $X \in \{\mathbf{A}, \mathbf{R}_a, \mathbf{R}_e\}$

$$Def(U) \stackrel{\text{def}}{=} Def_{\mathbf{A}}(U) \cup Def_{\mathbf{R}_a}(U) \cup Def_{\mathbf{R}_e}(U)$$

- *The set of supported elements $Sup(U)$ is recursively defined as follows:*[4]

$$Sup(U) \stackrel{\text{def}}{=} \mathbf{P} \cup \{\mathbf{t}(\alpha) | \exists \alpha \in \Delta \cap Sup(U \backslash \{\mathbf{t}(\alpha)\}), \mathbf{s}(\alpha) \subseteq (S \cap Sup(U \backslash \{\mathbf{t}(\alpha)\}))\}$$

Note that a standard element is supported if there is a "chain"[5] of supported supports leading to it, rooted in prima-facie arguments. Acceptability is more complex. Intuitively, an element is *acceptable* if it supported and in addition, every attack against it can be considered as "non-valid" because either the source or the attack itself is defeated or cannot be supported.
The elements that cannot be supported w.r.t. a structure U are called *unsupportable* w.r.t. U. An element is *supportable* w.r.t. U if there is a support for it which is non-defeated by U, with its source being non-defeated by U, and the support and its source being in turn supportable.
The elements that are defeated or unsupportable are called *unacceptable*.
Then an attack is said *unactivable* if either some argument in its source or itself is unacceptable.

Formally,

- The set of *unsupportable* elements w.r.t. U is:

$$UnSupp(U) \stackrel{\text{def}}{=} \overline{Sup(U')}$$

with $U' = (\overline{Def_{\mathbf{A}}(U)}, \mathbf{R}_a, \overline{Def_{\mathbf{R}_e}(U)})$.

[3] By $\overline{\Gamma} \stackrel{\text{def}}{=} \mathbf{R}_a \backslash \Gamma$ we denote the set complement of Γ w.r.t. $\mathbf{R}_a$. Similarly, by $\overline{\Delta} \stackrel{\text{def}}{=} \mathbf{R}_e \backslash \Delta$ we denote the set complement of Δ w.r.t. $\mathbf{R}_e$.

[4] By abuse of notation, we write $U \backslash T$ instead of $(S \backslash T, \Gamma \backslash T, \Delta \backslash T)$ with $T \subseteq (\mathbf{A} \cup \mathbf{R}_a \cup \mathbf{R}_e)$.

[5] Strictly speaking, it is not a chain, as each support may itself be the target of a support. However, we keep the word "chain" for simplicity.

- The set of *unacceptable* elements w.r.t. U is:
  $UnAcc(U) \stackrel{\text{def}}{=} Def(U) \cup UnSupp(U)$
- The set of *unactivable* attacks w.r.t. U is:
  $UnAct(U) \stackrel{\text{def}}{=} \{\alpha \in \mathbf{R}_a | \alpha \in UnAcc(U)$ or $\mathbf{s}(\alpha) \cap UnAcc(U) \neq \varnothing\}$

Definition 6 **[24]** *An element $x \in \mathbf{A} \cup \mathbf{R}_a \cup \mathbf{R}_e$ is said to be* acceptable *w.r.t. a structure U iff (i) $x \in Sup(U)$ and (ii) every attack $\alpha \in \mathbf{R}_a$ with $\mathbf{t}(\alpha) = x$ is unactivable, that is, $\alpha \in UnAct(U)$.*

$Acc(U)$ denotes the set containing all arguments, attacks and supports that are acceptable with respect to U.

The following order relations will help defining preferred structures: for any pair of structures $U = (S, \Gamma, \Delta)$ and $U' = (S', \Gamma', \Delta')$, we write $U \subseteq U'$ iff $(S \cup \Gamma \cup \Delta) \subseteq (S' \cup \Gamma' \cup \Delta')$. As usual, we say that a structure U is $\subseteq$-maximal (resp. $\subseteq$-minimal) iff every U' that satisfies $U \subseteq U'$ (resp. $U' \subseteq U$) also satisfies $U' \subseteq U$ (resp. $U \subseteq U'$).

Definition 7 **[24]** *A structure $U = (S, \Gamma, \Delta)$ is:*

1. self-supporting *iff* $(S \cup \Gamma \cup \Delta) \subseteq Sup(U)$,
2. conflict-free *iff* $X \cap Def_Y(U) = \varnothing$ *for any* $(X, Y) \in \{(S, \mathbf{A}), (\Gamma, \mathbf{R}_a), (\Delta, \mathbf{R}_e)\}$,
3. admissible *iff it is conflict-free and* $S \cup \Gamma \cup \Delta \subseteq Acc(U)$,
4. complete *iff it is conflict-free and* $Acc(U) = S \cup \Gamma \cup \Delta$,
5. grounded *iff it is a $\subseteq$-minimal complete structure,*[6]
6. preferred *iff it is a $\subseteq$-maximal admissible structure,*
7. stable[7] *iff* $(S \cup \Gamma \cup \Delta) = \overline{UnAcc(U)}$.

From the above definitions, it follows that if U is a conflict-free structure, unsupportable elements w.r.t. U are not supported w.r.t. U, that is $UnSupp(U) \subseteq \overline{Sup(U)}$.

Note that every admissible structure is also self-supporting. Moreover, the usual relations between extensions also hold for structures: every complete structure is also admissible, every preferred structure is also complete, and every stable structure is also preferred and so admissible. Other properties of REBAF are described in [24], which enable to prove for instance that there is a unique grounded structure.

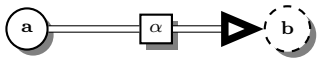
The previous definitions are illustrated on the following examples.

Example 4 *Consider two arguments a and b and a support from a to b. Following the set of prima-facie elements, different behaviours can be described.*

[6] The definition for the grounded extension is not given in [24] but can be easily proposed following the definition used in the AF case.

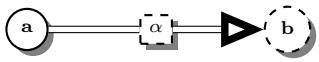
[7] Note that this is also equivalent to U is self-supporting, conflict-free and $\overline{S \cup \Gamma \cup \Delta} \subseteq UnAcc(U)$.

1. *The support and its source are assumed to be prima-facie. The target is not prima-facie.*



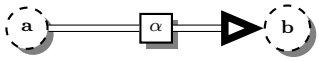
In this case, as α (resp. a) is prima-facie and not attacked, it is acceptable w.r.t. any structure. In contrast, b is not prime-facie, so b is supported w.r.t. a structure U implies that U contains the support α and its source a.
As a consequence, the structures $(\{a\}, \varnothing, \{\alpha\})$ and $(\{a, b\}, \varnothing, \{\alpha\})$ are admissible, whereas the structure $(\{b\}, \varnothing, \{\alpha\})$ is not admissible.

2. *Only the source of the support is assumed to be prima-facie.*



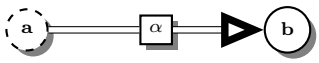
In this case, for any structure U, α is not supported w.r.t. U. It is the same for b. So the only admissible structures are $U = (\varnothing, \varnothing, \varnothing)$ and $U = (\{a\}, \varnothing, \varnothing)$.

3. *Only the support is assumed to be prima-facie.*



In this case, α is acceptable w.r.t. any structure. However, for any structure U, a is not supported w.r.t. U. So b cannot be supported. As a consequence, the only admissible structures are $U = (\varnothing, \varnothing, \varnothing)$ and $U = (\varnothing, \varnothing, \{\alpha\})$.

4. *The support and its target are assumed to be prima-facie. The source is not prima-facie.*



In this case, α (resp. b) is acceptable w.r.t. any structure. In contrast, a cannot be supported. So there are 4 admissible structures: $U = (\varnothing, \varnothing, \varnothing)$, $U = (\varnothing, \varnothing, \{\alpha\})$, $U = (\{b\}, \varnothing, \varnothing)$ and $U = (\{b\}, \varnothing, \{\alpha\})$.

□

In the next example, the support is itself the target of an attack.

Example 2 (cont'd): In this framework, neither β nor its source is attacked and β and its source are prima-facie. So, for any structure U, it holds that neither β nor its source c is unacceptable w.r.t. U. As a consequence, for any structure U, α is not acceptable w.r.t. U as α is attacked by β and β is not unactivable w.r.t. U.
As b is not prima-facie, and α is the only support to b, no admissible structure contains b. As a consequence, there is a unique complete, preferred and stable structure $U = (\{a, c\}, \{\beta\}, \varnothing)$. □

Finally, REBAF is a conservative generalization of RAF described in [22] with the addition of supports and joint attacks. Every RAF can be easily translated into a corresponding REBAF with no support and where every element (argument or attack) is prima-facie (see [24]).

3 Background on the Logical Description of a REBAF given in [25]

Note that the text (definitions, propositions and examples) of this section is extracted from [25].

Here, we recall the logical description of a REBAF proposed in [25], that allows an explicit representation of arguments, attacks, *evidential supports* and their properties. In [25] a variant of REBAF has been considered in which interactions are restricted to binary interactions (that is for any interaction α, $\mathbf{s}(\alpha)$ is a singleton) and the support relation is assumed to be acyclic. As a consequence, the definitions of $Def_X(U)$ and $Sup(U)$ given in Definition 5 can be simplified as follows:

Definition 8 (Definition 5.1 in [25]) *Given a structure $U = (S, \Gamma, \Delta)$,*

- $Def_X(U) \stackrel{\text{def}}{=} \{x \in X | \exists \alpha \in \Gamma, \mathbf{s}(\alpha) \in S$ *and* $\mathbf{t}(\alpha) = x\}$ *with* $X \in \{\mathbf{A}, \mathbf{R}_a, \mathbf{R}_e\}$.
- $Sup(U) \stackrel{\text{def}}{=} \mathbf{P} \cup \{\mathbf{t}(\alpha) | \exists \alpha \in (\Delta \cap Sup(U)), \mathbf{s}(\alpha) \in (S \cap Sup(U))\}$

Note that this new definition for $Sup(U)$ hides a difficult point of the general case: indeed, in the general case, $Sup(U)$ is defined recursively in order to avoid elements that cannot be supported without themselves. Trivially this recursion is useless when no support cycle exists in the REBAF. That justifies the new definition but that also explains why this definition cannot be used when support cycles exist.

3.1 Vocabulary

The following unary predicate symbols and unary functions symbols are used with the following meaning:

- $Arg(x)$ means "x is an argument"
- $Attack(x)$ means "x is an attack"
- $ESupport(x)$ means that "x is an evidential support"
- $T(x)$ (resp. $S(x)$) denotes the target (resp. source) of x, when x denotes an attack ou a support
- $PrimaFacie(x)$ means that "x is a prima-facie element"
- $Acc(x)$ (resp. $NAcc(x)$) means "x is accepted" (resp. "x cannot be accepted"), when x denotes an argument
- $Val(\alpha)$ means "α is valid" when α denotes an attack or a support

The binary equality predicate is also used. Note that the quantifiers $\exists$ and $\forall$ range over some domain D. To restrict them to subsets of D, bounded quantifiers will be used:

$\forall x \in E\ (P(x))$ means $\forall x\ (x \in E \rightarrow P(x))$ or equivalently $\forall x (E(x) \rightarrow P(x))$.

So we will use:

- $\forall x \in Attack\ (\Phi(x))$ (resp. $\exists x \in Attack\ (\Phi(x))$)
- $\forall x \in ESupport\ (\Phi(x))$ (resp. $\exists x \in ESupport\ (\Phi(x))$)
- and $\forall x \in Arg\ (\Phi(x))$ (resp. $\exists x \in Arg\ (\Phi(x))$).

Note that the meaning of $NAcc(x)$ is not "x is not accepted" but rather "x cannot be accepted" (for instance because x is the target of a valid attack whose source is accepted). Hence, $NAcc(x)$ is not logically equivalent to $\neg Acc(x)$. However, the logical theory will enable to deduce $\neg Acc(x)$ from $NAcc(x)$, as shown below.

Then we need symbols for denoting acceptability of elements. Let us recall that the purpose of [25] was to obtain a logical characterization of structures. As explained before, intuitively, a structure of REBAF represents the set of acceptable arguments (attacks and supports) w.r.t. the structure. And following Definition 6, acceptability w.r.t. a structure requires two conditions, one of them being a support by the structure, the other one making use of the notion of unsupportability. So the following unary predicate symbols are introduced in [25]:

- $Supp$ for denoting supported elements (argument, attack or support),
- $UnSupp$ for denoting unsupportable elements and
- $eAcc$ (resp. $eVal$) for denoting acceptability for arguments (resp. for interactions, attacks or supports).

Note that $eAcc(x)$ ("x is e-accepted") can be understood as "x is accepted and supported" and similarly $eVal(\alpha)$ ("α is e-valid") can be understood as "α is valid and supported".

3.2 Logical theory for describing REBAF

In [25], the formulae describing a given **REBAF** have been partitioned in two sets:

- The first set, denoted by Π, contains the formulae describing the general behaviour of an attack, possibly recursive, *i.e.* how an attack interacts with arguments and other attacks related to it, and also the formulae describing the general behaviour of an evidential support, possibly recursive, *i.e.* how a support interacts with arguments and other interactions related to it.
- The second set, denoted by Π(**REBAF**), contains the formulae encoding the specificities of the current framework.

The meaning of an attack is described under the form of constraints on its source (an argument) and its target (an argument or an attack). Moreover, as attacks may be attacked by other attacks, some attacks may not be valid. And finally supports must be taken into account in order to define this "validity". So we have:

- If an attack from an argument to an attack (or a support) is e-valid, then if its source is e-accepted, its target *is not* valid.

- If an attack between two arguments is e-valid and if its source is e-accepted, then its target *cannot be* accepted. In that case, the target *is not* accepted.

An evidential support can be described by the following constraints:

- If an element (argument or interaction) is prima-facie, it is supported.
- If an element is the target of an evidential support, it is supported if the source of the support is e-accepted and if the support is itself e-valid.

Using the vocabulary defined above,[8] these constraints have been expressed in [25] by the following formulae:

(1) $\forall x \in (Attack \cup ESupport)\ \forall y \in Attack$
$$\left(\begin{array}{l} (eVal(y) \wedge (t_y = x) \wedge eAcc(s_y)) \\ \rightarrow \neg Val(x) \end{array} \right)$$

(2) $\forall x \in Arg\ \forall y \in Attack$
$$\left(\begin{array}{l} (eVal(y) \wedge (t_y = x) \wedge eAcc(s_y)) \\ \rightarrow NAcc(x) \end{array} \right)$$

(3) $\forall x \in Arg\ (NAcc(x) \rightarrow \neg Acc(x))$

(1bis) $\forall x \in (Attack \cup ESupport \cup Arg)$
$$\left(\begin{array}{l} \left(\begin{array}{l} PrimaFacie(x) \vee \\ \exists y \in ESupport \\ \quad (eVal(y) \wedge (t_y = x) \wedge eAcc(s_y)) \end{array} \right) \\ \rightarrow Supp(x) \end{array} \right)$$

The following formulae define the e-acceptability (resp. e-validity). Recall that $eAcc(x)$ (resp. $eVal$) means "x is accepted (resp. valid) *and* supported":

(2bis) $\forall x \in Arg\ ((Acc(x) \wedge Supp(x)) \leftrightarrow eAcc(x))$

(3bis) $\forall x \in (Attack \cup ESupport)$
$((Val(x) \wedge Supp(x)) \leftrightarrow eVal(x))$

Other formulae limit the domain to arguments, attacks, supports.

(4) $\forall x\ (Attack(x) \rightarrow \neg Arg(x))$

(4bis) $\forall x\ (Attack(x) \rightarrow \neg ESupport(x))$

(4ter) $\forall x\ (ESupport(x) \rightarrow \neg Arg(x))$

[8] In the remainder of the paper, we will write s_α (resp. t_α) in place of $S(\alpha)$ (resp. $T(\alpha)$) for simplicity.

(5) $\forall x\ (Arg(x) \vee Attack(x) \vee ESupport(x))$

The logical theory Π consists of all the above formulae.
Then the logical encoding of specificities of a given **REBAF** leads to the set $\Pi(\textbf{REBAF})$ consisting of the following formulae. Let $\mathbf{A} = \{a_1, \ldots a_n\}$, $\mathbf{R}_a = \{\alpha_1, \ldots, \alpha_k\}$, $\mathbf{R}_e = \{\alpha_{k+1}, \ldots, \alpha_m\}$ and $\mathbf{P} = \{x_1, \ldots x_l\}$. [9]

(6) $(s_\alpha = a) \wedge (t_\alpha = b)$ for all $\alpha \in \mathbf{R}_a \cup \mathbf{R}_e$ with $\mathbf{s}(\alpha) = a$ and $\mathbf{t}(\alpha) = b$

(7) $\forall x\ (Arg(x) \leftrightarrow (x = a_1) \vee \ldots \vee (x = a_n))$

(8) $\forall x\ (Attack(x) \leftrightarrow (x = \alpha_1) \vee \ldots \vee (x = \alpha_k))$

(8bis) $\forall x\ (ESupport(x) \leftrightarrow (x = \alpha_{k+1}) \vee \ldots \vee (x = \alpha_m))$

(8ter) $\forall x\ (PrimaFacie(x) \leftrightarrow (x = x_1) \vee \ldots \vee (x = x_l))$

(9) $a_i \neq a_j$ for all $a_i, a_j \in \mathbf{A}$ with $i \neq j$

(10) $\alpha_i \neq \alpha_j$ for all $\alpha_i, \alpha_j \in \mathbf{R}_a \cup \mathbf{R}_e$ with $i \neq j$

Given **REBAF** a higher-order argumentation framework, $\Sigma(\textbf{REBAF})$ will denote the set of first-order logic formulae describing **REBAF**. And so the logical theory $\Sigma(\textbf{REBAF})$ is the union of Π and $\Pi(\textbf{REBAF})$. It is obviously consistent.

In the following examples, using the equality axioms, a simplified version of $\Sigma(\textbf{REBAF})$ is given.[10]

Example 4 (cont'd): Considering the version 1 of this example, we have:
$\Sigma(\textbf{REBAF}) = \{Supp(a)$ (from **(1bis)**, **(8ter)**),
$Supp(\alpha)$ (from **(1bis)**,**(8ter)**),
$(eAcc(a) \wedge eVal(\alpha)) \rightarrow Supp(b)$ (from **(1bis)**),
$(Supp(a) \wedge Acc(a)) \leftrightarrow eAcc(a)$ (from **(2bis)**),
$(Supp(b) \wedge Acc(b)) \leftrightarrow eAcc(b)$ (from **(2bis)**),
$(Supp(\alpha) \wedge Val(\alpha)) \leftrightarrow eVal(\alpha)$ (from **(3bis)**)$\}$

Considering the version 2 of this example, we have:
$\Sigma(\textbf{REBAF}) = \{Supp(a)$ (from **(1bis)**, **(8ter)**),
$(eAcc(a) \wedge eVal(\alpha)) \rightarrow Supp(b)$ (from **(1bis)**),
$(Supp(a) \wedge Acc(a)) \leftrightarrow eAcc(a)$ (from **(2bis)**),
$(Supp(b) \wedge Acc(b)) \leftrightarrow eAcc(b)$ (from **(2bis)**),
$(Supp(\alpha) \wedge Val(\alpha)) \leftrightarrow eVal(\alpha)$ (from **(3bis)**)$\}$

[9] We recall that $\mathbf{P} \subseteq \mathbf{A} \cup \mathbf{R}_a \cup \mathbf{R}_e$.
[10] We omit the formulae issued from **(4)** to **(10)** and the tautologies.

□

Example 2 (cont'd): Note that this example is a variant of the version 1 of Example 4 in which the attack β targeting α has been added.

$\Sigma(\mathbf{REBAF}) = \{(eVal(\beta) \wedge eAcc(c)) \rightarrow \neg Val(\alpha)$ (from **(1)**),
$Supp(a)$ (from **(1bis)**, **(8ter)**),
$Supp(c)$ (from **(1bis)**, **(8ter)**),
$Supp(\alpha)$ (from **(1bis)**,**(8ter)**),
$Supp(\beta)$ (from **(1bis)**, **(8ter)**),
$(eAcc(a) \wedge eVal(\alpha)) \rightarrow Supp(b)$ (from **(1bis)**),
$(Supp(a) \wedge Acc(a)) \leftrightarrow eAcc(a)$ (from **(2bis)**),
$(Supp(b) \wedge Acc(b)) \leftrightarrow eAcc(b)$ (from **(2bis)**),
$(Supp(c) \wedge Acc(c)) \leftrightarrow eAcc(c)$ (from **(2bis)**),
$(Supp(\alpha) \wedge Val(\alpha)) \leftrightarrow eVal(\alpha)$ (from **(3bis)**),
$(Supp(\beta) \wedge Val(\beta)) \leftrightarrow eVal(\beta)$ (from **(3bis)**)$\}$

□

3.3 Logical Formalization of REBAF semantics

In presence of higher-order attacks and supports, the conflict-freeness, defence, reinstatement and stability principles must take into account the fact that acceptability for an argument or an interaction requires that any attack against it is unactivable. Moreover acceptability requires support.

3.3.1 Conflict-freeness

In [25], the conflict-freeness principle has been formulated as follows:

- If there is an e-valid attack between two arguments, these arguments cannot be jointly e-accepted.
- If there is an e-valid attack from an e-accepted argument to an interaction (attack or support), this interaction cannot be e-valid.

Note that these properties are already expressed in $\Sigma(\mathbf{REBAF})$ (by the formulae **(1)**, **(2)**, **(3)**, **(2bis)**, **(3bis)**).

3.3.2 Self-supporting

The self-supporting principle states that each supported element must receive evidential support. In [25], it has been formulated as follows:

- If an element is supported then, either it is prima-facie, or it is the target of an e-valid support from an e-accepted source:

(17)

$$\forall x \in (Attack \cup ESupport \cup Arg)$$
$$\left(\begin{array}{l} Supp(x) \\ \rightarrow \left(\begin{array}{l} PrimaFacie(x) \vee \\ \exists y \in ESupport \\ \quad (eVal(y) \wedge (t_y = x) \wedge eAcc(s_y)) \end{array} \right) \end{array} \right)$$

- Supportability is a weaker notion, as elements that are not supportable (*i.e.* unsupportable) cannot be supported. An element is unsupportable iff it is not prima-facie and for each of its supports, either the support itself or its source is defeated, or the support or its source is in turn unsupportable:

(18)

$$\forall x \in (Attack \cup ESupport \cup Arg)$$
$$\left(\begin{array}{l} UnSupp(x) \\ \leftrightarrow \left(\begin{array}{l} \neg PrimaFacie(x) \wedge \\ \forall y \in ESupport(t_y = x \\ \quad \rightarrow \left(\begin{array}{l} \exists \beta \in Attack(t_\beta \in \{s_y, y\} \wedge \\ \quad eVal(\beta) \wedge eAcc(s_\beta))) \\ \vee UnSupp(s_y) \\ \vee UnSupp(y)) \end{array} \right) \end{array} \right) \end{array} \right)$$

Formulae **(17)** and **(18)** are added to the base $\Sigma(\mathbf{REBAF})$, thus producing the base $\Sigma_{ss}(\mathbf{REBAF})$.

3.3.3 Defence

As stated in Definition 6, an attacked element is acceptable if *(i)* it is supported and *(ii)* for each attack against it, either the source or the attack itself is defeated (by an e-valid attack from an e-accepted argument), or the source or the attack itself is unsupportable (w.r.t. e-valid elements and e-accepted arguments).
So, in [25], the principle corresponding to the previous item *(ii)* has been expressed by the following formulae that are associated with formulae **(17)** and **(18)**:

(11)

$$\forall \alpha \in Attack$$
$$\left(\begin{array}{l} Acc(t_\alpha) \\ \rightarrow \left(\begin{array}{l} \exists \beta \in Attack \\ \quad (t_\beta \in \{s_\alpha, \alpha\} \wedge eVal(\beta) \wedge eAcc(s_\beta)) \\ \vee UnSupp(s_\alpha) \\ \vee UnSupp(\alpha) \end{array} \right) \end{array} \right)$$

(12)

$$\forall \alpha \in Attack\ \forall \delta \in (Attack \cup ESupport)$$
$$\left(\begin{array}{l} ((\delta = t_\alpha) \wedge Val(\delta)) \\ \quad \rightarrow \left(\begin{array}{l} \exists \beta \in Attack \\ \quad\quad (t_\beta \in \{s_\alpha, \alpha\} \wedge eVal(\beta) \wedge eAcc(s_\beta)) \\ \vee\ UnSupp(s_\alpha) \\ \vee\ UnSupp(\alpha) \end{array} \right) \end{array} \right)$$

Formulae **(11)** and **(12)** are added to the base $\Sigma_{ss}(\mathbf{REBAF})$, thus producing the base $\Sigma_d(\mathbf{REBAF})$.

3.3.4 Reinstatement

In [25], the reinstatement principle has been expressed by the following formulae that are be associated with formulae **(17)** and **(18)**:

(13)

$$\forall c \in Arg$$
$$\left(\begin{array}{l} \left(\begin{array}{l} \forall \alpha \in Attack \\ \left(\begin{array}{l} t_\alpha = c \rightarrow \\ \left(\begin{array}{l} \exists \beta \in Attack(t_\beta \in \{s_\alpha, \alpha\} \wedge \\ \quad eVal(\beta) \wedge eAcc(s_\beta)) \\ \vee\ UnSupp(s_\alpha) \\ \vee\ UnSupp(\alpha) \end{array} \right) \end{array} \right) \end{array} \right) \\ \rightarrow Acc(c) \end{array} \right)$$

(14)

$$\forall \delta \in (Attack \cup ESupport)$$
$$\left(\begin{array}{l} \left(\begin{array}{l} (\forall \alpha \in Attack \\ \left(\begin{array}{l} t_\alpha = \delta \rightarrow \\ \left(\begin{array}{l} \exists \beta \in Attack(t_\beta \in \{s_\alpha, \alpha\} \wedge \\ \quad eVal(\beta) \wedge eAcc(s_\beta)) \\ \vee\ UnSupp(s_\alpha) \\ \vee\ UnSupp(\alpha) \end{array} \right) \end{array} \right) \end{array} \right) \\ \rightarrow Val(\delta) \end{array} \right)$$

Formulae **(13)** and **(14)** are added to the base $\Sigma_{ss}(\mathbf{REBAF})$, thus producing the base $\Sigma_r(\mathbf{REBAF})$.

3.3.5 Stability

In [25], the stability principle has been expressed by the three following formulae that are associated with formulae **(17)** and **(18)**:[11]

[11] Let us recall that a stable structure $U = (S, \Gamma, \Delta)$ satisfies: $\overline{S \cup \Gamma \cup \Delta} \subseteq UnAcc(U)$.

(15) $\forall c \in Arg$
$$\left(\begin{array}{l} \neg Acc(c) \\ \rightarrow \left(\begin{array}{l} \exists \beta \in Attack(t_\beta = c \wedge \\ \quad eVal(\beta) \wedge eAcc(s_\beta)) \end{array} \right) \end{array} \right)$$

(16) $\forall \alpha \in (Attack \cup ESupport)$
$$\left(\begin{array}{l} \neg Val(\alpha) \\ \rightarrow \left(\begin{array}{l} \exists \beta \in Attack(t_\beta = \alpha \wedge \\ \quad eVal(\beta) \wedge eAcc(s_\beta)) \end{array} \right) \end{array} \right)$$

(19) $\forall x \in (Arg \cup Attack \cup ESupport)$
$$(\neg Supp(x) \rightarrow UnSupp(x))$$

Formulae **(15)**, **(16)** and **(19)** are added to the base $\Sigma_{ss}(\textbf{REBAF})$, thus producing the base $\Sigma_s(\textbf{REBAF})$.

Example 4 (cont'd): Considering the version 1, $\Sigma_{ss}(\textbf{REBAF})$ is obtained from $\Sigma(\textbf{REBAF})$ by adding the following formulae:

$Supp(b) \rightarrow (eAcc(a) \wedge eVal(\alpha))$
$\neg UnSupp(a)$
$\neg UnSupp(\alpha)$
$Unsupp(b) \leftrightarrow (UnSupp(a) \vee UnSupp(\alpha))$

As there is no attack, $\Sigma_d(\textbf{REBAF})$ contains nothing more than $\Sigma_{ss}(\textbf{REBAF})$.

And finally $\Sigma_r(\textbf{REBAF})$ is obtained from $\Sigma_{ss}(\textbf{REBAF})$ by adding the formulae: $Acc(a)$, $Acc(b)$ and $Val(\alpha)$.

Considering the version 2, $\Sigma_{ss}(\textbf{REBAF})$ is obtained from $\Sigma(\textbf{REBAF})$ by adding the following formulae:

$Supp(b) \rightarrow (eAcc(a) \wedge eVal(\alpha))$
$\neg Supp(\alpha)$
$\neg UnSupp(a)$
$UnSupp(\alpha)$
$Unsupp(b) \leftrightarrow (UnSupp(a) \vee UnSupp(\alpha))$

Once again, $\Sigma_d(\textbf{REBAF})$ contains nothing more than $\Sigma_{ss}(\textbf{REBAF})$.

And $\Sigma_r(\textbf{REBAF})$ is obtained from $\Sigma_{ss}(\textbf{REBAF})$ by adding the formulae: $Acc(a)$, $Acc(b)$ and $Val(\alpha)$. □

Example 2 (cont'd): $\Sigma_{ss}(\textbf{REBAF})$ is obtained from $\Sigma(\textbf{REBAF})$ by adding formulae among which:

$Supp(b) \rightarrow (eAcc(a) \wedge eVal(\alpha))$
$\neg UnSupp(a)$
$\neg UnSupp(c)$
$\neg UnSupp(\alpha)$
$\neg UnSupp(\beta)$
$$Unsupp(b) \leftrightarrow \left(\begin{array}{l} (eVal(\beta) \wedge eAcc(c)) \\ \vee UnSupp(a) \\ \vee UnSupp(\alpha) \end{array} \right)$$

Then $\Sigma_d(\textbf{REBAF})$ is obtained from $\Sigma_{ss}(\textbf{REBAF})$ by adding formulae among which:

$Val(\alpha) \rightarrow (UnSupp(\beta) \vee UnSupp(c))$

$\Sigma_r(\textbf{REBAF})$ is obtained from $\Sigma_{ss}(\textbf{REBAF})$ by adding the formulae:

$Acc(a)$
$Acc(b)$
$Acc(c)$
$Val(\beta)$
$(UnSupp(c) \vee UnSupp(\beta)) \rightarrow Val(\alpha)$

$\Sigma_s(\textbf{REBAF})$ is obtained from $\Sigma_{ss}(\textbf{REBAF})$ by adding the formulae:

$Acc(a)$
$Acc(b)$
$Acc(c)$
$Val(\beta)$
$\neg Val(\alpha) \rightarrow eVal(\beta) \wedge eAcc(c)$
$\neg Supp(b) \rightarrow UnSupp(b)$ and also
$\neg Supp(x) \rightarrow UnSupp(x)$ for $x \in \{a, c, \alpha, \beta\}$ □

3.4 Characterizing Semantics of a REBAF

[25] proposed characterizations of the REBAF structures under different semantics in terms of models of the bases $\Sigma(\textbf{REBAF})$, $\Sigma_d(\textbf{REBAF})$, $\Sigma_r(\textbf{REBAF})$, $\Sigma_s(\textbf{REBAF})$. The common idea is that a structure gathers the acceptable elements w.r.t. it.

Let $\textbf{REBAF} = \langle \mathbf{A}, \mathbf{R}_a, \mathbf{R}_e, \mathbf{s}, \mathbf{t}, \mathbf{P} \rangle$. Given $\mathcal{I}$ an interpretation of $\Sigma(\textbf{REBAF})$, we define:

- $S_\mathcal{I} = \{x \in \mathbf{A} | \mathcal{I}(eAcc(x)) = \textit{true}\}$
- $\Gamma_\mathcal{I} = \{x \in \mathbf{R}_a | \mathcal{I}(eVal(x)) = \textit{true}\}$
- $\Delta_\mathcal{I} = \{x \in \mathbf{R}_e | \mathcal{I}(eVal(x)) = \textit{true}\}$

Moreover, let $\mathcal{I}$ be a model of $\Sigma(\textbf{REBAF})$:

- $\mathcal{I}$ is a $\subseteq$-maximal model of $\Sigma(\textbf{REBAF})$ iff there is no model $\mathcal{I}'$ of $\Sigma(\textbf{REBAF})$ with $(S_\mathcal{I} \cup \Gamma_\mathcal{I} \cup \Delta_\mathcal{I}) \subset (S_{\mathcal{I}'} \cup \Gamma_{\mathcal{I}'} \cup \Delta_{\mathcal{I}'})$.
- $\mathcal{I}$ is a $\subseteq$-minimal model of $\Sigma(\textbf{REBAF})$ iff there is no model $\mathcal{I}'$ of $\Sigma(\textbf{REBAF})$ with $(S_{\mathcal{I}'} \cup \Gamma_{\mathcal{I}'} \cup \Delta_{\mathcal{I}'}) \subset (S_\mathcal{I} \cup \Gamma_\mathcal{I} \cup \Delta_\mathcal{I})$.

The following characterizations are given in [25]:

Proposition 1 (Proposition 6.1 in [25]) *Let* $\textbf{REBAF} = \langle \mathbf{A}, \mathbf{R}_a, \mathbf{R}_e, \mathbf{s}, \mathbf{t}, \mathbf{P} \rangle$. *Let* $U = (S, \Gamma, \Delta)$ *be a structure on* **REBAF**.

1. *U is conflict-free iff there exists $\mathcal{I}$ model of $\Sigma(\textbf{REBAF})$ with $S_\mathcal{I} = S$, $\Gamma_\mathcal{I} = \Gamma$ and $\Delta_\mathcal{I} = \Delta$.*

2. *U is admissible iff there exists $\mathcal{I}$ model of $\Sigma_d(\mathbf{REBAF})$ with $S = S_\mathcal{I}$, $\Gamma_\mathcal{I} = \Gamma$ and $\Delta_\mathcal{I} = \Delta$.*

3. *U is complete iff there exists $\mathcal{I}$ model of $\Sigma_d(\mathbf{REBAF}) \cup \Sigma_r(\mathbf{REBAF})$ with $S = S_\mathcal{I}$, $\Gamma_\mathcal{I} = \Gamma$ and $\Delta_\mathcal{I} = \Delta$.*

4. *U is a stable structure iff there exists $\mathcal{I}$ model of $\Sigma_s(\mathbf{REBAF})$ with $S_\mathcal{I} = S$, $\Gamma_\mathcal{I} = \Gamma$ and $\Delta_\mathcal{I} = \Delta$.*

5. *U is a preferred structure iff there exists $\mathcal{I}$ $\subseteq$-maximal model of $\Sigma_d(\mathbf{REBAF})$ with $S_\mathcal{I} = S$, $\Gamma_\mathcal{I} = \Gamma$ and $\Delta_\mathcal{I} = \Delta$.*

6. *U is the grounded structure iff $S = S_\mathcal{I}$, $\Gamma_\mathcal{I} = \Gamma$ and $\Delta_\mathcal{I} = \Delta$ where $\mathcal{I}$ is a $\subseteq$-minimal model of $\Sigma_d(\mathbf{REBAF}) \cup \Sigma_r(\mathbf{REBAF})$.*[12]

The following examples illustrate the above proposition. The first one exemplifies the use of formula **(17)**. The second one exemplifies the case of an element which is attacked by a supported and unattacked attack (formulae **(12)** and **(18)**). The last two exemplify the case of an element which is attacked by an unattacked but unsupportable attack (formulae **(11)** and **(18)**).

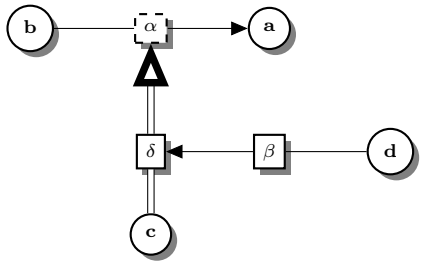
Example 4 (cont'd): Let consider the version 1. From $\Sigma_d(\mathbf{REBAF})$ it can be deduced that $eAcc(b) \rightarrow eAcc(a)$ and $eAcc(b) \rightarrow eVal(\alpha)$. That proves that each model of $\Sigma_d(\mathbf{REBAF})$ satisfying $eAcc(b)$ also satisfies $eAcc(a)$ and $eVal(\alpha)$. In other words, given $\mathcal{I}$ a model of $\Sigma_d(\mathbf{REBAF})$, if $b \in S_\mathcal{I}$ then $a \in S_\mathcal{I}$ and $\alpha \in \Delta_\mathcal{I}$. That corresponds to the fact that the structure $(\{b\}, \varnothing, \{\alpha\})$ is not admissible.
Moreover, there is a model of $\Sigma_d(\mathbf{REBAF})$ satisfying $eAcc(b)$ (and so $eAcc(a)$ and $eVal(\alpha)$). That corresponds to the fact that the structure $(\{a, b\}, \varnothing, \{\alpha\})$ is admissible.

Consider now the version 2. From $\Sigma_d(\mathbf{REBAF})$, it can be deduced that $\neg eVal(\alpha)$, $\neg Supp(b)$ and $\neg eAcc(b)$. Moreover there exists a model of $\Sigma_d(\mathbf{REBAF})$ satisfying $eAcc(a)$. That corresponds to the fact that the unique non-empty admissible structure is $(\{a\}, \varnothing, \varnothing)$.
Note that given $\mathcal{I}$ a model of $\Sigma_d(\mathbf{REBAF})$, it holds that $\mathcal{I}$ satisfies $\neg Supp(\alpha)$ and $\neg Supp(b)$. That corresponds to the fact that no admissible structure contains b (resp. α) because b (resp. α) lacks support. □

Example 2 (cont'd): From $\Sigma_d(\mathbf{REBAF})$, it can be deduced that $\neg Val(\alpha)$ then it can be deduced that $\neg eVal(\alpha)$, $\neg Supp(b)$ and $\neg eAcc(b)$. That corresponds to the fact that no admissible structure contains b (resp. α, though being supported).
Moreover there is a model of $\Sigma_d(\mathbf{REBAF})$ satisfying $eAcc(a)$, $eAcc(c)$ and $eVal(\beta)$. That corresponds to the fact that $(\{a, c\}, \varnothing, \{\beta\})$ is an admissible structure. □

Example 5 *Consider the following argumentation framework.*

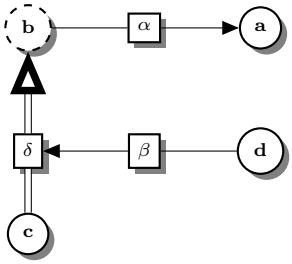
[12] It also holds that U is the grounded structure iff $U = (S_\mathcal{I}, \Gamma_\mathcal{I}, \Delta_\mathcal{I})$ where $\mathcal{I}$ is a $\subseteq$-minimal model of $\Sigma_r(\mathbf{REBAF})$.



From formula **(11)**, *it holds that the formula* $Acc(a) \rightarrow (UnSupp(\alpha) \vee UnSupp(b))$ *belongs to* Σ_d(**REBAF**). *So it can be deduced that* $Acc(a) \rightarrow UnSupp(\alpha)$ *as* b *is prima-facie. Then we can obtain the formula* $eAcc(a) \rightarrow UnSupp(\alpha)$. *Moreover, applying formula* **(18)** *yields* $UnSupp(\alpha) \leftrightarrow (eVal(\beta) \wedge eAcc(d))$ *as* δ *and* c *are prima-facie. As a consequence, we obtain* $eAcc(a) \rightarrow (eVal(\beta) \wedge eAcc(d))$.
Applying formula **(12)** *yields* $\neg Val(\delta)$, *as* β *and* d *are prima-facie, and as a consequence* $\neg eVal(\delta)$.
Finally, applying formula **(17)**, *we obtain* $eAcc(a) \rightarrow \neg Supp(\alpha)$ *as* α *is not prima-facie, and as a consequence* $eAcc(a) \rightarrow \neg eVal(\alpha)$.
That corresponds to the fact that if an admissible structure contains a, *then it also contains* d *and* β *and it does not contain* α. *Moreover no admissible structure contains* δ.

From Σ_r(**REBAF**) *it can be deduced that* $eAcc(d)$, $eVal(\beta)$ *and* $UnSupp(\alpha) \rightarrow Acc(a)$. *So, from* Σ_d(**REBAF**) $\cup$ Σ_r(**REBAF**) *it can be deduced that* $Acc(a)$ *and also* $eAcc(a)$ *as* a *is prima-facie.* Σ_r(**REBAF**) *also allows to deduce* $eAcc(b)$ *and* $eAcc(c)$.
That corresponds to the fact that the unique complete structure is $(\{a, b, c, d\}, \{\beta\}, \varnothing)$.
□

Example 6 *Consider the following argumentation framework.*



The same reasoning as the one presented for Example 5 can be used, exchanging the role of b *and* α.

So from formulae **(11)** *and* **(18)**, *we obtain the formula* $eAcc(a) \rightarrow (eVal(\beta) \wedge eAcc(d))$.
Then applying formula **(12)** *trivially yields the formula* $\neg eVal(\delta)$.
Finally, applying formula **(17)**, *we obtain* $eAcc(a) \rightarrow \neg eAcc(b)$.
That corresponds to the fact that if an admissible structure contains a, *then it also contains* d *and* β *and it does not contain* b. *Moreover no admissible structure contains* δ.

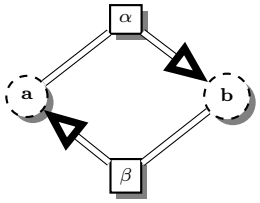
Considering $\Sigma_r(\textbf{REBAF}) \cup \Sigma_d(\textbf{REBAF})$*, we obtain* $(\{a, c, d\}, \{\alpha, \beta\}, \varnothing)$ *as the unique complete structure.* □

3.5 The case of support cycles

The logical representation proposed in [25] and recalled in the previous sections applies to a restricted variant of REBAF in which two constraints are given: first interactions are assumed to be binary and secondly *there is no cycle of supports*. This second restriction allows a direct encoding of the notions $Sup(U)$ and $UnSupp(U)$ and, in this case, it is worth to notice that formulae **(17)** and **(18)** are enough for checking that any element is supported without itself. Nevertheless, if this constraint is not satisfied (*i.e.* support cycles exist) then the use of these formulae does not prevent the acceptability of elements that cannot be supported without themselves.

However it could be interesting to see what happens in the case of a REBAF with support cycles and a first proposition has also been presented in [25] using a basic idea deduced from the following example.

Example 7 *This example corresponds to an even-length support cycle in which interactions are prima-facie and arguments are not.*



The interesting point is that, from Σ_{ss} *(and so from* Σ_d*), the following formulae can be entailed:* $Supp(a) \rightarrow eVal(\beta) \wedge eAcc(b)$ *and* $Supp(b) \rightarrow eVal(\alpha) \wedge eAcc(a)$*. So using formula* **(2bis)** *the following formulae can be entailed:* $Supp(a) \rightarrow eAcc(a)$ *and* $Supp(b) \rightarrow eAcc(b)$*. That means that* a *(resp.* b*) is supported only if it is accepted; thus these arguments cannot be supported without themselves.*

So, considering an existing model $\mathcal{I}$ *of* Σ_d *with* $S_\mathcal{I} = \{a, b\}$*,* $\Gamma_\mathcal{I} = \varnothing$ *and* $\Delta_\mathcal{I} = \{\alpha, \beta\}$*, we obtain a structure* $(S_\mathcal{I}, \Gamma_\mathcal{I}, \Delta_\mathcal{I})$ *that is not admissible since it is not self-supporting in the sense of Definition 5: there is no chain of supports leading to* a *(resp. to* b*) rooted in a prima-facie argument. Thus, in this case, Proposition 1 cannot be applied.*

Another important point is the fact that, if an element (argument or support) x cannot be supported without itself, it cannot be supportable w.r.t. a structure.

The above remarks lead to the following definition for "support-founded" interpretations given in [25]:

Definition 9 (Definition 6.1 in [25]) $\mathcal{I}$ *is a* support-founded interpretation *iff for each argument (resp. support)* x *s.t.* $\Sigma_{ss}(\textbf{REBAF})$ *entails* $Supp(x) \rightarrow eAcc(x)$ *(resp.* $Supp(x) \rightarrow eVal(x)$*), it holds that* $\mathcal{I}(eAcc(x)) =$ *false (resp.* $\mathcal{I}(eVal(x)) =$ *false) and* $\mathcal{I}(UnSupp(x)) =$ *true.*

Then a support-founded model *of* $\Sigma_d(\mathbf{REBAF})$ *is a support-founded interpretation which is a model of* $\Sigma_d(\mathbf{REBAF})$.

In [25], this definition has been used for characterizing admissible structures of a given **REBAF** with support cycles by a subclass of models of $\Sigma_d(\mathbf{REBAF})$:

Proposition 2 (Proposition 6.2 in [25]) *Let* $\mathbf{REBAF} = \langle \mathbf{A}, \mathbf{R}_a, \mathbf{R}_e, \mathbf{s}, \mathbf{t}, \mathbf{P} \rangle$. *Let* $U = (S, \Gamma, \Delta)$ *be a structure on* **REBAF**.

1. *U is admissible iff there exists $\mathcal{I}$ support-founded model of $\Sigma_d(\mathbf{REBAF})$ with $S_\mathcal{I} = S$, $\Gamma_\mathcal{I} = \Gamma$ and $\Delta_\mathcal{I} = \Delta$.*
2. *U is complete iff there exists $\mathcal{I}$ support-founded model of the union $(\Sigma_d(\mathbf{REBAF}) \cup \Sigma_r(\mathbf{REBAF}))$ with $S_\mathcal{I} = S$, $\Gamma_\mathcal{I} = \Gamma$ and $\Delta_\mathcal{I} = \Delta$.*
3. *U is a preferred structure iff there exists $\mathcal{I}$ $\subseteq$-maximal support-founded model of $\Sigma_d(\mathbf{REBAF})$ with $S_\mathcal{I} = S$, $\Gamma_\mathcal{I} = \Gamma$ and $\Delta_\mathcal{I} = \Delta$.*
4. *U is the grounded structure iff $S = S_\mathcal{I}$, $\Gamma_\mathcal{I} = \Gamma$ and $\Delta_\mathcal{I} = \Delta$ where $\mathcal{I}$ is a $\subseteq$-minimal support-founded model of $(\Sigma_d(\mathbf{REBAF}) \cup \Sigma_r(\mathbf{REBAF}))$.*

Let us illustrate the above results on the previous examples:

Example 7 (cont'd): The support-founded models of Σ_d yield all the admissible structures:

$(\varnothing, \varnothing, \varnothing)$
$(\varnothing, \varnothing, \{\alpha\})$
$(\varnothing, \varnothing, \{\beta\})$
$(\varnothing, \varnothing, \{\alpha, \beta\})$

The unique complete structure is $(\varnothing, \varnothing, \{\alpha, \beta\})$.

□

Nevertheless this characterisation has some weaknesses as it is shown in the next section.

4 REBAF with support cycles: analysis of [25] proposition

In this section, the definition of support-founded models is discussed using examples and we show that it leads to a characterisation of REBAF semantics that only holds when considering REBAF with specific cycles.

In order to do that, some formal definitions about supports cycles are first given.

4.1 Support cycles in a REBAF: Basic definitions and examples

Some notions related to directed cycles of supports must be defined before analysing the impact of such support cycles in the logical computation of structures for the REBAF.

Definition 10 *Let* REBAF $= \langle \mathbf{A}, \mathbf{R}_a, \mathbf{R}_e, \mathbf{s}, \mathbf{t}, \mathbf{P} \rangle$. *A* directed cycle of supports (DCS) *in this REBAF is a sequence* $\mathbf{C} = (x_0, \ldots, x_{n-1}, x_n)$ *such that:*[13]

- $n > 0$ *and* n *is the* size of the DCS,
- $\forall i = 0 \ldots n$, $x_i \in \mathbf{A} \cup \mathbf{R}_e$,
- $x_n = x_0$
- $\forall i = 0 \ldots n-1$, *if* $x_i \in \mathbf{A}$ *then* $x_{i+1} \in \mathbf{R}_e$ *and* $\mathbf{s}(x_{i+1}) = x_i$,
- $\forall i = 0 \ldots n-1$, *if* $x_i \in \mathbf{R}_e$ *then* $x_{i+1} = \mathbf{t}(x_i)$.

A simple DCS $\mathbf{C} = (x_0, \ldots, x_{n-1}, x_n)$ *is a DCS in which* $\forall i, j = 0 \ldots n-1$, *if* $i \neq j$ *then* $x_i \neq x_j$.

An input support of a DCS $\mathbf{C} = (x_0, \ldots, x_{n-1}, x_n)$ *is:*

- *either a support* $y \in \mathbf{R}_e$ *such that* $y \notin \mathbf{C}$ *and* $\exists x_i \in \mathbf{C}$ *and* $x_i = \mathbf{t}(y)$,
- *or an argument* $y \in \mathbf{A}$ *such that* $y \notin \mathbf{C}$ *and* $\exists x_i \in \mathbf{R}_e \cap \mathbf{C}$ *and* $y = \mathbf{s}(x_i)$.

The set of inputs of the DCS $\mathbf{C}$ *is denoted by* $\mathbf{C}^{In}$ *and it is partitioned into* $\mathbf{C}^{In}_{\mathbf{A}} = \mathbf{C}^{In} \cap \mathbf{A}$ *and* $\mathbf{C}^{In}_{\mathbf{R}_e} = \mathbf{C}^{In} \cap \mathbf{R}_e$.

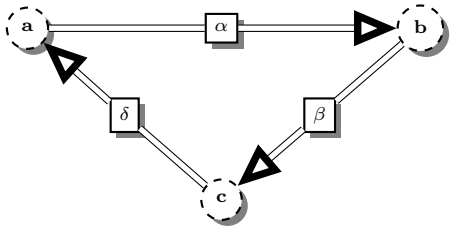
The previous definition is very general. Let us first consider examples of such a REBAF.

Example 7 (cont'd): There exists one DCS $\mathbf{C} = (a, \alpha, b, \beta, a)$ with $\mathbf{C}^{In} = \varnothing$.

Note that a DCS whose size is n can be represented by n different sequences obtained by a shift to the right or to the left. For instance, in this example, we also have:

$\mathbf{C} = (\alpha, b, \beta, a, \alpha)$ or $\mathbf{C} = (b, \beta, a, \alpha, b)$ □

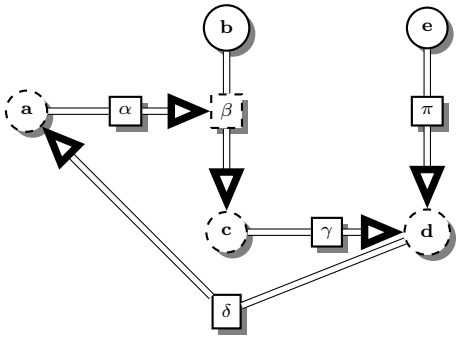
Example 8 *This example corresponds to an odd-length support cycle in which interactions are prima-facie and arguments are not. There exists one DCS* $\mathbf{C} = (a, \alpha, b, \beta, c, \delta, a)$ *with* $\mathbf{C}^{In} = \varnothing$.



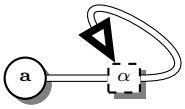
Example 9 *This example corresponds to a support loop in which the interaction is prima-facie and the argument is not. There exists one DCS* $\mathbf{C} = (a, \alpha, a)$ *with* $\mathbf{C}^{In} = \varnothing$.

[13] By abuse of language, the set of the elements composing $\mathbf{C}$ will be also denoted by $\mathbf{C}$. So $\mathbf{C}$ will be used with set operators as $\cap$ ou $\cup$ and will be comparable with other sets.

Example 10 *This example illustrates the fact that a support in a cycle can also be the target of another support in the cycle. Note that the source of the targeted support does not belong to the cycle. Here there exists one DCS* $\mathbf{C} = (a, \alpha, \beta, c, \gamma, d, \delta, a)$ *with* $\mathbf{C}^{In} = \{b, \pi\}$*. Note that the source of* π *is not considered as an input of the cycle.*



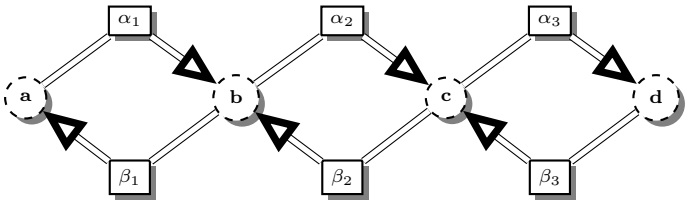
Example 11 *This example corresponds to a support that targets itself. There exists one DCS* $\mathbf{C} = (\alpha, \alpha)$ *with* $\mathbf{C}^{In} = \{a\}$.



4.2 Counterexamples of Proposition 6.2 in [25]

In this section, some counterexamples of Proposition 6.2 in [25] (numbered 2 in the current paper) are exhibed. Indeed, if several cycles exist and can be agglomerated into a non-simple directed cycle, Definition 9 is not enough for guaranting that any model in which an element that cannot be supported without itself is removed. Consider for instance the following example:

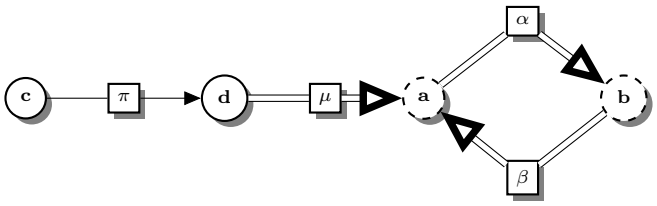
Example 12 *In this example, 3 simple DCS exist.*



Here Σ_{ss} does not entail $Supp(x) \rightarrow eAcc(x)$ for $x = a$, or $x = b$, or $x = c$, or $x = d$. So Definition 9 cannot be used in order to remove models in which a (resp. b, c, d) is supported by itself. The origin of this problem is the fact that, following Formula **(17)***, the existence of several supporters for b and c prevents the entailment described in Definition 9: $\Sigma_{ss} \models Supp(c) \rightarrow (eAcc(c) \vee eAcc(b))$ and not $\Sigma_{ss} \models Supp(c) \rightarrow eAcc(c)$.*

Another case showing the weakness of Definition 9 is the following example:

Example 13 *This example corresponds to an extension of Example 7: an attacked argument that supports an even-length support cycle $\mathbf{C} = (a, \alpha, b, \beta, a)$.*



Note that $\mathbf{C}^{In} = \{\mu\}$.

In this example, there exists a model $\mathcal{I}$ of Σ_d with $S_{\mathcal{I}} = \{a, b, c\}$, $\Gamma_{\mathcal{I}} = \{\pi\}$ and $\Delta_{\mathcal{I}} = \{\alpha, \beta, \mu\}$. The structure $(S_{\mathcal{I}}, \Gamma_{\mathcal{I}}, \Delta_{\mathcal{I}})$ is not admissible since it is not self-supporting in the sense of Definition 5: there is no chain of supported supports leading to a (resp. to b) rooted in a prima-facie argument that belongs to the structure (since d is attacked and not defended).

In fact, it seems that Proposition 6.2 in [25] can be applied only if the support cycles in the REBAF are simple DCS without input in the REBAF.

5 Support cycles in a REBAF: a new proposition

Since Definition 9 is not sufficient in the general case, and in order to identify the elements that cannot be supported without themselves, the main point must be to find the elements of the REBAF that would be able to play a role for supporting the elements of a cycle. That leads us to define the *impacting support chains* for each element of a REBAF. Unformally an impacting support chain for an element x is a sequence targeting x, originated in a prima-facie argument and composed alternatively by "an argument, a support, an argument, a support, ...". Moreover no repetition is authorized (so any element appears only one time in the sequence); and x cannot belong to the sequence. So Formally we have:

Definition 11 *Let $\mathbf{REBAF} = \langle \mathbf{A}, \mathbf{R}_a, \mathbf{R}_e, \mathbf{s}, \mathbf{t}, \mathbf{P} \rangle$. Let x be an element of this REBAF. An* impacting support chain *for x is a sequence $\mathbf{ISC} = (x_0, \ldots, x_n)$ with $n > 0$ and:*

- $\forall x_i, i \in [0 \ldots n]$, $x_i \in (\mathbf{A} \cup \mathbf{R}_e) \setminus \{x\}$
- $x_0 \in \mathbf{A} \cap \mathbf{P}$ *and* $x_n \in \mathbf{R}_e$ *such that* $\mathbf{t}(x_n) = x$

- $\forall i, j \in [0 \dots n]$, *if* $i \neq j$, *then* $x_i \neq x_j$
- $\forall i \in [1 \dots n]$, *if* $x_i \in \mathbf{R}_e$ *then* $x_{i-1} = \mathbf{s}(x_i)$
- $\forall i \in [2 \dots n-1]$, *if* $x_i \in \mathbf{A}$ *then* $x_i = \mathbf{t}(x_{i-1})$

It is obvious to see that if a DCS has some inputs then these inputs may belong to some impacting support chains of the elements of the DCS, if they are prima-facie or if they have at least one impacting support chain.

Another trivial property is the fact that, in a DCS without input and in which none argument is prima-facie, the elements of the DCS have no impacting support chains.

Example 12 (cont'd): Here, for any element, there is no impacting support chain. □

Example 13 (cont'd): Considering the arguments in the cycle, we have:

- for argument a, there is one impacting support chain: (d, μ),
- for argument b, there is one impacting support chain: (d, μ, a, α).

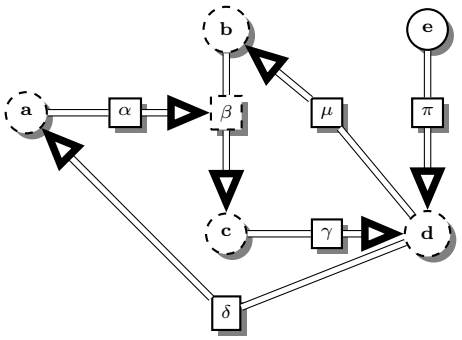
□

Example 10 (cont'd): Considering the impacting support chains of some elements of the DCS, we have for instance:

- For argument d, there exist two impacting support chains: (e, π) and (b, β, c, γ).
- For argument c, there exists only one impacting support chain: (b, β).
- For argument a, there exist two impacting support chains: (e, π, d, δ) and $(b, \beta, c, \gamma, d, \delta)$.
- For support β, there exists only one impacting support chain: $(e, \pi, d, \delta, a, \alpha)$.

□

Example 14 *This example extends Example 10 by adding a second cycle including the source of the support targeted in the first DCS.*



$\mathbf{C} = (a, \alpha, \beta, c, \gamma, d, \delta, a)$ *with* $\mathbf{C}^{In} = \{b, \pi\}$ *and* $\mathbf{C}' = (b, \beta, c, \gamma, d, \mu, b)$ *with* $\mathbf{C}'^{In} = \{\alpha, \pi\}$ *are the two simple DCS.*

Considering the impacting support chains of some elements of the DCS, we have for instance:

- *For argument* d*, there exists only one impacting support chain:* (e, π).
- *For argument* c*, there exists only one impacting support chain:* $(e, \pi, d, \mu, b, \beta)$.
- *For argument* a*, there exists only one impacting support chain:* (e, π, d, δ).
- *For support* β*, there exists only one impacting support chain:* $(e, \pi, d, \delta, a, \alpha)$.

In Example 14, it is worth to note that among the inputs of $\mathbf{C}$ we can find elements of $\mathbf{C}'$ (and vice-versa). So any element of $\mathbf{C}'$ could impact the supported status of the elements of $\mathbf{C}$ (and vice-versa). In this case, we must consider both cycles before deciding if an element can or cannot be supported without itself. This is why we must study the case where several simple directed cycles exist and can be aggregated (as in Example 12 or in Example 14). This notion of aggregation is formally defined as follows:

Definition 12 *Let* $\text{REBAF} = \langle \mathbf{A}, \mathbf{R}_a, \mathbf{R}_e, \mathbf{s}, \mathbf{t}, \mathbf{P} \rangle$. *Let* $\mathbf{C} = (x_0, \ldots, x_{n-1}, x_n)$ *and* $\mathbf{C}' = (x'_0, \ldots, x'_{m-1}, x'_m)$ *be two DCS of this REBAF such that there exist* $x_i \in \mathbf{C}$ *and* $x'_j \in \mathbf{C}'$ *and* $x_i = x'_j$.

The aggregation *of* $\mathbf{C}$ *and* $\mathbf{C}'$ *is the directed cycle corresponding to the union of the sets* $\{x_0, \ldots, x_{n-1}\}$ *and* $\{x'_0, \ldots, x'_{m-1}\}$. *This aggregation with be denoted by abuse of language* $\mathbf{C} \cup \mathbf{C}'$.

Using this notion of aggregation, a maximal DCS of a REBAF can be defined:

Definition 13 *Let* $\text{REBAF} = \langle \mathbf{A}, \mathbf{R}_a, \mathbf{R}_e, \mathbf{s}, \mathbf{t}, \mathbf{P} \rangle$. *Let* $\mathbf{C} = (x_0, \ldots, x_{n-1}, x_n)$ *be a DCS.* $\mathbf{C}$ *is a* maximal DCS *iff there does not exist another DCS that could be aggregated with* $\mathbf{C}$.

Example 12 (cont'd): In this example, 3 simple DCS exist: $\mathbf{C}_1 = (a, \alpha_1, b, \beta_1, a)$, $\mathbf{C}_2 = (b, \alpha_2, c, \beta_2, b)$ and $\mathbf{C}_3 = (c, \alpha_3, d, \beta_3, c)$ that can be aggregated together giving the only one maximal DCS:

$$\mathbf{C} = \mathbf{C}_1 \cup \mathbf{C}_2 \cup \mathbf{C}_3 = (a, \alpha_1, b, \alpha_2, c, \alpha_3, d, \beta_3, c, \beta_2, b, \beta_1, a)$$

□

Example 14 (cont'd): In this example, the two simple DCS can be aggregated since they share several elements (β, c, γ, d). And this aggregation is the only maximal DCS of this REBAF:

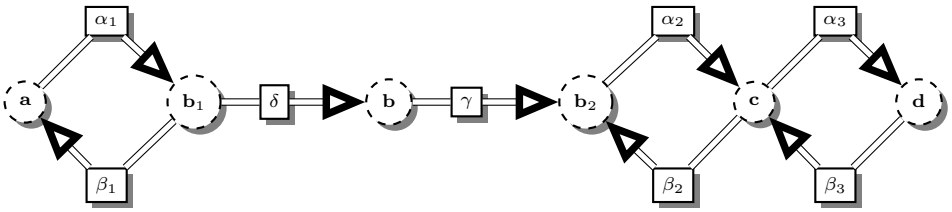
$$\mathbf{C}'' = (a, \alpha, \beta, c, \gamma, d, \mu, b, \beta, c, \gamma, d, \delta, a)$$

□

Example 15 *In this example, 3 simple DCS exist:*

$\mathbf{C}_1 = (a, \alpha_1, b_1, \beta_1, a)$, $\mathbf{C}_2 = (b_2, \alpha_2, c, \beta_2, b_2)$ *and* $\mathbf{C}_3 = (c, \alpha_3, d, \beta_3, c)$

But only two of them can be aggregated: $\mathbf{C}_2$ *and* $\mathbf{C}_3$.

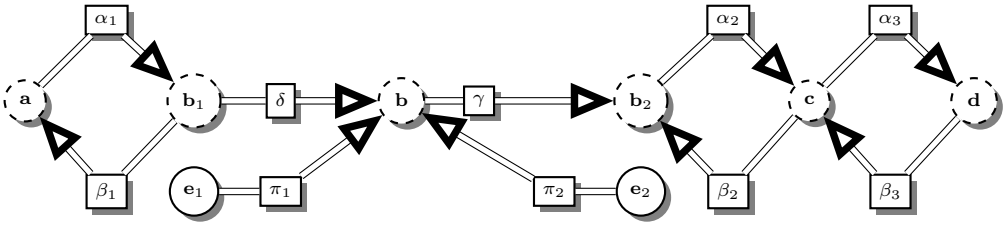


So two maximal DCS exist:

$$\mathbf{C} = \mathbf{C}_1 = (a, \alpha_1, b_1, \beta_1, a)$$
$$\mathbf{C}' = \mathbf{C}_2 \cup \mathbf{C}_3 = (b_2, \alpha_2, c, \alpha_3, d, \beta_3, c, \beta_2, b_2)$$

Note that, for any element of this REBAF, there is no impacting support chains.

Example 16 *This example is an extension of Example 15 with the same DCS but with some additionnal prima-facie arguments and supports.*



Considering the impacting supports chains, we have:

- *no impacting supports chain for the interactions (but they are all prima-facie);*
- *no impacting supports chain for e_1 and e_2 (but they are prima-facie);*
- *no impacting supports chain for a and b_1 since they cannot be supported without themselves;*
- *for the other arguments, there are two impacting supports chains, one containing e_1 and π_1 and the other containing e_2 and π_2; for instance for argument d we have: $(e_1, \pi_1, b, \gamma, b_2, \alpha_2, c, \alpha_3)$ and $(e_2, \pi_2, b, \gamma, b_2, \alpha_2, c, \alpha_3)$.*

Study now the use of these impacting support chains in order to identify the models of Σ_d we want to remove. The idea is very simple: we keep a model only if, for each element of a DCS, we can identify *in this model* a chain of supported supports leading to this argument; by definition, that means that the model we keep satisfies all the elements composing at least one impacting support chain for this element.

Moreover we must also take into account the fact that the existence of support cycles has an impact on the $UnSupp$ predicate. Consider for instance Example 7.

Example 7 (cont'd): Formula **(18)** gives:

$UnSupp(a) \rightarrow (UnSupp(\beta) \vee UnSupp(b))$

$UnSupp(b) \rightarrow (UnSupp(\alpha) \vee UnSupp(a))$
$\neg UnSupp(\alpha)$
$\neg UnSupp(\beta)$

So 2 models of Σ_{ss} exist, $\mathcal{I}_1$ and $\mathcal{I}_2$, with $\mathcal{I}_1(UnSupp(a)) = \mathcal{I}_1(UnSupp(b)) =$ *true* and $\mathcal{I}_2(UnSupp(a)) = \mathcal{I}_2(UnSupp(b)) =$ *false*.

Nevertheless, considering this REBAF, for any possible structure U, $UnSupp(U) = \{a, b\}$. So the model $\mathcal{I}_2$ does not reflect the reality concerning the "unsupportable" status of a and b and should be removed. □

The previous ideas lead to the following improvement of the notion of *support-founded interpretation* given in [25]:

Definition 14 *Let* **REBAF** $= \langle \mathbf{A}, \mathbf{R}_a, \mathbf{R}_e, \mathbf{s}, \mathbf{t}, \mathbf{P} \rangle$. $\mathcal{I}$ *is a* support-founded interpretation *iff the two following conditions hold:*

1. *for each argument (resp. support) x non prima-facie, belonging to a maximal DCS and such that $\mathcal{I}(eAcc(x)) =$ true (resp. $\mathcal{I}(eVal(x)) =$ true), there exists at least one impacting support chain* $\mathbf{ISC} = (x_0, \ldots, x_n)$ *for x that is satisfied by $\mathcal{I}$, i.e. $\forall x_i \in \mathbf{ISC}$, if $x_i \in \mathbf{A}$ then $\mathcal{I}(eAcc(x_i)) =$ true, otherwise $\mathcal{I}(eVal(x_i)) =$ true;*

2. *for each element x of* **REBAF**, $\mathcal{I}(UnSupp(x)) =$ *true iff $x \in UnSupp(U_{\mathcal{I}})$ with $U_{\mathcal{I}} = (S_{\mathcal{I}}, \Gamma_{\mathcal{I}}, \Delta_{\mathcal{I}})$.*

Let Σ_x be a base of formulae built over **REBAF**. *A* support-founded model *of Σ_x is a support-founded interpretation which is a model of Σ_x.*

Example 12 (cont'd): In this example there is no impacting support chain for argument a. So any model $\mathcal{I}$ of Σ_d that satisfies a (*i.e.* such that $\mathcal{I}(eAcc(a)) =$ *true*) is not support-founded. And we have the same result for arguments b, c and d. □

Example 13 (cont'd): Let consider argument a. This argument has only one impacting support chain (d, μ). So any model $\mathcal{I}$ of Σ_d that satisfies a will be support-founded iff it also satisfies d and μ (*i.e* we must have $\mathcal{I}(eAcc(a) = \mathcal{I}(eAcc(d) = \mathcal{I}(eVal(\mu) =$ *true*). □

Using this new definition, we can obtain the following characterization of admissible structures of a given **REBAF** with support cycles by a subclass of models of Σ_d(**REBAF**):

Proposition 3 *Let* **REBAF** $= \langle \mathbf{A}, \mathbf{R}_a, \mathbf{R}_e, \mathbf{s}, \mathbf{t}, \mathbf{P} \rangle$. *Let $U = (S, \Gamma, \Delta)$ be a structure on* **REBAF**.

1. *U is admissible iff there exists $\mathcal{I}$ support-founded model of Σ_d(***REBAF***) (in the sense of Definition 14) with $S_{\mathcal{I}} = S$, $\Gamma_{\mathcal{I}} = \Gamma$ and $\Delta_{\mathcal{I}} = \Delta$.*

2. *U is complete iff there exists $\mathcal{I}$ support-founded model of the union (Σ_d(***REBAF***) $\cup$ Σ_r(***REBAF***)) (in the sense of Definition 14) with $S_{\mathcal{I}} = S$, $\Gamma_{\mathcal{I}} = \Gamma$ and $\Delta_{\mathcal{I}} = \Delta$.*

3. *U is a preferred structure iff there exists $\mathcal{I}$ $\subseteq$-maximal support-founded model of $\Sigma_d(\textbf{REBAF})$ (in the sense of Definition 14) with $S_\mathcal{I} = S$, $\Gamma_\mathcal{I} = \Gamma$ and $\Delta_\mathcal{I} = \Delta$.*

4. *U is the grounded structure iff $S = S_\mathcal{I}$, $\Gamma_\mathcal{I} = \Gamma$ and $\Delta_\mathcal{I} = \Delta$ where $\mathcal{I}$ is a $\subseteq$-minimal support-founded model of $(\Sigma_d(\textbf{REBAF}) \cup \Sigma_r(\textbf{REBAF}))$ (in the sense of Definition 14).*

5. *U is stable iff there exists $\mathcal{I}$ support-founded model of $\Sigma_s(\textbf{REBAF})$ (in the sense of Definition 14) with $S_\mathcal{I} = S$, $\Gamma_\mathcal{I} = \Gamma$ and $\Delta_\mathcal{I} = \Delta$.*

Let us illustrate the above results on the previous examples:

Example 7 (cont'd): Apply Proposition 3 leads to the unique complete, preferred, stable and grounded structure $(\varnothing, \varnothing, \{\alpha, \beta\})$. □

Example 8 (cont'd): Apply Proposition 3 leads to the unique complete, preferred, stable and grounded structure $(\varnothing, \varnothing, \{\alpha, \beta, \delta\})$. □

Example 9 (cont'd): Apply Proposition 3 leads to the unique complete, preferred, stable and grounded structure $(\varnothing, \varnothing, \{\alpha\})$. □

Example 10 (cont'd): Apply Proposition 3 leads to the unique complete, preferred, stable and grounded structure $(\{a, b, c, d, e\}, \varnothing, \{\alpha, \beta, \delta, \gamma, \pi\})$. □

Example 11 (cont'd): Apply Proposition 3 leads to the unique complete, preferred, stable and grounded structure $(\{a\}, \varnothing, \varnothing)$. □

Example 12 (cont'd): Apply Proposition 3 leads to the unique complete, preferred, stable and grounded structure $(\varnothing, \varnothing, \{\alpha_1, \alpha_2, \alpha_3, \beta_1, \beta_2, \beta_3\})$. □

Example 13 (cont'd): Apply Proposition 3 leads to the unique complete, preferred, stable and grounded structure $(\{c\}, \{\pi\}, \{\alpha, \beta, \mu\})$. □

Example 14 (cont'd): Apply Proposition 3 leads to the unique complete, preferred, stable and grounded structure $(\{a, b, c, d, e\}, \varnothing, \{\alpha, \beta, \gamma, \delta, \pi, \mu\})$. □

Example 15 (cont'd): Apply Proposition 3 leads to the unique complete, preferred, stable and grounded structure $(\varnothing, \varnothing, \{\alpha_1, \alpha_2, \alpha_3, \beta_1, \beta_2, \beta_3, \delta, \gamma\})$. □

Example 16 (cont'd): Apply Proposition 3 leads to the unique complete, preferred, stable and grounded structure $(\{b, b_2, c, d, e_1, e_2\}, \varnothing, \{\alpha_1, \alpha_2, \alpha_3, \beta_1, \beta_2, \beta_3, \delta, \gamma, \pi_1, \pi_2\})$. □

6 Conclusion

In this work, we have solved a specific issue concerning the handling of support cycles in the logical translation of argumentation frameworks with higher-order attacks and evidential supports (REBAF). A first proposition has been presented in [25]. However, we have proven in this paper that this proposition does not hold in the general case.

So a new definition for the notion of support-founded models is proposed in this paper, for providing characterizations of admissible (resp. complete, preferred, stable and grounded) structures in the presence of all kinds of support cycles.

References

[1] I. Rahwan and G. Simari, *Argumentation in Artificial Intelligence*. Springer, 2009.

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

[3] N. Karacapilidis and D. Papadias, "Computer supported argumentation and collaborative decision making: the HERMES system," *Information systems*, vol. 26, no. 4, pp. 259–277, 2001.

[4] B. Verheij, "Deflog: on the logical interpretation of prima facie justified assumptions," *Journal of Logic in Computation*, vol. 13, pp. 319–346, 2003.

[5] C. Cayrol and M.-C. Lagasquie-Schiex, "Gradual valuation for bipolar argumentation frameworks," in *Proc. of ECSQARU*. Springer, 2005, pp. 366–377.

[6] G. Boella, D. M. Gabbay, L. van der Torre, and S. Villata, "Support in abstract argumentation," in *Proc. of COMMA*. IOS Press, 2010, pp. 111–122.

[7] F. Nouioua and V. Risch, "Bipolar argumentation frameworks with specialized supports," in *Proc. of ICTAI*. IEEE Computer Society, 2010, pp. 215–218.

[8] ——, "Argumentation frameworks with necessities," in *Proc. of SUM*. Springer-Verlag, 2011, pp. 163–176.

[9] N. Oren and T. J. Norman, "Semantics for evidence-based argumentation," in *Proc. of COMMA*. IOS Press, 2008, pp. 276–284.

[10] N. Oren, C. Reed, and M. Luck, "Moving between argumentation frameworks," in *Proc. of COMMA*. IOS Press, 2010, pp. 379–390.

[11] C. Cayrol and M.-C. Lagasquie-Schiex, "Bipolarity in argumentation graphs: towards a better understanding," *Intl. J. of Approximate Reasoning*, vol. 54, no. 7, pp. 876–899, 2013.

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

[13] H. Barringer, D. Gabbay, and J. Woods, "Temporal dynamics of support and attack networks : From argumentation to zoology," in *Mechanizing Mathematical Reasoning. LNAI 2605*, D. Hutter and W. Stephan, Eds. Springer Verlag, 2005, pp. 59–98.

[14] S. Modgil, "Reasoning about preferences in argumentation frameworks," *Artificial Intelligence*, vol. 173, pp. 901–934, 2009.

[15] P. Baroni, F. Cerutti, M. Giacomin, and G. Guida, "AFRA: Argumentation framework with recursive attacks," *Intl. Journal of Approximate Reasoning*, vol. 52, pp. 19–37, 2011.

[16] D. M. Gabbay, "Fibring argumentation frames," *Studia Logica*, vol. 93, pp. 231–295, 2009.

[17] S. Villata, G. Boella, D. M. Gabbay, and L. van der Torre, "Modelling defeasible and prioritized support in bipolar argumentation," *AMAI*, vol. 66, no. 1-4, pp. 163–197, 2012.

[18] A. Cohen, S. Gottifredi, A. J. García, and G. R. Simari, "An approach to abstract argumentation with recursive attack and support," *J. Applied Logic*, vol. 13, no. 4, pp. 509–533, 2015.

[19] R. Arisaka and K. Satoh, "Voluntary Manslaughter? A Case Study with Meta-Argumentation with Supports," in *Proc. of JSAI-isAI 2016. LNCS, vol 10247*. Springer, 2017, pp. 241–252.

[20] C. Cayrol, A. Cohen, and M.-C. Lagasquie-Schiex, "Towards a new framework for recursive interactions in abstract bipolar argumentation," in *Proc. of COMMA*. IOS Press, 2016, pp. 191–198.

[21] D. M. Gabbay, "Semantics for higher level attacks in extended argumentation frames," *Studia Logica*, vol. 93, pp. 357–381, 2009.

[22] C. Cayrol, J. Fandinno, L. Fariñas del Cerro, and M.-C. Lagasquie-Schiex, "Valid attacks in argumentation frameworks with recursive attacks," in *Proc. of Commonsense Reasoning*, vol. 2052. CEUR Workshop Proceedings, 2017.

[23] A. Cohen, S. Gottifredi, A. J. García, and G. R. Simari, "On the acceptability semantics of argumentation frameworks with recursive attack and support," in *Proc. of COMMA*. IOS Press, 2016, pp. 231–242.

[24] C. Cayrol, J. Fandinno, L. Fariñas del Cerro, and M.-C. Lagasquie-Schiex, "Argumentation frameworks with recursive attacks and evidence-based support," in *Proc. of FoIKS*, vol. LNCS 10833. Springer-Verlag, 2018, pp. 150–169.

[25] C. Cayrol and M.-C. Lagasquie-Schiex, "Logical encoding of argumentation frameworks with higher-order attacks and evidential supports," *International Journal on Artificial Intelligence Tools*, vol. 29, no. 3-4, pp. 2 060 003:1–2 060 003:50, June 2020.

[26] P. M. Dung, "On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games," *Artificial Intelligence*, vol. 77, pp. 321–357, 1995.

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

[28] D. M. Gabbay and M. Gabbay, "The attack as strong negation, part I," *Logic Journal of the IGPL*, vol. 23, no. 6, pp. 881–941, 2015.

[29] P. Besnard, S. Doutre, and A. Herzig, "Encoding argument graphs in logic," in *Proc of IPMU*. Springer, 2014, pp. 345–354.

[30] P. Besnard, S. Doutre, V. H. Ho, and D. Longin, "SESAME - A System for Specifying Semantics in Abstract Argumentation," in *Proc. of SAFA*, vol. 1672. CEUR Workshop Proceedings, 2016, pp. 40–51.

[31] D. Grossi, "On the logic of argumentation theory," in *Proc. of AAMAS*. IFAAMAS, 2010, pp. 409–416.

[32] O. Arieli and M. Caminada, "A QBF-based formalization of abstract argumentation semantics," *Journal of Applied Logic*, vol. 11, no. 2, pp. 229 – 252, 2013.

[33] J. P. Wallner, G. Weissenbacher, and S. Woltran, "Advanced SAT techniques for abstract argumentation," in *Proc. of CLIMA*. Springer, 2013, pp. 138–154.

[34] F. Cerutti, P. E. Dunne, M. Giacomin, and M. Vallati, "Computing preferred extensions in abstract argumentation: A SAT-based approach," in *Proc. of TAFA, Revised Selected papers*. Springer, 2014, pp. 176–193.

[35] B. Bogaerts, T. Janhunen, and S. Tasharrofi, "Declarative solver development: Case studies," in *Proc. of KR*. AAAI Press, 2016, pp. 74–83.

[36] G. Brewka and S. Woltran, "Abstract dialectical frameworks," in *Proc. of KR*. AAAI Press, 2010, pp. 102–111.

[37] F. Dupin de Saint-Cyr, P. Bisquert, C. Cayrol, and M.-C. Lagasquie-Schiex, "Argumentation update in YALLA (Yet Another Logic Language for Argumentation)," *Intl. J. of Approximate Reasoning*, vol. 75, pp. 57 – 92, 2016.

[38] C. Beierle, F. Brons, and N. Potyka, "A software system using a SAT solver for reasoning under complete, stable, preferred, and grounded argumentation semantics," in *Proc. of KI*. Springer, 2015, pp. 241–248.

[39] J. M. Lagniez, E. Lonca, and J. G. Mailly, "Coquiaas: A constraint-based quick abstract argumentation solver," in *Proc. of ICTAI*. IEEE Computer Society, 2015, pp. 928–935.

[40] C. Cayrol and M.-C. Lagasquie-Schiex, "The Grafix website," http://www.irit.fr/grafix.

A Proofs

We recalled here a notation and some lemmas given in [25] that will be useful for our proof.

Notation 1 (Notation Appendix A.1 in [25]) *Let $U = (S, \Gamma, \Delta)$ be a structure of* REBAF*, and $x \in \mathbf{A} \cup \mathbf{R}_a \cup \mathbf{R}_e$. x will be said to be* defended *by U, iff every attack $\alpha \in \mathbf{R}_a$ with $\mathbf{t}(\alpha) = x$ is unactivable w.r.t. U. $Defended(U)$ will denote the set of elements that are defended by U.*
Note that $x \in Acc(U)$ iff $x \in Sup(U)$ and $x \in Defended(U)$.

Lemma 1 (Lemma 7 in [24] and Lemma Appendix A.1 in [25]) *Any conflict-free self-supporting structure U satisfies:*
$Acc(U) \subseteq \overline{UnAcc(U)} \subseteq \overline{Def(U)}$.

Lemma 2 (Lemma Appendix A.2 in [25]) *Any stable structure U satisfies: $\overline{Sup(U)} = UnSupp(U)$.*

Lemma 3 (Lemma Appendix A.3, in [25]) *Let $U = (S, \Gamma, \Delta)$ be a structure and $x \notin \mathbf{P}$ be the target of a support y such that $y \in \Delta \cap Sup(U)$ and $s_y \in S \cap Sup(U)$. Then, there exists a support z such that $t_z = x$, $z \in \Delta \cap Sup(U \setminus \{x\})$ and $s_z \in S \cap Sup(U \setminus \{x\})$ and so $x \in Sup(U)$.*

Proof of Proposition 3.[14]

Let REBAF $= \langle \mathbf{A}, \mathbf{R}_a, \mathbf{R}_e, \mathbf{s}, \mathbf{t}, \mathbf{P} \rangle$.

1. (admissibility)

   $\Rightarrow$ Assume that the structure $U = (S, \Gamma, \Delta)$ is admissible. Let us define an interpretation $\mathcal{I}$ of Σ_d(REBAF). The idea is to define $\mathcal{I}$ by successively adding constraints that $\mathcal{I}$ should satisfy:
   - For all $x \in \mathbf{A} \cup \mathbf{R}_a \cup \mathbf{R}_e$,
     $\mathcal{I}(Arg(x)) =$ *true* iff $x \in \mathbf{A}$,
     $\mathcal{I}(Attack(x)) =$ *true* iff $x \in \mathbf{R}_a$ and
     $\mathcal{I}(ESupport(x)) =$ *true* iff $x \in \mathbf{R}_e$.
   - For all $x \in \mathbf{A} \cup \mathbf{R}_a \cup \mathbf{R}_e$, $\mathcal{I}(PrimaFacie(x)) =$ *true* iff $x \in \mathbf{P}$.
   - For all $x \in \mathbf{A} \cup \mathbf{R}_a \cup \mathbf{R}_e$, $\mathcal{I}(Supp(x)) =$ *true* iff $x \in Sup(U)$.
   - For all $x \in \mathbf{A} \cup \mathbf{R}_a \cup \mathbf{R}_e$, $\mathcal{I}(UnSupp(x)) =$ *true* iff $x \in UnSupp(U)$.
   - For all $x \in \mathbf{A}$, $\mathcal{I}(Acc(x)) =$ *true* iff $x \in S$ or ($x \notin S$, $x \notin Sup(U)$ and $x \in Defended(U)$).
   - For all $x \in \mathbf{A}$, $\mathcal{I}(NAcc(x)) =$ *true* iff $\mathcal{I}(Acc(x)) =$ *false*.
   - For all $x \in \mathbf{R}_a$ (resp. $\in \mathbf{R}_e$), $\mathcal{I}(Val(x)) =$ *true* iff $x \in \Gamma$ (resp. Δ) or ($x \notin \Gamma$ (resp. Δ), $x \notin Sup(U)$ and $x \in Defended(U)$).

[14] This proof is inspired by the proof of Proposition 6.1 in [25] (numbered Proposition 1 in the current paper).

- For all $x \in \mathbf{A}$, $\mathcal{I}(eAcc(x)) =$ *true* iff
  ($\mathcal{I}(Acc(x)) =$ *true* and $\mathcal{I}(Supp(x)) =$ *true*).
- For all $x \in \mathbf{R}_a \cup \mathbf{R}_e$, $\mathcal{I}(eVal(x)) =$ *true* iff
  ($\mathcal{I}(Val(x)) =$ *true* and $\mathcal{I}(Supp(x)) =$ *true*).

Note that in the current case, $Sup(U)$ refers to Definition 5.
We have to prove that $S_\mathcal{I} = S$, $\Gamma_\mathcal{I} = \Gamma$, $\Delta_\mathcal{I} = \Delta$, and that $\mathcal{I}$ is a support-founded model of Σ_d(**REBAF**). And for proving that $\mathcal{I}$ is a support-founded model of Σ_d(**REBAF**) it is sufficient to prove that $\mathcal{I}$ satisfies the formulae **(1)**, **(2)**, **(3)**, **(1bis)**, **(2bis)**, **(3bis)** and **(17)**, **(18)**, **(11)**, **(12)** and that is a support-founded interpretation.[15]

★ Let $x \in S_\mathcal{I}$. By definition of $S_\mathcal{I}$, $\mathcal{I}(eAcc(x)) =$ *true*, that is $\mathcal{I}(Acc(x)) =$ *true* and $\mathcal{I}(Supp(x)) =$ *true*. By definition of $\mathcal{I}(Acc)$ and $\mathcal{I}(Supp)$ it follows that $x \in S$. Conversely, given $x \in S$, it holds that $\mathcal{I}(Acc(x)) =$ *true*. As U is admissible, U is self-supporting, so $x \in Sup(U)$, then it holds that $\mathcal{I}(Supp(x)) =$ *true*. As a consequence, $\mathcal{I}(eAcc(x)) =$ *true* and $x \in S_\mathcal{I}$. Proving that $\Gamma_\mathcal{I} = \Gamma$ and $\Delta_\mathcal{I} = \Delta$ is similar.

★ Obviously $\mathcal{I}$ satisfies formulae **(3)**, **(2bis)**, **(3bis)**.

★ Let us first consider formula **(2)**. Let $y \in \mathbf{R}_a$ and $x \in \mathbf{A}$ with $x = t_y$, $\mathcal{I}(eVal(y)) =$ *true* and $\mathcal{I}(eAcc(s_y)) =$ *true*. Then $s_y \in S$ and $y \in \Gamma$. Let us assume that $\mathcal{I}(NAcc(x)) =$ *false*. Then $\mathcal{I}(Acc(x)) =$ *true*, by definition of $\mathcal{I}(NAcc)$. As U is admissible, U is conflict-free, so x cannot belong to S, and, by definition of $\mathcal{I}(Acc)$, it follows that $x \in Defended(U)$ and so $y \in UnAct(U)$, that is y or s_y belongs to $UnAcc(U)$. However, y and s_y being elements of the admissible structure U, due to Lemma 1, we obtain a contradiction. Hence, we have proved that $\mathcal{I}(NAcc(x)) =$ *true* and formula **(2)** is satisfied by $\mathcal{I}$. Proving that formula **(1)** is satisfied by $\mathcal{I}$ is similar.

★ Let us first consider formula **(17)**. Let x such that $\mathcal{I}(Supp(x)) =$ *true*. By definition of $\mathcal{I}(Supp)$, $x \in Sup(U)$. By definition of $Sup(U)$, either $x \in \mathbf{P}$ or x is the target of a support α such that $\alpha \in \Delta$, $\alpha \in Sup(U \setminus \{x\})$, $s_\alpha \in S$ and $s_\alpha \in Sup(U \setminus \{x\})$. In the first case, formula **(17)** is trivially satisfied by $\mathcal{I}$. In the second case, as $S = S_\mathcal{I}$ and $\Delta = \Delta_\mathcal{I}$ it holds that $\mathcal{I}(eAcc(s_\alpha)) =$ *true* and $\mathcal{I}(eVal(\alpha)) =$ *true*. Hence formula **(17)** is satisfied by $\mathcal{I}$.

★ Let us consider formula **(1bis)**. In the case when $\mathcal{I}(PrimaFacie(x)) =$ *true*, $x \in \mathbf{P}$, so $x \in Sup(U)$, hence $\mathcal{I}(Supp(x)) =$ *true* and formula **(1bis)** is satisfied. Let us consider the case when $x \notin \mathbf{P}$ and x is the target of a support y such that $\mathcal{I}(eAcc(s_y)) =$ *true* and $\mathcal{I}(eVal(y)) =$ *true*. We have to prove that $\mathcal{I}(Supp(x)) =$ *true*. As $S_\mathcal{I} = S$ and $\Delta_\mathcal{I} = \Delta$ it holds that $y \in \Delta$ and $s_y \in S$. Moreover, as U is admissible, U is self-supporting, so y and s_y belong to $Sup(U)$. From Lemma 3, it follows that $x \in Sup(U)$ hence $\mathcal{I}(Supp(x)) =$ *true*. So formula **(1bis)** is satisfied by $\mathcal{I}$.

[15] By definition, formulae **(4)** to **(10)** are satisfied by $\mathcal{I}$.

★ Let us now consider formula **(18)**. Consider x such that $\mathcal{I}(UnSupp(x))$ = *true*. By definition of $\mathcal{I}(UnSupp)$, $x \in UnSupp(U)$. And, since $UnSupp(U) = \overline{Sup(U')}$ (where $U' = (\overline{Def_{\mathbf{A}}(U)}, \mathbf{R}_a, \overline{Def_{\mathbf{R}_e}(U)})$), $x \in \overline{Sup(U')}$. So $x \notin \mathbf{P}$ and using the contrapositive of Lemma 3, applied to the structure U', it follows that for each support leading to x, either the support or its source is defeated by U, or the support or its source is itself not supported by U', hence belongs to $UnSupp(U)$. So the "only if" part of formula **(18)** is satisfied by $\mathcal{I}$.
For the "if" part, let us consider x such that $x \notin \mathbf{P}$ and for each support leading to x, either the support or its source is defeated by $(S_\mathcal{I}, \Gamma_\mathcal{I}, \Delta_\mathcal{I}) = U$, or the support or its source belongs to $UnSupp(U) = \overline{Sup(U')}$. As $U' \setminus \{x\} \subseteq U'$, it holds that $\overline{Sup(U')} \subseteq \overline{Sup(U' \setminus \{x\})}$. Hence, from Definition 5, it holds that $x \notin Sup(U')$, that is $x \in UnSupp(U)$ and so $\mathcal{I}(UnSupp(x)) =$ *true*. So formula **(18)** s satisfied by $\mathcal{I}$.

★ Let us now consider formula **(11)**. Let $\alpha \in \mathbf{R}_a$ and $x \in \mathbf{A}$ such that $x = t_\alpha$ and $\mathcal{I}(Acc(x)) =$ *true*. By definition of $\mathcal{I}(Acc)$, either $x \in S$ or ($x \notin S$, $x \notin Sup(U)$ and $x \in Defended(U)$). As U is admissible, in both cases, it holds that $\alpha \in UnAct(U)$. Then the fact that $\mathcal{I}$ satisfies formula **(11)** follows directly from the definition of $UnAct(U)$, the definition of $\mathcal{I}(Unsupp)$ and the fact that for an argument (resp. an attack) x, $\mathcal{I}(eAcc(x)) =$ *true* (resp. $\mathcal{I}(eVal(x)) =$ *true*) iff $x \in S$ (resp. $x \in \Gamma$).
Proving that formula **(12)** is satisfied by $\mathcal{I}$ is similar.

★ Finally, we have to prove that $\mathcal{I}$ is support-founded.
Condition 2 of Definition 14 is trivially satisfied.
Consider now Condition 1. Let $x \in \mathbf{A}$ such that x is non prima-facie and there exists a DCS $\mathbf{C}$ containing x. Assume that $\mathcal{I}(eAcc(x)) =$ *true*. So $x \in U$ and, since U is admissible (so self-supporting) and x non prima-facie, then there exists at least one chain of supported supports $(x_0, \ldots, x_n)$ leading to x and originated in x_0 that is prima-facie with any x_i belonging to U. Moreover, for all x_i, we have $x_i \in Sup(U)$. So following the definition of $\mathcal{I}$ we have either $\mathcal{I}(eAcc(x_i)) =$ *true* or $\mathcal{I}(eVal(x_i)) =$ *true* depending of the nature of x_i (argument or support). Thus there exists an impacting support chain $(x_0, \ldots, x_n)$ for x that is satisfied by $\mathcal{I}$. So $\mathcal{I}$ is a support-founded model. The proof for $x \in \mathbf{R}_e$ is similar.

⇐ Let $\mathcal{I}$ be a support-founded model of $\Sigma_d(\mathbf{REBAF})$. We have to prove that the structure $U = (S_\mathcal{I}, \Gamma_\mathcal{I}, \Delta_\mathcal{I})$ is admissible.

★ Let prove that U is conflict-free w.r.t. **REBAF**. If it is not the case, there exist $x \in S_\mathcal{I} \cup \Gamma_\mathcal{I} \cup \Delta_\mathcal{I}$ and $y \in \Gamma_\mathcal{I}$, with $s_y \in S_\mathcal{I}$ and $t_y = x$.
By definition, it holds that $\mathcal{I}(eAcc(s_y)) =$ *true* and $\mathcal{I}(eVal(y)) =$ *true*. Moreover, in the case when $x \in S_\mathcal{I}$, it holds that $\mathcal{I}(eAcc(x)) =$ *true* and so $\mathcal{I}(Acc(x)) =$ *true* (as $\mathcal{I}$ satisfies formula **(2bis)**). Then it holds that $\mathcal{I}(NAcc(x)) =$ *false* (as $\mathcal{I}$ satisfies formula **(3)**). As a consequence, formula **(2)** is falsified.
In the case when $x \in \Gamma_\mathcal{I} \cup \Delta_\mathcal{I}$, it holds that $\mathcal{I}(eVal(x)) =$ *true* and

so $\mathcal{I}(Val(x)) = \textit{true}$ (as $\mathcal{I}$ satisfies formula **(3bis)**). As a consequence, formula **(1)** is falsified.
In both cases, there is a contradiction with $\mathcal{I}$ being a model of $\Sigma(\textbf{REBAF})$.

★ Let us prove that U is self-supporting. Assume that $x \in S_{\mathcal{I}}$ (resp. $x \in \Gamma_{\mathcal{I}} \cup \Delta_{\mathcal{I}}$). By definition, it holds that $\mathcal{I}(eAcc(x)) = \textit{true}$ (resp. $\mathcal{I}(eVal(x)) = \textit{true}$). As $\mathcal{I}$ satisfies formula **(2bis)**, $\mathcal{I}(Supp(x)) = \textit{true}$. As $\mathcal{I}$ satisfies formula **(17)**, it holds that either $x \in \mathbf{P}$ or x is the target of a support x_n of source x_{n-1} such that $x_n \in \Delta_{\mathcal{I}}$ and $x_{n-1} \in S_{\mathcal{I}}$. In the first case, it holds that $x \in Sup(U)$. In the other case, it holds that $\mathcal{I}(eAcc(x_{n-1})) = \textit{true}$ and $\mathcal{I}(eVal(x_n)) = \textit{true}$, and formula **(17)** can still be used, thus enabling to build a chain of supports. As U is finite and $\mathcal{I}$ is support founded, this process will end with $x_1 \in \mathbf{P}$ and $x_0 = \mathbf{s}(x_1) \in \mathbf{P}$. And so it can still be proved that $x \in Sup(U)$ w.r.t. Definition 5. Hence U is self-supporting.

★ It remains to prove that, given x an element of the structure, if x is the target of an attack α, then α is unactivable w.r.t. U. Assume that $x \in S_{\mathcal{I}}$ is the target of an attack α. By definition, it holds that $\mathcal{I}(eAcc(x)) = \textit{true}$. It follows that $\mathcal{I}(Acc(x)) = \textit{true}$. As $\mathcal{I}$ satisfies formula **(11)**, it follows that either there exists an attack β targeting α (or s_α) with $\beta \in \Delta_{\mathcal{I}}$ and $s_\beta \in S_{\mathcal{I}}$, or $\mathcal{I}$ satisfies $UnSupp(\alpha)$ (or $\mathcal{I}$ satisfies $UnSupp(s_\alpha)$).

- In the first case, it holds that α (resp. s_α) belongs to $Def(U)$.
- In the second case, we prove that α (resp. s_α) belongs to $UnSupp(U)$. For that purpose, we must prove that for any element x, if $\mathcal{I}$ satisfies $UnSupp(x)$, then $x \in UnSupp(U)$, or equivalently, if $x \in Sup(U')$ then $\mathcal{I}$ does not satisfy $UnSupp(x)$. Let us consider $x \in Sup(U')$. There is a chain of supports leading to x, rooted in prima-facie elements such that each support (and its source) in the chain is not defeated by U. As $\mathcal{I}$ satisfies formula **(18)**, the contrapositive of the "only if" part of formula **(18)** can be used for proving that each supported element y in this chain is such that $\mathcal{I}(UnSupp(y)) = \textit{false}$. The proof starts with the prima-facie elements of the set, and goes on by induction. Thus it can be proved that $\mathcal{I}(UnSupp(x)) = \textit{false}$.

So, in both cases, α is unactivable w.r.t. U.
The same reasoning can be done for $x \in \Gamma_{\mathcal{I}} \cup \Delta_{\mathcal{I}}$ using formula **(12)**. Hence, we can prove that U is admissible.

2. (complete semantics)

⇒ Assume that the structure $U = (S, \Gamma, \Delta)$ is complete. Let us build an interpretation $\mathcal{I}$ of $\Sigma_d(\textbf{REBAF}) \cup \Sigma_r(\textbf{REBAF})$:

- We keep the same interpretation as the one used in Item 1 of the current proof except for Acc, Val.
- For all $x \in \mathbf{A}$, $\mathcal{I}(Acc(x)) = \textit{true}$ iff $x \in S$ or ($x \notin S$ and $x \in Defended(U)$).

- For all $x \in \mathbf{R}_a$ (resp. $\in \mathbf{R}_e$), $\mathcal{I}(Val(x)) =$ ***true*** if and only $x \in \Gamma$ (resp. Δ) or ($x \notin \Gamma$ (resp. Δ) and $x \in Defended(U)$).

We have to prove that $S_\mathcal{I} = S$, $\Gamma_\mathcal{I} = \Gamma$ and $\Delta_\mathcal{I} = \Delta$, and that $\mathcal{I}$ is a support-founded model of $\Sigma_d(\mathbf{REBAF}) \cup \Sigma_r(\mathbf{REBAF})$.

★ Note that if U is complete, for all $x \in \mathbf{A} \cup \mathbf{R}_a \cup \mathbf{R}_e$, if $x \notin S$ and $x \in Defended(U)$ then $x \notin Sup(U)$. So the above constraint expressed for the definition of $\mathcal{I}(Acc)$ (resp. $\mathcal{I}(Val)$), $x \notin S$ and $x \in Defended(U)$, is stronger than the one used for defining a model of an admissible structure ($x \notin S$, $x \notin Sup(U)$ and $x \in Defended(U)$).
Due to the above remark and the proof of Item 1 of this proof, it holds that $\mathcal{I}$ satisfies $S_\mathcal{I} = S$, $\Gamma_\mathcal{I} = \Gamma$, $\Delta_\mathcal{I} = \Delta$, and that $\mathcal{I}$ is a model of $\Sigma_d(\mathbf{REBAF})$.

★ Now let prove that $\mathcal{I}$ satisfies formulae **(13)** and **(14)**. Let us consider formula **(13)**. Let $x \in \mathbf{A}$ such that for each attack α targeting x, either $\mathcal{I}(UnSupp(\alpha)) =$ ***true***, or $\mathcal{I}(UnSupp(s_\alpha)) =$ ***true***, or α (or s_α) is attacked by β with $\beta \in \Gamma_\mathcal{I}$ and $s_\beta \in S_\mathcal{I}$. Due to the definition of $\mathcal{I}(UnSupp)$, for each attack α targeting x, either $\alpha \in UnSupp(U)$, or $s_\alpha \in UnSupp(U)$, or α (or s_α) belongs to $Def(U)$. In other words, for each attack α targeting x, $\alpha \in UnAct(U)$, so $x \in Defended(U)$. Now, by definition of $\mathcal{I}(Acc)$, it holds that $\mathcal{I}(Acc(x)) =$ ***true***. We have proved that $\mathcal{I}$ satisfies formula **(13)**. Proving that $\mathcal{I}$ satisfies formula **(14)** is similar.
So $\mathcal{I}$ is a model of $\Sigma_d(\mathbf{REBAF}) \cup \Sigma_r(\mathbf{REBAF})$.

★ It remains to prove that $\mathcal{I}$ is support-founded. For that purpose, the proof written in Item 1 of the current proof can be used as U is self-supporting.

⇐ Let $\mathcal{I}$ be a support-founded model of $\Sigma_d(\mathbf{REBAF}) \cup \Sigma_r(\mathbf{REBAF})$. We have to prove that the structure $U = (S_\mathcal{I}, \Gamma_\mathcal{I}, \Delta_\mathcal{I})$ is complete. For that purpose, it is enough to prove that $Acc(U)$ is included in $S_\mathcal{I} \cup \Gamma_\mathcal{I} \cup \Delta_\mathcal{I}$.
Consider $x \in \mathbf{A} \cap Acc(U)$. So $x \in Sup(U)$ and $x \in Defended(U)$. The first condition implies that $\mathcal{I}(Supp(x)) =$ ***true***, as $\mathcal{I}$ satisfies formula **(1bis)** and following the definition of $Sup(U)$. The second condition means that for each attack α targeting x, either $\alpha \in UnSupp(U)$, or $s_\alpha \in UnSupp(U)$, or α (or s_α) belongs to $Def(U)$ (*i.e.* α –or s_α– is attacked by $\beta \in U$ with $s_\beta \in U$). So, since $\mathcal{I}$ is a support-founded models (so Condition 2 of the definition of a support-founded model holds) and the fact that if an element β (resp. s_β) belongs to the structure then $\mathcal{I}(eVal(\beta))$ (resp. $\mathcal{I}(eAcc(s_\beta))$) is also ***true***, the premisse of formula **(13)** is ***true***, and as $\mathcal{I}$ satisfies formula **(13)**, it follows that $\mathcal{I}(Acc(x)) =$ ***true***. As $\mathcal{I}$ satisfies formula **(2bis)** it holds that $\mathcal{I}(eAcc(x)) =$ ***true***, so $x \in S_\mathcal{I}$. Similarly, it can be proved that for all $x \in \mathbf{R}_a \cap Acc(U)$ (resp. $x \in \mathbf{R}_e \cap Acc(U)$), $x \in \Gamma_\mathcal{I}$ (resp. $x \in \Delta_\mathcal{I}$). We have proved that U is a complete structure.
The proof is similar for any support or attack in $Acc(U)$.

3. (preferred semantics) Let $\mathcal{I}$ be an interpretation of a set of formulae Σ_x. Let $U_\mathcal{I}$ denote the structure $(S_\mathcal{I}, \Gamma_\mathcal{I}, \Delta_\mathcal{I})$. It is easy to see that $\mathcal{I}$ is a $\subseteq$-maximal

support-founded model of Σ_x iff the structure $U_{\mathcal{I}}$ is $\subseteq$-maximal among all the structures of the form $U_{\mathcal{J}} = (S_{\mathcal{J}}, \Gamma_{\mathcal{J}}, \Delta_{\mathcal{J}})$, where $\mathcal{J}$ denotes a support-founded model of Σ_x. Then taking $\Sigma_x = \Sigma_d(\textbf{REBAF})$, it follows that the preferred structures correspond to the structures $U_{\mathcal{I}}$ where $\mathcal{I}$ is a $\subseteq$-maximal support-founded model of $\Sigma_d(\textbf{REBAF})$.

4. (grounded semantics) Let $\mathcal{I}$ be an interpretation of a set of formulae Σ_x. Let $U_{\mathcal{I}}$ denote the structure $(S_{\mathcal{I}}, \Gamma_{\mathcal{I}}, \Delta_{\mathcal{I}})$. It is easy to see that $\mathcal{I}$ is a $\subseteq$-minimal support-founded model of Σ_x iff the structure $U_{\mathcal{I}}$ is $\subseteq$-minimal among all the structures of the form $U_{\mathcal{J}}$, where $\mathcal{J}$ denotes a support-founded model of Σ_x. Taking $\Sigma_x = \Sigma_d(\textbf{REBAF}) \cup \Sigma_r(\textbf{REBAF})$, it follows that the grounded structure correspond to the structure $U_{\mathcal{I}}$ where $\mathcal{I}$ is a $\subseteq$-minimal support-founded model of $\Sigma_d(\textbf{REBAF}) \cup \Sigma_r(\textbf{REBAF})$.

5. (stable semantics)

   $\Rightarrow$ Assume that the structure $U = (S, \Gamma, \Delta)$ is stable. Let us define an interpretation $\mathcal{I}$ of $\Sigma_s(\textbf{REBAF})$ as follows:
   - Once again, we keep the same interpretation as the one used in Item 1 of the current proof except for Acc, Val.
   - For all $x \in \mathbf{A}$, $\mathcal{I}(Acc(x)) =$ *true* iff $x \in S$ or $x \notin Def(U)$.
   - For all $x \in \mathbf{R}_a$ (resp. $\in \mathbf{R}_e$), $\mathcal{I}(Val(x)) =$ *true* iff $x \in \Gamma$ (resp. Δ) or $x \notin Def(U)$.

   We have to prove that $S_{\mathcal{I}} = S$, $\Gamma_{\mathcal{I}} = \Gamma$ and $\Delta_{\mathcal{I}} = \Delta$, and that $\mathcal{I}$ is a support-founded model of $\Sigma_s(\textbf{REBAF})$. And, for proving that $\mathcal{I}$ is a support-founded model of $\Sigma_s(\textbf{REBAF})$ it is sufficient to prove that $\mathcal{I}$ satisfies formulae **(1)**, **(2)**, **(3)**, **(1bis)**, **(2bis)**, **(3bis)** and **(17)**, **(18)**, **(15)**, **(16)**, **(19)** and that $\mathcal{I}$ is support-founded.

   ★ Let $x \in S_{\mathcal{I}}$. By definition, $\mathcal{I}(Acc(x)) =$ *true* and $\mathcal{I}(Supp(x)) =$ *true*. By definition of $\mathcal{I}(Acc)$ and $\mathcal{I}(Supp)$, it follows that $x \in Sup(U)$ and ($x \in S$ or $x \notin Def(U)$). Following Lemma 2, $x \notin UnSupp(U)$ and ($x \in S$ or $x \notin Def(U)$). If $x \notin S$, as U is stable, it follows that $x \in Def(U)$ or $x \in UnSupp(U)$. We obtain a contradiction, hence $x \in S$.
   Conversely, given $x \in S$, it holds that $\mathcal{I}(Acc(x)) =$ *true*. As U is stable, U is self-supporting, so $x \in Sup(U)$, then it holds that $\mathcal{I}(Supp(x)) =$ *true*. As a consequence, $\mathcal{I}(eAcc(x)) =$ *true* and $x \in S_{\mathcal{I}}$. Proving that $\Gamma_{\mathcal{I}} = \Gamma$ and $\Delta_{\mathcal{I}} = \Delta$ is similar.

   ★ Obviously $\mathcal{I}$ satisfies formulae **(3)**, **(2bis)**, **(3bis)**.

   ★ Let us first consider formula **(2)**. Let $y \in \mathbf{R}_a$ and $x \in \mathbf{A}$ with $x = t_y$, $\mathcal{I}(eVal(y) =$ *true* and $\mathcal{I}(eAcc(s_y) =$ *true*. Then $s_y \in S$ and $y \in \Gamma$, and it holds that $x \in Def(U)$. As U is stable, U is conflict-free, so x cannot belong to S. Hence we have $x \notin S$ and $x \in Def(U)$, or equivalently $\mathcal{I}(Acc(x)) =$ *false*, by definition of $\mathcal{I}(Acc)$ and then $\mathcal{I}(NAcc(x)) =$ *true*,

by definition of $\mathcal{I}(NAcc)$. We have proved that $\mathcal{I}$ satisfies formula **(2)**. Proving that formula **(1)** is satisfied by $\mathcal{I}$ is similar.

★ Proving that $\mathcal{I}$ satisfies formulae **(1bis)**, **(17)**, **(18)** can be done with exactly the same reasoning as the one used in Item 1 of the current proof.

★ Let us now consider formula **(15)**. Let $x \in \mathbf{A}$ such that $\mathcal{I}(Acc(x)) =$ *false*. By definition of $\mathcal{I}(Acc)$, it holds that $x \notin S$ and $x \in Def(U)$. So, there is $y \in \Gamma$ with $x = t_y$ and $s_y \in S$. Hence, there is $y \in \Gamma_{\mathcal{I}}$ with $x = t_y$ and $s_y \in S_{\mathcal{I}}$, or equivalently, there is $y \in \mathbf{R}_a$ with $x = t_y$ and $\mathcal{I}(eVal(y)) =$ *true* and $\mathcal{I}(eAcc(s_y)) =$ *true*. We have proved that $\mathcal{I}$ satisfies formula **(15)**. Proving that formula **(16)** is satisfied by $\mathcal{I}$ is similar.

★ Lastly, we consider formula **(19)**. Let $x \in \mathbf{A} \cup \mathbf{R}_a \cup \mathbf{R}_e$ such that $\mathcal{I}(Supp(x)) =$ *false*. By definition of $\mathcal{I}(Supp)$, $x \notin Sup(u)$. Due to Lemma 2, it follows that $x \in UnSupp(U)$, hence $\mathcal{I}(UnSupp(x)) =$ *true*, by definition of $\mathcal{I}(UnSupp)$. We have proved that $\mathcal{I}$ satisfies formula **(19)**. So $\mathcal{I}$ is a model of $\Sigma_s(\mathbf{REBAF})$.

★ It remains to prove that $\mathcal{I}$ is support-founded. For that purpose, the proof written in Item 1 of the current proof can be used as U is self-supporting.

$\Leftarrow$ Let $\mathcal{I}$ be a support-founded model of $\Sigma_s(\mathbf{REBAF})$. We have to prove that the structure $U = (S_{\mathcal{I}}, \Gamma_{\mathcal{I}}, \Delta_{\mathcal{I}})$ is stable.
As noted in Definition 7, it is sufficient to prove that U is conflict-free, self-supporting and satisfies $\overline{U} \subseteq UnAcc(U)$.
As $\Sigma_s(\mathbf{REBAF})$ contains $\Sigma(\mathbf{REBAF})$, from Proposition 1, we know that the structure U is conflict-free. Moreover, $\Sigma_s(\mathbf{REBAF})$ contains formulae **(17)**, **(18)**. So, with exactly the same reasoning as the one used in Item 1 for the admissible case, it can be proved that U is self-supporting.
It remains to prove that $\overline{U} \subseteq UnAcc(U)$. Let $x \in \mathbf{A}$ such that $x \in \overline{U}$. So $x \notin S_{\mathcal{I}}$ and by definition of $S_{\mathcal{I}}$, $\mathcal{I}(eAcc(x)) =$ *false*. As $\mathcal{I}$ satisfies formula **(2bis)**, it follows that $\mathcal{I}(Acc(x)) =$ *false* or $\mathcal{I}(Supp(x)) =$ *false*. In the case when $\mathcal{I}(Acc(x)) =$ *false*, as $\mathcal{I}$ satisfies formula **(15)**, it follows that $x \in Def(U)$. If $\mathcal{I}(Acc(x)) =$ *true*, it holds that $\mathcal{I}(Supp(x)) =$ *false*. As $\mathcal{I}$ satisfies formula **(19)**, it follows that $\mathcal{I}(UnSupp(x)) =$ *true*, so $x \in UnSupp(U)$ (following Condition 2 of Definition 14 since $\mathcal{I}$ is support-founded). In both cases, we have that $x \in UnAcc(U)$. We have proved that U is stable.