Logical Encoding of Argumentation Frameworks with Higher-order Attacks and Necessary Supports

Marie-Christine Lagasquie-Schiex

Logical Encoding of Argumentation Frameworks with Higher-order Attacks and Necessary Supports

M-Christine Lagasquie-Schiex

IRIT, Université de Toulouse, CNRS, Toulouse INP, UT3
118 route de Narbonne, 31062 Toulouse, France
{lagasq}@irit.fr

Tech. Report
IRIT/RR- -2021- -08- -FR

Novembre 2021

Abstract

In [23], we have proposed 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).

In this new work, we propose an extension of this previous works in order to take into account another kind of higher-order bipolar argumentation frameworks, those using the necessary meaning for the support relation.

Contents

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 [32]. Formal abstract frameworks have greatly eased the modelling and study of argumentation. The original Dung's argumentation framework (AF) [19] 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 [22,33]. In [12], 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 [4], necessary support [27, 28], evidential support [29, 30]). 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. [13] presents a comparative study in order to restate these proposals in a common setting, the bipolar argumentation framework (see also [15] for another survey).

Higher-order interactions. The idea of encompassing attacks to attacks in abstract argumentation frameworks has been first considered in [3] 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 [26]), or for modelling situations where an attack might be defeated by an argument, without contesting the acceptability of the source of the attack [2]. Attacks to attacks and supports have been first considered in [20] with higher level networks, then in [34]; and more generally, [16] 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 is an example of higher-order interactions (attacks and supports) in the legal field (this example is borrowed from [1] and modified in order to introduce some supports).

Example 1 *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 arguments a or b themselves. And so, the prosecutor's argumentation seems no longer sufficient for proving the intention to kill.*

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 [2, 5, 6, 16], 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 [7] or for higher-order attacks and supports (necessary supports: [10, 17], evidential supports: [8]). 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 [23], 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 the most general definition of REBAF (with support cycles and collective interactions). So the aim of the current work is to do the same thing but for argumentation frameworks with higher-order attacks and necessary supports (RAFN).

The paper is organized as follows: the necessary background about argumentation frameworks is given in Section 2; the logical encoding for REBAF is recalled in Section 3; the new proposition that can handle RAFN is given in Section 5; Section 6 concludes the paper. The proofs are given in Appendix B.

2 Background on argumentation frameworks

Note that the text (definitions, propositions and examples) of this section is extracted from [14].

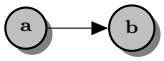
2.1 The Standard Abstract Framework

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

Definition 1 **[18]** *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 2 *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 **[18]** *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* **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* $S \subseteq \mathcal{F}(S)$*.*
- S *is* a complete extension *of* **AF** *iff it is conflict-free and a fixed point of* $\mathcal{F}$*.*
- S *is* the grounded extension *of* **AF** *iff it is the minimal (w.r.t.* $\subseteq$*) fixed point*[2] *of* $\mathcal{F}$*.*
- S *is* a preferred extension *of* **AF** *iff it is a maximal (w.r.t.* $\subseteq$*) complete extension.*
- S *is* a stable extension *of* **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.

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

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

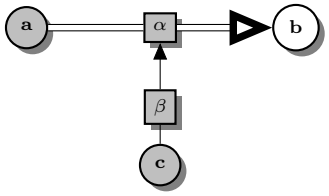
2.2 A Framework with Higher-Order Evidential Supports and Attacks

In this section, we recall the extension of [7] proposed in [8] for handling recursive attacks and evidence-based supports.

Definition 3 **[8]** *An* evidence-based recursive argumentation framework (REBAF) *is a sextuple* $\langle \mathbf{A}, \mathbf{R}, \mathbf{E}, \mathbf{s}, \mathbf{t}, \mathbf{P} \rangle$ *where* $\mathbf{A}$*,* $\mathbf{R}$ *and* $\mathbf{E}$ *are three (possible infinite) pairwise disjunct sets respectively representing arguments, attacks and supports names, and where* $\mathbf{P} \subseteq \mathbf{A} \cup \mathbf{R} \cup \mathbf{E}$ *is a set representing the* prima-facie *elements that do not need to be supported. Functions* $\mathbf{s} : (\mathbf{R} \cup \mathbf{E}) \longrightarrow 2^{\mathbf{A}} \setminus \varnothing$ *and* $\mathbf{t} : (\mathbf{R} \cup \mathbf{E}) \longrightarrow (\mathbf{A} \cup \mathbf{R} \cup \mathbf{E})$ *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 1 (cont'd): *The argumentation framework corresponding to the second example given in the introduction can be represented as follows (argument names are given in circular nodes, interaction names in square nodes, prima-facie elements are in grey nodes and non prima-facie element in white nodes; supports are represented by double edges):*



□

Semantics of REBAF are defined in [8] using the extension of the notion of structure introduced in [7]. 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 $\mathbf{REBAF} = \langle \mathbf{A}, \mathbf{R}, \mathbf{E}, \mathbf{s}, \mathbf{t}, \mathbf{P} \rangle$.

Definition 4 **[8]** *A triple* $U = (S, \Gamma, \Delta)$ *is said to be a* structure *of* **REBAF** *iff it satisfies:* $S \subseteq \mathbf{A}$*,* $\Gamma \subseteq \mathbf{R}$ *and* $\Delta \subseteq \mathbf{E}$*.*

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 [7] in order to take into account the evidential supports.

Definition 5 **[8]** *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 $X \in \{\mathbf{A}, \mathbf{R}, \mathbf{E}\}$

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

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

- *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,

Definition 6 *Let U be a structure.*

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

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

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

- *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:*

$$UnAct(U) \stackrel{\text{def}}{=} \{\alpha \in \mathbf{R} | \alpha \in UnAcc(U) \text{ or } \mathbf{s}(\alpha) \cap UnAcc(U) \neq \varnothing\}$$

Definition 7 **[8]** *An element $x \in \mathbf{A} \cup \mathbf{R} \cup \mathbf{E}$ is said to be* acceptable *w.r.t. a structure U iff (i) $x \in Sup(U)$ and (ii) every attack $\alpha \in \mathbf{R}$ 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 8 **[8]** *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* $(X, Y) \in \{(S, \mathbf{A}), (\Gamma, \mathbf{R}), (\Delta, \mathbf{E})\}$,
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]

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

[5] In fact, strictly speaking, this chain is in reality a "tree". That is due to several reasons. The first one is that each support can be in turn supported. And the second reason is the fact that interactions are collective; so the source of a support can be a set of arguments and in this case all elements in the source are needed for supporting the target.

[6] The definition for the grounded extension is not given in [8] but can be easily proposed following the definition used in the AF case.

6. preferred *iff it is a $\subseteq$-maximal admissible structure,*

7. stable[7] *iff* $(S \cup \Gamma \cup \Delta) = \overline{UnAcc(U)}$.

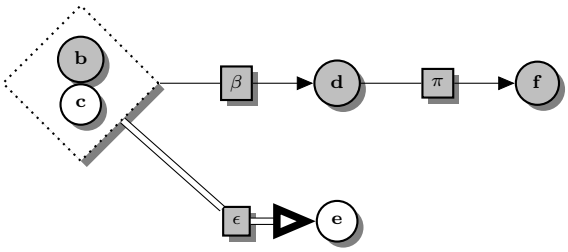
From the above definitions, it follows that if U is a conflict-free structure, unsupportable elements w.r.t. U are not supported w.r.t. U, that is $UnSupp(U) \subseteq \overline{Sup(U)}$.

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 [8], which enable to prove for instance that there is a unique grounded structure.

The previous definitions are illustrated on the following example.

Example 1 (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 $U = (\{a, c\}, \{\beta\}, \varnothing)$. □

Example 3 *In this example, there are a collective attack and a collective support using the same source. Arguments c and e are the only elements that are not prima-facie.*



Since c is unsupportable, then neither β nor ϵ can be activable and there is one preferred structure that is: $(\{b, d\}, \{\beta, \pi\}, \{\epsilon\})$. Trivially, an interaction can be activable w.r.t. a structure only if all the arguments in its source are in this structure.

Finally, REBAF is a conservative generalization of RAF described in [7] 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 [8]).

In [23], some notions related to directed cycles of supports have been defined in order to take into account support cycles and collective interactions in the logical computation of structures for the REBAF.

Definition 9 [23] *Let* REBAF $= \langle \mathbf{A}, \mathbf{R}, \mathbf{E}, \mathbf{s}, \mathbf{t}, \mathbf{P} \rangle$. *A* directed cycle of supports (DCS) *in this REBAF is a sequence* $\mathbf{C} = (x_0, \ldots, x_{n-1}, x_n)$ *such that:*[8]

- $n > 0$ *and* n *is the* size of the DCS
- $\forall i = 0 \ldots n$, *either* $x_i \in \mathbf{E}$, *or* $x_i = (a, S)$ *with* $S \in 2^{\mathbf{A}} \setminus \varnothing$ *and* $a \in \mathbf{A} \cap S$ *(a is called the "target field" of x_i and S is called the "source field" of x_i),*
- $x_n = x_0$
- $\forall i = 0 \ldots n-1$, *if* $x_i = (a, S) \in (\mathbf{A}, 2^{\mathbf{A}} \setminus \varnothing)$ *then* $x_{i+1} \in \mathbf{E}$ *and* $\mathbf{s}(x_{i+1}) = S$,
- $\forall i = 0 \ldots n-1$, *if* $x_i \in \mathbf{E}$ *then*

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

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

– *if* $x_{i+1} \in \mathbf{E}$ *then* $\mathbf{t}(x_i) = x_{i+1}$
– *if* $x_{i+1} = (a, S) \in (\mathbf{A}, 2^{\mathbf{A}} \setminus \varnothing)$ *then* $\mathbf{t}(x_i) = a$.

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* $y \in \mathbf{E}$ *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* $\exists x_i \in \mathbf{E} \cap \mathbf{C}$ *and* $y = \mathbf{s}(x_i)$.

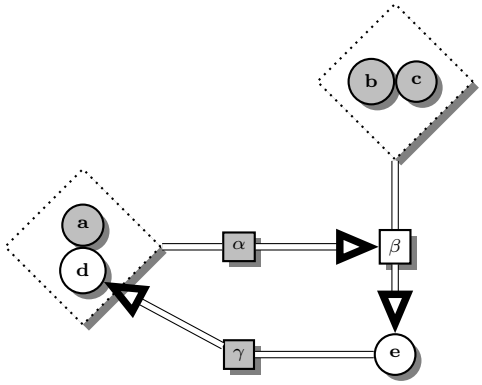
The set of inputs of the DCS $\mathbf{C}$ *is denoted by* $\mathbf{C}^{In}$ *and it is partitioned into* $\mathbf{C}^{In}_{\mathbf{A}} = \mathbf{C}^{In} \cap \mathbf{A}$ *and* $\mathbf{C}^{In}_{\mathbf{E}} = \mathbf{C}^{In} \cap \mathbf{E}$.

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 s.t. there exist* $x_i \in \mathbf{C}$ *and* $x'_j \in \mathbf{C}'$ *and* $x_i = x'_j$. *The* aggregation *of* $\mathbf{C}$ *and* $\mathbf{C}'$, *denoted by* $\mathbf{C} \cup \mathbf{C}'$, *is the directed cycle corresponding to the union of the sets* $\{x_0, \ldots, x_{n-1}\}$ *and* $\{x'_0, \ldots, x'_{m-1}\}$.

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* $\mathbf{C}$.

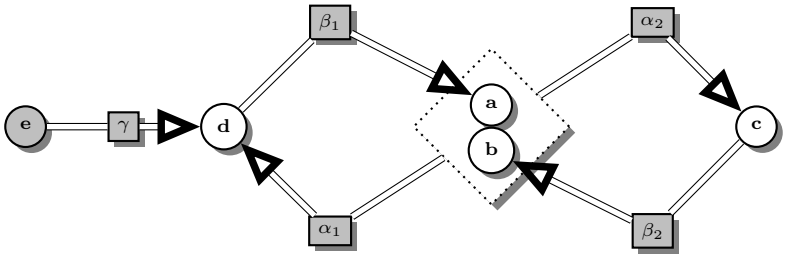
Note that a DCS is an "hybrid" sequence composed either with interactions, or with pairs (an argument, a non-empty set of arguments).

Example 4 *This example gives an example of support cycles with an higher-order support and some collective supports. Here* β, d *and* e *are not prima-facie.*



Here there is one DCS: $((d, \{a, d\}), \alpha, \beta, (e, \{e\}), \gamma, (d, \{a, d\}))$.
Note that the only preferred structure is $(\{a, b, c\}, \varnothing, \{\alpha, \gamma\})$.
It is also interesting to notice that a DCS can be represented by several sequences (n *sequences if* n *is the size of the DCS, each sequence being obtained by shifting to the right – or to the left). For instance, the DCS of this REBAF can also be expressed with:* $(\beta, (e, \{e\}), \gamma, (d, \{a, d\}), \alpha, \beta)$.

Example 5 *This example gives an example of several support cycles that can be aggregated*



Here there are three DCS (only the last one is a maximal DCS):

- $((d, \{d\}), \beta_1, (a, \{a, b\}), \alpha_1, (d, \{d\}))$

- $((c, \{c\}), \beta_2, (b, \{a, b\}), \alpha_2, (c, \{c\}))$
- $((d, \{d\}), \beta_1, (a, \{a, b\}), \alpha_2, (c, \{c\}), \beta_2, (b, \{a, b\}), \alpha_1, (d, \{d\}))$

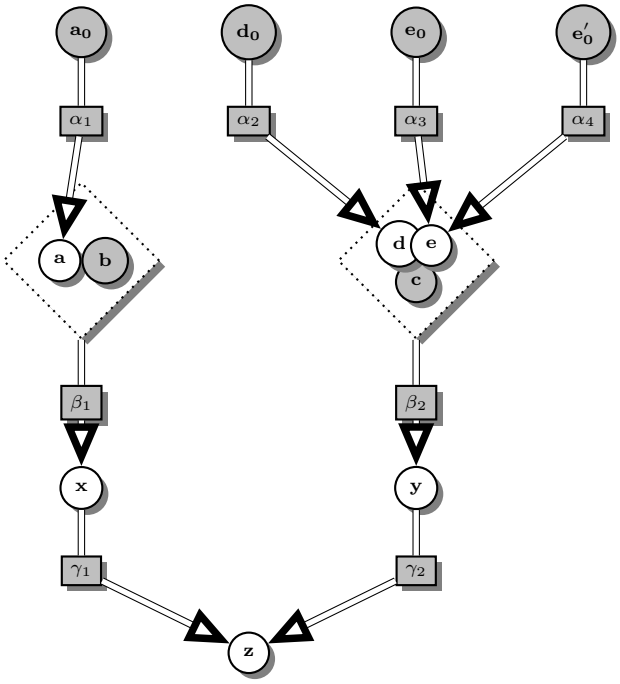
The interesting point is the fact that the set $\{a, b\}$ that is the source of α_1 and α_2 corresponds to two distinct elements in a DCS: $(a, \{a, b\})$ and $(b, \{a, b\})$; and each of them can be used as the preceding element of the supports α_1 or α_2 in the DCS.

Note that the only preferred structure is $(\{e, d, a\}, \varnothing, \{\alpha_1, \alpha_2, \beta_1, \beta_2, \gamma\})$.

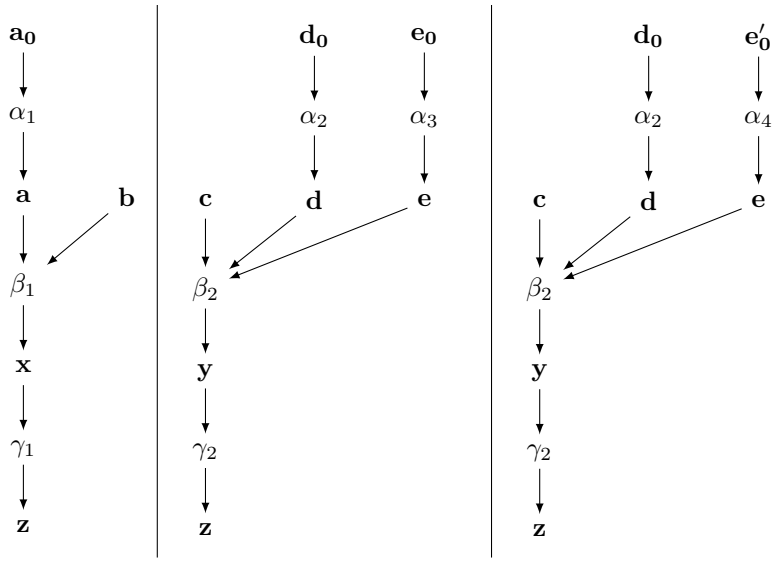
Another notion will be important in order to compute the logical characterization of REBAF semantics: the *impacting support elements* for an element of a REBAF.

Consider the following example:

Example 6 *Consider the following REBAF with only arguments and supports; among these, there exist 2 collective supports (one from $\{a, b\}$ to x and the another one from $\{c, d, e\}$ to y).*

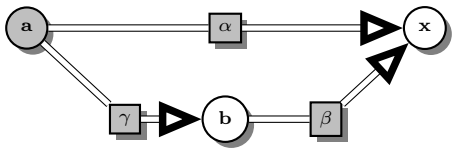


Let consider the elements that impact the supported status of argument z. Clearly, because of the collective interactions, we must use the notion of "trees"; indeed, any element of the source of a collective support must be supported if we want the target of this support to be also supported. Here, three "trees" must be taken into account for computing the supported status of z:



Other interesting examples show that some elements could appear several times in such a tree, since an argument can appeared in the source of several supports.

Example 7 *Consider the following REBAF:*



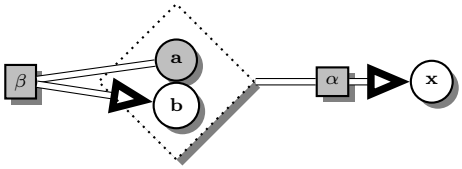
Let consider the elements that impact the supported status of argument x. Here 2 trees must be considered:

$$\mathbf{a} \longrightarrow \alpha \longrightarrow \mathbf{x}$$

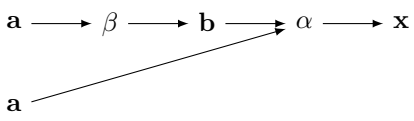
$$\mathbf{a} \longrightarrow \gamma \longrightarrow \mathbf{b} \longrightarrow \beta \longrightarrow \mathbf{x}$$

In this case, there is no repetition since there are 2 distinct trees.

Example 8 *Consider the following REBAF:*

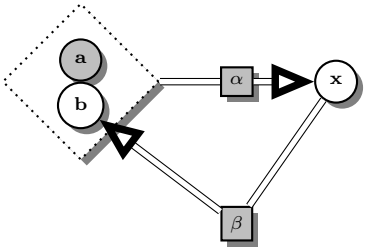


Let consider the elements that impact the supported status of argument x. Here a single tree must be taken into account:

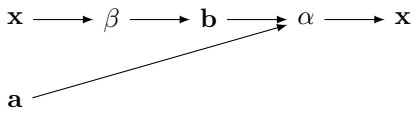


In this case, argument a must be repeated but in another branch and so as the source of another support.

Example 9 *Consider the following REBAF:*



Let consider the elements that impact the supported status of argument x. Here a single tree must be taken into account:



In this case, argument x must be repeated but in the same branch that leads to x. And here this is a clue of an existing problem: x cannot be supported without itself.

The previous examples give the main ideas for defining the notion of "impacting support tree" for an element of the REBAF:

Definition 10 [23] *Let* $\text{REBAF} = \langle \mathbf{A}, \mathbf{R}, \mathbf{E}, \mathbf{s}, \mathbf{t}, \mathbf{P} \rangle$. *Let x be an element of this REBAF. An* impacting support tree *for x is a set* $\mathbf{IST} = \{x_0, \ldots, x_n\}$ *with* $n > 0$ *s.t.:*

- $\forall x_i, i \in [0 \ldots n]$, $x_i \in (\mathbf{A} \cup \mathbf{E}) \setminus \{x\}$ *and is called a* node *of the tree;*
- *Let* $\mathbf{IST_P} = (\mathbf{IST} \cap \mathbf{P} \cap \mathbf{A})$. $\mathbf{IST_P} \neq \varnothing$*;*
- $!\exists x_i \in \mathbf{IST}$ *s.t.* $x_i \in \mathbf{E}$ *and* $\mathbf{t}(x_i) = x$*; x_i is called the* root *of the tree;*[9]
- $\forall x_i \in \mathbf{IST} \cap \mathbf{A}$, *either* $\exists x_j \in \mathbf{IST} \cap \mathbf{E}$ *s.t.* $x_i = \mathbf{t}(x_j)$, *or* $x_i \in \mathbf{IST_P}$ *(in this case x_i is called a* leaf *of the tree);*
- $\forall x_i \in \mathbf{IST} \cap \mathbf{E}$, $\forall x_j \in \mathbf{s}(x_i)$, $x_j \in \mathbf{IST}$.

Note that an element x cannot belong to its impacting support tree, and by definition repetition is not authorized since an $\mathbf{IST}$ is a set.

Example 3 (cont'd): *There is no* $\mathbf{IST}$ *for any interaction or argument. Indeed the ony element that is the target of a support is e and e has no* $\mathbf{IST}$*, because the support ϵ cannot belong to an* $\mathbf{IST}$ *for e (one argument of its source, c, cannot belong to an* $\mathbf{IST}$*; it is neither prima-facie, nor targeted by a support).* □

Example 4 (cont'd): *In this example, there are 3 non prima-facie elements that are in a DCS: d, e and β. Nevertheless, there exists an* $\mathbf{IST}$ *for d ($\{\gamma, e, \beta, b, c\}$) and another one for e ($\{\beta, b, c\}$) Let now consider β; there is no* $\mathbf{IST}$ *for β since it cannot be supported without itself.* □

Example 5 (cont'd): *In this example, considering the non prima-facie elements, there are an* $\mathbf{IST}$ *for d ($\{\gamma, e\}$) and another for a ($\{\beta_1, d, \gamma, e\}$), but nothing for c, nor b. Indeed, c needs the support of b and b needs the support of c, so its own support that is forbidden by Def. 5.* □

Example 6 (cont'd): *Considering a, there is one* $\mathbf{IST} = \{\alpha_1, a_0\}$. *Considering d, there is one* $\mathbf{IST} = \{\alpha_2, d_0\}$. *Considering e, there are two* $\mathbf{IST}$*, $\{\alpha_3, e_0\}$ and $\{\alpha_4, e'_0\}$. Considering x, there is one* $\mathbf{IST}$*, $\{\beta_1, a, b, \alpha_1, a_0\}$. Considering y, there are two* $\mathbf{IST}$*, $\{\beta_2, c, d, e, \alpha_2, d_0, \alpha_3, e_0\}$ and $\{\beta_2, c, d, e, \alpha_2, d_0, \alpha_4, e'_0\}$. Considering z, there are three* $\mathbf{IST}$*:*

- $\{\gamma_1, x, \beta_1, a, b, \alpha_1, a_0\}$ *(root: γ_1, leaves: a_0 and b),*

[9] $!\exists x_i \in \mathbf{IST}$ means "there exists *only one* $x_i \in \mathbf{IST}$.

- $\{\gamma_2, y, \beta_2, c, d, e, \alpha_2, d_0, \alpha_3, e_0\}$ *(root: γ_2, leaves: c, d_0 and e_0),*
- $\{\gamma_2, y, \beta_2, c, d, e, \alpha_2, d_0, \alpha_4, e'_0\}$ *(root: γ_2, leaves: c, d_0 and e'_0).*

The other elements of the REBAF have no $\mathbf{IST}$. □

Example 7 (cont'd): *In this example, the supports and a have no* $\mathbf{IST}$. *For b, the* $\mathbf{IST}$ *is the set $\{\gamma, a\}$ and for x there are two sets $\{a, b, \beta, \gamma\}$ and $\{a, \alpha\}$.* □

Example 8 (cont'd): *In this example, the supports and a have no* $\mathbf{IST}$. *For b, the* $\mathbf{IST}$ *is the set $\{\beta, a\}$ and for x the* $\mathbf{IST}$ *is $\{a, b, \alpha, \beta\}$.* □

Example 9 (cont'd): *In this example, the supports and a have no* $\mathbf{IST}$. *For x and for b, the same thing occurs but for a complete different reason: for the supports and a, they are not the target of a support; for b and x the non-existence of an* $\mathbf{IST}$ *is due to the fact that these elements cannot be supported without themselves and so an* $\mathbf{IST}$ *cannot be defined.* □

2.3 A Framework with Higher-Order Necessary Supports and Attacks

We recalled here the main definitions and properties concerning RAFN (see [10]), *i.e.* an higher-order bipolar argumentation framework using in the same time the RAF approach introduced in [11] and the necessary meaning for supports[10]

Definition 11 [10] *A* Recursive Argumentation Framework with Necessity (RAFN) *is a tuple $\langle \mathbf{A}, \mathbf{R}, \mathbf{N}, \mathbf{s}, \mathbf{t} \rangle$, where $\mathbf{A}$, $\mathbf{R}$ and $\mathbf{N}$ are three pairwise disjunct sets respectively representing arguments, attacks and necessary supports names, $\mathbf{s}$ is a function from $\mathbf{R} \cup \mathbf{N}$ to $(2^{\mathbf{A}} \setminus \varnothing)$ mapping each interaction to its source and $\mathbf{t}$ is a function from $\mathbf{R} \cup \mathbf{N}$ to $(\mathbf{A} \cup \mathbf{R} \cup \mathbf{N})$ mapping each interaction to its target. It is assumed that $\forall \alpha \in \mathbf{R}, \mathbf{s}(\alpha)$ is a singleton.*

Note that the source of a support in a RAFN is a *set* of arguments, whereas the source of an attack is a singleton (it is a difference with REBAF).

It is worth to notice that correspondences have been provided between frameworks using evidential supports and those using necessary supports when we consider first-order frameworks (see [31]). Nevertheless, in [10], it has been proven that these correspondences cannot be generalized in presence of higher-order interactions. So direct definitions for the semantics of RAFN have been provided in [10], in a similar way as for a REBAF.

The definition for a structure is kept as in Definition 4:

Definition 12 [10] *Let* $\mathbf{RAFN} = \langle \mathbf{A}, \mathbf{R}, \mathbf{N}, \mathbf{s}, \mathbf{t} \rangle$. *A triple $U = (S, \Gamma, \Delta)$ is a* structure *of* $\mathbf{RAFN}$ *iff $S \subseteq \mathbf{A}$, $\Gamma \subseteq \mathbf{R}$ and $\Delta \subseteq \mathbf{N}$. Moreover, for any pair of structures $U = (S, \Gamma, \Delta)$ and $U' = (S', \Gamma', \Delta')$, the definitions of "$U \subseteq U'$" and "$U$ is $\subseteq$-maximal (resp. $\subseteq$-minimal)" are those given in Definition 4.*

Some specific sets of elements w.r.t. a given structure must be redefined in order to take into account the necessary meaning for supports:

Definition 13 [10] *Let* $\mathbf{RAFN} = \langle \mathbf{A}, \mathbf{R}, \mathbf{N}, \mathbf{s}, \mathbf{t} \rangle$ *and given a structure $U = (S, \Gamma, \Delta)$.*

- *The sets of defeated elements w.r.t. U are:*
  $Def_X(U) \stackrel{\text{def}}{=} \{x \in X | \exists \alpha \in \Gamma, \mathbf{s}(\alpha) \in S \text{ and } \mathbf{t}(\alpha) = x\}$ *with* $X \in \{\mathbf{A}, \mathbf{R}, \mathbf{N}\}$
  $Def(U) \stackrel{\text{def}}{=} Def_{\mathbf{A}}(U) \cup Def_{\mathbf{R}}(U) \cup Def_{\mathbf{N}}(U)$
- *The set of supported elements $Sup(U)$ is recursively defined as follows:*
  $Sup(U) \stackrel{\text{def}}{=} \{x | \forall \alpha \in \Delta \text{ st } \mathbf{t}(\alpha) = x, \text{ if } \alpha \in Sup(U \setminus \{x\})$
  then $\mathbf{s}(\alpha) \cap (S \cap Sup(U \setminus \{x\})) \neq \varnothing\}$

[10] Binary necessary support was initially introduced in [28], then discussed in [6, 16, 17] in a more general context (particularly with higher-order interactions in the so-called ASAF).

- *The set of* unactivable *attacks w.r.t. U is:*
  $UnAct(U) \stackrel{\text{def}}{=} \{\alpha \in \mathbf{R} | \alpha \in UnAcc(U)$ *or* $\mathbf{s}(\alpha) \in UnAcc(U)\}$

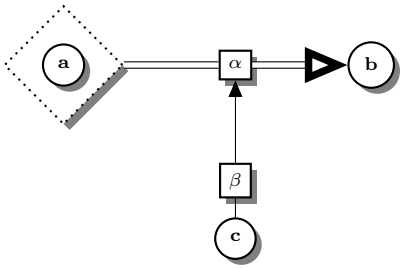
Note that the definitions for the sets of unsupportable elements ($UnSupp(U)$), acceptable elements ($Acc(U)$) and unacceptable elements ($UnAcc(U)$) are similar to the definitions given in Definition 5.

Then using all these definitions, semantics for RAFN can be defined. Note that the definitions for the self-supporting, conflict-free, admissible, complete and grounded structures are similar to the definitions given for REBAF (see Definition 8). So the only difference concerns the preferred and the stable semantics:[11]

Definition 14 [10] *Let* $\mathbf{RAFN} = \langle \mathbf{A}, \mathbf{R}, \mathbf{N}, \mathbf{s}, \mathbf{t} \rangle$ *and given a structure* $U = (S, \Gamma, \Delta)$*, U is*

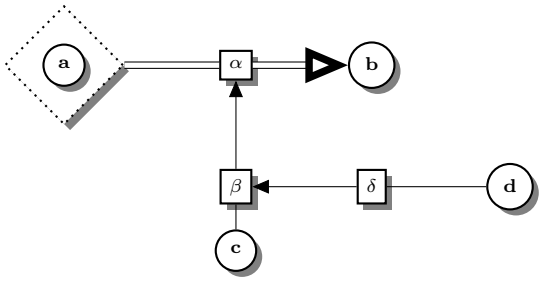
- preferred *iff it is a $\subseteq$-maximal complete structure*
- stable *iff it is complete and* $\overline{(S \cup \Gamma \cup \Delta)} = UnAcc(U)$

Example 1 (cont'd): *The argumentation framework corresponding to the example given in the introduction can be represented as follows (argument names are given in circular nodes, interaction names in square nodes, the source of a collective interaction is given in a "dotted diamond" ; attacks are represented by simple edges and supports by double edges):*



In this framework, neither β, nor its source c are attacked. Moreover they are not the source or the target of a support, so they can be supported by any structure. 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.
Concerning b and a, they are not attacked and, since the support α from a to b is not acceptable, they are also suppoted by any structure. So it is possible to have an admissible structure containing b and not a, or containing a and not b. In this example, there is a unique complete, grounded, preferred and stable structure $U = (\{a, b, c\}, \{\beta\}, \varnothing)$. □

Example 10 *Consider the following example:*



In this framework, β can be unactivable and so α can be acceptable. In this case, when α and b are in the structure, then a must also be in the structure. So we cannot have an admissible structure in which α and b are present, and a is not. In this example, there is a unique complete, grounded, preferred and stable structure $U = (\{a, b, c, d\}, \{\delta\}, \{\alpha\})$.

[11] Indeed, in [10], it has been proven that there is no Fundamental Lemma for RAFN since the function Sup is not monotonic. Due to this point, preferred and stable extensions are assumed to be complete sets.

3 Background on the Logical Description of a REBAF

Here, we recall the logical description of a REBAF proposed in [14] then improved in [23], that allows an explicit representation of arguments, attacks, *evidential supports* and their properties.

3.1 Vocabulary

In [23], the following unary predicate symbols and unary functions symbol are used with the following meaning:

Predicate symbol

- $Arg(x)$: x is an argument
- $Attack(x)$ (resp. $ESupport(x)$): x is an attack (resp. evidential support)
- $PrimaFacie(x)$: x is a prima-facie element
- $Acc(x)$: x is accepted, with $x \in \mathbf{A}$
- $NAcc(x)$: x cannot be accepted, with $x \in \mathbf{A}$
- $Val(\alpha)$: α is valid, with $\alpha \in (\mathbf{R} \cap \mathbf{E})$
- $Supp(x)$: x is supported, with $x \in (\mathbf{A} \cup \mathbf{R} \cap \mathbf{E})$
- $UnSupp(x)$: x is unsupportable, with $x \in (\mathbf{A} \cup \mathbf{R} \cap \mathbf{E})$
- $sAcc(x)$: x is accepted and supported, with $x \in \mathbf{A}$
- $sVal(\alpha)$: α is valid and supported, with $\alpha \in (\mathbf{R} \cap \mathbf{E})$
- $S(a, \alpha)$: "the argument a belongs to the source of α"

Function symbol

- $T(\alpha)$: denotes the target (resp. source) of α, with $\alpha \in (\mathbf{R} \cap \mathbf{E})$

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.

3.2 Logical theory for describing REBAF

In [23], several formulae describing a given **REBAF** are given using the following ideas.

First 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.

Moreover, 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, these constraints have been expressed in [23] by the following formulae:

(1) $\forall x \in (Attack \cup ESupport)\ \forall y \in Attack$
$$\left(\begin{array}{l} (sVal(y) \wedge (T(y) = x) \wedge \forall z \in Arg(S(z,y) \rightarrow sAcc(z))) \\ \rightarrow \neg Val(x) \end{array} \right)$$

Note that the subformula $\forall z \in Arg(S(z,y) \rightarrow sAcc(z))$ means that the source of y can be viewed as accepted and supported (so e-accepted).

(2) $\forall x \in Arg\ \forall y \in Attack$
$$\left(\begin{array}{l} (sVal(y) \wedge (T(y) = x) \wedge \forall z \in Arg(S(z,y) \rightarrow sAcc(z))) \\ \rightarrow NAcc(x) \end{array} \right)$$

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

(1bis) $\forall x \in (Attack \cup ESupport \cup Arg)$
$$\left(\begin{array}{l} \left(\begin{array}{l} PrimaFacie(x) \vee \\ \exists y \in ESupport \\ \quad (sVal(y) \wedge (T(y) = x) \wedge \forall z \in Arg(S(z,y) \rightarrow sAcc(z))) \end{array} \right) \\ \rightarrow Supp(x) \end{array} \right)$$

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

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

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

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

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

Then the logical encoding of specificities of a given **REBAF** leads to the following set of formulae. Let $\mathbf{A} = \{a_1, \ldots a_n\}$, $\mathbf{R} = \{\alpha_1, \ldots, \alpha_k\}$, $\mathbf{E} = \{\alpha_{k+1}, \ldots, \alpha_m\}$ and $\mathbf{P} = \{x_1, \ldots x_l\}$. [12]

[12] We recall that $\mathbf{P} \subseteq \mathbf{A} \cup \mathbf{R} \cup \mathbf{E}$.

(6) $S(a_1,\alpha)\wedge S(a_2,\alpha),\wedge\ldots\wedge S(a_n,\alpha)\wedge(T(\alpha)=b)$, for all $\alpha\in\mathbf{R}\cup\mathbf{E}$ with $\mathbf{s}(\alpha)=\{a_1,a_2,\ldots,a_n\}$ 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 $\alpha_i,\alpha_j\in\mathbf{R}\cup\mathbf{E}$ 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 all the previous formulae. It is obviously consistent.

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 [23], 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 $\Sigma(\textbf{REBAF})$ (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 [23], 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)

$$\forall x\in(Attack\cup ESupport\cup Arg)\left(\begin{array}{l}Supp(x)\\ \rightarrow\left(\begin{array}{l}PrimaFacie(x)\vee\\ \exists y\in ESupport\\ \quad(sVal(y)\wedge(T(y)=x)\wedge\forall z\in Arg(S(z,y)\rightarrow sAcc(z)))\end{array}\right)\end{array}\right)$$

- 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)

$$\forall x \in (Attack \cup ESupport \cup Arg)$$
$$\left(\begin{array}{l} UnSupp(x) \\ \leftrightarrow \left(\begin{array}{l} \neg PrimaFacie(x) \wedge \\ \forall y \in ESupport(T(y) = x \\ \quad \rightarrow \left(\begin{array}{l} \exists \beta \in Attack((T(\beta) = y \vee \exists z \in Arg(S(z,y) \wedge T(\beta) = z)) \wedge \\ \quad sVal(\beta) \wedge \forall z \in Arg(S(z,\beta) \rightarrow sAcc(z)))) \\ \vee \exists z \in Arg(S(z,y) \wedge UnSupp(z)) \\ \vee UnSupp(y)) \end{array} \right) \end{array} \right) \end{array} \right)$$

Note that the subformula $\exists z \in Arg(S(z,y) \wedge UnSupp(z))$ means that the source of y can be viewed as unsupportable. Moreover the suformula $(T(\beta) = y \vee \exists z \in Arg(S(z,y) \wedge T(\beta) = z))$ means that $T(\beta)$ is either y or belongs to the source of y.

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

3.3.3 Defence

As stated in Definition 7, 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 [23], the principle corresponding to the previous item *(ii)* has been expressed by the following formulae that are associated with formulae **(17)** and **(18)**:

(11)

$$\forall \alpha \in Attack$$
$$\left(\begin{array}{l} Acc(T(\alpha)) \\ \rightarrow \left(\begin{array}{l} \exists \beta \in Attack \\ \quad ((T(\beta) = \alpha \vee \exists z \in Arg(S(z,\alpha) \wedge T(\beta) = z)) \wedge sVal(\beta) \wedge \forall z \in Arg(S(z,\beta) \rightarrow sAcc(z))) \\ \vee \exists z \in Arg(S(z,\alpha) \wedge UnSupp(z)) \\ \vee UnSupp(\alpha) \end{array} \right) \end{array} \right)$$

(12)

$$\forall \alpha \in Attack\, \forall \delta \in (Attack \cup ESupport)$$
$$\left(\begin{array}{l} ((\delta = T(\alpha)) \wedge Val(\delta)) \\ \rightarrow \left(\begin{array}{l} \exists \beta \in Attack \\ \quad ((T(\beta) = \alpha \vee \exists z \in Arg(S(z,\alpha) \wedge T(\beta) = z)) \wedge sVal(\beta) \wedge \forall z \in Arg(S(z,\beta) \rightarrow sAcc(z))) \\ \vee \exists z \in Arg(S(z,\alpha) \wedge UnSupp(z)) \\ \vee UnSupp(\alpha) \end{array} \right) \end{array} \right)$$

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

3.3.4 Reinstatement

In [23], the reinstatement principle has been expressed by the following formulae that are be associated with formulae **(17)** and **(18)**:

(13)

$$\forall c \in Arg \left(\left(\left(\forall \alpha \in Attack \left(\left(\left(T(\alpha) = c \rightarrow \left(\left(\begin{array}{l} \exists \beta \in Attack((T(\beta) = \alpha \vee \exists z \in Arg(S(z,\alpha) \wedge T(\beta) = z)) \wedge \\ \quad sVal(\beta) \wedge \forall z \in Arg(S(z,\beta) \rightarrow sAcc(z))) \\ \vee \exists z \in Arg(S(z,\alpha) \wedge UnSupp(z)) \\ \vee UnSupp(\alpha) \end{array} \right) \right) \right) \right) \right) \right) \rightarrow Acc(c) \right)$$

(14)

$$\forall \delta \in (Attack \cup ESupport) \left(\left(\left((\forall \alpha \in Attack \left(\left(\left(T(\alpha) = \delta \rightarrow \left(\left(\begin{array}{l} \exists \beta \in Attack((T(\beta) = \alpha \vee \exists z \in Arg(S(z,\alpha) \wedge T(\beta) = z)) \wedge \\ \quad sVal(\beta) \wedge \forall z \in Arg(S(z,\beta) \rightarrow sAcc(z))) \\ \vee \exists z \in Arg(S(z,\alpha) \wedge UnSupp(z)) \\ \vee UnSupp(\alpha) \end{array} \right) \right) \right) \right) \right) \right) \rightarrow Val(\delta) \right)$$

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

3.3.5 Stability

In [23], the stability principle has been expressed by the three following formulae that are associated with formulae **(17)** and **(18)**:[13]

(15) $\forall c \in Arg \left(\neg Acc(c) \rightarrow \left(\begin{array}{l} \exists \beta \in Attack(T(\beta) = c \wedge \\ \quad sVal(\beta) \wedge \forall z \in Arg(S(z,\beta) \rightarrow sAcc(z))) \end{array} \right) \right)$

(16) $\forall \alpha \in (Attack \cup ESupport) \left(\neg Val(\alpha) \rightarrow \left(\begin{array}{l} \exists \beta \in Attack(T(\beta) = \alpha \wedge \\ \quad sVal(\beta) \wedge \forall z \in Arg(S(z,\beta) \rightarrow sAcc(z))) \end{array} \right) \right)$

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

$Supp(b) \rightarrow (sAcc(a) \wedge sVal(\alpha))$
$\neg UnSupp(a)$
$\neg UnSupp(c)$
$\neg UnSupp(\alpha)$

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

$\neg UnSupp(\beta)$

$$Unsupp(b) \leftrightarrow \left(\begin{array}{l} (sVal(\beta) \wedge sAcc(c)) \\ \vee\ UnSupp(a) \\ \vee\ UnSupp(\alpha) \end{array} \right)$$

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

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

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

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

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

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

3.4 Characterizing Semantics of a REBAF

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

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

- $S_\mathcal{I} = \{x \in \mathbf{A} | \mathcal{I}(sAcc(x)) = \textit{true}\}$
- $\Gamma_\mathcal{I} = \{x \in \mathbf{R} | \mathcal{I}(sVal(x)) = \textit{true}\}$
- $\Delta_\mathcal{I} = \{x \in \mathbf{E} | \mathcal{I}(sVal(x)) = \textit{true}\}$

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

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

Let recall that the existence of support cycles leads to the following definition in order to avoid the models in which some elements could be supported only with themselves:

Definition 15 *Let* $\mathbf{REBAF} = \langle \mathbf{A}, \mathbf{R}, \mathbf{E}, \mathbf{s}, \mathbf{t}, \mathbf{P} \rangle$. *$\mathcal{I}$ 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 s.t. $\mathcal{I}(sAcc(x)) =$ true (resp. $\mathcal{I}(sVal(x)) =$ true), there exists at least one impacting support tree $\mathbf{IST} = \{x_0, \ldots, x_n\}$ for x that is satisfied by $\mathcal{I}$, i.e. $\forall x_i \in \mathbf{IST}$, if $x_i \in \mathbf{A}$ then $\mathcal{I}(sAcc(x_i)) =$ true, otherwise $\mathcal{I}(sVal(x_i)) =$ true;*
2. *for each element x of* $\mathbf{REBAF}$, *$\mathcal{I}(UnSupp(x)) =$ true iff $x \in UnSupp(U_\mathcal{I})$ with $U_\mathcal{I} = (S_\mathcal{I}, \Gamma_\mathcal{I}, \Delta_\mathcal{I})$.*

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

Then using these support-founded models, the following characterization of REBAF semantics is given in [23]:

Proposition 1 **[23]** *Let* $\mathbf{REBAF} = \langle \mathbf{A}, \mathbf{R}, \mathbf{E}, \mathbf{s}, \mathbf{t}, \mathbf{P} \rangle$. *Let $U = (S, \Gamma, \Delta)$ be a structure on* **REBAF**.

1. *U is admissible iff there exists a support-founded model $\mathcal{I}$ of $\Sigma_d(\mathbf{REBAF})$ (in the sense of Def. 15) with $S_\mathcal{I} = S$, $\Gamma_\mathcal{I} = \Gamma$ and $\Delta_\mathcal{I} = \Delta$.*
2. *U is complete iff there exists a support-founded model $\mathcal{I}$ of $(\Sigma_d(\mathbf{REBAF}) \cup \Sigma_r(\mathbf{REBAF}))$ (in the sense of Def. 15) with $S_\mathcal{I} = S$, $\Gamma_\mathcal{I} = \Gamma$ and $\Delta_\mathcal{I} = \Delta$.*
3. *U is a preferred structure iff there exists a $\subseteq$-maximal support-founded model $\mathcal{I}$ of $\Sigma_d(\mathbf{REBAF})$ (in the sense of Def. 15) 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 $(\Sigma_d(\mathbf{REBAF}) \cup \Sigma_r(\mathbf{REBAF}))$ (in the sense of Def. 15).*
5. *U is stable iff there exists a support-founded model $\mathcal{I}$ of $\Sigma_s(\mathbf{REBAF})$ (in the sense of Def. 15) 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 3 (cont'd): *The $\subseteq$-maximal support-founded models $\mathcal{I}$ of $\Sigma_d(\mathbf{REBAF})$ (in the sense of Def. 15) correspond to the preferred structure $(\{b, d\}, \{\beta, \pi\}, \{\epsilon\})$.* □

Example 4 (cont'd): *The $\subseteq$-maximal support-founded models $\mathcal{I}$ of $\Sigma_d(\mathbf{REBAF})$ (in the sense of Def. 15) correspond to the preferred structure $(\{a, b, c\}, \varnothing, \{\alpha, \gamma\})$. The models that satisfy $sAcc(d)$ (resp. $sAcc(e)$, $sVal(\beta)$) are not kept since they are not support-founded (no* **IST** *for these arguments or this interaction that could be satisfied).* □

Example 5 (cont'd): *The $\subseteq$-maximal support-founded models $\mathcal{I}$ of $\Sigma_d(\mathbf{REBAF})$ (in the sense of Def. 15) correspond to the preferred structure $(\{e, d, a\}, \varnothing, \{\alpha_1, \alpha_2, \beta_1, \beta_2, \gamma\})$.* □

Example 6 (cont'd): *The $\subseteq$-maximal support-founded models $\mathcal{I}$ of $\Sigma_d(\mathbf{REBAF})$ (in the sense of Def. 15) correspond to the preferred structure $(\mathbf{A}, \mathbf{R}, \mathbf{E})$.* □

4 Towards a new version of RAFN

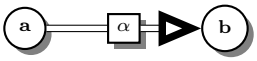
4.1 Analysis of the RAFN behaviour

Before to encode logically a RAFN we must identify more precisely its behaviour.

Of course, some results concerning REBAF can be reused since some definitions are common. For instance, let $\mathbf{RAFN} = \langle \mathbf{A}, \mathbf{R}, \mathbf{N}, \mathbf{s}, \mathbf{t} \rangle$, let U and U' be 2 structures of **RAFN**, if $U \subseteq U'$ then $Def(U) \subseteq Def(U')$.[14]

Nevertheless many other results cannot be reused due to the differences concerning the support relation. Indeed, consider the following very simple example.

Example 11 *In this example, we have two arguments a and b with a necessary support α from a to b.*



In this case, consider three different and admissible structures: $U_1 = (\varnothing, \varnothing, \varnothing)$, $U_2 = (\varnothing, \varnothing, \{\alpha\})$ and $U_3 = (\{a\}, \varnothing, \{\alpha\})$ Then we have:[15]

[14] First part of Lemma A.1 in [9], long version of [8]

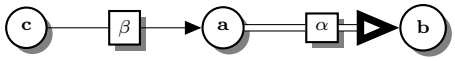
[15] Indeed, considering any structure, each element that is not a target of a support belonging to the structure is always supported by this structure; and an element that is the target of a support belonging to the structure is supported only if the source of the support is also in the structure.

- $Sup(U_1) = \{a, b, \alpha\}$
- $Sup(U_2) = \{a, \alpha\}$ *(b cannot belong to the set of supported elements by the structure since the support α is in the structure and its source a is not)*
- $Sup(U_3) = \{a, b, \alpha\}$

So $U \subseteq U'$ does not imply that $Sup(U) \subseteq Sup(U')$.[16] This non-monotonicity of the Sup relation implies the fact that there is no fundamental lemma for RAFN (see in [10]).

Another negative result is the following: in the general case and even if U is conflict-free and self-supporting, $Acc(U) \not\subseteq \overline{UnAcc(U)}$.[17] Indeed consider the following example.

Example 12 *In this example, we have three arguments a, b and c with a necessary support α from a to b and an attack β from c to a.*



Then consider the admissible structure $U = (\{b, c\}, \{\beta\}, \varnothing)$: in this case,

- $Sup(U) = \{a, b, c, \alpha, \beta\}$ *and* $Def(U) = \{a\}$,
- *so* $U' = (\overline{Def_{\mathbf{A}}(U)}, \mathbf{R}, \overline{Def_{\mathbf{N}}(U)}) = (\{b, c\}, \{\beta\}, \{\alpha\})$ *and* $UnSupp(U) = \{b\}$;
- *then* $UnAcc(U) = \{a, b\}$ *and* $UnAct(U) = \varnothing$;
- *this implies that* $Acc(U) = \{b, c, \alpha, \beta\}$ *(since β is not unactivable, a cannot be acceptable).*

An important point appears here: there may exist some structures U such that $Sup(U) \cap UnSupp(U) \neq \varnothing$ and $Acc(U) \cap UnAcc(U) \neq \varnothing$. So for a given structure it may exist an element that can be in the same time supported/acceptable and unsupportable/unacceptable by the structure.

This point is not a blocking point for the semantics computation since a such element is detected and so removed when we compare a structure and its acceptable elements for the admissibility property (in the previous example the structure $U = (\{b, c\}, \{\beta\}, \varnothing)$ is included in $Acc(U) = \{b, c, \alpha, \beta\}$ but the structure $U_1 = (\{b, c\}, \{\beta\}, \{\alpha\})$ is not included in $Acc(U_1) = \{c, \alpha, \beta\}$, so U is admissible and U_1 is not). Nevertheless it is a big problem when we try to encode with a classical logic such a behaviour.

Perhaps, it could be interesting to identify in which cases this happens.

Example 12 (cont'd): *Consider now the admissible structure $U = (\{b\}, \varnothing, \varnothing)$: in this case,*

- $Sup(U) = \{a, b, c, \alpha, \beta\}$ *and* $Def(U) = \varnothing$,
- *so* $U' = (\overline{Def_{\mathbf{A}}(U)}, \mathbf{R}, \overline{Def_{\mathbf{N}}(U)}) = (\{a, b, c\}, \{\beta\}, \{\alpha\})$ *and* $UnSupp(U) = \varnothing$;
- *then* $UnAcc(U) = \varnothing$ *and* $UnAct(U) = \varnothing$;
- *this implies that* $Acc(U) = \{b, \alpha, \beta\}$.

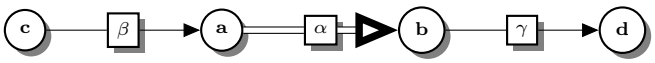
In this case, we have $Sup(U) \cap UnSupp(U) = \varnothing$ and $Acc(U) \cap UnAcc(U) = \varnothing$. □

Here the main difference between these 2 cases is the fact that, in the first case, we have in the structure a way to invalidate the source of the support, using an attack targeting this source. Nevertheless, it is not so obvious to identify precisely in *all possible cases* what is the reason explaining the fact that an element could be at the same time supported and unsupportable. Moreover these elements can also be at the same time acceptable and unacceptable but they are not the only ones. The following example illustrates this fact.

[16] So the second part of Lemma A.1 in [9], long version of [8] cannot be used in the case of RAFN. And it is the same for Lemma A.3 in [9].

[17] So Lemma A.7 in [9], long version of [8] cannot be applied for RAFN.

Example 13 *In this example, we have four arguments a, b, c and d with a necessary support α from a to b and two attacks, β from c to a and γ from b to d.*



Then consider the admissible structure $U = (\{b, c\}, \{\beta, \gamma\}, \varnothing)$: in this case,

- *$Sup(U) = \{a, b, c, d, \alpha, \beta, \gamma\}$ and $Def(U) = \{a, d\}$,*
- *so $U' = (\overline{Def_{\mathbf{A}}(U)}, \mathbf{R}, \overline{Def_{\mathbf{N}}(U)}) = (\{b, c\}, \{\beta, \gamma\}, \{\alpha\})$ and $UnSupp(U) = \{b\}$;*
- *then $UnAcc(U) = \{a, b, d\}$ and $UnAct(U) = \{\gamma\}$;*
- *this implies that $Acc(U) = \{b, c, d, \alpha, \beta, \gamma\}$ (since β is not unactivable, a cannot be acceptable and since γ is unactivable then d is acceptable).*

So $Sup(U) \cap UnSupp(U) = \{b\}$ and $Acc(U) \cap UnAcc(U) = \{b, d\}$. Here the fact that b can be at the same time supported and unsupportable has an impact on the status of d that becomes at the same time acceptable and unacceptable.

Another interesting point in this example is the following: $Acc(U) \cap Def(U) \neq \varnothing$ whereas U is conflict-free and self-supporting. This shows that Lemma A.2 given in [9] for REBAF cannot be applied for RAFN. And once again that seems to be a side effect of the existence of b as a supported and unsupportable element.

So, in conclusion, the RAFN defined in [10] is clearly a problematic framework when we are searching a logical encoding. So in this paper, we propose a new version of RAFN in order to avoid all these problems.

4.2 A new framework for RAFN

The previous examples clearly show that the problem is in the definition of the $UnSupp$ set. Recall that, in the framework proposed in [10], this set is defined as follows:

Definition 16 *Let* $\mathbf{RAFN} = \langle \mathbf{A}, \mathbf{R}, \mathbf{N}, \mathbf{s}, \mathbf{t} \rangle$. *Let U be a structure of* **RAFN**. *The set of* unsupportable *elements w.r.t. U is:*

$$UnSupp(U) \stackrel{\text{def}}{=} \overline{Sup(U')} \text{ with } U' = (\overline{Def_{\mathbf{A}}(U)}, \mathbf{R}, \overline{Def_{\mathbf{N}}(U)})$$

So we propose here a very simple solution:

Definition 17 *Let* $\mathbf{RAFN} = \langle \mathbf{A}, \mathbf{R}, \mathbf{N}, \mathbf{s}, \mathbf{t} \rangle$. *Let U be a structure of* **RAFN**. *The set of* unsupportable *elements w.r.t. U is:*

$$UnSupp(U) \stackrel{\text{def}}{=} \overline{Sup(U')} \setminus Sup(U) \text{ with } U' = (\overline{Def_{\mathbf{A}}(U)}, \mathbf{R}, \overline{Def_{\mathbf{N}}(U)})$$

The impact of this definition is illustrated with the following example:

Example 13 (cont'd): *In this example, with the new definition of the $UnSupp$ set and always with the admissible structure $U = (\{b, c\}, \{\beta, \gamma\}, \varnothing)$, we have the following results:*

- *$Sup(U) = \{a, b, c, d, \alpha, \beta, \gamma\}$ and $Def(U) = \{a, d\}$, (unchanged since the $UnSupp$ set is not used for defining the Sup and the Def sets)*
- *so $U' = (\overline{Def_{\mathbf{A}}(U)}, \mathbf{R}, \overline{Def_{\mathbf{N}}(U)}) = (\{b, c\}, \{\beta, \gamma\}, \{\alpha\})$ (unchanged) and $UnSupp(U) = \varnothing$ (changed of course);*
- *then $UnAcc(U) = \{a, d\}$ and $UnAct(U) = \varnothing$ (changed also);*
- *this implies that $Acc(U) = \{b, c, \alpha, \beta, \gamma\}$ (now γ is not unactivable so d cannot be acceptable).*

So $Sup(U) \cap UnSupp(U) = \varnothing$ but also $Acc(U) \cap UnAcc(U) = \varnothing$ and $Acc(U) \cap Def(U) = \varnothing$. □

So the only difference with the framework proposed in [10] is the fact that an element cannot be in the same time supported and unsupportable or acceptable and unacceptable.

Knowing that all other definitions remain unchanged, it would be interesting to verify if the results given by the semantics remain also unchanged. A first answer can be found through the following example:

Example 13 (cont'd): *Consider now another structure: $U = (\{b, c, d\}, \{\beta\}, \varnothing)$, we have the following results:*

- *$Sup(U) = \{a, b, c, d, \alpha, \beta, \gamma\}$ and $Def(U) = \{a\}$,*
- *so $U' = (\overline{Def_{\mathbf{A}}(U)}, \mathbf{R}, \overline{Def_{\mathbf{N}}(U)}) = (\{b, c, d\}, \{\beta, \gamma\}, \{\alpha\})$*
- *using Def 16, $UnSupp(U) = \{b\}$, whereas with Def 17, $UnSupp(U) = \varnothing$;*
- *so, with Def 16, $UnAcc(U) = \{a, b\}$ and $UnAct(U) = \{\gamma\}$, whereas, with Def 17, $UnAcc(U) = \{a\}$ and $UnAct(U) = \varnothing$;*
- *this implies that, with Def 16, $Acc(U) = \{b, c, d, \alpha, \beta, \gamma\}$, whereas, with Def 17, $Acc(U) = \{b, c, \alpha, \beta, \gamma\}$.*

So considering U, we can conclude that U is admissible if Def 16 is used, but it is not admissible if Def 17 is used.

Nevertheless, if we consider the complete semantics, we can see that the Acc set obtained with Def 16 being not conflict-free, it cannot be a complete structure, whereas that obtained with Def 17 will be a complete structure. □

And indeed the following proposition holds:

Proposition 2 *Let $\mathbf{RAFN} = \langle \mathbf{A}, \mathbf{R}, \mathbf{N}, \mathbf{s}, \mathbf{t} \rangle$. Let U be a structure of $\mathbf{RAFN}$. Let σ be one semantics among the following ones: conflict-free, self-supporting, complete, preferred, grounded and stable. U is a σ-structure obtained using Definition 16 for the $UnSupp(U)$ set iff U is a σ-structure obtained using Definition 17 for the $UnSupp(U)$ set.*

The new version of the RAFN framework will be called RAFN-v2.

5 Logical encoding for RAFN-v2

Considering the logical translation of a REBAF, and the fact that the similarities between the definitions of semantics for REBAF and for RAFN, we propose to adapt the previous encoding in order to take into account higher-order bipolar argumentation frameworks using the necessary meaning for the supports.

5.1 Vocabulary

The first adaptation concerns the vocabulary. It is almost unchanged, except that the predicate $ESupport$ is obviously replaced by the predicate $NSupport$ and the predicate $PrimaFacie$ is removed since it becomes useless. So we have:

Predicate symbol

- $Arg(x)$, $Attack(x)$, $Acc(x)$, $NAcc(x)$, $Val(\alpha)$, $Supp(x)$, $UnSupp(x)$, $sAcc(x)$, $sVal(\alpha)$, $S(a, \alpha)$: unchanged (see in Section 3)
- $ESupport(x)$, $PrimaFacie(x)$: removed
- $NSupport(x)$ means that "x is a necessary support"

Function symbol

- $T(\alpha)$: unchanged (see in Section 3)

5.2 Formulae

The second adaptation concerns the differences between necessary and evidential supports and their impact on the formulae encoding the framework and the semantics. These differences can be synthetized as follows:

- First, in the RAFN case, an element is supported iff, for *each* support targeting this element and belonging to the structure, *at least one* element of the source of this support is also supported by the structure. Whereas, in the REBAF case, an element is supported iff, for *at least one* support targeting this element and belonging to the structure, *each* element of the source of this support is also supported by the structure.
- Secondly, in the RAFN case, an attack is unactivable iff it is unacceptable or its source is *included in* the set of unacceptable elements. Whereas, in the REBAF case, an attack is unactivable iff it is unacceptable or *at least one element of* its source *belongs to* the set of unacceptable elements.
- Thirdly, the source of an attack is a singleton. This last point could induce some simplification in the formulae. Nevertheless, here we do not take it into account, since it would imply to distinguish between a source of an attack and a source of a support and so a modification of the vocabulary that is not really necessary.

A consequence of these différences is the fact that the source of a support, that is a set of arguments, is in the same time accepted and supported if *at least one* element in this set is in the same time accepted and supported. And we have of course the dual situation for the unsupportable status: the source of a support is unsupportable if *each* element of this set is unsupportable.

So the previous points lead to the following evolution of the formulae for the encoding of any **RAFN**.

Formula **(1)** is unchanged except that $NSupport$ is used in place of $ESupport$ (see Section 3)

Formula **(2)** is unchanged (see Section 3)

Formula **(3)** is unchanged (see Section 3)

Formula **(1bis)** becomes:

$$\forall x \in (Attack \cup NSupport \cup Arg)$$
$$\left(\begin{array}{l} \left(\begin{array}{l} \forall y \in NSupport \\ \quad ((sVal(y) \wedge (T(y) = x)) \rightarrow \exists z \in Arg(S(z, y) \wedge sAcc(z))) \end{array} \right) \\ \rightarrow Supp(x) \end{array} \right)$$

Formula **(2bis)** is unchanged (see Section 3)

Formula **(3bis)** is unchanged except that $NSupport$ is used in place of $ESupport$ (see Section 3)

Formula **(4)** is unchanged (see Section 3)

Formula **(4bis)** is unchanged except that $NSupport$ is used in place of $ESupport$ (see Section 3)

Formula **(4ter)** is unchanged except that $NSupport$ is used in place of $ESupport$ (see Section 3)

Formula **(5)** is unchanged except that $NSupport$ is used in place of $ESupport$ (see Section 3)

Then the logical encoding of specificities of a given **RAFN** leads to the following set of formulae. Let $\mathbf{A} = \{a_1, \ldots a_n\}$, $\mathbf{R} = \{\alpha_1, \ldots, \alpha_k\}$, $\mathbf{N} = \{\alpha_{k+1}, \ldots, \alpha_m\}$.

Formula **(6)** is unchanged (see Section 3)

Formula **(7)** is unchanged (see Section 3)

Formula **(8)** is unchanged (see Section 3)

Formula **(8bis)** is unchanged except that $NSupport$ is used in place of $ESupport$ (see Section 3)

Formula **(8ter)** is removed

Formula **(9)** is unchanged (see Section 3)

Formula **(10)** is unchanged (see Section 3)

Let consider now the encoding of the principles used in RAFN semantics.

Principle for conflict-freeness: As for REBAF, the conflict-freeness is already expressed by the formulae **(1)**, **(2)**, **(3)**, **(2bis)**, **(3bis)**.

Principle for self-supporting: For each support that is accepted and supported and that targets a supported element, at least one element of its source must be accepted and supported. And of course, elements that are unsupportable cannot be supported. That leads to the following evolution of formulae **(17)** and **(18)**:

Formula **(17)**:

$$\forall x \in (Attack \cup NSupport \cup Arg)$$
$$\big(\ Supp(x) \rightarrow \big(\ \forall y \in NSupport \big(\ (sVal(y) \wedge (T(y) = x)) \rightarrow \exists z \in Arg(S(z,y) \wedge sAcc(z))\ \big)\ \big)\ \big)$$

Formula **(18)**:

$$\forall x \in (Attack \cup NSupport \cup Arg)$$
$$\left(\begin{array}{l} UnSupp(x) \\ \\ \leftrightarrow \neg Supp(x) \wedge \left(\begin{array}{l} \exists y \in NSupport((T(y) = x) \\ \quad \wedge (\forall \alpha \in Attack((T(\alpha) = y \wedge sVal(\alpha)) \rightarrow \exists u \in Arg(S(u,\alpha) \wedge \neg sAcc(u)))) \\ \quad \wedge \neg UnSupp(y) \\ \quad \wedge \left(\begin{array}{l} \forall z \in Arg(S(z,y) \rightarrow \left(\begin{array}{l} \exists \beta \in Attack((T(\beta) = z) \wedge sVal(\beta) \wedge \\ \quad \forall u \in Arg(S(u,\beta) \rightarrow sAcc(u))) \end{array}\right) \\ \vee UnSupp(z) \end{array}\right) \end{array}\right) \end{array}\right)$$

Principle for defence: The formulae **(11)** and **(12)** related to this principle remain unchanged except that $NSupport$ is used in place of $ESupport$ (see Section 3).

Principle for reinstatement: The formulae **(13)** and **(14)** related to this principle remain unchanged except that $NSupport$ is used in place of $ESupport$ (see Section 3).

Principle for stability: The formulae **(15)**, **(16)** and **(19)** related to this principle remain unchanged except that $NSupport$ is used in place of $ESupport$ (see Section 3).

The logical bases for RAFN: The definitions of the 5 logical bases are similar to the ones given for REBAF (see Section 3). So we have:

- $\Sigma(\mathbf{RAFN}) = \{(\mathbf{1}), \ldots, (\mathbf{10})\}$
- $\Sigma_{ss}(\mathbf{RAFN}) = \Sigma(\mathbf{RAFN}) \cup \{(\mathbf{17}), (\mathbf{18})\}$
- $\Sigma_{d}(\mathbf{RAFN}) = \Sigma_{ss}(\mathbf{RAFN}) \cup \{(\mathbf{11}), (\mathbf{12})\}$
- $\Sigma_{r}(\mathbf{RAFN}) = \Sigma_{ss}(\mathbf{RAFN}) \cup \{(\mathbf{13}), (\mathbf{14})\}$
- $\Sigma_{s}(\mathbf{RAFN}) = \Sigma_{ss}(\mathbf{RAFN}) \cup \{(\mathbf{15}), (\mathbf{16}), (\mathbf{19})\}$

5.3 New definitions

Considering the definition of the supported elements, we can see that, in the case of RAFN as in the case of REBAF, the use of the logical bases is not sufficient for avoiding some problematic models when support cycles exist.

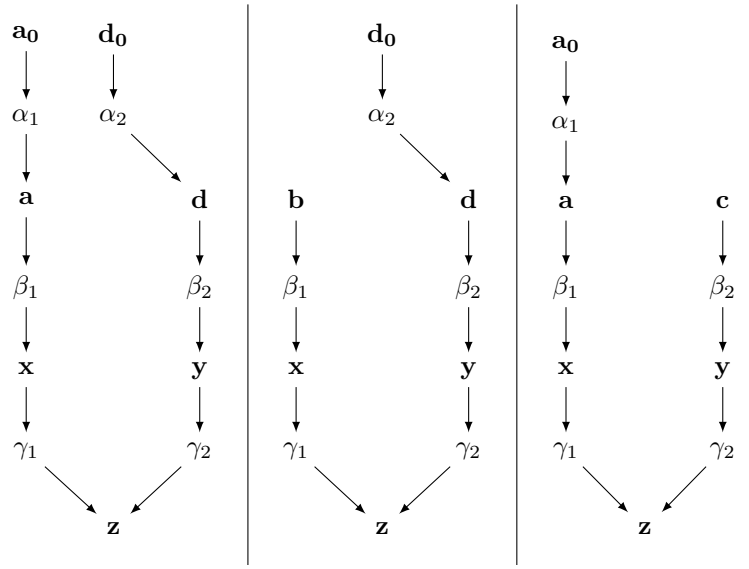
Even if the definition of such cycles can be kept the same, we must adapt the definition of support-founded models and consequently the definition of "impacting support elements" since the behaviour of the necessary supports is different to that of evidential supports.

This can be illustrated with the following example.

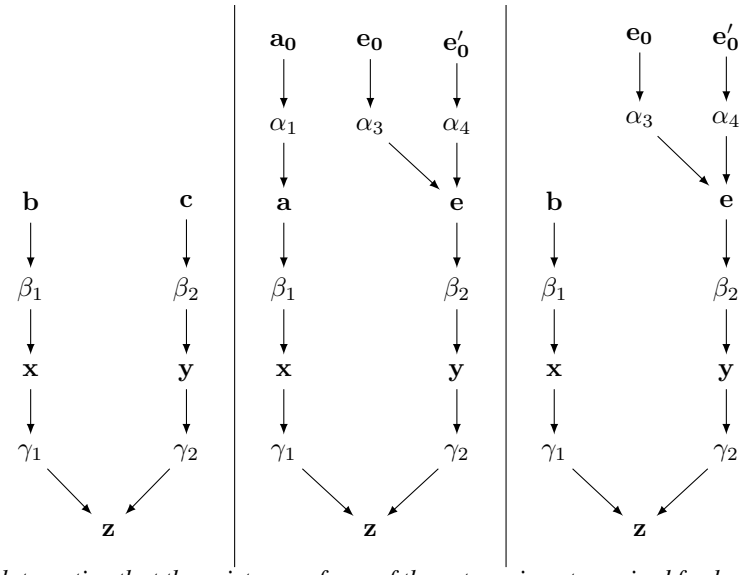
Example 6 (cont'd): *Using the definitions related to RAFN, the constraints related to the necessary supports induce the following results:*

- *Consider for instance z: z will be supported by a structure iff, for any support targeting z and belonging to the structure, at least an element of its source is also present in the structure; that means that if γ_1 (resp. γ_2) is in the structure then x (resp. y) must also be in the structure and if γ_1 (resp. γ_2) is not in the structure then there is no constraint about x (resp. y): it can be in the structure or not.*
- *Consider now x: x will be supported iff, if β_1 is in the structure then a* or *b are also in the structure.*

So if we want to identify the "impacting support elements", we also have trees but not the same trees as the ones proposed for the REBAF. For instance, considering z, there exist 6 trees giving the "impacting support elements" concerning z.[18]



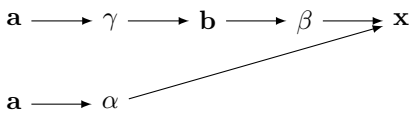
[18]Note that other trees are possible. For instance, the tree corresponding to the aggregation of the first and the second trees is also a possibility (in this case, for β_1, the two elements of the source are kept).



Moreover, it is worth to notice that the existence of one of these trees is not required for having z supported. Indeed if γ_1 and *γ_2 are not in the structure, z is obviously supported by the structure.* □

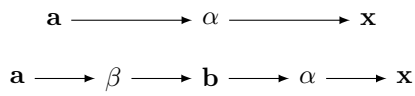
Concerning the possible repetitions, the behaviour for RAFN is once again different that the one for REBAF.

Example 7 (cont'd): *In this case, only one tree must be considered with the repetition of argument a:*



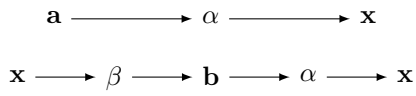
□

Example 8 (cont'd): *In this case, two distinct trees must be considered without any repetition:*



□

Example 9 (cont'd): *In this case, two distinct trees must be considered:*



The repetition in the second tree illustrates the fact that this tree cannot be used in order to show that x is supported. □

In the light of the previous example, we see that the definition and the use of such trees for RAFN are completely different from those for REBAF. So we propose a new definition for the notion of "impacting support tree" for an element of the RAFN:

Definition 18 *Let* $\mathbf{RAFN} = \langle \mathbf{A}, \mathbf{R}, \mathbf{N}, \mathbf{s}, \mathbf{t} \rangle$. *Let x be an element of this RAFN. An* impacting support tree *for x is a set* $\mathbf{IST} = \{x\} \cup \mathbf{IST}'$ *s.t.:*

- *x is called the* root *of the tree;*
- $\mathbf{IST}' = \varnothing$ *iff* $\nexists y \in \mathbf{N}$ *s.t.* $\mathbf{t}(y) = x$; *otherwise* $\mathbf{IST}' = \{x_0, \ldots, x_n\}$ *with* $n > 0$ *and* $\forall x_i, i \in [0 \ldots n]$, $x_i \in (\mathbf{A} \cup \mathbf{N}) \setminus \{x\}$ *and is called a* node *of the tree;*

- $\forall y \in \mathbf{N}$, *if* $x = \mathbf{t}(y)$ *then* $y \in \mathbf{IST}'$;[19]
- $\forall x_i \in (\mathbf{IST}' \cap \mathbf{A})$, *if* $\nexists x_j \in \mathbf{N}$ *s.t.* $x_i = \mathbf{t}(x_j)$ *then* x_i *is called a* leaf *of the tree; otherwise,* $\forall x_j \in \mathbf{N}$, *if* $x_i = \mathbf{t}(x_j)$ *then* $x_j \in \mathbf{IST}'$;
- $\forall x_i \in (\mathbf{IST}' \cap \mathbf{N})$, $\exists x_j \neq x, x_j \in \mathbf{s}(x_i)$ *such that* $x_j \in \mathbf{IST}'$.

Note that, contrary to the case for REBAF, an element x must belong to its impacting support tree. So in order to avoid the problems due to the support cycles, the selection of x as a source of a support is not authorized.

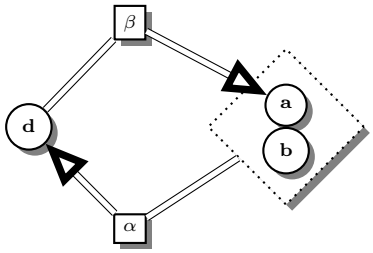
Example 3 (cont'd): *There are at least two* $\mathbf{IST}$ *for* e*:* $\{e, \epsilon, b\}$ *and* $\{e, \epsilon, c\}$ *(and a third one by aggregation). And for all the other elements of this RAFN, each of them has only one* $\mathbf{IST}$ *reduced to a singleton (so* $\{b\}$ *for* b*,* $\{c\}$ *for* c*, . . .).* □

Example 4 (cont'd): *Here, contrary to the REBAF case, there is one* $\mathbf{IST}$ *for* β*:* $\{\beta, \alpha, a\}$*. It cannot exist an* $\mathbf{IST}$ *for* β *using* d *since* d *is supported by* γ *whose source* e *is in turn supported by* β*, so a repetition that is prohibited by Def. 18. Nevertheless,* d *itself has at least two* $\mathbf{IST}$*:* $\{d, \gamma, e, \beta, b\}$ *and* $\{d, \gamma, e, \beta, c\}$ *.* □

Example 5 (cont'd): *Once again, the result for RAFN is different of that for REBAF. In the case of RAFN, there is no* $\mathbf{IST}$ *for* d*; indeed, even if one branch of an* $\mathbf{IST}$ *for* d *could be composed with* γ *and* e*, the other branch concerning the support* α_1 *cannot be built since, whatever we choose between* a *or* b*, we obtain a branch in which a repetition should appear. And the same thing occurs for* a*,* b *and* c*.*

In this example, only e *and the supports have an* $\mathbf{IST}$*, each time reduced to a singleton (so* $\{e\}$ *for* e*,* $\{\alpha_1\}$ *for* α_1*, . . .).* □

Example 14 *This example is a part of Ex. 5.*



In this case, each element has an $\mathbf{IST}$*:*

- *for* d*:* $\{d, \alpha, b\}$ *and for* a*:* $\{a, \beta, d, \alpha, b\}$*;*
- *for* b*:* $\{b\}$*, for* β*:* $\{\beta\}$ *and for* α*:* $\{\alpha\}$*.*

Example 7 (cont'd): *In this example, each element has only one* $\mathbf{IST}$*. For each support and for* a*, the* $\mathbf{IST}$ *is reduced to the singleton containing the corresponding element. For* b*, the* $\mathbf{IST}$ *is the set* $\{b, \gamma, a\}$ *and for* x *it is the set* $\{x, a, b, \alpha, \beta, \gamma\}$*.* □

Example 8 (cont'd): *In this example, each support has only one* $\mathbf{IST}$*, reduced to the singleton containing the corresponding support. The same thing occurs for* a*. For* b*, the* $\mathbf{IST}$ *is the set* $\{b, \beta, a\}$ *and for* x *there are two sets* $\{x, a, \alpha\}$ *and* $\{x, a, b, \alpha, \beta\}$*.* □

Example 9 (cont'd): *In this example,* a*,* α *and* β *have only one* $\mathbf{IST}$*, reduced to the singleton containing the corresponding element. For* x*, only one* $\mathbf{IST}$ *exists, the set* $\{x, \alpha, a\}$ *and for* b *the* $\mathbf{IST}$ *is also unique,* $\{x, a, b, \alpha, \beta\}$*. Indeed, the set* $\{x, b, \alpha, \beta\}$ *cannot be an* $\mathbf{IST}$ *for* b *since the source of* α *cannot be* b*, since* b *is the root of the* $\mathbf{IST}$*. Idem for* x*: the set* $\{x, b, \alpha, \beta\}$ *cannot be an* $\mathbf{IST}$ *for* x *since the source of* β *cannot be* x*, since* x *is the root of the* $\mathbf{IST}$*.* □

[19] Note that, if x is self-supported then this condition will be false and the $\mathbf{IST}'$ set cannot be built; and so there is not an $\mathbf{IST}$ for x.

5.4 Characterization: a new proposition

At this point, it is worth to note that if there is no support cycles in the RAFN then the use of the formulae bases is enough for characterizing the RAFN semantics without the removal of some models. So the difficulty comes with the existence of these cycles and implies that we are able to remove the models in which an element is supported only because it is satisfied by these models (so in contradiction Def. 13).

So using the definitions of DCS and IST, we are now able to propose a definition for support-founded interpretations and models adapted to the case of RAFN:

Definition 19 *Let* $\mathbf{RAFN} = \langle \mathbf{A}, \mathbf{R}, \mathbf{N}, \mathbf{s}, \mathbf{t} \rangle$ *and let use RAFN-v2 definitions.* $\mathcal{I}$ *is a* support-founded interpretation *iff the two following conditions hold:*

1. *for each argument (resp. support)* x *of* **RAFN**, *belonging to a maximal DCS and s.t.* $\mathcal{I}(sAcc(x)) =$ *true (resp.* $\mathcal{I}(sVal(x)) =$ *true), two cases must be considered:*
   - *if there does not exist an impacting support tree* $\mathbf{IST}$ *for* x *then,* $\forall \beta \in \mathbf{N}$ *belonging to the same DCS that* x *and s.t.* $\mathbf{t}(\beta) = x$, $\mathcal{I}$ *does not satisfy* β *(so* $\mathcal{I}(sVal(\beta)) =$ *false);*
   - *otherwise, among the existing impacting support trees for* x, *there exists at least one* $\mathbf{IST}$ *s.t., considering any support* $\beta \in \mathbf{IST}$ *s.t.* $\mathbf{t}(\beta) = x$, *if* $\mathcal{I}$ *satisfies* β *(so* $\mathcal{I}(sVal(\beta)) =$ *true) then* $\mathcal{I}$ *must also satisfy at least an element of the source of* β *that belongs to* $\mathbf{IST}$ *(so* $\exists x_i \in \mathbf{s}(\beta) \cap \mathbf{IST}$ *s.t.* $\mathcal{I}(sAcc(x_i)) =$ *true).*
2. *for each element* x *of* **RAFN**, $\mathcal{I}(UnSupp(x)) =$ *true iff* $x \in UnSupp(U_{\mathcal{I}})$ *with* $U_{\mathcal{I}} = (S_{\mathcal{I}}, \Gamma_{\mathcal{I}}, \Delta_{\mathcal{I}})$.

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

Then using these support-founded models, the following characterization of RAFN-v2 semantics can be given:

Proposition 3 *Let* $\mathbf{RAFN} = \langle \mathbf{A}, \mathbf{R}, \mathbf{N}, \mathbf{s}, \mathbf{t} \rangle$ *and let use RAFN-v2 definitions. Let* $U = (S, \Gamma, \Delta)$ *be a structure on* **RAFN**.

1. U *is admissible iff there exists a support-founded model* $\mathcal{I}$ *of* $\Sigma_d(\mathbf{RAFN})$ *(in the sense of Def. 19) with* $S_{\mathcal{I}} = S$, $\Gamma_{\mathcal{I}} = \Gamma$ *and* $\Delta_{\mathcal{I}} = \Delta$.
2. U *is complete iff there exists a support-founded model* $\mathcal{I}$ *of* $\Sigma_d(\mathbf{RAFN}) \cup \Sigma_r(\mathbf{RAFN})$ *(in the sense of Def. 19) with* $S_{\mathcal{I}} = S$, $\Gamma_{\mathcal{I}} = \Gamma$ *and* $\Delta_{\mathcal{I}} = \Delta$.
3. U *is a preferred structure iff there exists a* $\subseteq$*-maximal support-founded model* $\mathcal{I}$ *of* $\Sigma_d(\mathbf{RAFN}) \cup \Sigma_r(\mathbf{RAFN})$ *(in the sense of Def. 19) 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* $\Sigma_d(\mathbf{RAFN}) \cup \Sigma_r(\mathbf{RAFN})$ *(in the sense of Def. 19).*
5. U *is stable iff there exists a support-founded model* $\mathcal{I}$ *of* $\Sigma_s(\mathbf{RAFN})$ *(in the sense of Def. 19) with* $S_{\mathcal{I}} = S$, $\Gamma_{\mathcal{I}} = \Gamma$ *and* $\Delta_{\mathcal{I}} = \Delta$.

Using Prop. 3, consider the previous examples and the preferred semantics:

Example 3 (cont'd): *The* $\subseteq$*-maximal support-founded models* $\mathcal{I}$ *of* $\Sigma_d(\mathbf{RAFN}) \cup \Sigma_r(\mathbf{RAFN})$ *(in the sense of Def. 19) correspond to the preferred structure* $(\{b, c, e, f\}, \{\beta, \pi\}, \{\epsilon\})$. □

Example 4 (cont'd): *The* $\subseteq$*-maximal support-founded models* $\mathcal{I}$ *of* $\Sigma_d(\mathbf{RAFN}) \cup \Sigma_r(\mathbf{RAFN})$ *(in the sense of Def. 19) correspond to the preferred structure* $(\mathbf{A}, \mathbf{R}, \mathbf{N})$. *Even if a support cycle exists, all models are kept since they are compatible with Def. 13.* □

Example 5 (cont'd): *The* $\subseteq$*-maximal support-founded models* $\mathcal{I}$ *of* $\Sigma_d(\mathbf{RAFN}) \cup \Sigma_r(\mathbf{RAFN})$ *(in the sense of Def. 19) correspond to the preferred structure* $(\{e\}, \varnothing, \{\alpha_1, \alpha_2, \beta_1, \beta_2, \gamma\})$. □

Example 6 (cont'd): *The* $\subseteq$*-maximal support-founded models* $\mathcal{I}$ *of* $\Sigma_d(\mathbf{RAFN}) \cup \Sigma_r(\mathbf{RAFN})$ *(in the sense of Def. 19) correspond to the preferred structure* $(\mathbf{A}, \mathbf{R}, \mathbf{N})$. □

6 Conclusion

In this work, we have proposed an adaptation of the logical encoding presented for REBAF in order to take into account argumentation frameworks with higher-order attacks and necessary supports (RAFN).

This leads to a new notion of impacting support trees ($\mathbf{IST}$), in order to adapt the notion of support-founded models to RAFN.

Then using all these elements, we provided a characterization of admissible (resp. complete, preferred, stable and grounded) structures for the most general definition of RAFN.

References

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

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

[3] H. Barringer, D.M. Gabbay, and J. Woods. Temporal dynamics of support and attack networks : From argumentation to zoology. In D. Hutter and W. Stephan, editors, *Mechanizing Mathematical Reasoning. LNAI 2605*, pages 59–98. Springer Verlag, 2005.

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

[5] G. Boella, L. W. N. van der Torre, and S. Villata. Attack relations among dynamic coalitions. In *Proc. of BNAIC*, pages 25–32. Universiteit Twente, Enschede, 2008.

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

[7] 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*, volume 2052. CEUR Workshop Proceedings, 2017.

[8] 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*, volume LNCS 10833, pages 150–169. Springer-Verlag, 2018.

[9] C. Cayrol, J. Fandinno, L. Fariñas del Cerro, and M.-C. Lagasquie-Schiex. Argumentation Frameworks with Recursive Attacks and Evidence-Based Supports. Rapport de recherche IRIT/RR–2018–01–FR, IRIT, 2018.

[10] C. Cayrol, J. Fandinno, L. Fariñas del Cerro, and M.-C. Lagasquie-Schiex. Structure-based Semantics of Argumentation Frameworks with Higher-order Attacks and Supports. In C. I. Chesñevar et al, editor, *Argumentation-based Proofs of Endearment. Essays in Honor of Guillermo R. Simari on the Occasion of his 70 th Birthday*, volume 37 of *Tributes*, pages 43–72. College Publications, 2018.

[11] C. Cayrol, J. Fandinno, L. Fariñas del Cerro, and M.-C. Lagasquie-Schiex. Valid attacks in argumentation frameworks with recursive attacks. *Annals of Mathematics and Artificial Intelligence (Special Issue: Commonsense 2017)*, 89, 2020.

[12] C. Cayrol and M-C. Lagasquie-Schiex. Gradual valuation for bipolar argumentation frameworks. In *Proc. of ECSQARU*, pages 366–377. Springer, 2005.

[13] C. Cayrol and M-C. Lagasquie-Schiex. Bipolarity in argumentation graphs: towards a better understanding. *Intl. J. of Approximate Reasoning*, 54(7):876–899, 2013.

[14] Claudette Cayrol and Marie-Christine Lagasquie-Schiex. Logical encoding of argumentation frameworks with higher-order attacks and evidential supports. *International Journal on Artificial Intelligence Tools*, 29(3-4):2060003:1–2060003:50, June 2020.

[15] 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*, 29:513–550, 2014.

[16] A. Cohen, S. Gottifredi, A. J. García, and G. R. Simari. An approach to abstract argumentation with recursive attack and support. *J. Applied Logic*, 13(4):509–533, 2015.

[17] 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*, pages 231–242. IOS Press, 2016.

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

[19] Phan Minh Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial Intelligence*, 77:321–357, 1995.

[20] D. M. Gabbay. Fibring argumentation frames. *Studia Logica*, 93:231–295, 2009.

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

[22] N. Karacapilidis and D. Papadias. Computer supported argumentation and collaborative decision making: the HERMES system. *Information systems*, 26(4):259–277, 2001.

[23] Marie-Christine Lagasquie-Schiex. Handling support cycles and collective interactions in the logical encoding of higher-order bipolar argumentation frameworks. In Pietro Baroni, Christoph Benzmüller, and Yì Nicholas Wáng, editors, *Proceeding of the 4th International Conference on Logic and Argumentation (CLAR) in Lecture Notes in Computer Sciences (LNCS) 13040*, pages 244–265, Hangzhou, China, 2021. Springer.

[24] Marie-Christine Lagasquie-Schiex. Handling support cycles in the logical encoding of argumentation frameworks with higher-order attacks and evidential supports. Rapport de recherche IRIT/RR- -2021- -04- -FR, IRIT, France, May 2021.

[25] Marie-Christine Lagasquie-Schiex. Logical encoding of argumentation frameworks with higher-order attacks and evidential supports: Taking into account the collective interactions. Rapport de recherche IRIT/RR- -2021- -05- -FR, IRIT, France, May 2021.

[26] Sanjay Modgil. Reasoning about preferences in argumentation frameworks. *Artificial Intelligence*, 173:901–934, 2009.

[27] F. Nouioua and V. Risch. Bipolar argumentation frameworks with specialized supports. In *Proc. of ICTAI*, pages 215–218. IEEE Computer Society, 2010.

[28] F. Nouioua and V. Risch. Argumentation frameworks with necessities. In *Proc. of SUM*, pages 163–176. Springer-Verlag, 2011.

[29] N. Oren and T. J. Norman. Semantics for evidence-based argumentation. In *Proc. of COMMA*, pages 276–284. IOS Press, 2008.

[30] N. Oren, C. Reed, and M. Luck. Moving between argumentation frameworks. In *Proc. of COMMA*, pages 379–390. IOS Press, 2010.

[31] S. Polberg and N. Oren. Revisiting support in abstract argumentation systems. In *Proc. of COMMA*, pages 369–376, 2014.

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

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

[34] S. Villata, G. Boella, D. M. Gabbay, and L. van der Torre. Modelling defeasible and prioritized support in bipolar argumentation. *AMAI*, 66(1-4):163–197, 2012.

A Differences between REBAF and RAFN-v2

These differences are due to the interpretation of the meaning of the supports and to some constraints existing in a framework and not in the other framework. So we have an impact on the definition of the semantics (and so on the notions used for defining these semantics), and another impact in the logical encoding of these frameworks and the corresponding characterization.

A.1 In terms of semantics

Here two main differences:

- the first one could be neglicted: in REBAF and in RAFN, the source of an attack is a set of arguments, but in RAFN, this set is a singleton. This impacts the definition of the sets $Def(U)$ and $UnAct(U)$;
- the second difference concerns the meaning of the supports and impacts the definition of the sets $Sup(U)$, $UnSupp(U)$ and the definition of the preferred and stable semantics.

These differences are synthetized in the following table (the important points are given in a red box):

Let $U = (S, \Gamma, \Delta)$ be a structure		
Notions	REBAF case	RAFN case
$Def_X(U) \stackrel{\text{def}}{=}$	$\{x \in X \mid \exists \alpha \in \Gamma, \boxed{\mathbf{s}(\alpha) \subseteq S}$ and $\mathbf{t}(\alpha) = x\}$	$\{x \in X \mid \exists \alpha \in \Gamma, \boxed{\mathbf{s}(\alpha) \in S}$ and $\mathbf{t}(\alpha) = x\}$
$Sup(U) \stackrel{\text{def}}{=}$	$\boxed{\mathbf{P}\cup}$ $\{\mathbf{t}(\alpha) \mid \boxed{\exists}$ $\alpha \in \Delta \cap Sup(U \setminus \{\mathbf{t}(\alpha)\})$, $\boxed{\mathbf{s}(\alpha) \subseteq}$ $(S \cap Sup(U \setminus \{\mathbf{t}(\alpha)\}))\}$	$\{x \mid \boxed{\forall}$ $\alpha \in \Delta$ st $\mathbf{t}(\alpha) = x$, if $\alpha \in Sup(U \setminus \{x\})$ then $\boxed{\mathbf{s}(\alpha) \cap (S \cap Sup(U \setminus \{x\})) \neq \varnothing}\}$
$UnSupp(U) \stackrel{\text{def}}{=}$	$\overline{Sup(U')}$ with $U' = (\overline{Def_{\mathbf{A}}(U)}, \mathbf{R}, \overline{Def_{\mathbf{E}}(U)})$	$\overline{Sup(U')}$ $\boxed{\setminus Sup(U)}$ with $U' = (\overline{Def_{\mathbf{A}}(U)}, \mathbf{R}, \overline{Def_{\mathbf{E}}(U)})$
$UnAct(U) \stackrel{\text{def}}{=}$	$\{\alpha \in \mathbf{R} \mid \alpha \in UnAcc(U)$ or $\boxed{\mathbf{s}(\alpha) \cap UnAcc(U) \neq \varnothing}\}$	$\{\alpha \in \mathbf{R} \mid \alpha \in UnAcc(U)$ or $\boxed{\mathbf{s}(\alpha) \in} UnAcc(U)\}$
U is pref. iff	it is a $\subseteq$-maximal $\boxed{\text{admissible}}$ structure	it is a $\subseteq$-maximal $\boxed{\text{complete}}$ structure
U is stab. iff	$(S \cup \Gamma \cup \Delta) = \overline{UnAcc(U)}$	it is $\boxed{\text{complete and}}$ $\overline{(S \cup \Gamma \cup \Delta)} = UnAcc(U)$

A.2 In terms of logical encoding

The differences are of course induced by the differences in terms of semantics evoked in the previous section, particularly the second one, and are synthetized in the following table (the important points are given in a red box):

	REBAF case	RAFN case
(1bis)	$\forall x \in (Attack \cup ESupport \cup Arg)$ $\Big(\Big(\boxed{PrimaFacie(x) \vee}$ $\exists y \in ESupport$ $(sVal(y) \wedge (T(y) = x) \wedge$ $\boxed{\forall z} \in Arg(S(z,y) \boxed{\rightarrow} sAcc(z)))\Big) \rightarrow Supp(x)\Big)$	$\forall x \in (Attack \cup NSupport \cup Arg)$ $\Big(\Big(\boxed{\forall y} \in NSupport$ $((sVal(y) \wedge (T(y) = x)) \boxed{\rightarrow}$ $\boxed{\exists z} \in Arg(S(z,y) \boxed{\wedge} sAcc(z)))\Big) \rightarrow Supp(x)\Big)$
(17)	$\forall x \in (Attack \cup ESupport \cup Arg)$ $\Big(Supp(x) \rightarrow \Big(\boxed{PrimaFacie(x) \vee}$ $\boxed{\exists y} \in ESupport$ $(sVal(y) \wedge (T(y) = x) \boxed{\wedge}$ $\boxed{\forall z} \in Arg(S(z,y) \boxed{\rightarrow} sAcc(z)))\Big)\Big)$	$\forall x \in (Attack \cup NSupport \cup Arg)$ $\Big(Supp(x) \rightarrow \Big(\boxed{\forall y} \in NSupport$ $\Big((sVal(y) \wedge (T(y) = x)) \boxed{\rightarrow}$ $\boxed{\exists z} \in Arg(S(z,y) \boxed{\wedge} sAcc(z))\Big)\Big)\Big)$

	REBAF case	RAFN case
(18)	$\forall x \in (Attack \cup ESupport \cup Arg)$ $\Big(UnSupp(x) \leftrightarrow \Big(\Big(\neg PrimaFacie(x) \wedge \forall y \in ESupport(T(y) = x \rightarrow \Big(\exists \beta \in Attack((T(\beta) = y \vee \exists z \in Arg(S(z,y) \wedge T(\beta) = z)) \wedge sVal(\beta) \wedge \forall z \in Arg(S(z,\beta) \rightarrow sAcc(z)))) \vee \exists z \in Arg (S(z,y) \wedge UnSupp(z)) \vee UnSupp(y)) \Big) \Big) \Big)$ So an element is unsupportable iff it is not prima-facie and none support is effective	$\forall x \in (Attack \cup NSupport \cup Arg)$ $\Big(UnSupp(x) \leftrightarrow \neg Supp(x) \wedge \Big(\exists y \in NSupport((T(y) = x) \wedge \Big(\forall \alpha \in Attack((T(\alpha) = y \wedge sVal(\alpha)) \rightarrow \exists u \in Arg(S(u,\alpha) \wedge \neg sAcc(u))) \Big) \wedge \neg UnSupp(y) \wedge \Big(\forall z \in Arg(S(z,y) \rightarrow \Big(\exists \beta \in Attack ((T(\beta) = z) \wedge sVal(\beta) \wedge \forall u \in Arg (S(u,\beta) \rightarrow sAcc(u))) \vee UnSupp(z) \Big) \Big) \Big) \Big) \Big)$ So an element is unsupportable iff it is not supported and at least one of its supports is not unsupportable and not defeated and has no effective source

Moreover, the notion of IST completely differs between the two frameworks:

An *impacting support tree* for x is a set $\mathbf{IST}$ s.t.	
REBAF case	RAFN case
$\mathbf{IST} = \{x_0, \ldots, x_n\}$ with $n > 0$ s.t. ▪ $\forall x_i, i \in [0 \ldots n]$, $x_i \in (\mathbf{A} \cup \mathbf{E}) \setminus \{x\}$ and is called a *node* of the tree; ▪ Let $\mathbf{IST_P} = (\mathbf{IST} \cap \mathbf{P} \cap \mathbf{A})$. $\mathbf{IST_P} \neq \varnothing$; ▪ $!\exists x_i \in \mathbf{IST}$ s.t. $x_i \in \mathbf{E}$ and $\mathbf{t}(x_i) = x$; x_i is called the *root* of the tree; ▪ $\forall x_i \in \mathbf{IST} \cap \mathbf{A}$, either $\exists x_j \in \mathbf{IST} \cap \mathbf{E}$ s.t. $x_i = \mathbf{t}(x_j)$, or $x_i \in \mathbf{IST_P}$ (in this case x_i is called a *leaf* of the tree); ▪ $\forall x_i \in \mathbf{IST} \cap \mathbf{E}, \forall x_j \in \mathbf{s}(x_i)$, $x_j \in \mathbf{IST}$.	$\mathbf{IST} = \{x\} \cup \mathbf{IST'}$ s.t.: ▪ x is called the *root* of the tree; ▪ $\mathbf{IST'} = \varnothing$ iff $\nexists y \in \mathbf{N}$ s.t. $\mathbf{t}(y) = x$; otherwise $\mathbf{IST'} = \{x_0, \ldots, x_n\}$ with $n > 0$ and $\forall x_i, i \in [0 \ldots n]$, $x_i \in (\mathbf{A} \cup \mathbf{N}) \setminus \{x\}$ and is called a *node* of the tree; ▪ $\forall y \in \mathbf{N}$, if $x = \mathbf{t}(y)$ then $y \in \mathbf{IST'}$; ▪ $\forall x_i \in (\mathbf{IST'} \cap \mathbf{A})$, if $\nexists x_j \in \mathbf{N}$ s.t. $x_i = \mathbf{t}(x_j)$ then x_i is called a *leaf* of the tree; otherwise, $\forall x_j \in \mathbf{N}$, if $x_i = \mathbf{t}(x_j)$ then $x_j \in \mathbf{IST'}$; ▪ $\forall x_i \in (\mathbf{IST'} \cap \mathbf{N}), \exists x_j \neq x, x_j \in \mathbf{s}(x_i)$ such that $x_j \in \mathbf{IST'}$.

The last difference exists concerning the notion of support-founded interpretation:

$\mathcal{I}$ is a *support-founded interpretation* iff the two following conditions	
REBAF case	RAFN case
1. for each argument (resp. support) x non prima-facie, belonging to a maximal DCS and s.t. $\mathcal{I}(sAcc(x)) =$ *true* (resp. $\mathcal{I}(sVal(x)) =$ *true*), there exists at least one impacting support tree $\mathbf{IST} = \{x_0, \ldots, x_n\}$ for x that is satisfied by $\mathcal{I}$, *i.e.* $\forall x_i \in \mathbf{IST}$, if $x_i \in \mathbf{A}$ then $\mathcal{I}(sAcc(x_i)) =$ *true*, otherwise $\mathcal{I}(sVal(x_i)) =$ *true*; 2. for each element x of **REBAF**, $\mathcal{I}(UnSupp(x)) =$ *true* iff $x \in UnSupp(U_\mathcal{I})$ with $U_\mathcal{I} = (S_\mathcal{I}, \Gamma_\mathcal{I}, \Delta_\mathcal{I})$.	1. for each argument (resp. support) x, belonging to a maximal DCS and s.t. $\mathcal{I}(sAcc(x)) =$ *true* (resp. $\mathcal{I}(sVal(x)) =$ *true*), two cases must be considered: ▪ if there does not exist an impacting support tree $\mathbf{IST}$ for x then, $\forall \beta \in \mathbf{N}$ belonging to the same DCS that x and s.t. $\mathbf{t}(\beta) = x$, $\mathcal{I}$ does not satisfy β (so $\mathcal{I}(sVal(\beta)) =$ *false*); ▪ otherwise, among the existing impacting support trees for x, there exists at least one $\mathbf{IST}$ s.t., considering any support $\beta \in \mathbf{IST}$ s.t. $\mathbf{t}(\beta) = x$, if $\mathcal{I}$ satisfies β (so $\mathcal{I}(sVal(\beta)) =$ *true*) then $\mathcal{I}$ must also satisfy at least an element of the source of β that belongs to $\mathbf{IST}$ (so $\exists x_i \in \mathbf{s}(\beta) \cap \mathbf{IST}$ s.t. $\mathcal{I}(sAcc(x_i)) =$ *true*). 2. for each element x of **RAFN** $= \langle \mathbf{A}, \mathbf{R}, \mathbf{N}, \mathbf{s}, \mathbf{t} \rangle$, $\mathcal{I}(UnSupp(x)) =$ *true* iff $x \in UnSupp(U_\mathcal{I})$ with $U_\mathcal{I} = (S_\mathcal{I}, \Gamma_\mathcal{I}, \Delta_\mathcal{I})$.

B Proofs

B.1 Proofs of Section 4.2

Proof of Proposition 2

Recall that 2 definitions for the $UnSupp$ set exist for a given structure U:

1. either Def. 16: $UnSupp(U) = \overline{Sup(\overline{Def_{\mathbf{A}}(U)}, \mathbf{R}, \overline{Def_{\mathbf{N}}(U)})}$
2. or Def. 17: $UnSupp(U) = \overline{Sup(\overline{Def_{\mathbf{A}}(U)}, \mathbf{R}, \overline{Def_{\mathbf{N}}(U)})} \setminus Sup(U)$

In order to simplify the reading of this proof, we add an indice to a given semantics in order to identify which definition is used. So, a σ_1 structure will correspond to the definition of the σ semantics using Def. 16, whereas a σ_2 structure will correspond to the definition of the σ semantics using Def. 17.

Using this notation, the proof of Prop. 2 corresponds to: for each semantics σ, we must prove that U is a σ_1 structure iff U is a σ_2 structure, σ taken in $\{$conflict-free, self-supporting, admissible, complete, preferred, grounded, stable$\}$.

First of all, we can give some trivial results for any structure U:

- $Def_1(U) = Def_2(U)$ (the $UnSupp$ set is not used for defining the Def set)
- $Sup_1(U) = Sup_2(U)$ (the $UnSupp$ set is not used for defining the Sup set)
- $UnSupp_2(U) \subseteq UnSupp_1(U)$ (following Def. 16 and 17)
- $UnAcc_2(U) \subseteq UnAcc_1(U)$ (following the definition of the $UnAcc$ set and the previous items)
- $UnAct_2(U) \subseteq UnAct_1(U)$ (following the definition of the $UnAct$ set and the previous item)
- $Acc_2(U) \subseteq Acc_1(U)$ (following the definition of the Acc set and the previous item)

Note that Ex. 13 shows that $UnSupp_1(U) \not\subseteq UnSupp_2(U)$, $UnAcc_1(U) \not\subseteq UnAcc_2(U)$, $UnAct_1(U) \not\subseteq UnAct_2(U)$ and $Acc_1(U) \not\subseteq Acc_2(U)$.

Consider now the different semantics:

σ **is conflict-free:** since $Def_1(U) = Def_2(U)$, U is a conflict-free$_1$ structure iff U is a conflict-free$_2$ structure.

σ **is self-supporting:** since $Sup_1(U) = Sup_2(U)$, U is a self-supporting$_1$ structure iff U is a self-supporting$_2$ structure.

σ **is complete:** U is complete iff it is conflict-free and $U = Acc(U)$. The conflict-freeness being conserved between the two RAFN versions, it remains to prove that $U = Acc_1(U)$ iff $U = Acc_2(U)$.

- Consider first an element $x \in U = Acc_2(U)$: since $Acc_2(U) \subseteq Acc_1(U)$ then $x \in Acc_1(U)$. So $U = Acc_2(U) \subseteq Acc_1(U)$.
- Consider now an element $x \in U = Acc_1(U)$ and show that $x \in Acc_2(U)$. Assume that there exists an element x st $x \in U = Acc_1(U)$ and $x \notin Acc_2(U)$. Since $x \in Acc_1$ then $x \in Sup_1(U)$; so since $Sup_1(U) = Sup_2(U)$, $x \in Sup_2(U)$. Thus $x \notin Acc_2(U)$ corresponds to the existence of an attack α st $\mathbf{t}(\alpha) = x$ and $\alpha \notin UnAct_2(U)$; let denote α or its source by y, we have so $y \notin UnAcc_2(U)$; so $y \notin Def_2(U)$ and $y \notin UnSupp_2(U)$.
  Moreover, since $x \in Acc_1$ then $\alpha \in UnAct_1(U)$, so $y \in UnAcc_1(U)$ with two possible cases: either $y \in Def_1(U)$ or $y \in UnSupp_1(U)$. The first case leads to a contradiction since $Def_1(U) = Def_2(U)$. The second case leads to the fact that $y \in Unsupp_1(U) \setminus UnSupp_2(U) = Sup_1(U) = Sup_2(U)$. Let study the different constraints related to these conditions:
  - $y \in Sup_1(U)$ corresponds to the following cases:

1. either no supported support targets y
2. or for any supported support β that targets y either $\beta \notin U$ or β and a part of its source belong to U

– $y \in UnSupp_1(U)$ corresponds to the fact that there exists at least one supported support β not defeated by U whose source is defeated by U or not supported.

So considering the aggregation of constraints and the fact that U is conflict-free, we must be in the case where there exists at least one supported support β not defeated by U *but also not belonging to* U and whose source is defeated by U or not supported; however U is complete$_1$ then, since β is supported and not defeated, it should be in $Acc_1(U)$ and so in U; so contradiction. So $x \in Acc_2(U)$.

Thus U is complete$_1$ iff U is complete$_2$.

σ **is preferred:** following the previous item and the definition of the preferred semantics in the RAFN case (see Def. 14), it is obvious to see that U is preferred$_1$ iff U is preferred$_2$

σ **is grounded:** following the item about completeness and the definition of the grounded semantics in the RAFN case (similar to the one given for the REBAF case, see Def. 8), it is obvious to see that U is grounded$_1$ iff U is grounded$_2$

σ **is stable:** following the item about completeness and the definition of the stable semantics in the RAFN case (see Def. 14), for proving that U is stable$_1$ iff U is stable$_2$, it remains to prove that $\overline{U} = UnAcc_1(U)$ iff $\overline{U} = UnAcc_2(U)$, *i.e.* $U = \overline{UnAcc_1(U)}$ iff $U = \overline{UnAcc_2(U)}$. This proof is similar to the one given in the item about completeness.

- Consider first an element $x \in U = \overline{UnAcc_1(U)}$: since $UnAcc_2(U) \subseteq UnAcc_1(U)$ then $x \in \overline{UnAcc_2(U)}$. So $U = \overline{UnAcc_1(U)} \subseteq \overline{UnAcc_2(U)}$.
- Consider now an element $x \in U = \overline{UnAcc_2(U)}$ and show that $x \in \overline{UnAcc_1(U)}$. Assume that there exists an element x st $x \in U = \overline{UnAcc_2(U)}$ and $x \notin \overline{UnAcc_1(U)}$, *i.e.* $x \in U$, $x \notin UnAcc_2(U)$ and $x \in UnAcc_1(U)$. Since $UnAcc(U) = Def(U) \cup UnSupp(U)$, considering that $x \notin UnAcc_2(U)$ means that $x \notin Def_2(U)$ and $x \notin UnSupp_2(U)$ and considering that $x \in UnAcc_1(U)$ means that $x \in Def_1(U)$ or $x \in UnSupp_1(U)$; $x \in Def_1(U)$ is clearly in contradiction with $x \notin Def_2(U)$ since $Def_1(U) = Def_2(U)$; so the only possible case remains $x \in UnSupp_1(U)$, knowing that $x \notin UnSupp_2(U)$, so $x \in Sup_1(U)$. This is exactly the same configuration that the one described in the proof of the completeness item for the variable y and the same reasoning can be applied leading to a contradiction: there exists at least one supported support β for x not defeated by U *but also not belonging to* U and whose source is defeated by U or not supported; however U is complete$_1$ then, since β is supported and not defeated, it should be in $Acc_1(U)$ and so in U; so contradiction. So $x \in UnAcc_1(U)$.

Thus U is stable$_1$ iff U is stable$_2$.

B.2 Some additional results

For the next proofs, a notation and some lemmas will be useful.

Notation 1 *Let $U = (S, \Gamma, \Delta)$ be a structure of* **RAFN***, and $x \in \mathbf{A} \cup \mathbf{R} \cup \mathbf{N}$. x will be said to be* defended *by U, iff every attack $\alpha \in \mathbf{R}$ 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 *Let* **RAFN** $= \langle \mathbf{A}, \mathbf{R}, \mathbf{N}, \mathbf{s}, \mathbf{t} \rangle$ *and let use RAFN-v2 definitions. Any conflict-free structure U satisfies $Acc(U) \cap Def(U) = \varnothing$.*

Proof of Lemma 1 Consider an element $x \in Acc(U)$ and assume that $x \in Def(U)$. Since $x \in Def(U)$ then there exists an attack $\alpha \in U$ st $\mathbf{t}(\alpha) = x$ and $\mathbf{s}(\alpha) \in U$. But $x \in Acc(U)$ then α must be unactivable, so either α or $\mathbf{s}(\alpha)$ belong to $UnAcc(U)$; since $UnAcc(U) = Def(U) \cup UnSupp(U)$, then α (or $\mathbf{s}(\alpha)$) belongs to $Def(U)$ or to $UnSupp(U)$; the first case leads to a contradiction since U is conflict-free; the second case leads also to a contradiction since α and $\mathbf{s}(\alpha)$ belong to $Sup(U)$ (U being self-supporting) and, following Def. 17, they cannot belong to $UnSupp(U)$).

So $Acc(U) \cap Def(U) = \varnothing$.

Lemma 2 *Let* $\mathbf{RAFN} = \langle \mathbf{A}, \mathbf{R}, \mathbf{N}, \mathbf{s}, \mathbf{t} \rangle$ *and let use RAFN-v2 definitions. Any conflict-free and self-supporting structure U satisfies:*

$$Acc(U) \subseteq \overline{UnAcc(U)} \subseteq \overline{Def(U)}.$$

Proof of Lemma 2

- Proof for $\overline{UnAcc(U)} \subseteq \overline{Def(U)}$:
  since $UnAcc(U) = Def(U) \cup UnSupp(U)$, so $\overline{UnAcc(U)} = \overline{Def(U) \cup UnSupp(U)} = \overline{Def(U)} \cap \overline{UnSupp(U)}$; thus $\overline{UnAcc(U)} \subseteq \overline{Def(U)}$.
- Proof for $Acc(U) \subseteq \overline{UnAcc(U)}$:
  Consider an element $x \in Acc(U)$ and assume that $x \notin \overline{UnAcc(U)}$; so $x \in UnAcc(U)$; by definition $UnAcc(U) = Def(U) \cup UnSupp(U)$, so:
  - either $x \in Def(U)$: using Lemma 1, this is in contradiction with $x \in Acc(U)$;
  - or $x \in UnSupp(U)$: since, by definition of $Acc(U)$, each element in $Acc(U)$ is supported then $x \in Sup(U)$; so following Definition 17, we obtain a contradiction.

  So each case leading to a contradiction, we can conclude that $Acc(U) \subseteq \overline{UnAcc(U)}$.

Lemma 3 *Let* $\mathbf{RAFN} = \langle \mathbf{A}, \mathbf{R}, \mathbf{N}, \mathbf{s}, \mathbf{t} \rangle$ *and let use RAFN-v2 definitions. Any stable structure U satisfies:* $\overline{Sup(U)} = UnSupp(U)$.

Proof of Lemma 3

Assume that U is a stable structure. Following Def. 17, we already know that $UnSupp(U) \subseteq \overline{Sup(U)}$. It remains to prove the reverse inclusion.
Let us recall that $UnSupp(U) = \overline{Sup(U')} \setminus Sup(U)$ where $U' = (\overline{Def_{\mathbf{A}}(U)}, \mathbf{R}, \overline{Def_{\mathbf{N}}(U)})$. So we have to prove that $Sup(U') \cup Sup(U) \subseteq Sup(U)$, so that $Sup(U') \subseteq Sup(U)$.
Let $x \in Sup(U')$. So for any support $\alpha \in \overline{Def(U)}$ targeting x if $\alpha \in Sup(U' \setminus \{x\})$ then $s_\alpha \cap \overline{Def(U)} \cap Sup(U' \setminus \{x\}) \neq \varnothing$. Assume that $x \notin Sup(U)$ and consider the following cases:

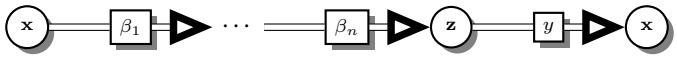
- Either there is no support α targeting x: so $x \in Sup(U)$ and contradiction with the assumption.
- Or there are some supports α targeting x but they are all defeated by U: since U is stable, U is conflict-free and so $\alpha \notin U$; thus $x \in Sup(U)$ and contradiction with the assumption.
- Or there exists at least one support α targeting x that is not defeated by U: so $\alpha \in U'$ and since $x \in Sup(U')$ there also exist in U' some elements y of $\mathbf{s}(\alpha)$; α and the corresponding y belong to $Sup(U' \setminus \{x\})$; so neither α, nor y need x to be supported; this implies that they also belong to $Sup(U')$; so they do not belong to $Unsupp(U)$; since α (resp. y) $\notin Def(U)$ and $\notin Unsupp(U)$ then, following the definition of the $UnAcc$ set, they belong to $\overline{UnAcc(U)}$; since U is stable, we have α (resp. y) $\in U$; so they are in $Sup(U)$; so $x \in Sup(U)$ and contradiction with the assumption.

Hence we have proved that $x \in Sup(U)$ and so that $\overline{Sup(U)} = UnSupp(U)$

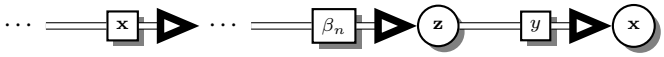
Lemma 4 *Let* $\mathbf{RAFN} = \langle \mathbf{A}, \mathbf{R}, \mathbf{N}, \mathbf{s}, \mathbf{t} \rangle$ *and let use RAFN-v2 definitions. Let* $U = (S, \Gamma, \Delta)$ *be a structure. Consider an element* x *such that, for any support* $y \in \Delta \cap Sup(U)$ *that targets* x*, then* $\mathbf{s}(y) \cap S \cap Sup(U) \neq \varnothing$*. Then,* $x \in Sup(U)$.

Proof of Lemma 4
Assume that $x \notin Sup(U)$. So, by the definition of the Sup set, $\exists y$ st $\mathbf{t}(y) = x$ and $y \in \Delta \cap Sup(U \setminus \{x\})$ and $\mathbf{s}(y) \cap S \cap Sup(U \setminus \{x\}) = \varnothing$. Since $\mathbf{s}(y) \cap S \cap Sup(U) \neq \varnothing$, there exists $z \in \mathbf{s}(y) \cap S$ and $z \in Sup(U) \setminus Sup(U \setminus \{x\})$. So, for any couple (z, y), we have a chain of supports (each support and a part of its source belong to U) leading to x and containing z and y. This chain can be schematically represented by:

- either

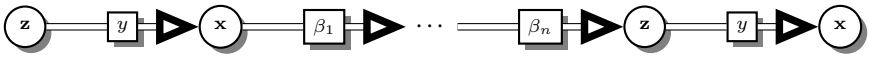


- or

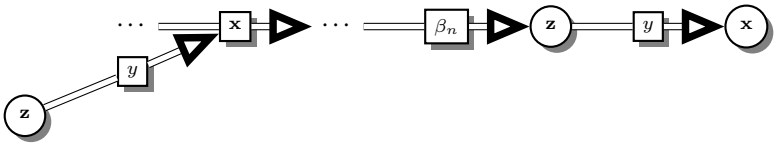


Let complete these chains by adding z and y that support x:

- either



- or



So it is obvious to see that, in each case and for any couple (z, y) with $z \in \mathbf{s}(y) \cap S \cap Sup(U)$, $z \notin Sup(U \setminus \{z\})$. So by definition of the Sup set, $z \notin Sup(U)$. Contradiction with the assumption saying that $\mathbf{s}(y) \cap S \cap Sup(U) \neq \varnothing$
So $x \in Sup(U)$

B.3 Proofs of Section 5.4

Proof of Proposition 3.[20]

Let $\mathbf{RAFN} = \langle \mathbf{A}, \mathbf{R}, \mathbf{N}, \mathbf{s}, \mathbf{t} \rangle$.

1. (admissibility)

   $\Rightarrow$ Assume that the structure $U = (S, \Gamma, \Delta)$ is admissible. Let us define an interpretation $\mathcal{I}$ of $\Sigma_d(\mathbf{RAFN})$. The idea is to define $\mathcal{I}$ by successively adding constraints that $\mathcal{I}$ should satisfy:
   - For all $x \in \mathbf{A} \cup \mathbf{R} \cup \mathbf{N}$,
     $\mathcal{I}(Arg(x)) =$ ***true*** iff $x \in \mathbf{A}$,
     $\mathcal{I}(Attack(x)) =$ ***true*** iff $x \in \mathbf{R}$ and
     $\mathcal{I}(NSupport(x)) =$ ***true*** iff $x \in \mathbf{N}$.
   - For all $x \in \mathbf{A} \cup \mathbf{R} \cup \mathbf{N}$, $\mathcal{I}(Supp(x)) =$ ***true*** iff $x \in Sup(U)$.
   - For all $x \in \mathbf{A} \cup \mathbf{R} \cup \mathbf{N}$, $\mathcal{I}(UnSupp(x)) =$ ***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 $x \in Defended(U)$).

[20] This proof is inspired by the proof of the corresponding propositions in [24, 25].

- For all $x \in \mathbf{A}$, $\mathcal{I}(NAcc(x)) =$ *true* iff $\mathcal{I}(Acc(x)) =$ *false*.
- For all $x \in \mathbf{R}$ (resp. $\in \mathbf{N}$), $\mathcal{I}(Val(x)) =$ *true* iff $x \in \Gamma$ (resp. Δ) or ($x \notin \Gamma$ (resp. Δ), $x \notin Sup(U)$ and $x \in Defended(U)$).
- For all $x \in \mathbf{A}$, $\mathcal{I}(sAcc(x)) =$ *true* iff ($\mathcal{I}(Acc(x)) =$ *true* and $\mathcal{I}(Supp(x)) =$ *true*).
- For all $x \in \mathbf{R} \cup \mathbf{N}$, $\mathcal{I}(sVal(x)) =$ *true* iff ($\mathcal{I}(Val(x)) =$ *true* and $\mathcal{I}(Supp(x)) =$ *true*).

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 $\Sigma_d(\mathbf{RAFN})$. And for proving that $\mathcal{I}$ is a support-founded model of $\Sigma_d(\mathbf{RAFN})$ 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.[21]

★ Let $x \in S_\mathcal{I}$. By definition of $S_\mathcal{I}$, $\mathcal{I}(sAcc(x)) =$ *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, $\mathcal{I}(sAcc(x)) =$ *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 $y \in \mathbf{R}$ and $x \in \mathbf{A}$ with $x = T(y)$, $\mathcal{I}(sVal(y)) =$ *true* and $\mathcal{I}(\forall z \in Arg(S(z, y) \rightarrow sAcc(z))) =$ *true* (so, since in the case of RAFN the source of any attack is a singleton, here denoted by x_y, we have $\mathcal{I}(sAcc(x_y)) =$ *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 belong to $UnAcc(U)$. However, y and s_y being elements of the admissible structure U, due to Lemma 2, 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 $Sup(U)$, for any support α targeting x and such that $\alpha \in \Delta$ and $\alpha \in Sup(U \setminus \{x\})$, then $s_\alpha \cap S \cap Sup(U \setminus \{x\}) \neq \varnothing$. As $S = S_\mathcal{I}$ and $\Delta = \Delta_\mathcal{I}$ it holds that there exists an element z in $\mathbf{s}_\alpha$ st $\mathcal{I}(sAcc(z)) =$ *true* and $\mathcal{I}(sVal(\alpha)) =$ *true*. Hence formula **(17)** is satisfied by $\mathcal{I}$.

★ Let us consider formula **(1bis)**. If x is not the target of a support belonging to U then $x \in Sup(U)$ and formula **(1bis)** is trivially satisfied by $\mathcal{I}$. Consider now the case of any support y that targets x and such that $\mathcal{I}(\exists z \in Arg(S(z, y) \wedge sAcc(z))) =$ *true* and $\mathcal{I}(sVal(y)) =$ *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 $s_y \subseteq S$. Moreover, as U is admissible, U is self-supporting, so y and some elements of s_y belong to $Sup(U)$. From Lemma 4, it follows that $x \in Sup(U)$ hence $\mathcal{I}(Supp(x)) =$ *true*. So formula **(1bis)** is satisfied by $\mathcal{I}$.

★ Let us now consider formula **(18)**. First, for the "only if part", consider x such that $\mathcal{I}(UnSupp(x))$ $=$ *true*. By definition of $\mathcal{I}(UnSupp)$, $x \in UnSupp(U)$. So $x \notin Supp(U)$ and $\mathcal{I}(\neg Supp(x)) =$ *true*. Moreover, since $UnSupp(U) = \overline{Sup(U')}$ (where $U' = (\overline{Def_\mathbf{A}(U)}, \mathbf{R}, \overline{Def_\mathbf{N}(U)})$), $x \in \overline{Sup(U')}$. So using the contrapositive of Lemma 4, applied to the structure U', it follows that for at least one support y leading to x, $y \in \overline{Def(U)} \cap Sup(U')$ and $\mathbf{s}(y) \cap \overline{Def(U)} \cap Sup(U') = \varnothing$. so we have:

- if $y \in \overline{Def(U)}$: so $y \notin Def(U)$; that implies that for any attack α in U and targeting y, $\mathbf{s}(\alpha)$ is not included in U; so the subformula $\forall \alpha \in Attack((T(\alpha) = y \wedge sVal(\alpha)) \rightarrow \exists u \in Arg(S(u, \alpha) \wedge \neg sAcc(u)))$ is satisfied by $\mathcal{I}$;
- if $y \in Sup(U')$: so $y \notin UnSup(U)$; this implies that the formula $\neg UnSupp(y)$ is satisfied by $\mathcal{I}$;
- if $\mathbf{s}(y) \cap \overline{Def(U)} \cap Sup(U') = \varnothing$: so for any $z \in \mathbf{s}(y)$, either z is defeated by U (so the subformula $\exists \beta \in Attack((T(\beta) = z) \wedge sVal(\beta) \wedge \forall u \in Arg(S(u, \beta) \rightarrow sAcc(u)))$ is satisfied by $\mathcal{I}$), or $z \notin Sup(U')$ (so the subformula $UnSupp(z)$ is satisfied by $\mathcal{I}$).

So the "only if" part of formula **(18)** is satisfied by $\mathcal{I}$.
For the "if" part, let us consider x such that $x \notin Sup(U)$ and there exists a support that is not defeated by

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

U, that is not unsupportable by U and whose source components are either defeated or unsupportable by U. So $x \notin Sup(U')$ and $x \notin Sup(U)$; thus, by the definition of the $UnSupp$ set, we have $x \in UnSupp(U)$ and so $\mathcal{I}(UnSupp(x)) = \textit{true}$.
So formula **(18)** is satisfied by $\mathcal{I}$.

★ Let us now consider formula **(11)**. Let $\alpha \in \mathbf{R}$ and $x \in \mathbf{A}$ such that $x = T(\alpha)$ and $\mathcal{I}(Acc(x)) = \textit{true}$. 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, $\mathcal{I}(sAcc(x)) = \textit{true}$ (resp. $\mathcal{I}(sVal(x)) = \textit{true}$) 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 19 is trivially satisfied.
Consider now Condition 1. Let x an element such that there exists a DCS $\mathbf{C}$ containing x. Assume that $\mathcal{I}(sAcc(x))$ (or $\mathcal{I}(sVal(x))) = \textit{true}$. So $x \in U$ and, since U is admissible (so self-supporting), two cases are possible:

- Either there is no $\mathbf{IST}$ for x: so by the definition of $\mathbf{IST}$, there exist some supports targeting x; moreover, the only reason explaining the non-existence of an $\mathbf{IST}$ for x is the fact that, when we build the $\mathbf{IST}$, an element that should be added to the $\mathbf{IST}'$ part of the $\mathbf{IST}$ is x and, this addition being impossible following the definition of $\mathbf{IST}'$, the building of the $\mathbf{IST}$ fails; moreover, it is trivial to see that this problem occurs only when we consider the branches of the tree originated in the supports targeting x and belonging to the same DCS that x; thus it is not possible to support x with U if at least one of these supports is present; so no support targeting x and belonging to the same DCS can belong to U and so $\mathcal{I}$ does not satisfy these supports.
- Or there are some $\mathbf{IST}$ for x: so two subcases are possible:
  - either there is no support targeting x in U, so $\mathcal{I}$ cannot satisfy any support for x and the condition is trivially satisfied;
  - or there are some supports β targeting x in U: by definition of $\mathbf{IST}$, $\beta \in \mathbf{IST}$; moreover since $\beta \in U$, then at least one element x_i of its source is also in U; so consider the $\mathbf{IST}$ containing at the same time β and x_i; since they are in U and U is self-supporting and following the definition of $\mathcal{I}$, we have $\mathcal{I}(sVal(\beta)) = \textit{true}$ and $\mathcal{I}(sAcc(x_i)) = \textit{true}$; so Condition 1 is satisfied.

So, in both cases, $\mathcal{I}$ is a support-founded model.

⇐ Let $\mathcal{I}$ be a support-founded model of $\Sigma_d(\mathbf{RAFN})$. 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. $\mathbf{RAFN}$. 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 $s_y \subseteq S_\mathcal{I}$ and $T(y) = x$.
By definition, it holds that $\mathcal{I}(\forall z \in Arg(S(z, y) \rightarrow sAcc(z))) = \textit{true}$ and $\mathcal{I}(sVal(y)) = \textit{true}$. Moreover, in the case when $x \in S_\mathcal{I}$, it holds that $\mathcal{I}(sAcc(x)) = \textit{true}$ and so $\mathcal{I}(Acc(x)) = \textit{true}$ (as $\mathcal{I}$ satisfies formula **(2bis)**). Then it holds that $\mathcal{I}(NAcc(x)) = \textit{false}$ (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 $\mathcal{I}(sVal(x)) = \textit{true}$ 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 $\Sigma(\mathbf{RAFN})$.

★ 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 $\mathcal{I}(sAcc(x)) = \textit{true}$ (resp. $\mathcal{I}(sVal(x)) = \textit{true}$). As $\mathcal{I}$ satisfies formula **(2bis)**, $\mathcal{I}(Supp(x)) = \textit{true}$. As $\mathcal{I}$ satisfies formula **(17)**, two cases must be considered:

- either there does not exist a support targeting x: so x is trivially supported and belongs to $Sup(U)$;
- or there exist some supports y targeting x: if none of the y belong to U then x is trivially supported and belongs to $Sup(U)$; consider now a support y that belongs to U; since formula **(17)** is satisfied, there exists an element z in the source of y that is also in U; so it holds that $\mathcal{I}(sAcc(z)) = \textit{true}$ and

$\mathcal{I}(sVal(y)) =$ *true*, and formula **(17)** can still be used, thus enabling to build a tree of supports. As U is finite and $\mathcal{I}$ is support founded, this process will end and an **IST** can be exhibited for x showing that x is supported without itself and so belongs to $Sup(U)$.

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 $\mathcal{I}(sAcc(x)) =$ *true*. It follows that $\mathcal{I}(Acc(x)) =$ *true*. As $\mathcal{I}$ satisfies formula **(11)**, it follows that either there exists an attack β targeting α (or an element of s_α) with $\beta \in \Delta_\mathcal{I}$ and $s_\beta \subseteq S_\mathcal{I}$, or $\mathcal{I}$ satisfies $UnSupp(\alpha)$ (or $\mathcal{I}$ satisfies $\exists z \in Arg(S(z,\alpha) \wedge UnSupp(z))$, *i.e.* at least for one element x_i of s_α we have $\mathcal{I}(UnSupp(x_i)) =$ *true*).

- In the first case, it holds that α (resp. an element of s_α) belongs to $Def(U)$.
- In the second case, we prove that α (resp. an element of s_α) belongs to $UnSupp(U)$. For that purpose, we must prove the following intermediary result:

  Property: For any element u, if $\mathcal{I}$ satisfies $UnSupp(u)$, then $u \in UnSupp(U)$.
  Proof: The proof is done taking the contrapositive: if $u \in Sup(U') \cup Sup(U)$, with $U' = (Def_\mathbf{A}(U), \mathbf{R}, Def_\mathbf{N}(U))$, then $\mathcal{I}$ does not satisfy $UnSupp(u)$. Let us consider $u \in Sup(U)$. Several cases must be taken into account:

  - no support targets u: so following formula **(18)**, we trivially obtain that $\mathcal{I}(UnSupp(u))$ is *false*;
  - a support y targets u but it is defeated by U: so following formula **(18)**, we trivially obtain that $\mathcal{I}(UnSupp(u))$ is *false*;
  - the supports targeting u are not defeated by U: so they belong to U'; let y be one of these supports, two subcases appear here:
    - either there is at least one element of the source of y that is not defeated, so that belongs to U'; then following formula **(18)**, we trivially obtain that $\mathcal{I}(UnSupp(u))$ is *false*;
    - or any element in the source of y is defeated; this implies that $u \notin Sup(U')$, so $u \in Sup(U)$; considering that for any y that supports u, its source cannot belong to U (since U is conflict-free, see a previous item in this proof), then we can conclude that y cannot belong to U otherwise $u \notin Sup(U)$; then, following formula **(17)**, we can deduce that $\mathcal{I}(Supp(u))$ is *true* and so, following formula **(18)**, we obtain that $\mathcal{I}(UnSupp(u))$ is *false*.

So, in every case, 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 $\Sigma_d(\mathbf{RAFN}) \cup \Sigma_r(\mathbf{RAFN})$:

- 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 ($x \notin S$ and $x \in Defended(U)$).
- For all $x \in \mathbf{R}$ (resp. $\in \mathbf{N}$), $\mathcal{I}(Val(x)) =$ *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 $\Sigma_d(\mathbf{RAFN}) \cup \Sigma_r(\mathbf{RAFN})$.

★ Note that if U is complete, for all $x \in \mathbf{A} \cup \mathbf{R} \cup \mathbf{N}$, 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 $\Sigma_d(\mathbf{RAFN})$.

★ 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 $\mathcal{I}(\exists z \in Arg(S(z, \alpha) \wedge UnSupp(z))) =$ ***true***, or α (or an element of s_α) is attacked by β with $\beta \in \Gamma_{\mathcal{I}}$ and $s_\beta \subseteq S_{\mathcal{I}}$. Due to the definition of $\mathcal{I}(UnSupp)$, for each attack α targeting x, either $\alpha \in UnSupp(U)$, or an element of $s_\alpha \in UnSupp(U)$, or α (or an element of 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 $\Sigma_d(\mathbf{RAFN}) \cup \Sigma_r(\mathbf{RAFN})$.

★ 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 $\Sigma_d(\mathbf{RAFN}) \cup \Sigma_r(\mathbf{RAFN})$. 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 an element of $s_\alpha \in UnSupp(U)$, or α (or an element of s_α) belongs to $Def(U)$ (*i.e.* α –or an element of s_α– is attacked by $\beta \in U$ with $s_\beta \subseteq U$). 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 belongs to (resp. s_β is included in) the structure then $\mathcal{I}(sVal(\beta))$ (resp. $\mathcal{I}(\forall z \in Arg(S(z, \beta) \rightarrow sAcc(z))))$ 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 $\mathcal{I}(sAcc(x)) =$ ***true***, so $x \in S_{\mathcal{I}}$. Similarly, it can be proved that for all $x \in \mathbf{R} \cap Acc(U)$ (resp. $x \in \mathbf{N} \cap Acc(U)$), $x \in \Gamma_{\mathcal{I}}$ (resp. $x \in \Delta_{\mathcal{I}}$). We have proved that U is a complete structure.

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 $\Sigma_x = \Sigma_d(\mathbf{RAFN}) \cup \Sigma_r(\mathbf{RAFN})$, it follows that the preferred structures correspond to the structures $U_{\mathcal{I}}$ where $\mathcal{I}$ is a $\subseteq$-maximal support-founded model of $\Sigma_d(\mathbf{RAFN}) \cup \Sigma_r(\mathbf{RAFN})$.

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 $U_{\mathcal{J}} = (S_{\mathcal{J}}, \Gamma_{\mathcal{J}}, \Delta_{\mathcal{J}})$, where $\mathcal{J}$ denotes a support-founded model of Σ_x. Taking $\Sigma_x = \Sigma_d(\mathbf{RAFN}) \cup \Sigma_r(\mathbf{RAFN})$, it follows that the grounded structure correspond to the structure $U_{\mathcal{I}}$ where $\mathcal{I}$ is a $\subseteq$-minimal support-founded model of $\Sigma_d(\mathbf{RAFN}) \cup \Sigma_r(\mathbf{RAFN})$.

5. (stable semantics)

   ⇒ Assume that the structure $U = (S, \Gamma, \Delta)$ is stable. Let us define an interpretation $\mathcal{I}$ of $\Sigma_s(\mathbf{RAFN})$ 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 $x \in \mathbf{R}$ (resp. $\in \mathbf{N}$), $\mathcal{I}(Val(x)) = \textit{true}$ 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 $\Sigma_s(\mathbf{RAFN})$. And, for proving that $\mathcal{I}$ is a support-founded model of $\Sigma_s(\mathbf{RAFN})$ 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, $\mathcal{I}(Acc(x)) = \textit{true}$ and $\mathcal{I}(Supp(x)) = \textit{true}$. 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 3, $x \notin UnSupp(U)$ 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 $\mathcal{I}(Acc(x)) = \textit{true}$. As U is stable, U is self-supporting, so $x \in Sup(U)$, then it holds that $\mathcal{I}(Supp(x)) = \textit{true}$. As a consequence, $\mathcal{I}(sAcc(x)) = \textit{true}$ 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 $y \in \mathbf{R}$ and $x \in \mathbf{A}$ with $x = T(y)$, $\mathcal{I}(sVal(y) = \textit{true}$ and $\mathcal{I}(\forall z \in Arg(S(z,y) \rightarrow sAcc(z)) = \textit{true}$. Then $s_y \subseteq S$ 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 $\mathcal{I}(Acc(x)) = \textit{false}$, by definition of $\mathcal{I}(Acc)$ and then $\mathcal{I}(NAcc(x)) = \textit{true}$, 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 $\mathcal{I}(Acc(x)) = \textit{false}$. 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 $s_y \subseteq S$. Hence, there is $y \in \Gamma_\mathcal{I}$ with $x = T(y)$ and $s_y \subseteq S_\mathcal{I}$, or equivalently, there is $y \in \mathbf{R}$ with $x = T(y)$ and $\mathcal{I}(sVal(y)) = \textit{true}$ and $\mathcal{I}(\forall z \in Arg(S(z,y) \rightarrow sAcc(z))) = \textit{true}$. 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 $x \in \mathbf{A} \cup \mathbf{R} \cup \mathbf{N}$ such that $\mathcal{I}(Supp(x)) = \textit{false}$. By definition of $\mathcal{I}(Supp)$, $x \notin Sup(u)$. Due to Lemma 3, it follows that $x \in UnSupp(U)$, hence $\mathcal{I}(UnSupp(x)) = \textit{true}$, by definition of $\mathcal{I}(UnSupp)$. We have proved that $\mathcal{I}$ satisfies formula **(19)**. So $\mathcal{I}$ is a model of $\Sigma_s(\mathbf{RAFN})$.

★ 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 $\Sigma_s(\mathbf{RAFN})$. We have to prove that the structure $U = (S_\mathcal{I}, \Gamma_\mathcal{I}, \Delta_\mathcal{I})$ is stable.
$\Sigma_s(\mathbf{RAFN})$ contains $\Sigma(\mathbf{RAFN})$ and 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 conflict-free and self-supporting.
It remains to prove that $U = Acc(U) = \overline{UnAcc(U)}$.
First, consider $x \in \mathbf{A}$ such that $x \in \overline{U}$. So $x \notin S_\mathcal{I}$ and by definition of $S_\mathcal{I}$, $\mathcal{I}(sAcc(x)) = \textit{false}$. As $\mathcal{I}$ satisfies formula **(2bis)**, it follows that $\mathcal{I}(Acc(x)) = \textit{false}$ or $\mathcal{I}(Supp(x)) = \textit{false}$. In the case when $\mathcal{I}(Acc(x)) = \textit{false}$, as $\mathcal{I}$ satisfies formula **(15)**, it follows that $x \in Def(U)$. If $\mathcal{I}(Acc(x)) = \textit{true}$, it holds that $\mathcal{I}(Supp(x)) = \textit{false}$. As $\mathcal{I}$ satisfies formula **(19)**, it follows that $\mathcal{I}(UnSupp(x)) = \textit{true}$, so $x \in UnSupp(U)$ (following Condition 2 of Definition 19 since $\mathcal{I}$ is support-founded). In both cases, we have that $x \in UnAcc(U)$. The same proof can be done for $x \in \mathbf{R} \cap \mathbf{N}$. So $\overline{UnAcc(U)} \subseteq U$. Moreover following Lemma 2, we have $Acc(U) \subseteq \overline{UnAcc(U)} \subseteq U$.

Second, let prove that $U \subseteq Acc(U)$. Consider $x \in \mathbf{A}$ such that $x \in U$. So $x \in S_\mathcal{I}$ and by definition of $S_\mathcal{I}$, $\mathcal{I}(sAcc(x)) = \textit{true}$. As $\mathcal{I}$ satisfies formula **(2bis)**, it follows that $\mathcal{I}(Acc(x)) = \textit{true}$ and $\mathcal{I}(Supp(x)) = \textit{true}$. Since $\mathcal{I}(Acc(x)) = \textit{true}$, then using the contrapositive of formula **(3)**, we obtain that $\mathcal{I}(NAcc(x)) = \textit{false}$; then following the contrapositive of formula **(2)**, we must have that, for any attack y targeting x, $\mathcal{I}(sVal(y) \wedge sAcc(s_y)) = \textit{false}$ (here we simplify the original formula since in a RAFN the source of an attack is a singleton); at this step, four cases are possible: either $\mathcal{I}(Val(y)) = \textit{false}$, or

$\mathcal{I}(Supp(y)) =$ *false*, or $\mathcal{I}(Acc(s_y)) =$ *false*, or $\mathcal{I}(Supp(s_y)) =$ *false*; consider for instance the first case: since $\mathcal{I}(Val(y)) =$ *false* then using formula **(16)**, we can conclude that $y \in Def(U)$, so $y \in UnAcc(U)$ and thus $y \in UnAct(U)$; this reasoning can be applied for the three other cases, using either formula **(15)** (for the third case), or formula **(19)** (for the two other cases) and leads to the same conclusion: $y \in UnAct(U)$; so in each case, $x \in Acc(U)$ (recalled that, U being self-supporting, $x \in Sup(U)$). The same proof can be done for $x \in \mathbf{R} \cap \mathbf{N}$. And so $U \subseteq Acc(U)$.
In conclusion, $U = Acc(U) = \overline{UnAcc(U)}$. So U is stable.