RAFDivider

Sylvie Doutre, Marie-Christine Lagasquie-Schiex

▶ **To cite this version:**

Sylvie Doutre, Marie-Christine Lagasquie-Schiex. RAFDivider. [Research Report] IRIT/RR–2022–07–FR, IRIT : Institut de Recherche en Informatique de Toulouse. 2022, pp.1-48. hal-03719439

HAL Id: hal-03719439

https://ut3-toulouseinp.hal.science/hal-03719439

Submitted on 11 Jul 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

RAFDivider

Sylvie DOUTRE
Marie-Christine LAGASQUIE-SCHIEX

(special acknowledgement to Mickaël LAFAGES)

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

Tech. Report
IRIT/RR- -2022- -07- -FR

June 2022

Abstract

The topic of this work is related to a computational issue concerning an enriched abstract argumentation framework called RAF ("Recursive Argumentation Framework"). A RAF is composed of a set of arguments and a binary relation modelling the attacks as in Dung's framework. The main difference between Dung's framework and RAF is the fact that a RAF is able to take into account *higher-order* interactions (*i.e.* an attack can target an attack and not only an argument). Since this kind of framework is relatively recent, the efficient computation of the main semantics remains an open question.

In this paper, we propose one of the first algorithms dedicated to this issue. We also prove the soundness and the completeness of our algorithms.

Contents

1 Motivation **4**

2 Background **4**
- 2.1 Dung Argumentation Framework (AF) . . . 4
- 2.2 AFDivider: an algorithm for AF . . . 6
- 2.3 RAF: a Higher-order Argumentation Framework . . . 9
  - 2.3.1 RAF structure-based semantics . . . 9
  - 2.3.2 RAF labellings . . . 10
  - 2.3.3 Decomposability of RAF and RAF semantics . . . 11

3 Some Preliminary Definitions **14**

4 A Generic Algorithm for RAF: Presentation by Example **16**
- 4.1 Pretreatment: Removing the Trivial Part . . . 16
- 4.2 Identifying Clusters . . . 21
- 4.3 Computing the Labellings . . . 21
- 4.4 Reunifying the Results . . . 31
  - 4.4.1 Component labelling reunification . . . 32
  - 4.4.2 Whole RAF labelling reunification . . . 35
- 4.5 Synthesis of the running example . . . 35

5 *RAFDivider*: Algorithms and Properties **35**

6 A Clustering Method **38**

7 Conclusion and Future Works **40**

A Proofs **41**

1 Motivation

Argumentation, by considering arguments and their interactions, is a way of reasoning that has proven successful in many contexts, multi-agent applications for instance (e.g. [9]). Considering a formal representation of this reasoning model, argumentation frameworks with higher-order attacks (e.g. [7, 23, 24, 5, 6]) are a rich extension of the classical Argumentation Framework (AF) by [17]: not only they consider arguments and attacks between arguments, but also attacks on attacks (see for instance [5, 6]). Among these frameworks, the Recursive Argumentation Framework (RAF) by [11] proposes a direct approach regarding acceptability, which outputs sets of arguments and/or attacks (defined under the notion of structure), keeping the full expressiveness of higher-order attacks. A correspondence between Dung's extension-based semantics for AF and structure-based semantics of RAF without any attack on attacks has been shown in [11], proving that RAF are a conservative generalisation of AF. This characteristic makes RAF particularly interesting to consider.

The computation of semantics of RAF has not been addressed so far but a simple way to do so can be to extend what is done for AF: some of the most efficient algorithms for computing AF semantics are based on a cutting of the AF and then on a distributed and parallel computation (see [12, 19, 22, 16]) using the notion of AF labellings (e.g. [8, 4])[1] and the fact that such semantics are decomposable (see [3]). Indeed, RAF labellings already exist and classical decision problems for AF were also adapted to RAF with an interesting result (see [15]): even if the expressive power of the frameworks with higher-order attacks is higher, the complexity of their decision problems keeps the same as in AF. Moreover, it has been proven in [21] that RAF semantics, as AF semantics, are decomposable. Thus, following the line of the *AFDivider* algorithm designed for AF [16], all the mandatory elements are now present for the definition of some efficient algorithms for computing RAF labellings using a distributed and parallel method; this is the topic of the present paper.

The paper is organised as follows: the basics of Dung's argumentation framework are recalled in Section 2.1; an existing algorithm proposing a distributed and parallel computation for AF semantics is described in Section 2.2. Recursive Argumentation Frameworks (RAF) and their semantics are recalled in Section 2.3. Section 3 gives some additional definitions mandatory before the presentation of the algorithm itself in Sections 4 and 5. An example of a clustering method is provided in Section 6. Section 7 draws conclusions and opens future perspectives. The proofs of the soundness and completeness of the approach can be found in Appendix A.

2 Background

In this section, we first recall the framework proposed by Dung in 1995. Then a distributed and parallel algorithm for the computation of AF semantics is presented (our own algorithms will be strongly inspired by this one). The last part of this section is related to the main definitions concerning RAFs: basics, labellings and decomposability of the semantics.

2.1 Dung Argumentation Framework (AF)

In the original setting [17], an argumentation framework can be identified with a directed graph.

Definition 1 (Dung's framework -AF)**.** *A* Dung's Argumentation Framework (AF) *is an ordered pair* $\mathcal{AF} = (A, K)$ *s.t. A is a given set and K is a binary relation over A:* $K \subseteq A \times A$.

Each element $a \in A$ is called an *argument* and aKb means that *a attacks b*. For $S \subseteq A$, we say that S attacks $a \in A$ iff bKa for some $b \in S$.

As an illustration, Figure 1 depicts such an Argumentation Framework.

The main asset of Dung's approach is the definition of semantics using some basic properties in order to define sets of acceptable arguments, as follows.

[1] Whereas an extension assigns to its elements an accepted or a rejected status, a labelling considers a third status, undecided, which applies to arguments which are neither accepted, nor rejected. This enrichment has proven useful for the computation of acceptance statuses in AF (see [14] for a survey).

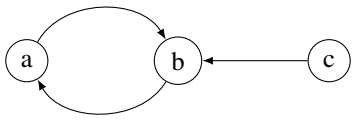


Figure 1: Example of an Argumentation Framework (AF)
with $A = \{a, b, c\}$ and $K = \{(a,b), (b,a), (c,b)\}$

Definition 2 (acceptability)**.** *Given $\mathcal{AF} = (A, K)$, an argument $a \in A$ is* acceptable *wrt $S \subseteq A$ iff for all $b \in A$, if bKa then cKb for some $c \in S$.*

Definition 3 (Characteristic function)**.** *The* characteristic function *of $\mathcal{AF} = (A, K)$ is $F_{\mathcal{AF}} : 2^A \to 2^A$ s.t. $F_{\mathcal{AF}}(S) = \{a \in A \mid a \text{ is acceptable wrt } S\}$ for all $S \subseteq A$.*

The semantics originally defined in [17] are as follows.

Definition 4 (Semantics)**.** *Given $\mathcal{AF} = (A, K)$, a subset S of A is said to be:*

- conflict-free *iff there are no a and b in S s.t. a attacks b,*
- admissible *iff S is conflict-free and for all $a \in S$, a is acceptable wrt S,*
- complete *iff S is admissible and for all $a \in A$, if a is acceptable wrt S then $a \in S$,*
- preferred *iff S is maximal (in the sense of set inclusion) admissible,*[2]
- grounded *iff S is the least fixpoint for $F_{\mathcal{AF}}$,*
- stable *iff S is conflict-free and S attacks all $a \in A \setminus S$.*

In this document, σ will denote the semantics with $\sigma \in \{$conflict-free, admissible, complete, preferred, grounded, stable$\}$ and the set $\sigma(\mathcal{AF})$ will denote the set of all the extensions produced using σ.

Back to Figure 1, the complete semantics induces a singleton containing the unique extension $\{a, c\}$, that is also the unique preferred, grounded and stable extension.

Some properties have been proven in [17] establishing a link between the different semantics. For instance:

Proposition 1. *Given $\mathcal{AF} = (A, K)$,*

- *There exists at least a preferred extension.*
- *Every preferred extension is complete, but not vice-versa.*
- *Every stable extension is preferred, but not vice-versa.*
- *The grounded extension is the least (with respect to set-inclusion) complete extension.*

Dung-like semantics can also be defined in terms of labellings as introduced in [8]. A labelling maps to each argument of an AF a value representing its acceptability status. This status may be accepted (`in`), rejected (`out`) or in an undecided state (`und`). Formally:

Definition 5 (Labelling)**.** *Let $\mathcal{AF} = \langle A, K \rangle$ be an AF, and $S \subseteq A$. A* labelling *of S is a total function $\ell : S \to \{\texttt{in}, \texttt{out}, \texttt{und}\}$. A* labelling *of $\mathcal{AF}$ is a labelling of A. The set of all labellings of $\mathcal{AF}$ is denoted as $\mathscr{L}(\mathcal{AF})$. The set of all labellings of a set of arguments S is denoted as $\mathscr{L}(S)$.*

We write $\texttt{in}(\ell)$ for $\{a | \ell(a) = \texttt{in}\}$, $\texttt{out}(\ell)$ for $\{a | \ell(a) = \texttt{out}\}$ and $\texttt{und}(\ell)$ for $\{a | \ell(a) = \texttt{und}\}$.

[2] We write $\subseteq$-*maximal*.

Definition 6 (Legally Labelled argument). *Let $\mathcal{AF} = \langle A, K \rangle$ be an AF, and $\ell \in \mathscr{L}(\mathcal{AF})$ be a Labelling.*

- *An $\mathtt{in}$-labelled argument is said to be legally $\mathtt{in}$ iff all its attackers are labelled $\mathtt{out}$.*
- *An $\mathtt{out}$-labelled argument is said to be legally $\mathtt{out}$ iff at least one of its attackers is labelled $\mathtt{in}$.*
- *An $\mathtt{und}$-labelled argument is said to be legally $\mathtt{und}$ iff it does not have an attacker that is labelled $\mathtt{in}$ and one of its attackers is not labelled $\mathtt{out}$.*

Definition 7 (Reinstatement labelling). *Let $\mathcal{AF} = \langle A, K \rangle$ be an AF, and $\ell \in \mathscr{L}(\mathcal{AF})$ be a labelling. ℓ is a reinstatement labelling of $\mathcal{AF}$ iff it satisfies the following conditions for any $a \in A$:*

- *For each $a \in \mathtt{in}(\ell)$, a is legally $\mathtt{in}$.*
- *For each $a \in \mathtt{out}(\ell)$, a is legally $\mathtt{out}$.*

Then using the reinstatement labellings and some specific restrictions, it is possible to define many other labellings corresponding to some extension-based semantics.

Restriction on AF reinstatement labelling	Semantics
no restrictions	*complete* semantics
empty $\mathtt{und}$	*stable* semantics
maximal $\mathtt{in}$	*preferred* semantics
maximal $\mathtt{out}$	
maximal $\mathtt{und}$	*grounded* semantics
minimal $\mathtt{in}$	
minimal $\mathtt{out}$	

Table 1: Reinstatement labellings and extension-based semantics correspondence

2.2 AFDivider: an algorithm for AF

Finding all the possible solutions of a semantics for a given AF can be very time consuming. Many AF instances, particularly large,[3] are too hard to be solved in an acceptable amount of time, as shown by the results of the ICCMA argumentation solver competition.[4] Formally, the so-called *enumeration problem* is defined as follows:

Definition 8 (Enumeration Problem). *Given $\mathcal{AF} = \langle A, K \rangle$ and a semantics σ, compute the set $\sigma(\mathcal{AF})$ corresponding to the AF solutions.*

The hardness of this problem is not relative to the current state of the art but rather to the intrinsic theoretical complexity of the semantics that are tackled.[5]

Enhancing the computational time of enumerating the solutions of an AF has been the object of study of many works, resulting in the elaboration of several recent algorithms such as [1, 13, 22, 2] (see [14] for an overview). During his thesis [21], Mickaël Lafages addressed this issue with the proposal of an algorithm, the so-called: *AFDivider*.

[3] This notion of largeness of an AF is not so simple to define. It is related to the fact that the computation of the solutions is complex either because of the number of arguments, or of the number of interactions, or because of the structure of the AF.

[4] `http://argumentationcompetition.org`

[5] Notice that in the literature it is the decision problem versions of this problem that are studied. Nevertheless, the complexity of their decision versions is sufficient to give a good idea of their hardness. See [18] for an overview.

The idea that led to his algorithm is that argumentation frameworks constructed from real data should have a particular structure. Indeed, people have themes and goals while arguing. It is thus a reasonable conjecture to say that the AFs obtained from real argumentation are not random and that they have a relatively low density of relations between arguments (it would be very surprising if, in some argumentation, any argument attacks a large part of the other ones).

The *AFDivider* algorithm takes advantage of this sparsity[6] of AF graphs. To do so, it uses methods that have not yet been considered for this purpose (namely clustering methods used in an original way), combined with techniques that have already been applied in other existing algorithms (distributed and parallel methods). Note that this kind of algorithms can be defined since AF semantics are decomposable (*i.e.* the computation of the results of a given semantics can be built using the results of this same semantics on sub-parts of the initial AF, see [3]).

The *AFDivider* algorithm solves the enumeration problem using labelling-based semantics. It has been designed for Dung original semantics: the ***complete***, the ***stable*** and the ***preferred*** semantics. Given that the *grounded* semantics can be computed in linear time and that it gives only one labelling, computing it with the *AFDivider* is unappropriated.

Given an argumentation framework $\mathcal{AF} = \langle A, K \rangle$ and a semantics $\sigma \in \{complete, stable, preferred\}$, the *AFDivider* algorithm, rather than building labellings that cover the whole AF (which could be time consuming), computes the semantics labellings using a distributed and clustering-based method. Here are its four major steps graphically represented in Figure 2:

1. A pretreatment on $\mathcal{AF}$ removes "trivial" parts of it using the grounded labelling.
2. Clusters in $\mathcal{AF}$ are identified.
3. The labellings under semantics σ in each of these clusters are computed **in parallel**.
4. The results of each cluster are reunified to get the labellings of $\mathcal{AF}$.

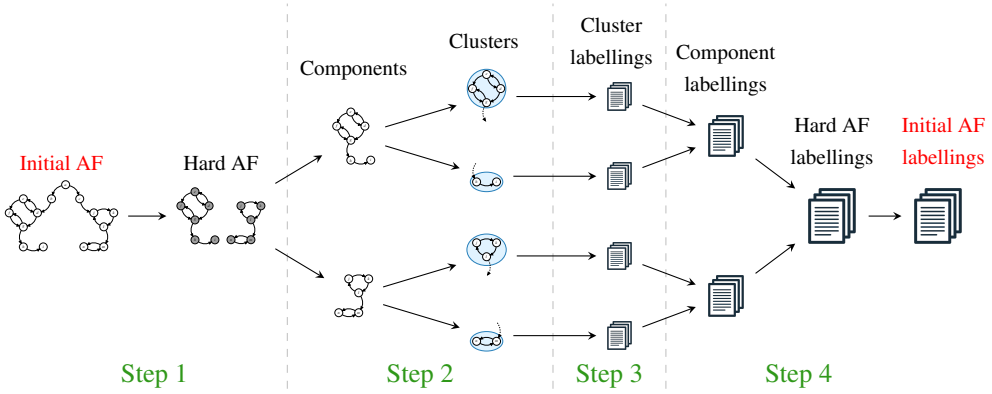


Figure 2: *AFDivider* operating diagram

Algorithms 1 and 2 give the formal definition of the *AFDivider* algorithm. They are said to be generic algorithms in the sense that:

- Any clustering method can be used to split the AF.
- Any sound and complete procedure that computes the semantics σ can be used to compute the labellings of the different clusters.

[6] A graph is said to be Sparse when its density is low.

Algorithm 1: *AFDivider* algorithm.

Input: Let $\mathcal{AF} = \langle A, K \rangle$ be an AF and σ be a semantics
Result: $\mathscr{L}_\sigma \in 2^{\mathscr{L}(\mathcal{AF})}$: the set of the σ-labellings of $\mathcal{AF}$
Local variables:
- ℓ'_{gr}: the *grounded* labelling restricted to the arguments labelled $\mathtt{in}$ and $\mathtt{out}$
- *CCSet*: the set of connected components of $\mathcal{AF}_{hard}$ ($\mathcal{AF}$ in which the trivial part has been removed)
- *ClustSet*: the set of cluster structures of af_i
- $\mathscr{L}_\sigma(af_i)$: the set of all σ-labellings of af_i

$\ell'_{gr} \leftarrow \texttt{ComputeGroundedLabelling}(\mathcal{AF})$
$CCSet \leftarrow \texttt{SplitConnectedComponents}(\mathcal{AF}, \ell'_{gr})$
for all $af_i \in CCSet$ **do in parallel**
  $ClustSet \leftarrow \texttt{ComputeClusters}(af_i)$
  $\mathscr{L}_\sigma(af_i) \leftarrow \texttt{ComputeCompLabs}(\sigma, ClustSet)$
$\mathscr{L}_\sigma \leftarrow \varnothing$
if $\nexists af_i \in CCSet$ *s.t.* $\mathscr{L}_\sigma(af_i) = \varnothing$ **then** $\mathscr{L}_\sigma \leftarrow \{\ell'_{gr}\} \times \prod_{af_i \in CCSet} \mathscr{L}_\sigma(af_i)$
return $\mathscr{L}_\sigma$

Algorithm 2: `ComputeCompLabs` algorithm.

Input: Let *ClustSet* be a set of cluster structures for a component af, σ be a semantics
Result: $\mathscr{L}_\sigma \in 2^{\mathscr{L}(af)}$: the set of the σ-labellings of af
Local variables:
- κ_j: a cluster structure
- $\mathscr{L}_\sigma^{\kappa_j}$: the set of all σ-labellings of κ_j
- $\mathscr{P}^{\kappa_j}$: the set of configurations corresponding to the σ-labellings of κ_j
- $\mathscr{P}$: the set of all reunified labelling profiles

for all $\kappa_j \in ClustSet$ **do in parallel**
  $\mathscr{L}_\sigma^{\kappa_j} \leftarrow \texttt{ComputeClustLabs}(\sigma, \kappa_j)$
  $\mathscr{P}^{\kappa_j} \leftarrow \texttt{IdentifyConfigs}(\mathscr{L}_\sigma^{\kappa_j}, \kappa_j)$
$\mathscr{L}_\sigma = \varnothing$
$\mathscr{P} = \texttt{ReunifyCompConfigs}(\bigcup_{\kappa_j \in ClustSet} \mathscr{P}^{\kappa_j}, ClustSet)$
for all $p \in \mathscr{P}$ **do**
  $\mathscr{L}_\sigma \leftarrow \mathscr{L}_\sigma \cup \left(\prod_{\xi \in p} \{\ell | \ell \in \texttt{ProfileLabellings}(\xi, \bigcup_{\kappa_j \in ClustSet} \mathscr{L}_\sigma^{\kappa_j})\} \right)$
if $\sigma = pr$ **then** $\mathscr{L}_\sigma \leftarrow \{\ell | \ell \in \mathscr{L}_\sigma \text{ s.t. } \nexists \ell' \in \mathscr{L}_\sigma \text{ s.t. } \mathtt{in}(\ell) \subset \mathtt{in}(\ell')\}$
return $\mathscr{L}_\sigma$

2.3 RAF: a Higher-order Argumentation Framework

Higher-order attacks (that is, possibly targeting attacks as well as arguments) has been introduced in [7] then developed in several papers among which one can cite the AFRA (Argumentation Framework with Recursive Attacks) approach described in [6] and the RAF (Recursive Argumentation Framework) approach introduced in [11]. RAF and AFRA differ upon the way these attacks are handled despite the fact that there is no difference in the structure of the graph. This paper is concerned by the RAF approach. Note that AFRA and RAF give the same results in terms of semantics even if some intermediate results are different (see in [11] for a comparison between these approaches).

Definition 9. (Recursive argumentation framework - RAF). *A* Recursive Argumentation Framework (RAF) $\mathcal{RAF} = \langle A, K, s, t \rangle$ *is a quadruple where A and K are (possibly infinite) disjoint sets respectively representing* arguments *and* attack *names, and where* $s : K \to A$ *and* $t : K \to A \cup K$ *are functions respectively mapping each attack name to its* source *and to its* target.

Figure 5 shows an example of a RAF. There are two different possibilities for defining the semantics of a RAF: either by selecting some specific structures (a pair composed of a set of arguments and a set of attacks) [11] or by using labellings [15].[7]

2.3.1 RAF structure-based semantics

What differs from AF to RAF is that in a RAF an attack can have an attack for target. As a consequence, an attack is not always "*valid*". In order to express this fact, a RAF "*structure*-based semantic", that is, a function that defines the solutions of a RAF produces *structures*: a couple whose first element is a set of arguments and the second, a set of attacks. The idea behind the notion of *structure* is that when presented together, the elements of the structure (*i.e.* arguments plus attacks) win the argumentation.

A *structure* is thus defined as follows:

Definition 10 (Structure)**.** *A pair* $\mathcal{U} = \langle S, Q \rangle$ *is said to be a* structure *of some* $\mathcal{RAF} = \langle A, K, s, t \rangle$ *if it satisfies:* $S \subseteq A$ *and* $Q \subseteq K$*. Notice that by* $x \in \mathcal{U}$ *we mean:* $x \in S \cup Q$*.*

Intuitively, the set S represents the set of "acceptable arguments" *w.r.t.* the structure $\mathcal{U}$, while Q represents the set of "valid attacks" *w.r.t.* $\mathcal{U}$. Any attack that does not belong to Q is understood as non-valid and, in this sense, it cannot defeat its target.

Definition 11 (Defeat and Inhibition in RAF)**.** *Let* $\mathcal{U} = \langle S, Q \rangle$ *be a structure. The set of all arguments* defeated *by* $\mathcal{U}$*, denoted RAF-Def*$(\mathcal{U})$*, is defined as follows:*

$$\textit{RAF-Def}(\mathcal{U}) = \{a \in A | \exists \alpha \in Q \text{ s.t. } s(\alpha) \in S \text{ and } t(\alpha) = a\}$$

The set of all attacks inhibited *by* $\mathcal{U}$*, denoted RAF-Inh*$(\mathcal{U})$*, is defined as follows:*

$$\textit{RAF-Inh}(\mathcal{U}) = \{\alpha \in K | \exists \beta \in Q \text{ s.t. } s(\beta) \in S \text{ and } t(\beta) = \alpha\}$$

The counterpart of defeat/inhibition is the notion of *acceptance*:

Definition 12 (RAF Acceptability)**.** *An element* $x \in (A \cup K)$ *is said to be acceptable w.r.t. some structure* $\mathcal{U}$ *iff every attack* $\alpha \in K$ *with* $t(\alpha) = x$ *satisfies one of the two following conditions:*

- $s(\alpha) \in \textit{RAF-Def}(\mathcal{U})$
- $\alpha \in \textit{RAF-Inh}(\mathcal{U})$

By RAF-Acc$(\mathcal{U})$ *we denote the set containing all acceptable arguments and attacks with respect to* $\mathcal{U}$*.*

[7] Relations between labelling-based semantics and structure-based semantics have been exhibited in [15].

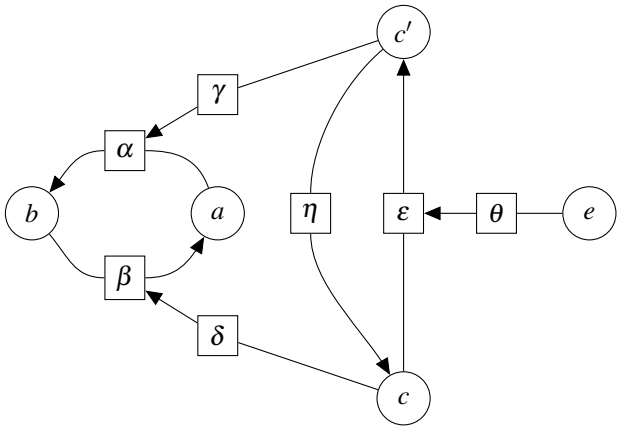


Figure 3: A example of RAF (arguments are in circles and the Greek letters labelling attacks are within squares)

For any pair of structures $\mathcal{U} = \langle S, Q \rangle$ and $\mathcal{U}' = \langle S', Q' \rangle$, we write $\mathcal{U}' \sqsubseteq \mathcal{U}'$ *iff* $(S \cup Q) \subseteq (S' \cup Q')$ and we write $\mathcal{U} \sqsubseteq_{ar} \mathcal{U}'$ *iff* $S \subseteq S'$. As usual, we say that a structure $\mathcal{U}$ is $\sqsubseteq$-maximal (resp. $\sqsubseteq_{ar}$-maximal) *iff* every $\mathcal{U}'$ that satisfies $\mathcal{U} \sqsubseteq \mathcal{U}'$ (resp. $\mathcal{U} \sqsubseteq_{ar} \mathcal{U}'$) also satisfies $\mathcal{U}' \sqsubseteq \mathcal{U}$ (resp. $\mathcal{U}' \sqsubseteq_{ar} \mathcal{U}$).

Inspired by Dung's AF semantics, the first RAF structure-based semantics, that have been defined in [11], are the following ones:

Definition 13 (RAF structure-based semantics)**.** *Let $\mathcal{U} = \langle S, Q \rangle$ be a structure over some RAF $\mathcal{RAF} = \langle A, K, s, t \rangle$. $\mathcal{U}$ is said to be:*

1. *RAF*-conflict-free *iff* $S \cap \mathit{RAF\text{-}Def}(\mathcal{U}) = \varnothing$ *and* $Q \cap \mathit{RAF\text{-}Inh}(\mathcal{U}) = \varnothing$.
2. *RAF*-admissible *iff it is RAF*-conflict-free *and* $(S \cup Q) \subseteq \mathit{RAF\text{-}Acc}(\mathcal{U})$.
3. *RAF*-complete *iff it is RAF*-conflict-free *and* $(S \cup Q) = \mathit{RAF\text{-}Acc}(\mathcal{U})$.
4. *RAF*-grounded *iff it is a $\sqsubseteq$-minimal RAF*-complete *structure.*
5. *RAF*-preferred *iff it is a $\sqsubseteq$-maximal RAF*-admissible *structure.*
6. *RAF*-stable *iff* $S = A \setminus \mathit{RAF\text{-}Def}(\mathcal{U})$ *and* $Q = K \setminus \mathit{RAF\text{-}Inh}(\mathcal{U})$.

Example 1. *Let consider the RAF shown in Figure 3. The semantics mentioned in Definition 13 produce a singleton containing a unique structure that is the grounded, complete, preferred and stable structure:* $\langle \{b, c', e\}, \{\beta, \delta, \gamma, \eta, \theta\} \rangle$.

In [11], the reader will find several interesting properties of these structure-based semantics (existence, cardinality for instance).

2.3.2 RAF labellings

As for AF, a second way for defining semantics exists: the use of labellings, see [15].[8]

Definition 14. (RAF labelling). Let $\mathcal{RAF} = \langle A, K, s, t \rangle$ be a recursive argumentation framework. A RAF labelling is a total function $\mathcal{L} : A \cup K \to \{\mathtt{in}, \mathtt{out}, \mathtt{und}\}$. We define $\mathtt{in}(\mathcal{L})$ (resp. $\mathtt{out}(\mathcal{L})$, $\mathtt{und}(\mathcal{L})$) as the set $\{x \in A \cup K | \mathcal{L}(x) = \mathtt{in}$ (resp. $\mathtt{out}$, $\mathtt{und}$)$\}$.

[8] The original definition for the RAF labelling is a pair of a labelling over arguments and a labelling over attacks. Here we give an equivalent definition that does not make difference between arguments and attacks. Moreover we also simplify the terminology using "labelling" in place of "structure labelling".

Definition 15. (Reinstatement RAF labelling). Let $\mathcal{RAF} = \langle A, K, s, t \rangle$ be a recursive argumentation framework and $\mathcal{L}$ be a RAF labelling.

$\mathcal{L}$ is a *reinstatement RAF labelling iff* it satisfies: $\forall x \in (A \cup K)$,

- $(\mathcal{L}(x) = \texttt{out}) \iff (\exists \alpha \in K \text{ s.t. } t(\alpha) = x, \mathcal{L}(\alpha) = \texttt{in} \text{ and } \mathcal{L}(s(\alpha)) = \texttt{in})$
- $(\mathcal{L}(x) = \texttt{in}) \iff (\forall \alpha \in K \text{ s.t. } t(\alpha) = x, \mathcal{L}(\alpha) = \texttt{out} \text{ or } \mathcal{L}(s(\alpha)) = \texttt{out})$

An equivalent definition of reinstatement RAF labelling can be made, as for AF, using the notion of "*legally labelled argument*". An $\texttt{in}$-labelled element is said to be *legally* $\texttt{in}$ *iff* all its attackers or their involved attacks are labelled $\texttt{out}$. An $\texttt{out}$-labelled element is said to be *legally* $\texttt{out}$ *iff* at least one of its attackers and the involved attacks are labelled $\texttt{in}$. An $\texttt{und}$-labelled element is said to be *legally* $\texttt{und}$ *iff* it does not have any attacker and its involved attack that are labelled $\texttt{in}$ and one of its attackers and the involved attack are not labelled $\texttt{out}$. Formally, "*valid labellings*" (notion equivalent to reinstatement RAF labellings) are defined as follows:

Definition 16 (Legally labelled elements, valid RAF labelling)**.**
Let $\mathcal{RAF} = \langle A, K, s, t \rangle$ be a recursive argumentation framework and $\mathcal{L}$ be a RAF labelling over $\mathcal{RAF}$.

Let $x \in A \cup K$ be an element of $\mathcal{RAF}$. x is said to be legally labelled *in $\mathcal{L}$ iff it is a reinstatement labelling and $x \in \texttt{und}(\mathcal{L})$ iff $((\nexists \alpha \in K \text{ s.t. } t(\alpha) = x, \mathcal{L}(\alpha) = \texttt{in} \text{ and } \mathcal{L}(s(\alpha)) = \texttt{in})$ and $(\exists \alpha \in K \text{ s.t. } t(\alpha) = x, \mathcal{L}(\alpha) \neq \texttt{out} \text{ and } \mathcal{L}(s(\alpha)) \neq \texttt{out}))$. $\mathcal{L}$ is said to be a* valid RAF labelling *if all its elements are legally labelled.*

Regarding the RAF of Figure 5, Example 2 shows its grounded labelling, Example 20 gives one of its complete labellings.

As for AF, there exists a one-to-one mapping between RAF labellings and structure-based semantics. Table 2 sums up these relations.

Restriction on Reinstatement RAF labelling	Structure-based Semantics
no restrictions	*complete* semantics
empty $\texttt{und}$	*stable* semantics
maximal $\texttt{in}$	*preferred* semantics
maximal $\texttt{out}$	
maximal $\texttt{und}$	*grounded* semantics
minimal $\texttt{in}$	
minimal $\texttt{out}$	

Table 2: Links between Reinstatement RAF labellings and structure-based semantics

2.3.3 Decomposability of RAF and RAF semantics

In order to define an algorithm able to answer the enumeration problem in the case of a RAF, similarly to the one given for an AF (*AFDivider* [16]), we must be able to split a RAF. Based on the notion introduced in [3], any AF can be split into several sub-frameworks by simply ignoring some attacks (that are always valid). Nevertheless, it is not the case for RAF. Attacks, as arguments, can be labelled $\texttt{in}$, $\texttt{out}$ or $\texttt{und}$. As a consequence, we cannot just ignore attacks to split a RAF. So, if we do not suppress attacks while splitting RAFs, we will have attacks without targets or without sources. Thus the result of such a split does not produce a RAF but a *partial RAF*.

Definition 17. (Partial RAF). Let $\mathcal{RAF} = \langle A, K, s, t \rangle$ be a RAF. A *partial RAF* $\widetilde{\mathcal{RAF}} = \langle \tilde{A}, \tilde{K}, \tilde{s}, \tilde{t}, s, t \rangle$ of $\mathcal{RAF}$ is a tuple where $\tilde{A} \subseteq A$ (resp. $\tilde{K} \subseteq K$) is a set representing arguments (resp. attacks) and:

- $\tilde{s} : \tilde{K} \rightarrow \{\texttt{true}, \texttt{false}\}$ is a boolean function that indicates whether or not an attack in $\tilde{K}$ has its source in $\tilde{A}$ defined as following:
$$\forall \alpha \in \tilde{K}, \tilde{s}(\alpha) = \texttt{true} \text{ if } s(\alpha) \in \tilde{A} \text{ otherwise } \texttt{false}$$
- $\tilde{t} : \tilde{K} \rightarrow \{\texttt{true}, \texttt{false}\}$ is a boolean function that indicates whether or not an attack in $\tilde{K}$ has its target in $\tilde{A} \cup \tilde{K}$
$$\forall \alpha \in \tilde{K}, \tilde{t}(\alpha) = \texttt{true} \text{ if } t(\alpha) \in \tilde{A} \cup \tilde{K} \text{ otherwise } \texttt{false}$$

Then, using the notion of partial RAF, a partition of a RAF can be defined:

Definition 18. (RAF partition). Let $\mathcal{RAF} = \langle A, K, s, t \rangle$ be a RAF. Let $\Omega = \{\omega_1, ..., \omega_n\}$ be a partition[9] of $(A \cup K)$. A *RAF partition* of $\mathcal{RAF}$ is a set of partial RAFs $\{\widetilde{\mathcal{RAF}}_1, ..., \widetilde{\mathcal{RAF}}_n\}$ s.t.:
$$\forall i, \widetilde{\mathcal{RAF}}_i = \left\langle \tilde{A}_i, \tilde{K}_i, \tilde{s}_i, \tilde{t}_i, s, t \right\rangle \text{ with:}$$

- $\tilde{A}_i = \omega_i \cap A$ and $\tilde{K}_i = \omega_i \cap K$
- $\tilde{s}_i : \tilde{K}_i \rightarrow \{\texttt{true}, \texttt{false}\}$ is a boolean function that indicates whether or not an attack in $\tilde{K}_i$ has its source in $\tilde{A}_i$ defined as following:
$$\forall \alpha \in \tilde{K}_i, \tilde{s}_i(\alpha) = \texttt{true} \text{ if } s(\alpha) \in \tilde{A}_i \text{ otherwise } \texttt{false}$$
- $\tilde{t}_i : \tilde{K}_i \rightarrow \{\texttt{true}, \texttt{false}\}$ is a boolean function that indicates whether or not an attack in $\tilde{K}_i$ has its target in $\tilde{A}_i \cup \tilde{K}_i$
$$\forall \alpha \in \tilde{K}_i, \tilde{t}_i(\alpha) = \texttt{true} \text{ if } t(\alpha) \in \tilde{A}_i \cup \tilde{K}_i \text{ otherwise } \texttt{false}$$

Considering a partial RAF implies to consider also its "inputs" and their labellings (note that several partial RAF with input can be built from a given partial RAF since several labellings can exist for its inputs):

Definition 19. (Partial RAF with input). Let $\mathcal{RAF} = \langle A, K, s, t \rangle$ be a RAF and $\widetilde{\mathcal{RAF}} = \langle \tilde{A}, \tilde{K}, \tilde{s}, \tilde{t}, s, t \rangle$ be a partial RAF of $\mathcal{RAF}$. The *input* $\mathcal{I}$ of $\widetilde{\mathcal{RAF}}$ is a tuple $\left\langle S^{inp}, Q^{inp} \right\rangle$ where:

- $S^{inp} = \{s(\alpha) \in (A \setminus \tilde{A}) | \alpha \in K \text{ and } t(\alpha) \in (\tilde{A} \cup \tilde{K})\}$
- $Q^{inp} = \{\alpha \in (K \setminus \tilde{K}) | t(\alpha) \in (\tilde{A} \cup \tilde{K})\}$

The tuple $\left\langle \widetilde{\mathcal{RAF}}, \mathcal{I}, \mathcal{L}^{inp} \right\rangle$ is called a *partial RAF with input*, where $\mathcal{L}^{inp}$ is a labelling of $\mathcal{I}$.

Then a standard RAF is the RAF that can be built from a partial RAF with inputs in order to simulate what happens if we take into account the labellings of these inputs (for instance, see in Figure 12 the standard RAF corresponding to the partial RAF given in Figure 9(b)):

Definition 20 (Standard RAF)**.** *Let $\mathcal{RAF} = \langle A, K, s, t \rangle$ be a RAF. Let $\left\langle \widetilde{\mathcal{RAF}}, \mathcal{I}, \mathcal{L}^{inp} \right\rangle$ be a partial RAF with input s.t. $\widetilde{\mathcal{RAF}} = \left\langle \tilde{A}, \tilde{K}, \tilde{s}, \tilde{t}, s, t \right\rangle$ is a partial RAF of $\mathcal{RAF}$. The standard RAF w.r.t. $\left\langle \widetilde{\mathcal{RAF}}, \mathcal{I}, \mathcal{L}^{inp} \right\rangle$ is a RAF defined as $\widetilde{\mathcal{RAF}}_s = \langle A_s, K_s, s_s, t_s \rangle$ where:*

- $A_s = \tilde{A} \cup S^{inp} \cup \{a_\upsilon, a_\rho, a_\zeta\}$
- $K_s = \tilde{K} \cup Q^{inp} \cup N \cup \{\alpha_\theta\}$*, with:*
  - $N = \{\alpha_x | x \in (Und \cup Out)\}$
  - $Out = \texttt{out}(\mathcal{L}^{inp})$
  - $Und = \texttt{und}(\mathcal{L}^{inp})$

And where $s_s : K_s \rightarrow A_s$ and $t_s : K_s \rightarrow (A_s \cup K_s)$ are functions respectively mapping each attack to its source and to its target and s.t.:

[9] So the following property holds for Ω: $\forall (i, j) \in \{1, ..., n\}$ s.t. $i \neq j, \omega_i \cap \omega_j = \varnothing$ and $\bigcup_{i=1}^{n} \omega_i = A \cup K$.

- $\forall \alpha \in (\tilde{K} \cup Q^{inp})$, $s_s(\alpha) = s(\alpha)$
- $\forall \alpha \in Q^{inp} \cup (\tilde{K} \setminus \{\alpha | \alpha \in \tilde{K}$ *s.t.* $\tilde{t}(\alpha)$ *is* `false`$\})$, $t_s(\alpha) = t(\alpha)$
- $\forall \alpha \in \{\alpha | \alpha \in \tilde{K}$ *s.t.* $\tilde{t}(\alpha)$ *is* `false`$\}$, $t_s(\alpha) = a_\zeta$
- $\forall \alpha_x \in \{\alpha_x \in N | x \in Out\}$, $s_s(\alpha_x) = a_\rho$
- $\forall \alpha_x \in \{\alpha_x \in N | x \in Und\}$, $s_s(\alpha_x) = a_\upsilon$
- $\forall \alpha_x \in N$, $t_s(\alpha_x) = x$
- $s_s(\alpha_\theta) = a_\upsilon$
- $t_s(\alpha_\theta) = a_\upsilon$

The intuition behind the new elements added in the standard RAF is the following:

- a_υ is the argument that will serve to label `und` an element of the RAF input.
- α_θ is the attack whose source and target is a_υ, making a_υ a self attacking argument and thus an argument that will be labelled `und`.
- a_ρ is the argument that will serve to label `out` an element of the RAF input.
- N is the set of attacks that will link a_υ and a_ρ to all elements of the RAF input that should be labelled `out` or `und`.
- a_ζ is an argument that will serve as the target of all attacks of the partial RAF whose target does not belong to the partial RAF.

Note that there is a one-to-one correspondence between a partial RAF with input and its standard RAF (and so potentially several standard RAF for a given partial RAF).

The canonical local function associates any partial RAF with input with a set of labellings built using its standard RAF:

Definition 21. (RAF canonical local function). Let σ be a semantics. A local function $\mathscr{F}_\sigma^{raf}$ assigns to any partial RAF with input $\left\langle \widetilde{\mathcal{RAF}}, \mathcal{I}, \mathcal{L}^{inp} \right\rangle$ a (possibly empty) set of labellings of $\widetilde{\mathcal{RAF}}$ under σ, *i.e.* $\mathscr{F}_\sigma^{raf}(\widetilde{\mathcal{RAF}}, \mathcal{I}, \mathcal{L}^{inp}) \in 2^{\{\mathcal{L} | \mathcal{L} \text{ being any labelling over } \widetilde{\mathcal{RAF}}\}}$.

The canonical local function $\mathscr{F}_\sigma^{raf}$ is a local function s.t. $\mathscr{F}_\sigma^{raf}(\widetilde{\mathcal{RAF}}, \mathcal{I}, \mathcal{L}^{inp}) = \{\mathcal{L} \downarrow_{\langle \tilde{A} \cup \tilde{K} \rangle} | \mathcal{L} \in \mathscr{L}_\sigma(\widetilde{\mathcal{RAF}}_s)\}$ ($\widetilde{\mathcal{RAF}}_s$ being the standard RAF associated with $\widetilde{\mathcal{RAF}}$).

The notion of semantics decomposability is thus as follows:

Definition 22. (Semantics decomposability).

A semantics σ is *full decomposable* iff there is a local function $\mathscr{F}_\sigma^{raf}$ s.t., for any RAF $\mathcal{RAF} = \langle A, K, s, t \rangle$ and any partition $\{\widetilde{\mathcal{RAF}}_1, ..., \widetilde{\mathcal{RAF}}_n\}$ of $\mathcal{RAF}$, the set of all possible labellings under the semantics σ of $\mathcal{RAF}$, denoted by $\mathscr{L}_\sigma(\mathcal{RAF})$, satisfies:

$$\mathscr{L}_\sigma(\mathcal{RAF}) = \{\mathcal{L}_1 \cup ... \cup \mathcal{L}_n | \forall i \in \{1, ..., n\}, \mathcal{L}_i \in \mathscr{F}_\sigma^{raf}(\widetilde{\mathcal{RAF}}_i, \mathcal{I}_i, \mathcal{L}_i^{inp})\}$$

with $\widetilde{\mathcal{RAF}}_i = \langle \tilde{A}_i, \tilde{K}_i, \tilde{s}_i, \tilde{t}_i, s, t \rangle$ and $\mathcal{I}_i = \left\langle S_i^{inp}, Q_i^{inp} \right\rangle$ and $\mathcal{L}_i^{inp}$ defined as following:

- $S_i^{inp} = \{s(\alpha) \notin \tilde{A}_i | \exists \alpha \in K \text{ s.t. } t(\alpha) \in (\tilde{A}_i \cup \tilde{K}_i)\}$
- $Q_i^{inp} = \{\alpha \notin \tilde{K}_i | \exists \alpha \in K \text{ s.t. } t(\alpha) \in (\tilde{A}_i \cup \tilde{K}_i)\}$

- $\mathcal{L}_i^{inp} = (\bigcup_{j\in\{1,...,n\} \text{ s.t. } j\neq i} \mathcal{L}_j)\downarrow_{\langle S_i^{inp}, Q_i^{inp}\rangle}$ [10]

A semantics σ is said to be *top-down* (resp. *bottom-up*) *decomposable* iff:

$$\mathscr{L}_\sigma(\mathcal{RAF}) \subseteq \text{ (resp. } \supseteq)\{\mathcal{L}_1 \cup ... \cup \mathcal{L}_n | \forall i \in \{1,...,n\}, \mathcal{L}_i \in \mathscr{F}_\sigma^{raf}(\widetilde{\mathcal{RAF}}_i, \mathcal{I}_i, \mathcal{L}_i^{inp})\}$$

In [21], a specific RAF partition selector has been defined that produces a partition respecting the strong connected components (SCC) of a RAF:[11]

Definition 23. (USCC RAF partition selector). Let $\mathcal{RAF}$ be a RAF. Let $\mathscr{S}_{raf\text{-}USCC}$ be the RAF partition selector s.t.:

$$\mathscr{S}_{raf\text{-}USCC}(\mathcal{RAF})$$
$$=$$
$$\{\Omega | \ \Omega \text{ is a partition of } \mathcal{RAF} \text{ and } \forall S \in SCCS_{raf}(\mathcal{RAF}), \exists \omega_i \in \Omega \text{ s.t. } \omega_i \cap S \neq \varnothing \implies S \subseteq \omega_i\}$$

Let $S \subseteq A \cup K$ s.t. $S \in \mathscr{S}_{raf\text{-}USCC}(\mathcal{RAF})$, S is called an "$USCC_{raf}$".

Then, in [21], the following proposition has been proven:

Proposition 2. *Let $\mathcal{RAF} = \langle A, K, s, t\rangle$ be any RAF. The semantics properties in Table 3 hold.*

	RAF semantics			
	complete	grounded	preferred	stable
Full decomposability	✓	×	×	✓
Top-down decomposability	✓	✓	✓	✓
Bottom-up decomposability	✓	×	×	✓
Full decomposability *w.r.t.* $\mathscr{S}_{raf\text{-}USCC}$ (so Top-down and Bottom-up decomposability)	✓	✓	✓	✓

"✓" (resp. "×") means that the semantics on the column has (resp. does not have) the property on the row.

Table 3: RAF Semantics decomposability properties

3 Some Preliminary Definitions

Before explaining our algorithm, some additional definitions are needed. First of all, let define the notions of "*non-directed RAF-walk*" and "*non-directed RAF-path*".

Definition 24 (Directed and Non-directed RAF-walk and RAF-path)**.** *Let $\mathcal{RAF} = \langle A, K, s, t\rangle$ be a RAF and $e_1, e_n \in (A \cup K)$ be elements of $\mathcal{RAF}$. A [non-]directed RAF-walk is a sequence $(e_1, ..., e_n)$ with $n \in \mathbb{N}^*$ s.t.:*[12]

- $\forall i \in \{1,...,n\}, e_i \in (A \cup K)$

[10] $\downarrow$ is the classical generic operator of restriction that allows the selection of a sub-part of a given "object" wrt to a given set of "elements". Here for instance, it produces the sub-part of the labellings concerning only the elements belonging to $S_i^{inp} \cup Q_i^{inp}$.

[11] See in [21] the details about the method for defining the SCC of a RAF.

[12] The definition for the non-directed case implies additional constraints given between brackets.

- *If* $n > 1$, $\forall i \in \{1,...,n-1\}$, $e_i \in A \implies e_{i+1} \in K$ *and* $e_i = t(e_{i+1}))$ *[or* $(e_i = s(e_{i+1})]$
- *If* $n > 1$, $\forall i \in \{1,...,n-1\}$, $e_i \in K \implies t(e_i) = e_{i+1}$ *[or* $(e_{i+1} \in A$ *and* $s(e_i) = e_{i+1})$ *or* $(e_{i+1} \in K$ *and* $e_i = t(e_{i+1}))]$

A non-directed RAF-path is a non-directed RAF-walk in which all the elements are distinct.

From these notions we can define the notions of "*connected*" and "*disconnected*" RAFs:

Definition 25 (Connected and disconnected RAF)**.** *Let* $\mathcal{RAF} = \langle A, K, s, t \rangle$ *be a RAF.* $\mathcal{RAF}$ *is a* connected *RAF if, for all distinct elements* $x_i \in A \cup K$ *and* $x_j \in A \cup K$*, there exists a non-directed RAF-path* p *in* $\mathcal{RAF}$ *s.t.* x_i *is the first element of* p *and* x_j *is the last element of* p*. Otherwise the RAF is* disconnected*.*

Now, let extend these notions for partial RAFs. A non-directed partial-RAF-walk is a sequence of arguments and interactions respecting some constraints:

Definition 26. (Non-directed Partial-RAF-walk and Partial-RAF-path) Let $\widetilde{\mathcal{RAF}} = \langle \tilde{A}, \tilde{K}, \tilde{s}, \tilde{t}, s, t \rangle$ be a partial RAF and $e_1, e_n \in (\tilde{A} \cup \tilde{K})$ be elements of $\widetilde{\mathcal{RAF}}$. A *non-directed partial-RAF-walk* is a sequence $(e_1, ..., e_n)$ with $n \in \mathbb{N}^*$ and $\forall i, e_i \in (A \cup K)$ s.t.:

- If $n > 1, \forall i \in \{1,...,n-1\}, e_i \in A \implies e_{i+1} \in K$ and $(\tilde{s}(e_{i+1}) = \texttt{true}$ and $e_i = s(e_{i+1})$ or $(\tilde{t}(e_{i+1}) = \texttt{true}$ and $e_i = t(e_{i+1})))$
- If $n > 1, \forall i \in \{1,...,n-1\}, e_i \in K \implies (e_{i+1} \in A$ and $((\tilde{s}(e_i) = \texttt{true}$ and $s(e_i) = e_{i+1})$ or $(\tilde{t}(e_i) = \texttt{true}$ and $t(e_i) = e_{i+1})))$ or $(e_{i+1} \in K$ and $((\tilde{t}(e_{i+1}) = \texttt{true}$ and $e_i = t(e_{i+1}))$ or $(\tilde{t}(e_i) = \texttt{true}$ and $t(e_i) = e_{i+1})))$

A *non-directed partial-RAF-path* is a non-directed partial-RAF-walk in which all the elements are distinct.

Definition 27. (Connected and disconnected partial RAF). Let $\widetilde{\mathcal{RAF}} = \langle \tilde{A}, \tilde{K}, \tilde{s}, \tilde{t}, s, t \rangle$ be a partial RAF. $\widetilde{\mathcal{RAF}}$ is a *connected* partial RAF if, for all distinct elements $x_i \in \tilde{A} \cup \tilde{K}$ and $x_j \in \tilde{A} \cup \tilde{K}$, there exists a non-directed partial-RAF-path p in $\widetilde{\mathcal{RAF}}$ s.t. x_i is the first element of p and x_j is the last element of p. Otherwise the partial RAF is *disconnected*.

As for AF, let define the operator $\downarrow$ for partial RAFs:

Definition 28 (Partial RAF restriction $\downarrow$)**.** *Let* $\mathcal{RAF} = \langle A, K, s, t \rangle$ *be a RAF,* $\widetilde{\mathcal{RAF}} = \langle \tilde{A}_1, \tilde{K}_1, \tilde{s}_1, \tilde{t}_1, s, t \rangle$ *be a partial RAF of* $\mathcal{RAF}$ *and* $S \subseteq \tilde{A}_1 \cup \tilde{K}_1$ *be a set of elements. The* restriction *of* $\widetilde{\mathcal{RAF}}$ *to S, denoted* $\widetilde{\mathcal{RAF}} \downarrow_S$*, is the partial RAF of* $\mathcal{RAF}$ *defined as follows:*

$$\widetilde{\mathcal{RAF}} \downarrow_S = \langle \tilde{A}_2, \tilde{K}_2, \tilde{s}_2, \tilde{t}_2, s, t \rangle$$

where:

- $\tilde{A}_2 = \tilde{A}_1 \cap S$ *and* $\tilde{K}_2 = \tilde{K}_1 \cap S$
- $\tilde{s}_2 : \tilde{K}_2 \to \{\texttt{true}, \texttt{false}\}$ *s.t.* $\forall \alpha \in \tilde{K}_2, \tilde{s}_2(\alpha) = \texttt{true}$ *if* $s(\alpha) \in \tilde{A}_2$ *otherwise* $\texttt{false}$
- $\tilde{t}_2 : \tilde{K}_2 \to \{\texttt{true}, \texttt{false}\}$ *s.t.* $\forall \alpha \in \tilde{K}_2, \tilde{t}_2(\alpha) = \texttt{true}$ *if* $t(\alpha) \in \tilde{A}_2 \cup \tilde{K}_2$ *otherwise* $\texttt{false}$

Finally, with the help of $\downarrow$ operator, we can define the notion of "Partial RAF Connected Component":

Definition 29. (Partial RAF Connected Component). Let $\widetilde{\mathcal{RAF}} = \langle \tilde{A}, \tilde{K}, \tilde{s}, \tilde{t}, s, t \rangle$ be a partial RAF. Let $S \subseteq \tilde{A} \cup \tilde{K}$ be a set of elements. The partial RAF $\widetilde{\mathcal{RAF}} \downarrow_S$ is a *connected component* of $\widetilde{\mathcal{RAF}}$ iff:

- $\widetilde{\mathcal{RAF}} \downarrow_S$ is connected.
- There exists no set $S' \subseteq \tilde{A} \cup \tilde{K}$ s.t. $S \subset S'$ and $\widetilde{\mathcal{RAF}} \downarrow_{S'}$ is connected.

Note: By sake of brevity, we say "connected component" instead of "partial RAF Connected Component" when there is no ambiguity.

Figure 9 shows an example of several (here 4) partial RAF connected components issued from the partial RAF described in Figure 7.

4 A Generic Algorithm for RAF: Presentation by Example

The *RAFDivider* algorithm we propose in the following is an adaptation of the *AFDivider* algorithm proposed in [16]. As for *AFDivider* , it addresses the enumeration problem for the *complete*, *stable* and *preferred* semantics . It follows the same four major steps (see Figure 4):

1. A pretreatment on $\mathcal{RAF}$ removes "trivial" parts of it.
2. Clusters in $\mathcal{RAF}$ are identified.
3. The labellings under semantics σ in each cluster are computed in parallel.
4. The results of each cluster are reunified to get the labellings of $\mathcal{RAF}$.

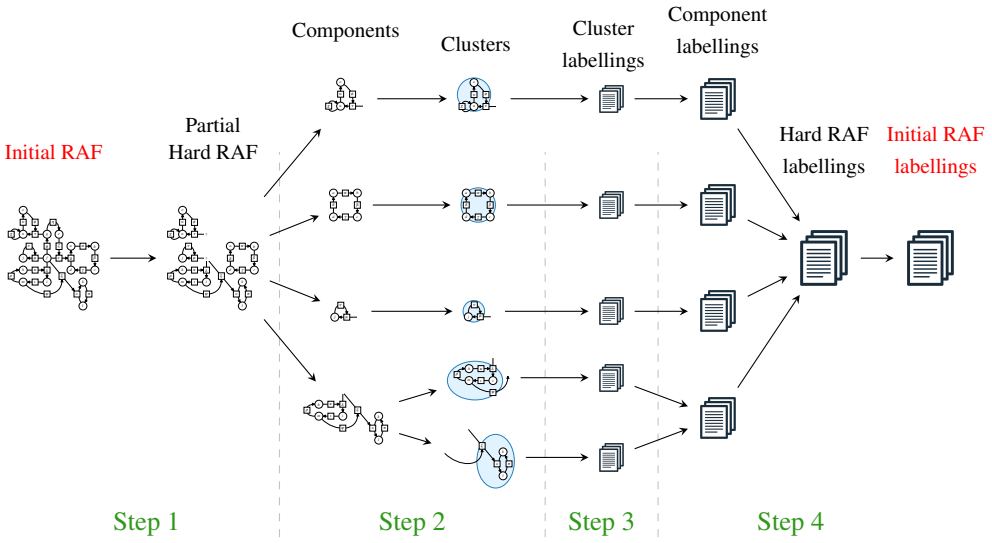


Figure 4: *RAFDivider* operating diagram

Before giving the formal definition of the algorithm, we describe its wanted behaviour on a running example.

4.1 Pretreatment: Removing the Trivial Part

The first step is to identify the *trivial* part to remove. As for AF, it is built using the in and out elements produced by the *grounded* semantics. In the case of an AF, these elements are all removed. However, for a RAF, this complete removal is not possible.

Example 2. *Consider $\mathcal{RAF} = \langle A, K, s, t \rangle$, the RAF illustrated in Figure 5. Its* grounded *structure labelling is the following one:*

$$\mathcal{L}_{gr} = \left\{ \begin{array}{c} (a, und), (b, und), (c, und), (d, und), (e, in), (f, in), (g, out), \\ (h, und), (i, und), (j, und), (k, und), (l, und), (m, und), (n, und), (o, und) \\ (\alpha, in), (\beta, in), (\gamma, in), (\delta, in), (\varepsilon, out), (\zeta, in), (\eta, in), (\theta, in), (\iota, in), \\ (\kappa, in), (\lambda, und), (\mu, und), (\nu, in), (\xi, und), (\mathrm{o}, und), (\pi, in), (\rho, in), (\tau, und), \\ (\upsilon, in), (\phi, in), (\chi, in), (\psi, in) \end{array} \right\}$$

Figure 6 illustrates the labelled RAF corresponding to the running example.

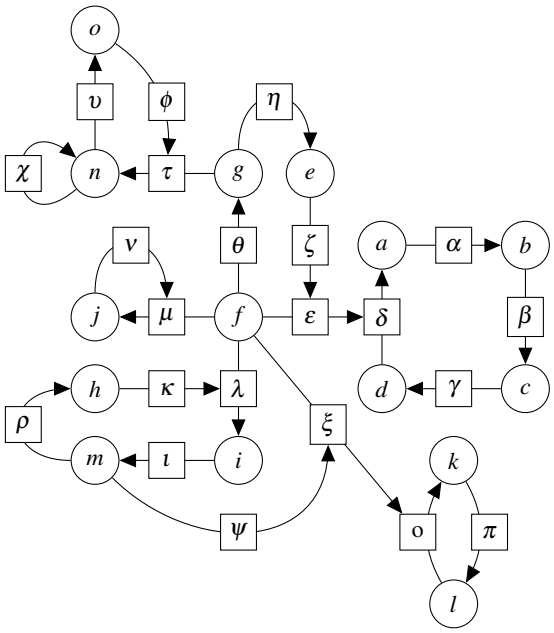


Figure 5: Running example

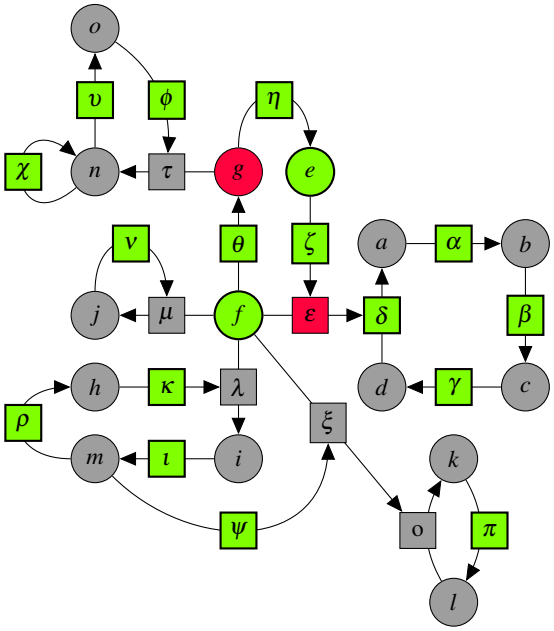


Figure 6: Running example with its grounded labelling (in green the *in* elements, in red the *out* elements and in grey the *und* elements)

We cannot remove all elements that are not labelled `und` *as it was done in* AFDivider *algorithm. Otherwise, the resultant partial RAF would not capture the initial relations between the elements (for instance, the relation between the arguments a and b is expressed by the attack* α *labelled* `in`*, so the removal of* α *is not possible). We would like all elements labelled* `out`*, all arguments labelled* `in` *and all attacks labelled* `in` *whose source is labelled* `out` *or* `in` *to be removed.*

This leads to the following definition of the *RAF trivial part*:

Definition 30. (RAF trivial part). Let $\mathcal{RAF} = \langle A, K, s, t\rangle$ be a RAF and let $\mathcal{L}_{gr}$ be the *grounded* structure labelling of $\mathcal{RAF}$. The "*trivial part*" of $\mathcal{RAF}$ is the structure of $\mathcal{RAF}$, denoted by $\mathcal{U}_{triv} = \langle S_{triv}, Q_{triv}\rangle$, with

$$\mathcal{U}_{triv} = \left\langle \left\{a \in A | a \notin \texttt{und}(\mathcal{L}_{gr})\right\}, \left\{\alpha \in K | \alpha \in \texttt{out}(\mathcal{L}_{gr}) \text{ or } (\alpha \in \texttt{in}(\mathcal{L}_{gr}) \text{ and } s(\alpha) \notin \texttt{und}(\mathcal{L}_{gr}))\right\}\right\rangle$$

Example 3. *Following Example 2, the trivial part of* $\mathcal{RAF}$ *is:*

$$\mathcal{U}_{triv} = \langle \{e, f, g\}, \{\varepsilon, \zeta, \eta, \theta\}\rangle$$

Then the *partial hard RAF* and *partial hard RAF with input* are defined as follows:

Definition 31. (Partial hard RAF).
Let $\mathcal{RAF} = \langle A, K, s, t\rangle$ be a RAF and let $\mathcal{U}_{triv} = \langle S_{triv}, Q_{triv}\rangle$ be the RAF trivial part of $\mathcal{RAF}$. The *partial hard RAF* of $\mathcal{RAF}$, denoted by $\widetilde{\mathcal{RAF}}_{hard}$, is defined as

$$\widetilde{\mathcal{RAF}}_{hard} = \left\langle \tilde{A}, \tilde{K}, \tilde{s}, \tilde{t}, s_h, t_h\right\rangle \text{ with } \tilde{A} = A \setminus S_{triv},\ \tilde{K} = K \setminus Q_{triv} \text{ and}$$

$$s_h : \tilde{K} \to A \text{ and } \forall \alpha \in \tilde{K}, s_h(\alpha) = s(\alpha),\ t_h : \tilde{K} \to (A \cup K) \text{ and } \forall \alpha \in \tilde{K}, t_h(\alpha) = t(\alpha)$$
$$\tilde{s} : \tilde{K} \to \{\texttt{true}, \texttt{false}\} \text{ s.t. } \forall \alpha \in \tilde{K}, \tilde{s}(\alpha) = \texttt{true} \text{ if } s_h(\alpha) \in \tilde{A} \text{ otherwise } \texttt{false}$$
$$\tilde{t} : \tilde{K} \to \{\texttt{true}, \texttt{false}\} \text{ s.t. } \forall \alpha \in \tilde{K}, \tilde{t}(\alpha) = \texttt{true} \text{ if } t_h(\alpha) \in \tilde{A} \cup \tilde{K} \text{ otherwise } \texttt{false}$$

Although a partial hard RAF is a partial RAF, we want to consider only the input labelling that coincides with the *grounded* labelling of the initial RAF. This leads to the following definition of "*partial hard RAF with input*":

Definition 32 (Partial hard RAF with input)**.** *Let* $\mathcal{RAF} = \langle A, K, s, t\rangle$ *be a RAF,* $\mathcal{L}_{gr}$ *be the* grounded *structure labelling of* $\mathcal{RAF}$*,* $\mathcal{U}_{triv}$ *be the trivial part of* $\mathcal{RAF}$*. Let* $\widetilde{\mathcal{RAF}}_{hard} = \left\langle \tilde{A}, \tilde{K}, \tilde{s}, \tilde{t}, s_h, t_h\right\rangle$ *be the partial hard RAF of* $\mathcal{RAF}$ *taking into account* $\mathcal{U}_{triv}$*. The "*partial hard RAF with input*" corresponding to* $\mathcal{RAF}$ *is the partial RAF with input* $\left\langle \widetilde{\mathcal{RAF}}_{hard}, \mathcal{I} = \langle S^{inp}, Q^{inp}\rangle, \mathcal{L}^{inp}\right\rangle$*, where*

- $S^{inp} = \{s(\alpha) \in (A \setminus \tilde{A}) | \alpha \in \tilde{K} \text{ s.t. } \tilde{s}(\alpha) = \texttt{false}\}$ *and* $Q^{inp} = \varnothing$[13]
- $\mathcal{L}^{inp}$ *is astructure labelling of the elements in* S^{inp} *and* Q^{inp} *st:*

$$\forall x \in S^{inp} \cup Q^{inp},\ \mathcal{L}^{inp}(x) = \mathcal{L}_{gr}(x)$$

The partial hard RAF with input is unique since the labelling of its inputs is unique.

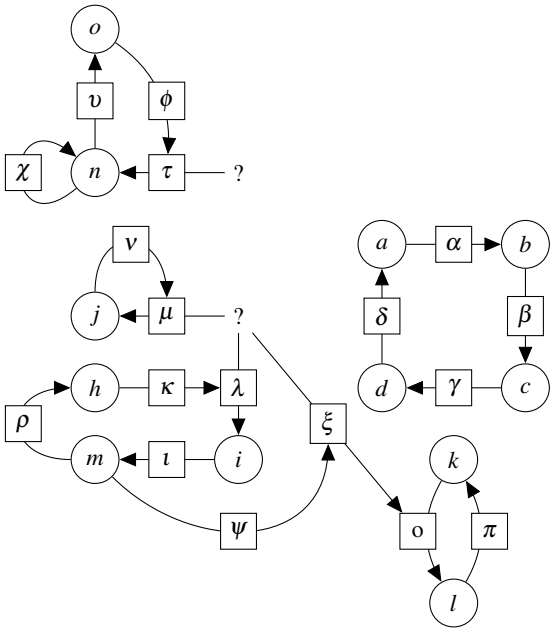


Figure 7: Partial Hard RAF (the attacks τ, μ, λ and ξ have no source)

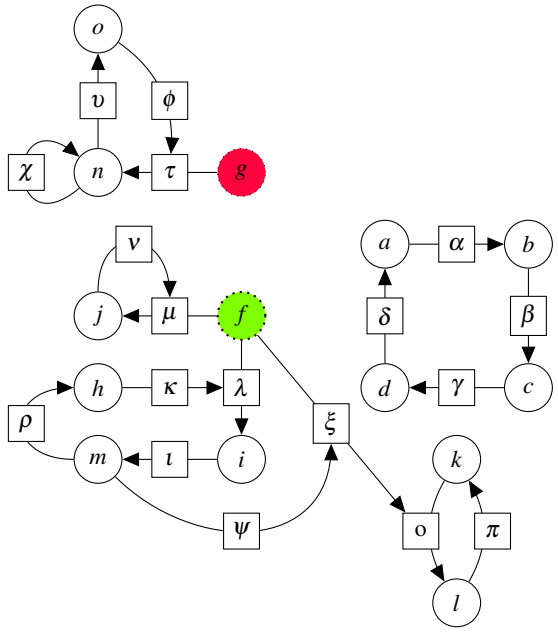
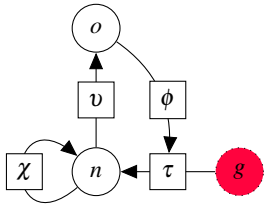


Figure 8: Partial Hard RAF with input (the sources of the attacks τ, μ, λ and ξ are the inputs, given with their labelling issued to $\mathcal{L}_{gr}$)

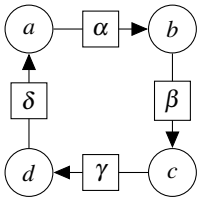
Example 4. *Figure 7 illustrates $\widetilde{\mathcal{RAF}}_{hard}$, the partial hard RAF corresponding to the RAF shown in Figure 5. In this partial hard RAF, 4 attacks have no source (τ, μ, λ and ξ). Figure 8 illustrates the partial hard RAF with input corresponding to $\mathcal{RAF}$: $\langle \widetilde{\mathcal{RAF}}_{hard}, \mathcal{I} = \{f,g\}, \mathcal{L}^{inp} = \{(f,\texttt{in}),(g,\texttt{out})\}\rangle$. Notice that the arguments f and g are the input arguments of $\widetilde{\mathcal{RAF}}_{hard}$; they do not belong to $\widetilde{\mathcal{RAF}}_{hard}$. Nevertheless, considering these inputs, any attack in the partial hard RAF with input has now a source.*

Note that taking into account the inputs is not the guarantee to transform the partial hard RAF with input into a RAF. Indeed, let consider the RAF defined by $A = \{a,b,c\}$, $K = \{\alpha,\beta,\gamma\}$, $s = \{(\alpha,b),(\beta,c),(\gamma,c)\}$, $t = \{(\alpha,a),(\beta,a),(\gamma,c)\}$; in this case, the grounded labelling is: $\{(a,\texttt{out}),(b,\texttt{in}),(c,\texttt{und}),(\alpha,\texttt{in}),(\beta,\texttt{in}),(\gamma,\texttt{in})\}$; and so the trivial part of the RAF is $\mathcal{U}_{triv} = \langle \{a,b\},\{\alpha\}\rangle$ and the partial hard RAF only contains c, γ and β (but not the target of β).

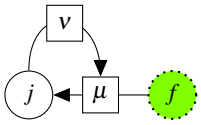
For each input element, several cases have to be considered and this can be very time consuming. In order to avoid this cost for elements that are in the "trivial part", we simply cut that part from the RAF and, only after that, look for clusters. So, given a RAF $\mathcal{RAF} = \langle A,K,s,t\rangle$, the *RAFDivider* algorithm starts by computing $\mathcal{L}_{gr}$, the *grounded* labelling of $\mathcal{RAF}$ and $\mathcal{U}_{triv}$ the trivial part corresponding to it. Once the trivial part has been computed, the algorithm removes it from $\mathcal{RAF}$ to produce $\widetilde{\mathcal{RAF}}_{hard}$ the partial hard RAF of $\mathcal{RAF}$, as well as $\widetilde{\mathcal{RAF}}_{hard}$ input elements $\mathcal{I}$ with their labelling issued from $\mathcal{L}_{gr}$. Then, if possible, $\widetilde{\mathcal{RAF}}_{hard}$ is split into several connected components (see Definition 27), producing the set *CCSet*. If there is only one connected component we have so: $CCSet = \left\{\widetilde{raf}_4\right\}$ with $\widetilde{raf}_4 = \widetilde{\mathcal{RAF}}_{hard}$. *CCSet* is a set of partial hard RAFs (with input).



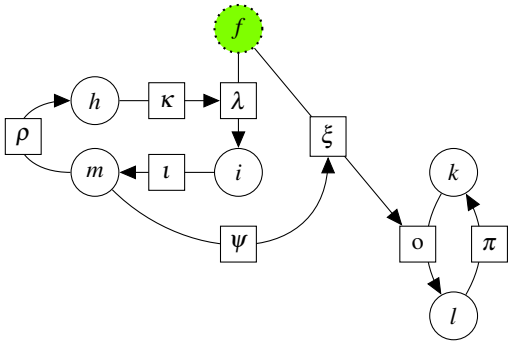
(a) Component 4: $\widetilde{raf}_4$



(b) Component 1: $\widetilde{raf}_1$



(c) Component 3: $\widetilde{raf}_3$



(d) Component 2: $\widetilde{raf}_2$

Figure 9: The connected components of $\widetilde{\mathcal{RAF}}_{hard}$ with their inputs ($\widetilde{raf}_1$ has no input; $\widetilde{raf}_3$ and $\widetilde{raf}_2$ have the same input f)

Example 5. *Figure 9 illustrates the four connected components of $\widetilde{\mathcal{RAF}}_{hard}$, the partial hard RAF illustrated in Figure 7. The CCSet will then contain these four partial RAF.*

[13] Indeed, let consider α being an attack in Q_{triv} s.t. its target is in $\widetilde{\mathcal{RAF}}_{hard}$; either α is labelled with $\texttt{out}$ or α is labelled with $\texttt{in}$ and its source is either labelled with $\texttt{in}$ or $\texttt{out}$: in any case, the fact that its target is in $\widetilde{\mathcal{RAF}}_{hard}$ shows that α is useless for determining the label of its target.

4.2 Identifying Clusters

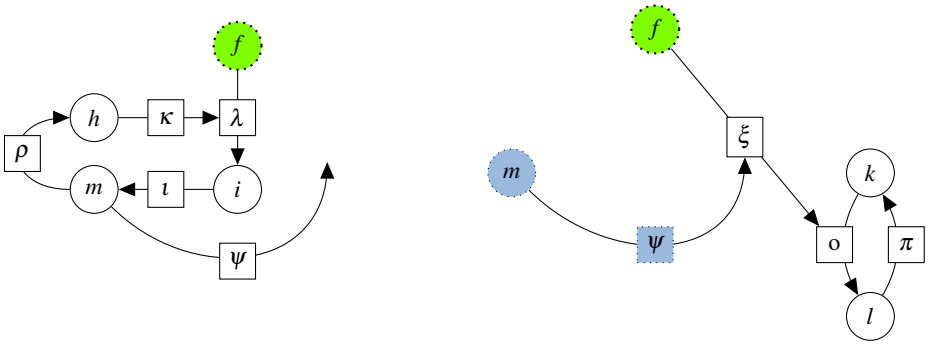
For each connected component, a clustering can be made using any clustering method partitioning the partial RAF (even a random partition method). See in Section 6 such a method that returns the set of clusters identified, that is a set of partial RAFs.

Example 6. *Following Example 5, let consider that the chosen clustering method produces only one cluster for components 1, 3 and 4. Let* $\widetilde{raf}_2 = \langle \tilde{A}_2, \tilde{K}_2, \tilde{s}_2, \tilde{t}_2, s_{h2}, t_{h2} \rangle$ *be the fourth component and let the following partition be the one produced by the chosen clustering method:*

$$\Omega = \{\omega_1 = \{h, i, m, \iota, \kappa, \lambda, \rho, \psi\}, \omega_2 = \{k, l, \xi, \mathrm{o}, \pi\}\}$$

Figure 10 illustrates the partial RAFs corresponding to the partitioning of $\widetilde{raf}_2$*, that is* $\widetilde{raf}_2 \downarrow_{\omega_1}$[14] *(also denoted by* $\widetilde{raf}_{2.1}$*) and* $\widetilde{raf}_2 \downarrow_{\omega_2}$ *(also denoted by* $\widetilde{raf}_{2.2}$*).*

Note that, following this clustering, m *and* ψ *become inputs for* $\widetilde{raf}_{2.2}$*. Note also that this clustering must also take into account attacks and not only arguments (as it is the case for the AF divider). See in Section 6 an example of such a clustering.*



(a) Cluster 1: $\widetilde{raf}_{2.1} = \widetilde{raf}_2 \downarrow_{\omega_1}$ (b) Cluster 2: $\widetilde{raf}_{2.2} = \widetilde{raf}_2 \downarrow_{\omega_2}$

Figure 10: Clusters of $\widetilde{raf}_2$ (the inputs m and ψ for $\widetilde{raf}_{2.2}$ are given in blue since their labellings are unknown at this time; whereas f is in green since its labelling is known: following $\mathcal{L}_{gr}$, f must be in)

4.3 Computing the Labellings

The next step is the computation of the component labellings in a distributed way relying on the clustering made. The σ-labellings of each cluster are computed simultaneously. Unlike the case of connected components, the partial RAF corresponding to the computed clusters may admit several input labellings. In order to compute all the possible σ-labellings of a given cluster, every possible case concerning its input elements has to be considered.

Note: In the worst case the number of input labellings to consider for an partial RAF is 3^n*, with n being the number of its input element. When choosing a clustering, there is thus a threshold between the size of the clusters and the number of attack cuts to consider as it effects the overall solving time.* We call "context" a particular input labelling of a partial RAF:[15]

Definition 33. (Context).

Let $\widetilde{\mathcal{RAF}} = \langle \tilde{A}, \tilde{K}, \tilde{s}, \tilde{t}, s, t \rangle$ be a partial RAF and $\mathcal{I} = \langle S^{inp}, Q^{inp} \rangle$ be the input of $\widetilde{\mathcal{RAF}}$. A context μ of a partial RAF is a labelling of $\mathcal{I}$.

[14] $\widetilde{raf}_2 \downarrow_{\omega_1}$ produces a partial RAF built from $\widetilde{raf}_2$ keeping only the elements of ω_1.

[15] Obviously, each context of a partial RAF will induce a specific set of labellings of this partial RAF.

Example 7. *Following Figure 10 and in the worst case, the contexts of* $\widetilde{raf}_{2.2}$ *would be the following ones (3 inputs with 3 possible values, so* $3^3 = 27$ *contexts):*

- $\mu_{2.2.1} = \{(m, in), (f, in), (\psi, in)\}$
- $\mu_{2.2.2} = \{(m, out), (f, in), (\psi, in)\}$
- $\mu_{2.2.3} = \{(m, und), (f, in), (\psi, in)\}$
- $\mu_{2.2.4} = \{(m, in), (f, out), (\psi, in)\}$
- $\mu_{2.2.5} = \{(m, out), (f, out), (\psi, in)\}$
- $\mu_{2.2.6} = \{(m, und), (f, out), (\psi, in)\}$
- $\mu_{2.2.7} = \{(m, in), (f, und), (\psi, in)\}$
- $\mu_{2.2.8} = \{(m, out), (f, und), (\psi, in)\}$
- $\mu_{2.2.9} = \{(m, und), (f, und), (\psi, in)\}$
- $\mu_{2.2.10} = \{(m, in), (f, in), (\psi, out)\}$
- $\mu_{2.2.11} = \{(m, out), (f, in), (\psi, out)\}$
- $\mu_{2.2.12} = \{(m, und), (f, in), (\psi, out)\}$
- $\mu_{2.2.13} = \{(m, in), (f, out), (\psi, out)\}$
- $\mu_{2.2.14} = \{(m, out), (f, out), (\psi, out)\}$
- $\mu_{2.2.15} = \{(m, und), (f, out), (\psi, out)\}$
- $\mu_{2.2.16} = \{(m, in), (f, und), (\psi, out)\}$
- $\mu_{2.2.17} = \{(m, out), (f, und), (\psi, out)\}$
- $\mu_{2.2.18} = \{(m, und), (f, und), (\psi, out)\}$
- $\mu_{2.2.19} = \{(m, in), (f, in), (\psi, und)\}$
- $\mu_{2.2.20} = \{(m, out), (f, in), (\psi, und)\}$
- $\mu_{2.2.21} = \{(m, und), (f, in), (\psi, und)\}$
- $\mu_{2.2.22} = \{(m, in), (f, out), (\psi, und)\}$
- $\mu_{2.2.23} = \{(m, out), (f, out), (\psi, und)\}$
- $\mu_{2.2.24} = \{(m, und), (f, out), (\psi, und)\}$
- $\mu_{2.2.25} = \{(m, in), (f, und), (\psi, und)\}$
- $\mu_{2.2.26} = \{(m, out), (f, und), (\psi, und)\}$
- $\mu_{2.2.27} = \{(m, und), (f, und), (\psi, und)\}$

Nevertheless, some of these 27 contexts are not compatible with the labelling $\mathcal{L}_{gr}$ *(*f *must be labelled* in*). So only 9 contexts are compatible:* $\mu_{2.2.1}$ *to* $\mu_{2.2.3}$*,* $\mu_{2.2.10}$ *to* $\mu_{2.2.12}$*,* $\mu_{2.2.19}$ *to* $\mu_{2.2.21}$*.*

The labellings "induced" by a certain context of a partial RAF are defined as follows:

Definition 34 (Induced labellings)**.** *Let* σ *be a semantics,* $\widetilde{\mathcal{RAF}} = \langle \tilde{A}, \tilde{K}, \tilde{s}, \tilde{t}, s, t \rangle$ *be a partial RAF,* $\mathcal{I}$ *be the input of* $\widetilde{\mathcal{RAF}}$ *and* μ *be a context of* $\widetilde{\mathcal{RAF}}$*. Let* $\widetilde{\mathcal{RAF}}_s$ *be the standard RAF corresponding to the partial RAF with input* $\left\langle \widetilde{\mathcal{RAF}}, \mathcal{I}, \mu \right\rangle$*. The set of "induced labellings"* $\mathscr{L}_\sigma^{\mu(\widetilde{\mathcal{RAF}})}$ *of* $\widetilde{\mathcal{RAF}}$ *under the context* μ *is defined as follows:*

$$\mathscr{L}_\sigma^{\mu(\widetilde{\mathcal{RAF}})} = \{\mathcal{L} \downarrow_{\tilde{A} \cup \tilde{K}} | \mathcal{L} \in \mathscr{L}_\sigma(\widetilde{\mathcal{RAF}}_s)\}$$

$\mathscr{L}_\sigma(\widetilde{\mathcal{RAF}})$ *denotes the set of all induced labelling of* $\widetilde{\mathcal{RAF}}$ *under all possible contexts.*

As one can notice from Example 7, it may be useless to consider some cluster contexts. So three optimizations can enhance the computation time:

- **First optimization**: Given a cluster, if one of its input elements is also an input element of the partial hard RAF then this element should only be labelled as in the *grounded* labelling $\mathcal{L}_{gr}$ (see Example 7).

  At this step we can illustrate how the *RAFDivider* algorithm will compute the labellings for the components 4, 1 and 3 as well as for the first cluster of component 2:

  Example 8. *Following Example 5, components 4, 1 and 3 only have one cluster. Furthermore, labellings of each of them have to be computed for only one context. Indeed, whether it is empty (the case of component 1) or it has no input element that do not belong to the trivial part. Figure 11 to 13 illustrate the unique standard RAF to compute for each of them.*

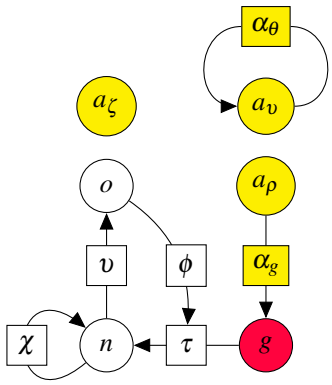


Figure 11: Standard RAF corresponding to $\widetilde{raf}_4$ and $\mathcal{L}^{inp}(g) = \texttt{out}$

Notice that the labellings of $\widetilde{raf}_4$ have to be computed for only one context (that is $\mathcal{L}^{inp}(g) = \texttt{out}$) as this partial RAF has no input element that does not belong to the trivial part. In yellow, the additional elements for building the standard RAF.

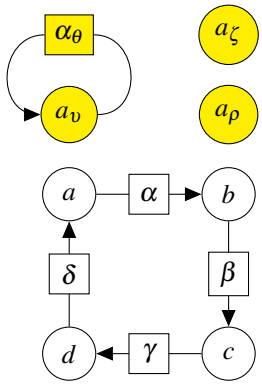


Figure 12: Standard RAF corresponding to $\widetilde{raf}_1$

Notice that the labellings of $\widetilde{raf}_1$ have to be computed for only one context (the empty one) as this partial RAF has no input elements. In yellow, the additional elements for building the standard RAF.

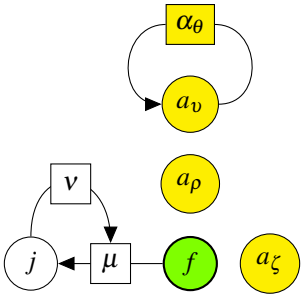


Figure 13: Standard RAF corresponding to $\widetilde{raf}_3$

Notice that the labellings of $\widetilde{raf}_3$ have to be computed for only one context (that is $\mathcal{L}^{inp}(f) = \texttt{in}$) as this partial RAF has no input element that does not belong to the trivial part. In yellow, the additional elements for building the standard RAF.

Example 9. *Let consider the* complete *semantics. Following Example 8, the* complete *labellings of the components 4, 1 and 3 are as follows. Notice that the elements added while constructing their corresponding standard RAFs have been removed:*[16]

$$\mathscr{L}_{co}(\widetilde{raf}_4) = \left\{ \mathcal{L}_{4.1} = \left\{ (n, und), (o, und), (\tau, und), (\upsilon, in), (\phi, in), (\chi, in) \right\} \right\}$$

$$\mathscr{L}_{co}(\widetilde{raf}_1) = \left\{ \begin{array}{l} \mathcal{L}_{1.1} = \left\{ \begin{array}{c} (a, in), (b, out), (c, in), (d, out), \\ (\alpha, in), (\beta, in), (\gamma, in), (\delta, in) \end{array} \right\}, \\ \mathcal{L}_{1.2} = \left\{ \begin{array}{c} (a, out), (b, in), (c, out), (d, in), \\ (\alpha, in), (\beta, in), (\gamma, in), (\delta, in) \end{array} \right\}, \\ \mathcal{L}_{1.3} = \left\{ \begin{array}{c} (a, und), (b, und), (c, und), (d, und), \\ (\alpha, in), (\beta, in), (\gamma, in), (\delta, in) \end{array} \right\} \end{array} \right\}$$

$$\mathscr{L}_{co}(\widetilde{raf}_3) = \left\{ \begin{array}{l} \mathcal{L}_{3.1} = \left\{ (j, in), (\mu, out), (\nu, in) \right\}, \\ \mathcal{L}_{3.2} = \left\{ (j, out), (\mu, in), (\nu, in) \right\}, \\ \mathcal{L}_{3.3} = \left\{ (j, und), (\mu, und), (\nu, in) \right\} \end{array} \right\}$$

Example 10. *Following Example 6, the component 2 has two clusters in it. Figure 14 illustrates the only standard RAF to compute for* $\widetilde{raf}_{2.1}$.

Example 11. *Let consider the* complete *semantics. Following Example 10, the* complete *labellings of the first cluster of component 2 are as follows. Notice that the elements added while constructing their corresponding standard RAFs have been removed:*

$$\mathscr{L}_{co}(\widetilde{raf}_{2.1}) = \left\{ \begin{array}{l} \mathcal{L}_{2.1.1} = \left\{ \begin{array}{c} (i, out), (h, out), (m, in), \\ (\iota, in), (\kappa, in), (\lambda, in), (\rho, in), (\psi, in) \end{array} \right\}, \\ \mathcal{L}_{2.1.2} = \left\{ \begin{array}{c} (i, in), (h, in), (m, out), \\ (\iota, in), (\kappa, in), (\lambda, out), (\rho, in), (\psi, in) \end{array} \right\}, \\ \mathcal{L}_{2.1.3} = \left\{ \begin{array}{c} (i, und), (h, und), (m, und), \\ (\iota, in), (\kappa, in), (\lambda, und), (\rho, in), (\psi, in) \end{array} \right\} \end{array} \right\}$$

- **Second optimization**: If an input attack is unattacked (a so-called valid attack) in the initial RAF, then this attack will always be labelled in following all semantics we are interested in.

Example 12. *The attack* Ψ*, being an input of* $\widetilde{raf}_{2.2}$ *and being a valid one in* $\mathcal{RAF}$ *or in* $\widetilde{\mathcal{RAF}}_{hard}$*, it is useless to consider contexts where* Ψ *is not labelled* in.

[16] As an example the set of labellings of the unique standard RAF of component $\widetilde{raf}_4$ is the following:

$$\left\{ \left\{ \begin{array}{c} (g, out), (n, und), (o, und), (a_\zeta, in), (a_\rho, in), (a_\upsilon, und), \\ (\tau, und), (\upsilon, in), (\phi, in), (\chi, in), (\alpha_g, in), (\alpha_\theta, in) \end{array} \right\} \right\}$$

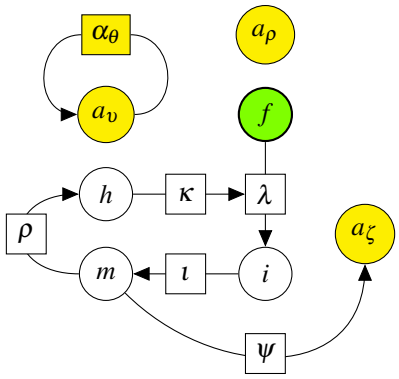


Figure 14: Standard RAF corresponding to $\widetilde{raf}_{2.1}$

Notice that the labellings of $\widetilde{raf}_{2.1}$ have to be computed for only one context (that is $\mathcal{L}^{inp}(f) = in$) as this partial RAF has no input element that does not belong to the trivial part. In yellow, the additional elements for building the standard RAF.

Those restrictions over possible contexts may decrease a lot the number of cases to consider. As an illustration, consider the following example:

Example 13. *Example 7 gives the list of all possible contexts (even the useless ones) of* $\widetilde{raf}_{2.2}$*. There are 27 in total. Indeed, as there are three input elements we have* 3^3 *contexts. Now, with the two restrictions mentioned above, this number drops to three. Only the first three contexts have to be kept (those with* f *and* ψ *labelled with* in*):*

- $\mu_{2.2.1} = \{(m, in), (f, in), (\psi, in)\}$
- $\mu_{2.2.2} = \{(m, out), (f, in), (\psi, in)\}$
- $\mu_{2.2.3} = \{(m, und), (f, in), (\psi, in)\}$

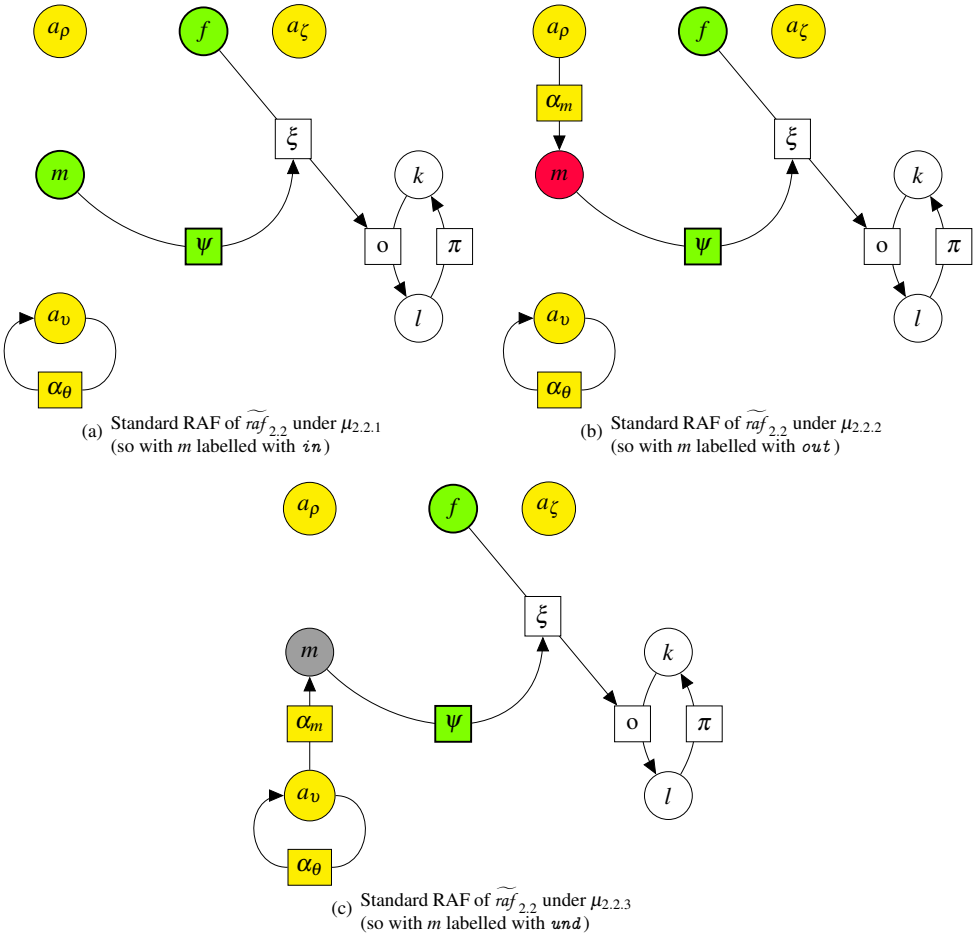
The standard RAFs corresponding to those three contexts are illustrated in Figure 15.

The labellings corresponding to $\widetilde{raf}_{2.2}$ *are thus as follows:*[17]

$$\mathscr{L}_{co}(\widetilde{raf}_{2.2}) = \left\{ \begin{array}{l} \mathcal{L}_{2.2.1} = \left\{ (k, out), (l, in), (\mathrm{o}, out), (\pi, in), (\xi, in) \right\}, \\ \mathcal{L}_{2.2.2} = \left\{ (k, in), (l, out), (\mathrm{o}, in), (\pi, in), (\xi, out) \right\}, \\ \mathcal{L}_{2.2.3} = \left\{ (k, out), (l, in), (\mathrm{o}, in), (\pi, in), (\xi, out) \right\}, \\ \mathcal{L}_{2.2.4} = \left\{ (k, und), (l, und), (\mathrm{o}, in), (\pi, in), (\xi, out) \right\}, \\ \mathcal{L}_{2.2.5} = \left\{ (k, out), (l, in), (\mathrm{o}, und), (\pi, in), (\xi, und) \right\}, \\ \mathcal{L}_{2.2.6} = \left\{ (k, und), (l, und), (\mathrm{o}, und), (\pi, in), (\xi, und) \right\} \end{array} \right\}$$

- **Third optimization**: As it cannot be illustrated with the running example, let consider that the attack Ψ is not always valid in $\widetilde{\mathcal{RAF}}_{hard}$. As a consequence the first two optimizations would not allow to reduce the number of contexts of $\widetilde{raf}_{2.2}$ *w.r.t.* m and Ψ to consider. So, $\widetilde{raf}_{2.2}$ would admit 9 contexts:

[17] $\mathcal{L}_{2.2.1}$ corresponds to $\mu_{2.2.2}$; $\mathcal{L}_{2.2.2}$ to $\mathcal{L}_{2.2.4}$ correspond to $\mu_{2.2.1}$; $\mathcal{L}_{2.2.5}$ to $\mathcal{L}_{2.2.6}$ correspond to $\mu_{2.2.3}$.



(a) Standard RAF of $\widetilde{raf}_{2.2}$ under $\mu_{2.2.1}$ (so with m labelled with $\mathtt{in}$)

(b) Standard RAF of $\widetilde{raf}_{2.2}$ under $\mu_{2.2.2}$ (so with m labelled with $\mathtt{out}$)

(c) Standard RAF of $\widetilde{raf}_{2.2}$ under $\mu_{2.2.3}$ (so with m labelled with $\mathtt{und}$)

Figure 15: Standard RAFs of $\widetilde{raf}_{2.2}$. In yellow, the additional elements for building these standard RAFs.

- $\mu_1 = \{(m, \mathit{in}), (\Psi, \mathit{in}), (f, \mathit{in})\}$
- $\mu_2 = \{(m, \mathit{in}), (\Psi, \mathit{out}), (f, \mathit{in})\}$
- $\mu_3 = \{(m, \mathit{in}), (\Psi, \mathit{und}), (f, \mathit{in})\}$
- $\mu_4 = \{(m, \mathit{out}), (\Psi, \mathit{in}), (f, \mathit{in})\}$
- $\mu_5 = \{(m, \mathit{out}), (\Psi, \mathit{out}), (f, \mathit{in})\}$
- $\mu_6 = \{(m, \mathit{out}), (\Psi, \mathit{und}), (f, \mathit{in})\}$
- $\mu_7 = \{(m, \mathit{und}), (\Psi, \mathit{in}), (f, \mathit{in})\}$
- $\mu_8 = \{(m, \mathit{und}), (\Psi, \mathit{out}), (f, \mathit{in})\}$
- $\mu_9 = \{(m, \mathit{und}), (\Psi, \mathit{und}), (f, \mathit{in})\}$

The 9 standard RAF induced by these contexts are illustrated in Figure 16, 17 and 18.

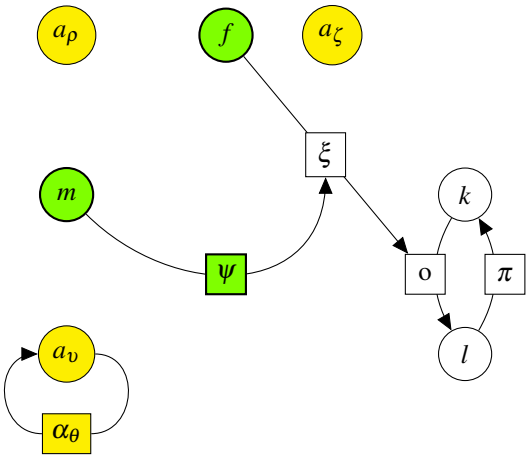


Figure 16: $\mathcal{L}^{inp}(\psi) = \mathit{in}$ and $\mathcal{L}^{inp}(m) = \mathit{in}$

However, we can observe that the labelling computation can be reduced to three cases. Indeed there are:

- One context where the attack relation is effective, corresponding to μ_1, that is the target of Ψ must be labelled out.
- Five contexts where the attack relation has no effect, corresponding to μ_2, μ_4, μ_5, μ_6 and μ_8 (Ψ or m is labelled out).
- Three contexts where the attack relation prevents the target of Ψ from being labelled in, corresponding to μ_3, μ_7, μ_9.

For each context of same type the partial RAF labellings generated will be thus the same. As a consequence, the labellings of the partial RAF can be computed considering only one context of each type.

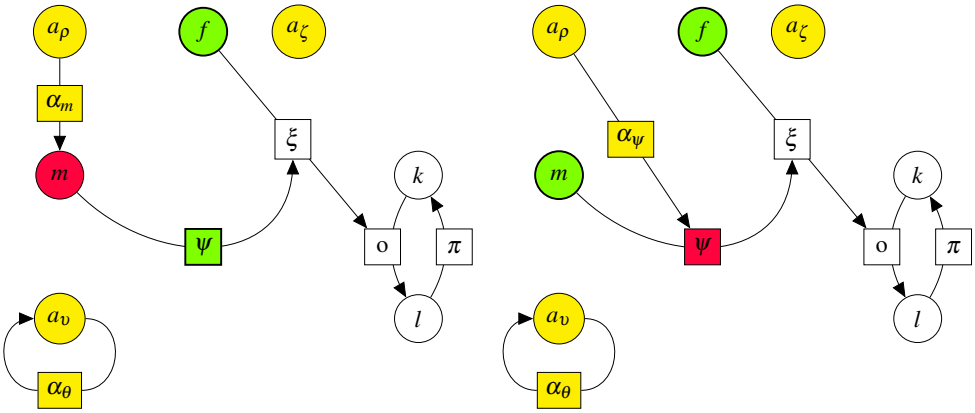
Example 14. *As an illustration:*

- *For* $\mu \in \{\mu_2, \mu_4, \mu_5, \mu_6, \mu_8\}$*, we have:*

$$\mathscr{L}_{co}^{\mu(\widetilde{raf}_{2.2})} = \left\{ \left\{ (k, \mathit{out}), (l, \mathit{in}), (\mathrm{o}, \mathit{out}), (\pi, \mathit{in}), (\xi, \mathit{in}) \right\} \right\}$$

- *For* $\mu \in \{\mu_3, \mu_7, \mu_9\}$*, we have:*

$$\mathscr{L}_{co}^{\mu(\widetilde{raf}_{2.2})} = \left\{ \begin{array}{l} \left\{ (k, \mathit{out}), (l, \mathit{in}), (\mathrm{o}, \mathit{und}), (\pi, \mathit{in}), (\xi, \mathit{und}) \right\}, \\ \left\{ (k, \mathit{und}), (l, \mathit{und}), (\mathrm{o}, \mathit{und}), (\pi, \mathit{in}), (\xi, \mathit{und}) \right\} \end{array} \right\}$$



(a) $\mathcal{L}^{inp}(\psi) = \mathtt{in}$ and $\mathcal{L}^{inp}(m) = \mathtt{out}$

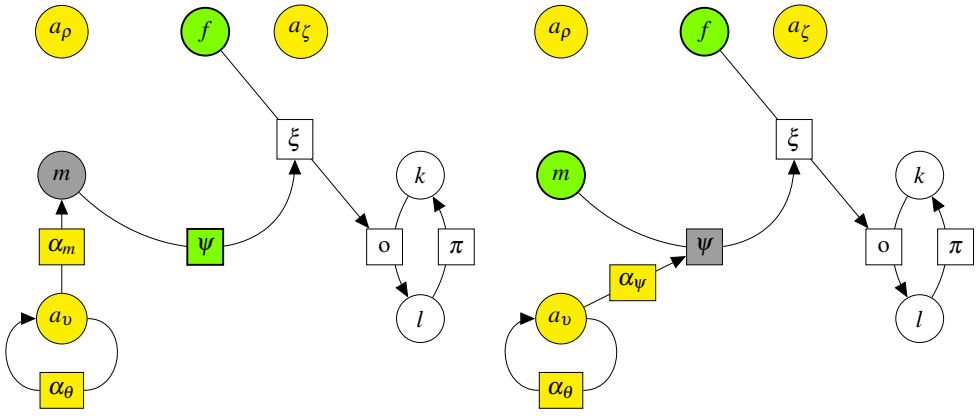
(b) $\mathcal{L}^{inp}(\psi) = \mathtt{out}$ and $\mathcal{L}^{inp}(m) = \mathtt{in}$

(c) $\mathcal{L}^{inp}(\psi) = \mathtt{out}$ and $\mathcal{L}^{inp}(m) = \mathtt{out}$

(d) $\mathcal{L}^{inp}(\psi) = \mathtt{out}$ and $\mathcal{L}^{inp}(m) = \mathtt{und}$

(e) $\mathcal{L}^{inp}(\psi) = \mathtt{und}$ and $\mathcal{L}^{inp}(m) = \mathtt{out}$

Figure 17: Standard RAFs of $\widetilde{raf}_{2.2}$ when m or ψ is rejected
Notice that in any context $\mathcal{L}^{inp}(f) = \mathtt{in}$ as f belongs to the trivial part. In yellow the additional elements for building the standard RAF.



(a) $\mathcal{L}^{inp}(\psi) = \texttt{in}$ and $\mathcal{L}^{inp}(m) = \texttt{und}$

(b) $\mathcal{L}^{inp}(\psi) = \texttt{und}$ and $\mathcal{L}^{inp}(m) = \texttt{in}$

(c) $\mathcal{L}^{inp}(\psi) = \texttt{und}$ and $\mathcal{L}^{inp}(m) = \texttt{und}$

Figure 18: Standard RAFs of $\widetilde{raf}_{2.2}$ when neither m nor ψ is $\texttt{out}$ and one of them is $\texttt{und}$ Notice that in any context $\mathcal{L}^{inp}(f) = \texttt{in}$ as f belongs to the trivial part. In yellow the additional elements for building the standard RAF.

– *For μ_1, we have:*

$$\mathscr{L}_{co}^{\mu(\widetilde{raf}_{2.2})} = \left\{ \begin{array}{l} \left\{ \ (k,\mathtt{in}),(l,\mathtt{out}),(\mathrm{o},\mathtt{in}),(\pi,\mathtt{in}),(\xi,\mathtt{out}) \ \right\}, \\ \left\{ \ (k,\mathtt{out}),(l,\mathtt{in}),(\mathrm{o},\mathtt{in}),(\pi,\mathtt{in}),(\xi,\mathtt{out}) \ \right\}, \\ \left\{ \ (k,\mathtt{und}),(l,\mathtt{und}),(\mathrm{o},\mathtt{in}),(\pi,\mathtt{in}),(\xi,\mathtt{out}) \ \right\} \end{array} \right\}$$

Notice that the union of those three sets coincide with the labellings computed in Example 13.

The following proposition then holds:[18]

Proposition 3. *Let σ be a semantics, $\widetilde{\mathcal{RAF}} = \langle \tilde{A}, \tilde{K}, \tilde{s}, \tilde{t}, s, t \rangle$ be a partial RAF, $\mathcal{I} = \langle S^{inp}, Q^{inp} \rangle$ be the input of $\widetilde{\mathcal{RAF}}$ and let $\alpha \in Q^{inp}$ s.t. $s(\alpha) \in S^{inp}$. Let μ_1, μ_2 be contexts of $\widetilde{\mathcal{RAF}}$ s.t.:*

1. $\forall x \in (S^{inp} \cup Q^{inp}) \setminus \{\alpha, s(\alpha)\}, \mu_1(x) = \mu_2(x)$
2. *and*
   - *either* $(\mu_1(\alpha) = \mathtt{in}$ *and* $\mu_1(s(\alpha)) = \mathtt{in})$ *and* $(\mu_2(\alpha) = \mathtt{in}$ *and* $\mu_2(s(\alpha)) = \mathtt{in})$
   - *or* $(\mu_1(\alpha) = \mathtt{out}$ *or* $\mu_1(s(\alpha)) = \mathtt{out})$ *and* $(\mu_2(\alpha) = \mathtt{out}$ *or* $\mu_2(s(\alpha)) = \mathtt{out})$
   - *or* $\big([(\mu_1(\alpha) = \mathtt{und}$ *or* $\mu_1(s(\alpha)) = \mathtt{und})$ *and* $\mu_1(\alpha) \neq \mathtt{out}$ *and* $\mu_1(s(\alpha)) \neq \mathtt{out}]$ *and* $[(\mu_2(\alpha) = \mathtt{und}$ *or* $\mu_2(s(\alpha)) = \mathtt{und})$ *and* $\mu_2(\alpha) \neq \mathtt{out}$ *and* $\mu_2(s(\alpha)) \neq \mathtt{out}]\big)$

The following property holds:

$$\mathscr{L}_{\sigma}^{\mu_1(\widetilde{\mathcal{RAF}})} = \mathscr{L}_{\sigma}^{\mu_2(\widetilde{\mathcal{RAF}})}$$

Following Proposition 3, only three cases per attack relation (whether the attack element itself and/or an attack source) has to be consider to compute the labellings of any partial RAF.

Moreover, in order to prepare the reunification of all these labellings, we must also identify the labelling "*configurations*" of each partial RAF belonging to a given partition. These configurations are the labellings of the inputs of this partial RAF and of any other input in this partition that can impact this partial RAF. So these configurations are built using the contexts of each cluster of the partition.

Definition 35 (Configuration ξ)**.** *Let σ be a semantics and $\widetilde{\mathcal{RAF}} = \langle \tilde{A}, \tilde{K}, \tilde{s}, \tilde{t}, s, t \rangle$ be a partial RAF. Let $\Omega = \{\omega_1, ..., \omega_n\}$ of $(\tilde{A} \cup \tilde{K})$ and $\left\{ \widetilde{raf}_1, ..., \widetilde{raf}_n \right\}$ be the partition of $\widetilde{\mathcal{RAF}}$ corresponding to Ω s.t. for all $i \in \{1, ..., n\}$, $\widetilde{raf}_i = \langle \tilde{A}_i, \tilde{K}_i, \tilde{s}_i, \tilde{t}_i, s, t \rangle$ and $\mathcal{I}_i = \left\langle S_i^{inp}, Q_i^{inp} \right\rangle$ is the input of $\widetilde{raf}_i$. Let $B = \left\langle B_A = \bigcup_1^n S_i^{inp}, B_K = \bigcup_1^n Q_i^{inp} \right\rangle$ be the structure corresponding the union of all input elements following the partition Ω.*

Let $\widetilde{raf}_i \in \left\{ \widetilde{raf}_1, ..., \widetilde{raf}_n \right\}$ be a partial RAF. Let μ be a context of $\widetilde{raf}_i$, and $\mathcal{L} \in \mathscr{L}_{\sigma}^{\mu(\widetilde{raf}_i)}$ be a computed labelling of $\widetilde{raf}_i$ under μ. Given $\mathcal{L}$, a configuration is astructure labelling over $S_i^{inp} \cup (\tilde{A}_i \cap B_A)$ and $Q_i^{inp} \cup (\tilde{K}_i \cap B_K)$ s.t.:

$$\xi(x) = \begin{cases} \mathcal{L}(x) & \text{if } x \in \omega_i \\ \mu(x) & \text{if } x \in \mathcal{I}_i \end{cases}$$

Example 15. *Following Example 11, each labelling $\mathcal{L}_{2.1.i}$ has its own configuration $\xi_{2.1.i}$. In any configuration, we have $(f, \mathtt{in})$ and $(\psi, \mathtt{in})$, but the label of m varies ($\mathtt{in}$ for $\xi_{2.1.1}$, $\mathtt{out}$ for $\xi_{2.1.2}$, $\mathtt{und}$ for $\xi_{2.1.3}$). Note that m and ψ are not the inputs of $\widetilde{\mathcal{RAF}}_{2.1}$ but they are the inputs of $\widetilde{\mathcal{RAF}}_{2.2}$ that is in the same partition as $\widetilde{\mathcal{RAF}}_{2.1}$. Moreover they belong to $\widetilde{\mathcal{RAF}}_{2.1}$. So any label of these elements can impact the labels inside $\widetilde{\mathcal{RAF}}_{2.1}$.*

[18] The proof is given in Appendix A.

Note: Distinct labellings can have the same configuration. See Example 16.

Example 16. *Following Example 13, the following labellings have the same configuration: is:*

- $\mathcal{L}_{2.2.2} = \{(k, in), (l, out), (o, in), (\pi, in), (\xi, out)\}$
- $\mathcal{L}_{2.2.3} = \{(k, out), (l, in), (o, in), (\pi, in), (\xi, out)\}$
- $\mathcal{L}_{2.2.4} = \{(k, und), (l, und), (o, in), (\pi, in), (\xi, out)\}$

Which is:

$$\xi_{2.2.1} = \{(m, in), (f, in), (\psi, in)\}$$

The configuration corresponding to these 3 labellings is unique, since the label of f must respect the grounded labelling (so the other configurations in which f is labelled either with out or with und are not possible).

Definition 36 (Distinct configuration set)**.** *Let σ be a semantics and $\widetilde{\mathcal{RAF}} = \langle \tilde{A}, \tilde{K}, \tilde{s}, \tilde{t}, s, t \rangle$ be a partial RAF. Let $\mathscr{L}_\sigma(\widetilde{\mathcal{RAF}})$ be the set of labellings of $\widetilde{\mathcal{RAF}}$. We denote by "$\xi^{\widetilde{\mathcal{RAF}}}$" the set of distinct configurations corresponding to $\mathscr{L}_\sigma(\widetilde{\mathcal{RAF}})$.*

Example 17. *Following examples 11 and 13, we have:*

$$\xi^{\widetilde{raf}_{2.1}} = \left\{ \begin{array}{c} \xi_{2.1.1} = \{(m, in), (f, in), (\psi, in)\} \\ \xi_{2.1.2} = \{(m, out), (f, in), (\psi, in)\} \\ \xi_{2.1.3} = \{(m, und), (f, in), (\psi, in)\} \end{array} \right\}$$

With:

- $\xi_{2.1.1}$ *corresponding to* $\mathcal{L}_{2.1.1}$
- $\xi_{2.1.2}$ *corresponding to* $\mathcal{L}_{2.1.2}$
- $\xi_{2.1.3}$ *corresponding to* $\mathcal{L}_{2.1.3}$

And:

$$\xi^{\widetilde{raf}_{2.2}} = \left\{ \begin{array}{c} \xi_{2.2.1} = \{(m, in), (f, in), (\psi, in)\}, \\ \xi_{2.2.2} = \{(m, out), (f, in), (\psi, in)\}, \\ \xi_{2.2.3} = \{(m, und), (f, in), (\psi, in)\} \end{array} \right\}$$

With:

- $\xi_{2.2.1}$ *corresponding to* $\mathcal{L}_{2.2.2}$, $\mathcal{L}_{2.2.3}$ *and* $\mathcal{L}_{2.2.4}$
- $\xi_{2.2.2}$ *corresponding to* $\mathcal{L}_{2.2.1}$
- $\xi_{2.2.3}$ *corresponding to* $\mathcal{L}_{2.2.5}$ *and* $\mathcal{L}_{2.2.6}$

Notice that the fact that $\xi^{\widetilde{raf}_{2.1}}$ coincide with $\xi^{\widetilde{raf}_{2.2}}$ is a particular case. As an example, if $\widetilde{raf}_{2.1}$ had an input element that is not an input of $\widetilde{raf}_{2.2}$, the sets would contain different configurations.

4.4 Reunifying the Results

The labelling reunifying process is made in two steps: first, the reunification of the component labellings (*i.e.* the reunification of their cluster labellings together) and second, the reunification of the whole RAF labellings (*i.e.* the reunification of its component labellings together).

4.4.1 Component labelling reunification

To go further in the explanation of the *RAFDivider* algorithm the notion of "*reunified labelling profile*" is needed:

Definition 37 (Reunified labelling profiles)**.** *Let σ be a semantics and $\widetilde{\mathcal{RAF}} = \langle \tilde{A}, \tilde{K}, \tilde{s}, \tilde{t}, s, t \rangle$ be a partial RAF. Let $\Omega = \{\omega_1, ..., \omega_n\}$ of $(\tilde{A} \cup \tilde{K})$ and $\left\{ \widetilde{raf}_1, ..., \widetilde{raf}_n \right\}$ the partition of $\widetilde{\mathcal{RAF}}$ corresponding to Ω s.t. for all $i \in \{1, ..., n\}$, $\widetilde{raf}_i = \langle \tilde{A}_i, \tilde{K}_i, \tilde{s}_i, \tilde{t}_i, s, t \rangle$ and $\mathcal{I}_i = \left\langle S_i^{inp}, Q_i^{inp} \right\rangle$ is the input of $\widetilde{raf}_i$. Let $B = \left\langle B_A = \bigcup_1^n S_i^{inp}, B_K = \bigcup_1^n Q_i^{inp} \right\rangle$ be the structure corresponding to the union of all input elements following the partition Ω.*

Let $\{\mathscr{L}_\sigma(\widetilde{raf}_1), ..., \mathscr{L}_\sigma(\widetilde{raf}_n)\}$ be the set of the labelling sets of each partial RAF of $\mathcal{RAF}$ and $\{\xi^{\widetilde{raf}_1}, ..., \xi^{\widetilde{raf}_n}\}$ be the set of their corresponding distinct configuration sets. Let $p = \{\xi_1, ..., \xi_n\}$ be a set of configurations s.t., for all $i \in \{1, ..., n\}$, $\xi_i \in \xi^{\widetilde{raf}_i}$. p is a reunified labelling profile *(or equivalently, the configurations $\xi_1, ..., \xi_n$ are said to be compatible together) iff:*

$$\forall x \in B, \ \forall (\xi_j, \xi_k) \in \{\xi_i | \xi_i \in p \text{ s.t. } x \in \xi_i\}^2, \ \xi_j(x) = \xi_k(x)$$

So a reunified labelling profile is a set of labellings s.t. the labels assigned to each common element by the configurations are the same. As for *AFDivider* algorithm, this reunifying problem is transformed into a CSP.

Here are the four steps of the transformation process. Let $\widetilde{raf}_i = \langle \tilde{A}_i, \tilde{K}_i, \tilde{s}_i, \tilde{t}_i, s_i, t_i \rangle$ be a partial RAF corresponding to a connected component of a partial hard RAF. Let $\Omega = \{\omega_1, ..., \omega_n\}$ of $(\tilde{A}_i \cup \tilde{K}_i)$ and $\left\{ \widetilde{raf}_{i.1}, ..., \widetilde{raf}_{i.n} \right\}$ be the partition of $\widetilde{raf}_i$ corresponding to Ω s.t. for all $j \in \{1, ..., n\}$, $\widetilde{raf}_{i.j} = \langle \tilde{A}_{i.j}, \tilde{K}_{i.j}, \tilde{s}_{i.j}, \tilde{t}_{i.j}, s_{i.j}, t_{i.j} \rangle$ and $\mathcal{I}_{i.j} = \left\langle S_{i.j}^{inp}, Q_{i.j}^{inp} \right\rangle$ is the input of $\widetilde{raf}_{i.j}$. Let $B = \left\langle B_A = \bigcup_{j=1}^n S_{i.j}^{inp}, B_K = \bigcup_{j=1}^n Q_{i.j}^{inp} \right\rangle$.

1. For each partial RAF $\widetilde{raf}_{i.j} = \langle \tilde{A}_{i.j}, \tilde{K}_{i.j}, \tilde{s}_{i.j}, \tilde{t}_{i.j}, s_{i.j}, t_{i.j} \rangle$, a variable V_j is created. For each of them, the domain is the set of the distinct configurations corresponding to their computed labellings, *i.e.* $\xi^{\widetilde{raf}_{i.j}}$.
2. For each input element $x_k \in B$, a variable V_{x_k} is created with a domain corresponding to their possible labels, *i.e.* $\{\texttt{in}, \texttt{out}, \texttt{und}\}$ taking into account the optimizations evoked in Section 4.3 (so, in our example, the domain of the variable V_f is $\{\texttt{in}\}$).
3. For each variable V_j with $j \in \{1, ..., n\}$ corresponding to a partial RAF, for each ξ in the domain of V_j, constraints are added to map the configuration with the labels of its corresponding elements. The constraints are defined as follows:
$$\forall j \in \{1, ..., n\}, \ \forall \xi \in \xi^{\widetilde{raf}_{i.j}}, \ V_j = \xi \implies (\forall x_k \in S_{i.j}^{inp} \cup Q_{i.j}^{inp} \cup (\omega_j \cap (B_A \cup B_K)), V_{x_k} = \xi(x_k))$$

`Note:` *The constraints have to be seen as declarative rules. For example the rule: $V_j = \xi \implies (\forall x_k \in S_{i.j}^{inp} \cup Q_{i.j}^{inp} \cup (\omega_j \cap (B_A \cup B_K)), V_{x_k} = \xi(x_k))$ as to be understand as "If the variable V_j has the value ξ, then any variable V_{x_k} s.t. $x_k \in S_{i.j}^{inp} \cup Q_{i.j}^{inp} \cup (\omega_j \cap (B_A \cup B_K)$ must have the value corresponding to $\xi(x_k)$".*

The solutions of that CSP are the reunified labelling profiles (corresponding to values of the V_j variables).

Example 18. *Following Example 17, let illustrate this CSP modelling for the reunification of $\widetilde{raf}_2$. Let $\langle X, D, C \rangle$ be that CSP. It is defined as follows:*

- $X = \{V_1, V_2, V_f, V_m, V_\Psi\}$
  - *V_1 corresponds to $\widetilde{raf}_{2.1}$*
  - *V_2 corresponds to $\widetilde{raf}_{2.2}$*

- V_f *corresponds to the argument* f
- V_m *corresponds to the argument* m
- V_Ψ *corresponds to the attack* Ψ

- $D = \left\{ \begin{array}{l} D(V_1) = \left\{ \begin{array}{l} \xi_{2.1.1} = \{(m, \mathit{in}), (f, \mathit{in}), (\psi, \mathit{in})\} \\ \xi_{2.1.2} = \{(m, \mathit{out}), (f, \mathit{in}), (\psi, \mathit{in})\} \\ \xi_{2.1.3} = \{(m, \mathit{und}), (f, \mathit{in}), (\psi, \mathit{in})\} \end{array} \right\}, \\ D(V_2) = \left\{ \begin{array}{l} \xi_{2.2.1} = \{(m, \mathit{in}), (f, \mathit{in}), (\psi, \mathit{in})\} \\ \xi_{2.2.2} = \{(m, \mathit{out}), (f, \mathit{in}), (\psi, \mathit{in})\} \\ \xi_{2.2.3} = \{(m, \mathit{und}), (f, \mathit{in}), (\psi, \mathit{in})\} \end{array} \right\}, \\ D(V_f) = \{\mathit{in}\} \\ D(V_m) = \{\mathit{in}, \mathit{out}, \mathit{und}\}, \\ D(V_\Psi) = \{\mathit{in}, \mathit{out}, \mathit{und}\} \end{array} \right\}$

- $C = \{c_1, c_2, c_3, c_4, c_5, c_6\}$ *is a set of constraints, with*
  - c_1 *being defined as follows:*
  $$V_1 = \xi_{2.1.1} \implies (V_m = \mathit{in} \wedge V_f = \mathit{in} \wedge V_\psi = \mathit{in})$$
  - c_2 *being defined as follows:*
  $$V_1 = \xi_{2.1.2} \implies (V_m = \mathit{out} \wedge V_f = \mathit{in} \wedge V_\psi = \mathit{in})$$
  - c_3 *being defined as follows:*
  $$V_1 = \xi_{2.1.3} \implies (V_m = \mathit{und} \wedge V_f = \mathit{in} \wedge V_\psi = \mathit{in})$$
  - c_4 *being defined as follows:*
  $$V_2 = \xi_{2.2.1} \implies (V_m = \mathit{in} \wedge V_f = \mathit{in} \wedge V_\psi = \mathit{in})$$
  - c_5 *being defined as follows:*
  $$V_2 = \xi_{2.2.2} \implies (V_m = \mathit{out} \wedge V_f = \mathit{in} \wedge V_\psi = \mathit{in})$$
  - c_6 *being defined as follows:*
  $$V_2 = \xi_{2.2.3} \implies (V_m = \mathit{und} \wedge V_f = \mathit{in} \wedge V_\psi = \mathit{in})$$

The solutions of $\langle X, D, C \rangle$ *are the following ones:*

$$\left\{ \begin{array}{l} \left\{(V_1, \xi_{2.1.1}), (V_2, \xi_{2.2.1}), (V_f, \mathit{in}), (V_m, \mathit{in}), (V_\Psi, \mathit{in})\right\}, \\ \left\{(V_1, \xi_{2.1.2}), (V_2, \xi_{2.2.2}), (V_f, \mathit{in}), (V_m, \mathit{out}), (V_\Psi, \mathit{in})\right\}, \\ \left\{(V_1, \xi_{2.1.3}), (V_2, \xi_{2.2.3}), (V_f, \mathit{in}), (V_m, \mathit{und}), (V_\Psi, \mathit{in})\right\} \end{array} \right\}$$

As consequence the reunified labellings profiles produced by the CSP are the following ones:

$$\mathscr{P} = \left\{ \begin{array}{l} p_{2.1} = \{\xi_{2.1.1}, \xi_{2.2.1}\}, \\ p_{2.2} = \{\xi_{2.1.2}, \xi_{2.2.2}\}, \\ p_{2.3} = \{\xi_{2.1.3}, \xi_{2.2.3}\} \end{array} \right\}$$

Then, each reunified labelling profile computed corresponds to some labelling parts.

Example 19. *Following examples 11, 13 and 18, the labellings corresponding to the reunified profiles are as follows:*

- *Corresponding to* $p_{2.1}$*:*

$$\left\{\begin{array}{l} \mathcal{L}_{2.1.1} \cup \mathcal{L}_{2.2.2} = \left\{\begin{array}{c}(i,out),(h,out),(m,in),(k,in),(l,out),\\ (\imath,in),(\kappa,in),(\lambda,in),(\rho,in),(\psi,in),(\xi,out),(\mathrm{o},in),(\pi,in)\end{array}\right\},\\ \mathcal{L}_{2.1.1} \cup \mathcal{L}_{2.2.3} = \left\{\begin{array}{c}(i,out),(h,out),(m,in),(k,out),(l,in),\\ (\imath,in),(\kappa,in),(\lambda,in),(\rho,in),(\psi,in),(\xi,out),(\mathrm{o},in),(\pi,in)\end{array}\right\},\\ \mathcal{L}_{2.1.1} \cup \mathcal{L}_{2.2.4} = \left\{\begin{array}{c}(i,out),(h,out),(m,in),(k,und),(l,und),\\ (\imath,in),(\kappa,in),(\lambda,in),(\rho,in),(\psi,in),(\xi,out),(\mathrm{o},in),(\pi,in)\end{array}\right\}\end{array}\right\}$$

- *Corresponding to* $p_{2.2}$*:*

$$\left\{\begin{array}{l} \mathcal{L}_{2.1.2} \cup \mathcal{L}_{2.2.1} = \left\{\begin{array}{c}(i,in),(h,in),(m,out),(k,out),(l,in),\\ (\imath,in),(\kappa,in),(\lambda,out),(\rho,in),(\psi,in),(\xi,in),(\mathrm{o},out),(\pi,in)\end{array}\right\}\end{array}\right\}$$

- *Corresponding to* $p_{2.3}$*:*

$$\left\{\begin{array}{l} \mathcal{L}_{2.1.3} \cup \mathcal{L}_{2.2.5} = \left\{\begin{array}{c}(i,und),(h,und),(m,und),(k,out),(l,in),\\ (\imath,in),(\kappa,in),(\lambda,und),(\rho,in),(\psi,in),(\xi,und),(\mathrm{o},und),(\pi,in)\end{array}\right\},\\ \mathcal{L}_{2.1.3} \cup \mathcal{L}_{2.2.6} = \left\{\begin{array}{c}(i,und),(h,und),(m,und),(k,und),(l,und),\\ (\imath,in),(\kappa,in),(\lambda,und),(\rho,in),(\psi,in),(\xi,und),(\mathrm{o},und),(\pi,in)\end{array}\right\}\end{array}\right\}$$

We have so:

$$\mathscr{L}_{co}(\widetilde{raf}_2) = \left\{\begin{array}{l} \mathcal{L}_{2.1} = \left\{\begin{array}{c}(i,out),(h,out),(m,in),(k,in),(l,out),\\ (\imath,in),(\kappa,in),(\lambda,in),(\rho,in),(\psi,in),(\xi,out),(\mathrm{o},in),(\pi,in)\end{array}\right\},\\ \mathcal{L}_{2.2} = \left\{\begin{array}{c}(i,out),(h,out),(m,in),(k,out),(l,in),\\ (\imath,in),(\kappa,in),(\lambda,in),(\rho,in),(\psi,in),(\xi,out),(\mathrm{o},in),(\pi,in)\end{array}\right\},\\ \mathcal{L}_{2.3} = \left\{\begin{array}{c}(i,out),(h,out),(m,in),(k,und),(l,und),\\ (\imath,in),(\kappa,in),(\lambda,in),(\rho,in),(\psi,in),(\xi,out),(\mathrm{o},in),(\pi,in)\end{array}\right\},\\ \mathcal{L}_{2.4} = \left\{\begin{array}{c}(i,in),(h,in),(m,out),(k,out),(l,in),\\ (\imath,in),(\kappa,in),(\lambda,out),(\rho,in),(\psi,in),(\xi,in),(\mathrm{o},out),(\pi,in)\end{array}\right\},\\ \mathcal{L}_{2.5} = \left\{\begin{array}{c}(i,und),(h,und),(m,und),(k,out),(l,in),\\ (\imath,in),(\kappa,in),(\lambda,und),(\rho,in),(\psi,in),(\xi,und),(\mathrm{o},und),(\pi,in)\end{array}\right\},\\ \mathcal{L}_{2.6} = \left\{\begin{array}{c}(i,und),(h,und),(m,und),(k,und),(l,und),\\ (\imath,in),(\kappa,in),(\lambda,und),(\rho,in),(\psi,in),(\xi,und),(\mathrm{o},und),(\pi,in)\end{array}\right\}\end{array}\right\}$$

A special step has to be done for the *preferred* semantics as this reunifying process does not ensure the maximality (*w.r.t.* $\sqsubseteq$) of the set of in-labelled elements. Indeed, the preferred semantics is not bottom-up decomposable (see [3]). A maximality check must be done in order to keep only the wanted labellings. *Note: When computing the* stable *semantics, the set of labellings $\mathscr{L}_\sigma$ returned by the algorithm may be empty. It happens when one of the component clusters has no* stable *labelling.*

4.4.2 Whole RAF labelling reunification

Now that all the component labellings are built, we can reunify the labellings of the whole AF. Indeed, given that the trivial part is a fixed part of all σ-labellings of $\mathcal{RAF}$ and that each connected component has a unique context (these contexts being compatible with each other), the σ-labellings of the whole AF are built by performing a simple cartesian product between the labellings of all the components and the trivial part labelling. If one of the components has no labelling then the whole AF has no labelling (so $\mathscr{L}_\sigma = \varnothing$).

Example 20. *Following examples 9 and 19, the* complete *semantics produces 54 labellings for $\mathcal{RAF}$, with for instance corresponding to $\mathcal{L}_{gr}\downarrow_{\mathcal{U}_{triv}} \cup \mathcal{L}_{4.1} \cup \mathcal{L}_{1.1} \cup \mathcal{L}_{3.1} \cup \mathcal{L}_{2.1}$:*

$$\left\{\begin{array}{c}(e,in),(f,in),(g,out),(n,und),(o,und),(a,in),(b,out),(c,in),\\(d,out),(j,in),(i,out),(h,out),(m,in),(k,in),(l,out),\\(\varepsilon,out),(\zeta,in),(\eta,in),(\theta,in),(\tau,und),(\upsilon,in),(\phi,in),(\chi,in),\\(\alpha,in),(\beta,in),(\gamma,in),(\delta,in),(\mu,out),(\nu,in),(\iota,in),(\kappa,in),\\(\lambda,in),(\rho,in),(\psi,in),(\xi,out),(\mathrm{o},in),(\pi,in)\end{array}\right\}$$

4.5 Synthesis of the running example

Figure 19 gives the synthesis of the different objects used and built by *RAFDivider* for the computation of the complete labellings in the running example.

5 *RAFDivider*: Algorithms and Properties

Algorithms 3 and 4 give the formal definition of the *RAFDivider* algorithm. As for *AFDivider* , they are said to be generic algorithms in the sense that any clustering method can be used to split the AF and any sound and complete procedure that computes the semantics σ, can be used to compute the labellings of the different clusters.

The *RAFDivider* algorithm gives all the expected labellings (so it is complete) and only good labellings (it is sound) for the *complete*, *stable* and *preferred* semantics. The proof of the following propositions are given inSection A. They are very similar to the proofs given for *AFDivider* in [21].

First, concerning Algorithm 4 the two following properties hold:

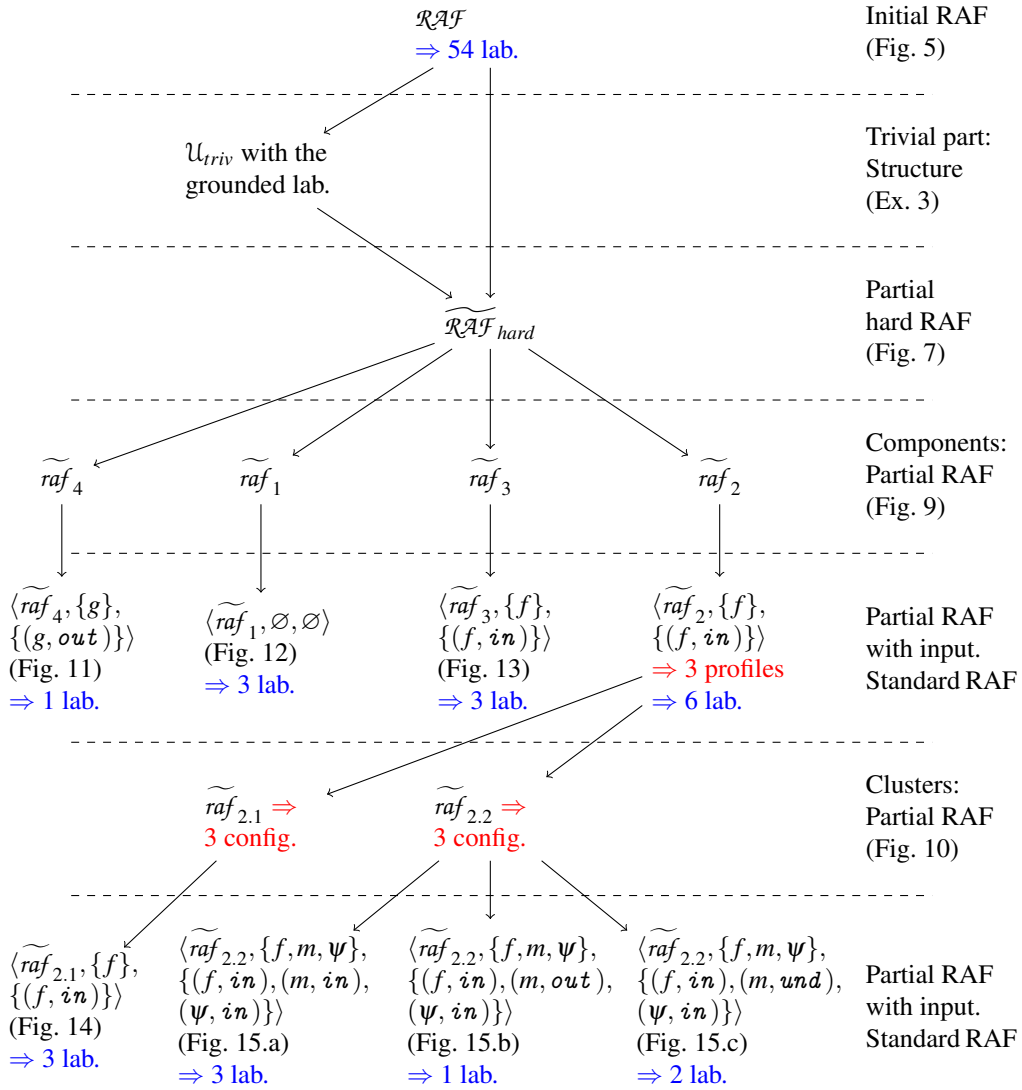
Proposition 4 (Completeness of Algorithm 4)**.** *Algorithm 4 is complete for the* stable*,* complete *and* preferred *semantics.*

Proposition 5 (Soundness of Algorithm 4)**.** *The following properties hold:*

1. *Algorithm 4 is sound for the* stable *and* complete *semantics*
2. *Algorithm 4 is sound for the* preferred *semantics*

Note: Proposition 5 and afterward Proposition 7 are separated into two assertions because the proofs for the preferred *semantics are different.*

Regarding Algorithm 3, two similar properties can be established:



In blue, the number of labellings computed at the level of a partial RAF with input that is not clustered.
In red, the number of distinct configurations obtained for preparing the reunification at the cluster level, then the number of reunified labelling profiles at the level of the partial RAF with input that is clustered; these profiles induce the number of labellings (in blue) for each clustered partial RAF.
And finally, in blue, the total number of labellings of the initial RAF obtained by a cartesian product.

Figure 19: Objects generated by *RAFDivider* in the running example for the complete semantics

Algorithm 3: *RAFDivider* algorithm.

Input: Let $\mathcal{RAF} = \langle A, K, s, t \rangle$ be an RAF and σ be a semantics
Result: $\mathscr{L}_\sigma \in 2^{\mathscr{L}(\mathcal{RAF})}$: the set of the σ-labellings of $\mathcal{RAF}$
Local variables:

- $\mathcal{U}_{triv}$: The trivial part of $\mathcal{RAF}$
- $\mathcal{L}_{gr}$: The *grounded* labelling of $\mathcal{RAF}$
- $\mathcal{I}$: The input of $\widetilde{\mathcal{RAF}}_{hard}$
- *CCSet*: The set of connected components of $\widetilde{\mathcal{RAF}}_{hard}$
- *PartRAFSet*: The set of partial RAFs of $\widetilde{raf}_i$
- $\mathscr{L}_\sigma(raf_i)$: the set of all σ-labellings of $\widetilde{raf}_i$

$\mathcal{U}_{triv}, \mathcal{L}_{gr} \leftarrow$ `ComputeRAFTrivialPart`$(\mathcal{RAF})$
$\widetilde{\mathcal{RAF}}_{hard}, \mathcal{I} \leftarrow$ `RemoveRAFTrivialPart`$(\mathcal{RAF}, \mathcal{U}_{triv})$
$CCSet \leftarrow$ `SplitPartialRAFConnectedComponents`$(\widetilde{\mathcal{RAF}}_{hard})$
for all $\widetilde{raf}_i \in CCSet$ **do in parallel**
  $PartRAFSet \leftarrow$ `ComputePartRAFs`$(\widetilde{raf}_i)$ // clustering
  $\mathscr{L}_\sigma(\widetilde{raf}_i) \leftarrow$ `ComputeRAFCompLabs`$(\sigma, PartRAFSet, \mathcal{I}, \mathcal{L}_{gr})$
$\mathscr{L}_\sigma \leftarrow \{\mathcal{L}_{gr} \downarrow_{\mathcal{U}_{triv}}\} \times \prod_{\widetilde{raf}_i \in CCSet} \mathscr{L}_\sigma(\widetilde{raf}_i)$
return $\mathscr{L}_\sigma$

Algorithm 4: `ComputeRAFCompLabs` algorithm.

Input: Let σ be a semantics, *PartRAFSet* be a set of clusters (partial RAFs) for a component $\widetilde{raf}_i$, $\mathcal{I}$ be the input of the partial hard RAF and $\mathcal{L}_{gr}$ be the grounded labelling of the initial RAF
Result: $\mathscr{L}_\sigma \in 2^{\mathscr{L}(\widetilde{raf}_i)}$: the set of the σ-labellings of $\widetilde{raf}_i$
Local variables:

- $\widetilde{raf}_{i.j}$: a partial RAF
- $\mathscr{L}_\sigma^{\widetilde{raf}_{i.j}}$: the set of all σ-structure labellings of $\widetilde{raf}_{i.j}$
- $\mathscr{P}^{\widetilde{raf}_{i.j}}$: the set of configurations corresponding to the σ-labellings of $\widetilde{raf}_{i.j}$
- $\mathscr{P}$: the set of all reunified labelling profiles

for all $\widetilde{raf}_{i.j} \in PartRAFSet$ **do in parallel**
  $\mathscr{L}_\sigma^{\widetilde{raf}_{i.j}} \leftarrow$ `ComputePartRAFLabs`$(\sigma, \widetilde{raf}_{i.j}, \mathcal{I}, \mathcal{L}_{gr})$ // external solver call
  $\mathscr{P}^{\widetilde{raf}_{i.j}} \leftarrow$ `IdentifyConfigs`$(\mathscr{L}_\sigma^{\widetilde{raf}_{i.j}}, \widetilde{raf}_{i.j})$
$\mathscr{P} \leftarrow$ `ReunifyCompConfigs`$(\{\mathscr{P}^{\widetilde{raf}_{i.j}} | \widetilde{raf}_{i.j} \in PartRAFSet\}, PartRAFSet)$
for all $p \in \mathscr{P}$ **do**
  $\mathscr{L}_\sigma \leftarrow \mathscr{L}_\sigma \cup \left\{ \mathcal{L} \,\middle|\, \mathcal{L} \in$ `ReunifyProfileLabellings`$(p, \{\mathscr{L}_\sigma^{\widetilde{raf}_{i.j}} | \widetilde{raf}_{i.j} \in PartRAFSet\}) \right\}$
if $\sigma = pr$ **then** $\mathscr{L}_\sigma \leftarrow \{\mathcal{L} | \mathcal{L} \in \mathscr{L}_\sigma$ s.t. $\nexists \mathcal{L}' \in \mathscr{L}_\sigma$ s.t. $in(\mathcal{L}) \sqsubset in(\mathcal{L}')\}$
return $\mathscr{L}_\sigma$

Proposition 6 (Completeness of Algorithm 3)**.** *Algorithm 3 is complete for the* stable*,* complete *and* preferred *semantics.*

Proposition 7 (Soundness of Algorithm 3)**.** *The following properties hold:*

1. *Algorithm 3 is sound for the* stable *and* complete *semantics.*
2. *Algorithm 3 is sound for the* preferred *semantics.*

6 A Clustering Method

The main idea of the clustering presented in this section is to ensure that the Strongly Connected Components (SCC)[19] are not split into different clusters. The following method is inspired by those proposed in [16, 21] for testing the *AFDivider* algorithms. Given a RAF, the so-called "*USCC-clustering*" forms clusters as follows (each cluster being an $USCC_{raf}$, see Definition 23). First, the set of SCC is computed. Then neighbour SCC singletons are joined together in order to form a cluster using the following definition of neighbourhood:

Definition 38. (Neighbourhood).
Let $\mathcal{RAF} = \langle A, K, s, t \rangle$. Let x and y be two elements of $\mathcal{RAF}$. x and y are considered as *neighbour* iff:

- either $x \in K$ and ($y = s(x)$ or $y = t(x)$
- or $x \in A$ and $y \in K$ and ($s(y) = x$ or $t(y) = x$)

Note that, by definition, two arguments cannot be considered as neighbours.

In the third step, each SCC that is not a singleton is joined with its neighbour SCC singletons (those that are neighbours with at least an element in the SCC non singleton) producing a cluster. This merging must respect the following constraint (the idea is to put in the same cluster the attacks and their source in order to have all the necessary elements for identifying the status of the target):

> *Let $USCC_{raf}$ be a cluster. Let x be an SCC singleton that is a neighbour of $USCC_{raf}$. x will be joined with $USCC_{raf}$ if*
>
> - *either $x \in K$ and $s(x) \in USCC_{raf}$*
> - *or $x \in A$ and $\exists y \in USCC_{raf}$ s.t. $s(y) = x$*

The last step is to join clusters together so that there are not too many clusters of little size. This is done in an iterative way. The smallest group is merged to its smallest neighbour group, and that until there is no group of less than a certain number of arguments. Some experiments would be necessary in order to identify this threshold wrt the RAF we take into account.

Before illustrating this clustering on the running example, let recall informally what is an SCC in the case of RAF (see the precise formal definitions in [21]). Two elements (arguments or attacks) are in the same SCC iff there exists a RAF-closed-walk that contains and attacks them (a RAF-closed-walk being a directed RAF-walk $(e_1, \ldots, e_n)$ with $e_1 = e_n$).

For instance in the RAF given in Figure 5, one can found the following directed RAF-walks:

- $(\tau, n, \upsilon, o, \phi, \tau)$: a directed RAF-walk that is closed.
- $(f, \theta, g, \eta, e, \zeta, \varepsilon)$: a directed RAF-walk that is not closed.

Then considering the RAF given in Figure 5, we can compute the following set of SCC:

[19] See in [21] the details about the definition of the SCCs for a RAF.

- each unattacked element produces an SCC that is the singleton containing only this element; in the running example, 16 elements are concerned: f and all the attacks except τ, μ, λ, ε, δ, ξ, o;
- there are 5 directed RAF-closed-walk, each of them producing an SCC:
  - $(\tau, n, \upsilon, o, \phi, \tau)$ produces the SCC $\{\tau, n, o\}$
  - (μ, j, ν, μ) produces the SCC $\{\mu, j\}$
  - $(\lambda, i, \iota, m, \rho, h, \kappa, \lambda)$ produces the SCC $\{\lambda, i, m, h\}$
  - $(a, \alpha, b, \beta, c, \gamma, d, \delta, a)$ produces the SCC $\{a, b, c, d\}$
  - $(k, \pi, l, \mathrm{o}, k)$ produces the SCC $\{k, l\}$
- the remaining elements s.t., even if they are attacked, they do not belong to an SCC that could contain another element; so they also produce SCC corresponding to a singleton; in the running example, 6 elements are concerned: g, e, ε, δ, ξ, o.

Then the second step of the *USCC-clustering* (aggregation of neighbour singletons) gives the following results (2 clusters):

- $\{f, \theta, g, \eta, e, \zeta, \varepsilon, \delta, \xi, \mathrm{o}\}$
- $\{\psi, \xi, \mathrm{o}\}$

Considering the 5 SCC that are non singletons, the third step produces the 5 following clusters:

- $\{\tau, n, \upsilon, o, \phi, \chi\}$
- $\{\mu, j, \nu\}$
- $\{\lambda, i, \iota, m, \rho, h, \kappa, \psi\}$
- $\{a, \alpha, b, \beta, c, \gamma, d, \delta\}$
- $\{k, \pi, l, \mathrm{o}\}$

Then in the fourth step, we identify the singletons that belong to several clusters and we remove them to the clusters that do not satisfy the constraints. In our example, 4 elements are concerned: ψ, ξ, δ and o.

- for ψ: we keep it in the cluster containing m
- for ξ: we keep it in the cluster containing f
- for δ: we keep it in the cluster containing d
- for o: we keep it in the cluster containing l

The final result of the fourth step is the 6 following clusters:

- $\{f, \theta, g, \eta, e, \zeta, \varepsilon, \xi\}$
- $\{\tau, n, \upsilon, o, \phi, \chi\}$
- $\{\mu, j, \nu\}$
- $\{\lambda, i, \iota, m, \rho, h, \kappa, \psi\}$
- $\{a, \alpha, b, \beta, c, \gamma, d, \delta\}$
- $\{k, \pi, l, \mathrm{o}\}$

Then in the final step, if we consider a threshold of 6 in order to avoid too small clusters, then the clusters $\{\mu, j, \nu\}$ and $\{k, \pi, l, \mathrm{o}\}$ would be aggregated with the first cluster producing a big cluster containing 15 elements.

Note that, in the real example and following our algorithms, the clustering step happens only after the removal of the trivial part (so on the components of the partial hard RAF) and so the real clustering is not the one described in this section. Indeed, following the size of each component, the clustering is done only of $\widetilde{\mathcal{RAF}}_2$ in which only two SCC exist ($\{\lambda, i, m, h\}$ and $\{k, l\}$) and the *USCC-clustering* produces two clusters: $\{\lambda, i, \iota, m, \rho, h, \kappa, \psi\}$ and $\{k, \pi, l, \mathrm{o}\}$ (see Figure 10).

7 Conclusion and Future Works

This paper presents *RAFDivider*, one of the first algorithms for the enumeration of acceptable sets in a Recursive Argumentation Framework (RAF), an argumentation framework enriched with higher-order interactions. This algorithm, proven sound and complete, is based on a cutting of the framework which allows a distributed and parallel computation, technique successfully used for the enumeration of acceptable sets in an AF by *AFDivider*. An example of a clustering method, USCC-clustering, which can be used with this algorithm, is provided. An implementation of *RAFDivider* is to come.

The extension of the algorithmic approach to other kinds of enriched argumentation frameworks may be investigated: argumentation frameworks which consider support interactions in addition to attacks, notably (see [10] for an overview of such enrichments).

To go further, such algorithms for argumentation frameworks with higher-order attacks may encourage the extension to RAFs of the reasoning tasks proposed for AFs at the International Competition on Computational Models of Argumentation (ICCMA) [20].

References

[1] Gianvincenzo Alfano, Sergio Greco, and Francesco Parisi. Efficient computation of extensions for dynamic abstract argumentation frameworks: An incremental approach. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI*, pages 49–55, 2017.

[2] Mario Alviano. The pyglaf argumentation reasoner. In *OASIcs-OpenAccess Series in Informatics*, volume 58. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2018.

[3] Pietro Baroni, Guido Boella, Federico Cerutti, Massimiliano Giacomin, Leendert Van Der Torre, and Serena Villata. On the input/output behavior of argumentation frameworks. *Artificial Intelligence*, 217:144–197, 2014.

[4] Pietro Baroni, Martin Caminada, and Massimiliano Giacomin. An introduction to argumentation semantics. *Knowledge Eng. Review*, 26(4):365–410, 2011.

[5] Pietro Baroni, Federico Cerutti, Paul E. Dunne, and Massimiliano Giacomin. Computing with infinite argumentation frameworks: The case of AFRAs. In *Proc. of TAFA, Revised Selected Papers*, pages 197–214, 2011.

[6] Pietro Baroni, Federico Cerutti, Massimiliano Giacomin, and Giovanni Guida. AFRA: Argumentation framework with recursive attacks. *Intl. Journal of Approximate Reasoning*, 52:19–37, 2011.

[7] Howard Barringer, Dov Gabbay, and John Woods. Temporal dynamics of support and attack networks : From argumentation to zoology. In *Mechanizing Mathematical Reasoning, Essays in Honor of Jörg H. Siekmann on the Occasion of His 60th Birthday. LNAI 2605*, pages 59–98. Springer Verlag, 2005.

[8] Martin Caminada. On the issue of reinstatement in argumentation. In *JELIA*, pages 111–123, 2006.

[9] Álvaro Carrera and Carlos A Iglesias. A systematic review of argumentation techniques for multi-agent systems research. *Artificial Intelligence Review*, 44(4):509–535, 2015.

[10] Claudette Cayrol, Andrea Cohen, and Marie-Christine Lagasquie-Schiex. Higher-order interactions (bipolar or not) in abstract argumentation: A state of the art. In *Handbook of Formal Argumentation, Volume 2*. 2021.

[11] Claudette Cayrol, Jorge Fandinno, Luis Fariñas del Cerro, and Marie-Christine Lagasquie-Schiex. Valid attacks in argumentation frameworks with recursive attacks. *AMAI Journal*, 89(1):53–101, November 2020.

[12] Federico Cerutti, Massimiliano Giacomin, Mauro Vallati, and Marina Zanella. An SCC recursive meta-algorithm for computing preferred labellings in abstract argumentation. In *Proc. of KR*. AAAI Press, 2014.

[13] Federico Cerutti, Ilias Tachmazidis, Mauro Vallati, Sotirios Batsakis, Massimiliano Giacomin, and Grigoris Antoniou. Exploiting parallelism for hard problems in abstract argumentation. In *AAAI*, pages 1475–1481, 2015.

[14] Günther Charwat, Wolfgang Dvořák, Sarah Alice Gaggl, Johannes Peter Wallner, and Stefan Woltran. Methods for solving reasoning problems in abstract argumentation — A survey. *Artificial Intelligence*, 220:28–63, 2015.

[15] Sylvie Doutre, Mickaël Lafages, and Marie-Christine Lagasquie-Schiex. Argumentation Frameworks with Higher-Order Attacks: Labellings and Complexity. In Miltos Alamaniotis and Shimei Pan, editors, *Proc. of ICTAI*, pages 1210–1217. IEEE, November 2020.

[16] Sylvie Doutre, Mickaël Lafages, and Marie-Christine Lagasquie-Schiex. A distributed and clustering-based algorithm for the enumeration problem in abstract argumentation. In *Proc. of PRIMA*, pages 87–105. Springer, 2019.

[17] Phan Minh Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and *n*-person games. *Artificial Intelligence*, 77(2):321–357, 1995.

[18] Wolfgang Dvorak and Paul E. Dunne. Computational problems in formal argumentation and their complexity. In *Handbook of formal argumentation*, pages 631–688. College publication, 2018.

[19] Wolfgang Dvořák, Reinhard Pichler, and Stefan Woltran. Towards fixed-parameter tractable algorithms for abstract argumentation. *Artificial Intelligence*, 186:1–37, 2012.

[20] ICCMA. International Competition on Computational Models of Argumentation.

[21] Mickaël Lafages. *Algorithms for Enriched Abstract Argumentation Frameworks for Large-scale Cases*. Phd thesis, Toulouse, France, 2021.

[22] Beishui Liao. Toward incremental computation of argumentation semantics: A decomposition-based approach. *Annals of Mathematics and Artificial Intelligence*, 67(3-4):319–358, 2013.

[23] Sanjay Modgil. An abstract theory of argumentation that accommodates defeasible reasoning about preferences. In *Proc. of ECSQARU*, pages 648–659, 2007.

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

A Proofs

Proposition 3 Let σ be a semantics, $\widetilde{\mathcal{RAF}} = \langle \tilde{A}, \tilde{K}, \tilde{s}, \tilde{t}, s, t \rangle$ be a partial RAF, $\mathcal{I} = \langle S^{inp}, Q^{inp} \rangle$ be the input of $\widetilde{\mathcal{RAF}}$ and let $\alpha \in Q^{inp}$ such that $s(\alpha) \in S^{inp}$. Let μ_1, μ_2 be contexts of $\widetilde{\mathcal{RAF}}$ such that:

1. $\forall x \in (S^{inp} \cup Q^{inp}) \setminus \{\alpha, s(\alpha)\}, \mu_1(x) = \mu_2(x)$
2. and
   - either ($\mu_1(\alpha) = \texttt{in}$ and $\mu_1(s(\alpha)) = \texttt{in}$) and ($\mu_2(\alpha) = \texttt{in}$ and $\mu_2(s(\alpha)) = \texttt{in}$)
   - or ($\mu_1(\alpha) = \texttt{out}$ or $\mu_1(s(\alpha)) = \texttt{out}$) and ($\mu_2(\alpha) = \texttt{out}$ or $\mu_2(s(\alpha)) = \texttt{out}$)
   - or $\big($[($\mu_1(\alpha) = \texttt{und}$ or $\mu_1(s(\alpha)) = \texttt{und}$) and $\mu_1(\alpha) \neq \texttt{out}$ and $\mu_1(s(\alpha)) \neq \texttt{out}$] and
     [($\mu_2(\alpha) = \texttt{und}$ or $\mu_2(s(\alpha)) = \texttt{und}$) and $\mu_2(\alpha) \neq \texttt{out}$ and $\mu_2(s(\alpha)) \neq \texttt{out}$]$\big)$

The following property holds:

$$\mathscr{L}_{\sigma}^{\mu_1}(\widetilde{\mathcal{RAF}}) = \mathscr{L}_{\sigma}^{\mu_2}(\widetilde{\mathcal{RAF}})$$

Proof (of Prop. 3) Consider $x = t(\alpha) \in \widetilde{\mathcal{RAF}}$ and let study its label.
Using μ_1, 3 cases must be studied:

- either ($\mu_1(\alpha) = \texttt{in}$ and $\mu_1(s(\alpha)) = \texttt{in}$): so, if α is the only attack targeting x, x is labelled with $\texttt{out}$; moreover, following the assumption saying that, in this case, we also have ($\mu_2(\alpha) = \texttt{in}$ and $\mu_2(s(\alpha)) = \texttt{in}$), then using μ_2, we obtain the same label for x;
- or ($\mu_1(\alpha) = \texttt{out}$ or $\mu_1(s(\alpha)) = \texttt{out}$): so, if α is the only attack targeting x, x is labelled with $\texttt{in}$; moreover, following the assumption saying that, in this case, we also have ($\mu_2(\alpha) = \texttt{out}$ or $\mu_2(s(\alpha)) = \texttt{out}$), then using μ_2, we obtain the same label for x;
- or $\big([(\mu_1(\alpha) = \texttt{und}$ or $\mu_1(s(\alpha)) = \texttt{und})$ and $\mu_1(\alpha) \neq \texttt{out}$ and $\mu_1(s(\alpha)) \neq \texttt{out}]$: so, if α is the only attack targeting x, x is labelled with $\texttt{und}$; moreover, following the assumption saying that, in this case, we also have $[(\mu_2(\alpha) = \texttt{und}$ or $\mu_2(s(\alpha)) = \texttt{und})$ and $\mu_2(\alpha) \neq \texttt{out}$ and $\mu_2(s(\alpha)) \neq \texttt{out}]\big)$, then using μ_2, we obtain the same label for x.

Moreover, all other things being equal, for any other element y in $\widetilde{\mathcal{RAF}}$, the label obtained using μ_1 is equal to the label obtained using μ_2. So, if α is not the only attack targeting x, the label of x is computed with exactly the same information using μ_1 or μ_2.
So $\mathscr{L}_{\sigma}^{\mu_1}(\widetilde{\mathcal{RAF}}) = \mathscr{L}_{\sigma}^{\mu_2}(\widetilde{\mathcal{RAF}})$ □

Lemma 1. *Definition 22 on page 14 and Proposition 2 on page 14 can be applied on partial RAF with input.*

Proof (of Lemma 1) From any partial RAF with input $\langle \widetilde{\mathcal{RAF}} = \langle \tilde{A}, \tilde{K}, \tilde{s}, \tilde{t}, s, t \rangle, \mathcal{I} = \langle S^{inp}, Q^{inp} \rangle, \mathcal{L}^{inp} \rangle$, a specific RAF $\langle A', K', s', t' \rangle$ can be built considering that:

- $A' = \tilde{A} \cup S^{inp} \cup \{t(\alpha) | \alpha \in \tilde{K} \text{ and } \tilde{t}(\alpha) = \text{false}\}$
- $K' = \tilde{K} \cup Q^{inp}$
- $s' = \{(x,y) \in s | x \in K' \text{ and } y \in A'\}$
- $t' = \{(x,y) \in t | x \in K' \text{ and } y \in A' \cup K'\}$

The idea is here to built the RAF from the partial RAF adding the inputs and the missing targets. Then, since the decomposability of semantics has been defined and proven for RAF (see Definition 22 on page 14 and Proposition 2 on page 14), it is also defined and proven for partial RAF with input. □

Proposition 4 Algorithm 4 is complete for the *stable*, *complete* and *preferred* semantics.

Proof (of Prop. 4) Let $\widetilde{\mathcal{RAF}} = \langle \tilde{A}, \tilde{K}, \tilde{s}, \tilde{t}, s, t \rangle$ be a partial RAF, $\Omega = \{\omega_1, ..., \omega_n\}$ be a partition of $\widetilde{\mathcal{RAF}}$ and $\{\widetilde{\mathcal{RAF}} \downarrow_{\omega_1}, ..., \widetilde{\mathcal{RAF}} \downarrow_{\omega_n}\}$ be the set of partial RAFs corresponding to Ω, with for each $\widetilde{\mathcal{RAF}} \downarrow_{\omega_i}$, denoted by $\widetilde{raf}_i$, their inputs $\mathcal{I}_i = (S_i^{inp}, Q_i^{inp})$ and the labellings $\mathcal{L}_i^{inp}$ of these inputs.

Let σ be a top-down decomposable semantics.

Let $\mathscr{L}_{\sigma}(\widetilde{\mathcal{RAF}})$ be the set of distinct labellings of $\widetilde{\mathcal{RAF}}$ according to the semantics σ.

Let $\mathscr{L}_{\sigma}^{*}(\widetilde{\mathcal{RAF}})$ be the set of labellings of $\widetilde{\mathcal{RAF}}$ according to σ obtained by Algorithm 4.

We must show that Algorithm 4 produces all the possible labellings corresponding to the semantics σ, so that $\mathscr{L}_{\sigma}(\widetilde{\mathcal{RAF}}) \subseteq \mathscr{L}_{\sigma}^{*}(\widetilde{\mathcal{RAF}})$.

Let $\mathscr{L}_\sigma^{\widetilde{raf}_i}$ be the set of labellings of $\widetilde{raf}_i$ according to the semantics σ.

Let $\mathscr{L}_\sigma^{*\mu(\widetilde{raf}_i)}$ be the set of labellings of $\widetilde{raf}_i$ under the context μ. These labellings are obtained in Algorithm 4 using an external solver assumed to be sound and complete.

By definition of a semantic top-down decomposable, we have (Definition 22 on page 14 and Lemma 1):

$$\mathscr{L}_\sigma(\widetilde{\mathcal{RAF}}) \subseteq \{\mathcal{L}_1 \cup ... \cup \mathcal{L}_n | \forall i \in \{1,...,n\}, \mathcal{L}_i \in \mathscr{F}_\sigma^{raf}(\widetilde{raf}_i, \mathcal{I}_i, \mathcal{L}_i^{inp})\} \tag{1}$$

Given that the labellings of all partial RAF $\widetilde{raf}_i$ are computed for every possible context, we have, by definition of the context and of the input arguments:

$$\forall i, \forall \mathcal{L}_i^{inp}, \exists \mu^{\widetilde{raf}_i} \text{ s.t. } \mu^{\widetilde{raf}_i} = \mathcal{L}_i^{inp} \tag{2}$$

Given that the external solver that computes the labellings of $\widetilde{raf}_i$ according to the semantics σ is sound and complete, and considering $\widetilde{\mathcal{RAF}}_s$ being the standard RAF w.r.t. the partial RAF with input $\left\langle \widetilde{raf}_i, \mathcal{I}_i, \mu^{\widetilde{raf}_i} \right\rangle$, we have:

$$\forall i, \forall \mu^{\widetilde{raf}_i}, \forall \mathcal{L} \in \mathscr{L}_\sigma(\widetilde{\mathcal{RAF}}_s), \exists \mathcal{L}' \in \mathscr{L}_\sigma^{*\mu(\widetilde{raf}_i)} \text{ s.t. } \mathcal{L}' = \mathcal{L} \downarrow_{\omega_i} \tag{3}$$

So we have:

$$\forall i, \forall \mu^{\widetilde{raf}_i}, \forall \mathcal{L} \in \mathscr{L}_\sigma(\widetilde{\mathcal{RAF}}_s), \mathcal{L} \downarrow_{\omega_i} \in \mathscr{L}_\sigma^{\widetilde{raf}_i} \tag{4}$$

And so (following Definition 21 on page 13):

$$\forall i, \mathscr{F}_\sigma^{raf}(\widetilde{raf}_i, \mathcal{I}_i, \mathcal{L}_i^{inp}) \subseteq \mathscr{L}_\sigma^{\widetilde{raf}_i} \tag{5}$$

As a consequence and because of Equation (1) we have ($\prod$ denoting the cartesian product):

$$\mathscr{L}_\sigma(\widetilde{\mathcal{RAF}}) \subseteq \prod_{i=1...n} \mathscr{L}_\sigma^{\widetilde{raf}_i} \tag{6}$$

Let $\chi = \{\mathcal{L} | \mathcal{L} \in \prod_{i=1...n} \mathscr{L}_\sigma^{\widetilde{raf}_i}$ and $\exists x \in A \cup K$ s.t. x is illegally labelled in $\mathcal{L}\}$ be the set of all possible incorrect labellings (*i.e.* the set of labellings in which there exists an element that is not legally labelled).

We have, by definition of σ:

$$\mathscr{L}_\sigma(\widetilde{\mathcal{RAF}}) \subseteq (\prod_{i=1...n} \mathscr{L}_\sigma^{\widetilde{raf}_i}) \setminus \chi \tag{7}$$

Given that, for all computed labellings, we keep only the compatible configuration, that is the most flexible possible configuration, our CSP modelling does not add extra constraints. The proposed reunification removes, thus, only the labellings belonging to χ.

As a consequence, we have:

$$\mathscr{L}_\sigma(\widetilde{\mathcal{RAF}}) \subseteq \mathscr{L}_\sigma^*(\widetilde{\mathcal{RAF}}) \tag{8}$$

We prove so that for any top-down decomposable semantics σ our algorithm is complete, and so for the complete, stable and preferred semantics following Lemma 1 and so Proposition 2 on page 14.

□

Proposition 5

1. Algorithm 4 is sound for the *stable* and *complete* semantics
2. Algorithm 4 is sound for the *preferred* semantics

Proof (of Prop. 5) Let $\widetilde{\mathcal{RAF}} = \left\langle \tilde{A}, \tilde{K}, \tilde{s}, \tilde{t}, s, t \right\rangle$ be a partial RAF, $\Omega = \{\omega_1, ..., \omega_n\}$ be a partition of $\widetilde{\mathcal{RAF}}$ and $\{\widetilde{\mathcal{RAF}} \downarrow_{\omega_1}, ..., \widetilde{\mathcal{RAF}} \downarrow_{\omega_n}\}$ be the set of partial RAFs corresponding to Ω, with for each $\widetilde{\mathcal{RAF}} \downarrow_{\omega_i}$, denoted by $\widetilde{raf}_i$, the inputs $\mathcal{I}_i = (S_i^{inp}, Q_i^{inp})$ and their labellings $\mathcal{L}_i^{inp}$.

Let σ be a semantics.

Let $\mathscr{L}_\sigma(\widetilde{\mathcal{RAF}})$ be the set of distinct labellings of $\widetilde{\mathcal{RAF}}$ according to the semantics σ.

Let $\mathscr{L}^*_\sigma(\widetilde{\mathcal{RAF}})$ be the set of labellings of $\widetilde{\mathcal{RAF}}$ according to σ obtained by Algorithm 4.

We must show that Algorithm 4 produces only the labellings corresponding to the semantics σ, so that $\mathscr{L}^*_\sigma(\widetilde{\mathcal{RAF}}) \subseteq \mathscr{L}_\sigma(\widetilde{\mathcal{RAF}})$.

- **Assertion 1:** Assume that σ be a fully decomposable and complete-based semantics and let $\mathcal{L}^*$ be a labelling of $\widetilde{\mathcal{RAF}}$ according to σ obtained by Algorithm 4 (so $\mathcal{L}^* \in \mathscr{L}^*_\sigma(\widetilde{\mathcal{RAF}})$).

  Let suppose that $\mathcal{L}^* \notin \mathscr{L}_\sigma(\widetilde{\mathcal{RAF}})$. We will prove that it is impossible with a reductio ad absurdum.

  As σ is a complete-based and fully decomposable semantics we can say that (Definition 22 on page 14 and Lemma 1):

  $$\mathcal{L}^* \notin \{\mathcal{L}_1 \cup ... \cup \mathcal{L}_n | \mathcal{L}_i \in \mathscr{F}^{raf}_\sigma(\widetilde{raf}_i, \mathcal{I}_i, \mathcal{L}_i^{inp})\} \tag{9}$$

  And so:

  $$\exists \omega_i \in \Omega \text{ s.t. } \mathcal{L}^* \downarrow_{\omega_i} \notin \mathscr{F}^{raf}_\sigma(\widetilde{raf}_i, \mathcal{I}_i, \mathcal{L}_i^{inp}) \tag{10}$$

  Let $\left\langle \widetilde{raf}_i, \mathcal{I}_i, \mathcal{L}_i^{inp} \right\rangle$ be the partial RAF with input corresponding to this ω_i.

  Let μ be a context of $\widetilde{raf}_i$ such that $\mu = (\bigcup_{j \in \{1,...,n\} \text{ s.t. } j \neq i} \mathcal{L}_j) \downarrow_{\mathcal{I}_i}$.

  Let $\mathscr{L}^{*\mu(\widetilde{raf}_i)}_\sigma$ be the set of labellings of $\widetilde{raf}_i$ under the context μ produced by Algorithm 4.

  Let $\mathcal{L}'^* \in \mathscr{L}^{*\mu(\widetilde{raf}_i)}_\sigma$ be the labelling coinciding with $\mathcal{L}^* \downarrow_{\omega_i}$ (*i.e.* $\mathcal{L}'^* = \mathcal{L}^* \downarrow_{\omega_i}$).

  We have so:

  $$\mathcal{L}'^* \in \mathscr{L}^{*\mu(\widetilde{raf}_i)}_\sigma \tag{11}$$

  Whereas:

  $$\mathcal{L}'^* \notin \mathscr{F}^{raf}_\sigma(\widetilde{raf}_i, \mathcal{I}_i, \mathcal{L}_i^{inp}) \tag{12}$$

  And so:

  $$\mathscr{L}^{*\mu(\widetilde{raf}_i)}_\sigma \neq \mathscr{F}^{raf}_\sigma(\widetilde{raf}_i, \mathcal{I}_i, \mathcal{L}_i^{inp}) \tag{13}$$

  Nevertheless, according to Definition 34 on page 22 we must have:

  $$\mathscr{L}^{*\mu(\widetilde{raf}_i)}_\sigma = \mathscr{F}^{raf}_\sigma(\widetilde{raf}_i, \mathcal{I}_i, \mathcal{L}_i^{inp}) \tag{14}$$

  Thus, there is a contradiction between Equation (13) and Equation (14).

  From this contradiction we can conclude that:

  $$\mathscr{L}^*_\sigma(\widetilde{\mathcal{RAF}}) \subseteq \mathscr{L}_\sigma(\widetilde{\mathcal{RAF}}) \tag{15}$$

  We prove so that for any fully decomposable and complete-based semantics σ our algorithm is sound, and so for the *complete* and *stable* semantics, following Lemma 1 and so Proposition 2 on page 14.

- **Assertion 2:** Let $\widetilde{\mathcal{RAF}}$ be a partial RAF and σ be the preferred semantics. Given that Algorithm 4 is *complete* for the *preferred* semantics (see Proposition 4 on page 35), $\mathscr{L}^*_{pr}$, the set of all labellings reunified from the different clusters obtained in Algorithm 4 line 7, contains all the preferred labellings of $\widetilde{\mathcal{RAF}}$.

  In Algorithm 4 line 8, we keep from $\mathscr{L}^*_{pr}$ only the maximal (w.r.t $\subseteq$ of $\mathbf{in}$-labelled arguments) labellings, that are by definition the preferred labellings. As a consequence, $\mathscr{L}^*_{pr}$ contains only and all the *preferred* labellings of $\widetilde{\mathcal{RAF}}$.

  Algorithm 4 is, thus, sound and complete for the *preferred* semantics.

□

Proposition 6 Algorithm 3 is complete for the *stable*, *complete* and *preferred* semantics.

Proof (of Prop. 6) Let $\mathcal{RAF} = \langle A, K, s, t\rangle$ be a RAF, $\mathcal{L}_{gr}$ be its grounded labelling, $\widetilde{\mathcal{RAF}}_{hard}$ be the partial hard RAF of $\mathcal{RAF}$ and $\{\widetilde{raf}_1, ..., \widetilde{raf}_n\}$ be the set of partial RAFs obtained from $\widetilde{\mathcal{RAF}}_{hard}$ (so its components).

Let σ be the *complete*, *stable* or *preferred* semantics.

Let $\mathscr{L}^*_\sigma(\mathcal{RAF})$ be the set of labellings obtained from Algorithm 3.

Let $\mathscr{L}_\sigma(\mathcal{RAF})$ be the set of labellings of $\mathcal{RAF}$ according to the semantics σ.

We must prove that $\mathscr{L}_\sigma(\mathcal{RAF}) \subseteq \mathscr{L}^*_\sigma(\mathcal{RAF})$.

Let Ω be the partition of $\mathcal{RAF}$ corresponding to the trivial part and each component. Note that the trivial part induces a partial RAF denoted $\widetilde{raf}_{gr}$ with eventually a set of some inputs denoted $\mathcal{I}_{gr}$ (so this partial RAF can also be considered as a component).

Let $\mathscr{L}^*_\sigma(\widetilde{raf}_i)$ be the set of labellings obtained from Algorithm 4 for the component $\widetilde{raf}_i$.

Let $\mathscr{L}_\sigma(\widetilde{raf}_i)$ be the set of labellings for the component $\widetilde{raf}_i$ according to the semantics σ.

Consider $\mathcal{L} \in \mathscr{L}_\sigma(\mathcal{RAF})$, a labelling of $\mathcal{RAF}$ according to σ.

Given that (following Definition 21 on page 13):

$$\mathscr{F}^{raf}_\sigma(\widetilde{raf}_{gr}, \mathcal{I}_{gr}, (\bigcup_{i\in\{1,...,n\}} \mathcal{L}_\sigma(\widetilde{raf}_i)) \downarrow_{\mathcal{I}_{gr}}) = \{\mathcal{L}_{gr}\} \tag{16}$$

We have by definition of top-down decomposable semantics (following Definition 22 on page 14):

$$\begin{aligned} \mathscr{L}_\sigma(\mathcal{RAF}) &\subseteq \{\mathcal{L}_{gr} \cup \bigcup_{i\in\{1,...,n\}} \mathcal{L}_\sigma(\widetilde{raf}_i)\} \\ \text{with } \mathcal{L}_\sigma(\widetilde{raf}_i) &\in \mathscr{F}^{raf}_\sigma(\widetilde{raf}_i, \mathcal{I}_i, (\bigcup_{j\in\{1,...,n\} \text{ s.t. } j\neq i} \mathcal{L}_\sigma(\widetilde{raf}_j)) \downarrow_{\mathcal{I}_i}) \end{aligned} \tag{17}$$

Given that Algorithm 4 is complete for top-down decomposable semantics (*i.e.* $\forall i, \mathscr{L}_\sigma(\widetilde{raf}_i) \subseteq \mathscr{L}^*_\sigma(\widetilde{raf}_i)$),

$$\forall i, \mathcal{L}_\sigma(\widetilde{raf}_i) \in \mathscr{L}^*_\sigma(\widetilde{raf}_i) \tag{18}$$

Furthermore:

$$\forall \mathcal{L}^* \in \mathscr{L}^*_\sigma(\mathcal{RAF}), \mathcal{L}^* = \mathcal{L}_{gr} \cup \bigcup \mathcal{L}^*_i, \text{ with } \mathcal{L}^*_i \in \mathscr{L}^*_\sigma(\widetilde{raf}_i) \tag{19}$$

We have so:

$$\{\mathcal{L}_{gr} \cup \bigcup_{i=1...n} \mathcal{L}_\sigma(\widetilde{raf}_i)\} \subseteq \mathscr{L}^*_\sigma(\mathcal{RAF}) \tag{20}$$

Finally, we have:

$$\mathscr{L}_\sigma(\mathcal{RAF}) \subseteq \mathscr{L}^*_\sigma(\mathcal{RAF}) \tag{21}$$

We prove so that our algorithm is complete for the *complete*, *stable* and *preferred* semantics. □

Proposition 7

1. Algorithm 3 is sound for the *stable* and *complete* semantics.
2. Algorithm 3 is sound for the *preferred* semantics.

Proof (of Prop. 7)

- **Assertion 1:** Algorithm 3 is sound for the *stable* and *complete* semantics.

  Let $\mathcal{RAF} = \langle A, K, s, t \rangle$ be a RAF, $\mathcal{L}_{gr}$ be its grounded labelling, $\widetilde{\mathcal{RAF}}_{hard}$ be the partial hard RAF of $\mathcal{RAF}$ and $\{\widetilde{raf}_1, ..., \widetilde{raf}_n\}$ be the set of partial RAFs obtained from $\widetilde{\mathcal{RAF}}_{hard}$ (so its components).

  Let σ be the *complete* or *stable* semantics.

  Let $\mathscr{L}^*_\sigma(\mathcal{RAF})$ be the set of labellings of $\mathcal{RAF}$ obtained from Algorithm 3.

  Let $\mathscr{L}_\sigma(\mathcal{RAF})$ be the set of labellings of $\mathcal{RAF}$ according to the semantics σ.

  We must show that $\mathscr{L}^*_\sigma(\mathcal{RAF}) \subseteq \mathscr{L}_\sigma(\mathcal{RAF})$.

  Let $\mathcal{L}^* \in \mathscr{L}^*_\sigma(\mathcal{RAF})$ be a labelling of $\mathcal{RAF}$ computed by Algorithm 3.

  Let Ω be the partition of $\mathcal{RAF}$ corresponding to the trivial part and each component of $\widetilde{\mathcal{RAF}}_{hard}$. Note that the trivial part induces a partial RAF denoted $\widetilde{raf}_{gr}$ with eventually a set of some inputs denoted $\mathcal{I}_{gr}$ (and this partial RAF can be considered as a component).

  Let $\mathscr{L}^*_\sigma(\widetilde{raf}_i)$ be the set of labellings of $\widetilde{raf}_i$ obtained from Algorithm 4.

  Following Algorithm 3, we have:

$$\mathcal{L}^* = \mathcal{L}_{gr} \cup \bigcup \mathcal{L}^*_i, \text{ with } \mathcal{L}^*_i \in \mathscr{L}^*_\sigma(\widetilde{raf}_i) \tag{22}$$

  We have (following Definition 21 on page 13):

$$\mathscr{F}^{raf}_\sigma(\widetilde{raf}_{gr}, \mathcal{I}_{gr}, (\bigcup_{i \in \{1,...,n\}} \mathcal{L}_\sigma(\widetilde{raf}_i)) \downarrow_{\mathcal{I}_{gr}}) = \{\mathcal{L}_{gr}\} \tag{23}$$

  Because σ is a fully decomposable semantics we have so (Definition 22 on page 14):

$$\begin{aligned} \mathscr{L}_\sigma(\mathcal{RAF}) &= \{\mathcal{L}_{gr} \cup \bigcup_{i=1...n} \mathcal{L}_\sigma(\widetilde{raf}_i)\} \\ \text{with } \mathcal{L}_\sigma(\widetilde{raf}_i) &\in \mathscr{F}^{raf}_\sigma(\widetilde{raf}_i, \mathcal{I}_i, (\bigcup_{j \in \{1,...,n\} \text{ s.t. } j \neq i} \mathcal{L}_\sigma(\widetilde{raf}_j)) \downarrow_{\mathcal{I}_i}) \end{aligned} \tag{24}$$

  Given that Equation (24) holds and that Algorithm 4 is sound for fully decomposable semantics (*i.e.* $\forall i$, $\mathscr{L}^*_\sigma(\widetilde{raf}_i) \subseteq \mathscr{L}_\sigma(\widetilde{raf}_i)$), we have:

$$\mathcal{L}^* \in \mathscr{L}_\sigma(\mathcal{RAF}) \tag{25}$$

  And thus:

$$\mathscr{L}^*_\sigma(\mathcal{RAF}) \subseteq \mathscr{L}_\sigma(\mathcal{RAF}) \tag{26}$$

  We prove so that for the *complete* and *stable* semantics our algorithm is sound.

- **Assertion 2:** Algorithm 3 is sound for the *preferred* semantics.

  Let $\mathcal{RAF} = \langle A, K, s, t \rangle$ be a RAF, $\mathcal{L}_{gr}$ be its grounded labelling, $\widetilde{\mathcal{RAF}}_{hard}$ be the partial hard RAF of $\mathcal{RAF}$ and $\{\widetilde{raf}_1, ..., \widetilde{raf}_n\}$ be the set of partial RAFs obtained from $\widetilde{\mathcal{RAF}}_{hard}$ (so its components).

  Let $\mathscr{L}^*_{pr}(\mathcal{RAF})$ be the set of labellings of $\mathcal{RAF}$ obtained from Algorithm 3.

  Let $\mathscr{L}_{pr}(\mathcal{RAF})$ be the set of labellings of $\mathcal{RAF}$ according to the preferred semantics.

  We must show that $\mathscr{L}^*_{pr}(\mathcal{RAF}) \subseteq \mathscr{L}_{pr}(\mathcal{RAF})$.

  Let $\mathcal{L}^* \in \mathscr{L}^*_{pr}(\mathcal{RAF})$ be a labelling of $\mathcal{RAF}$ computed by Algorithm 3.

  Let $\mathscr{L}^*_{pr}(\widetilde{raf}_i)$ be the set of labellings of $\widetilde{raf}_i$ obtained from Algorithm 4.

  Following Algorithm 3, we have:

$$\mathcal{L}^* = \mathcal{L}_{gr} \cup \bigcup \mathcal{L}^*_i, \text{ with } \mathcal{L}^*_i \in \mathscr{L}^*_{pr}(\widetilde{raf}_i) \tag{27}$$

  Let Ω be the partition of $\mathcal{RAF}$ corresponding to the trivial part and each component of $\widetilde{\mathcal{RAF}}_{hard}$. Note that the trivial part induces a partial RAF denoted $\widetilde{raf}_{gr}$ with eventually a set of some inputs denoted $\mathcal{I}_{gr}$ (and this partial RAF can be considered as a component).

  By definition of the grounded labelling, we have:

$$\begin{gathered} \exists x \in A \cup K \text{ s.t. } \mathcal{L}_{gr}(x) = \texttt{und} \implies \\ (\forall \alpha \in K \text{ s.t. } t(\alpha) = x, \mathcal{L}_{gr}(\alpha) \neq \texttt{in} \text{ or } \mathcal{L}_{gr}(s(\alpha)) \neq \texttt{in}) \end{gathered} \tag{28}$$

  Given that:

$$\texttt{und}(\mathcal{L}_{gr}) \cap \widetilde{raf}_{gr} = \varnothing \tag{29}$$

  And that by construction of $\widetilde{raf}_{gr}$:

$$\forall i \in \{1, ..., n\}, \forall x \in \widetilde{raf}_i, \mathcal{L}_{gr}(x) = \texttt{und} \tag{30}$$

  The consequence of Equation (28) is:

$$\forall i \in \{1, ..., n\}, \forall \alpha \in K \text{ s.t. } s(\alpha) \in \widetilde{raf}_{gr} \text{ and } t(\alpha) \in \widetilde{raf}_i, \mathcal{L}_{gr}(s(\alpha)) = \texttt{out} \tag{31}$$

  Let $\mathcal{RAF}'$ be the RAF constructed by removing from $\mathcal{RAF}$ the attacks between its trivial part and its partial hard RAF. As in $\mathcal{RAF}$ all arguments in the trivial part attacking arguments outside the trivial part is labelled $\texttt{out}$ (Equation (31)) their attacks have no effect. The consequence is the following:

$$\mathscr{L}_{pr}(\mathcal{RAF}') = \mathscr{L}_{pr}(\mathcal{RAF}) \tag{32}$$

  Notice that $\mathcal{RAF}'$ has $n+1$ connected components corresponding to the partition Ω, each component being a partial RAF denoted by $\widetilde{raf}_i$, for $i = 0 \ldots n$ with its own inputs denoted by $\mathcal{I}_i$ (we consider that $\widetilde{raf}_{gr}$ is denoted by $\widetilde{raf}_0$). Given that there is no connection (attack) between those connected components, each $\omega_i \in \Omega$ is an $USCC_{raf}$ (see Definition 23 on page 14). As a consequence, following the definition of $\mathscr{S}_{raf\text{-}USCC}$ (Definition 23 on page 14), we have:

$$\Omega \in \mathscr{S}_{raf\text{-}USCC}(\mathcal{RAF}') \tag{33}$$

  As the preferred semantics is fully decomposable w.r.t. $\mathscr{S}_{raf\text{-}USCC}$ (Proposition 2 on page 14), we have:

$$\mathscr{L}_{pr}(\mathcal{RAF}') = \{\mathcal{L}_{pr}(\widetilde{raf}_0) \cup ... \cup \mathcal{L}_{pr}(\widetilde{raf}_n) | \\ \mathcal{L}_{pr}(\widetilde{raf}_i) \in \mathscr{F}_{pr}^{raf}(\widetilde{raf}_i, \mathcal{I}_i, (\bigcup_{j \in \{0,...,n\} \text{ s.t. } j \neq i} \mathcal{L}_{pr}(\widetilde{raf}_j)) \downarrow_{\mathcal{I}_i})\} \tag{34}$$

Notice that:

$$\mathscr{F}_{pr}^{raf}(\widetilde{raf}_0, \mathcal{I}_0, (\bigcup_{j \in \{1,...,n\}} \mathcal{L}_{pr}(\widetilde{raf}_j)) \downarrow_{\mathcal{I}_0}) = \{\mathcal{L}_{gr}\} \tag{35}$$

Notice also that, given Algorithm 4 is sound and complete for the *preferred* semantics (Propositions 4 and 5 on page 35), we have:

$$\forall i \in \{1,...,n\}, \mathscr{F}_{pr}^{raf}(\widetilde{raf}_i, \mathcal{I}_i, (\bigcup_{j \in \{1,...,n\} \text{ s.t. } j \neq i} \mathcal{L}_{pr}(\widetilde{raf}_j)) \downarrow_{\mathcal{I}_i}) = \mathscr{L}_{pr}^*(\widetilde{raf}_i) \tag{36}$$

From the Equations (34) to (36), we have:

$$\mathscr{L}_{pr}(\mathcal{RAF}') = \{\mathcal{L}_{gr} \cup \mathcal{L}_{pr}(\widetilde{raf}_1) \cup ... \cup \mathcal{L}_{pr}(\widetilde{raf}_n) | \mathcal{L}_{pr}(\widetilde{raf}_i) \in \mathscr{L}_{pr}^*(\widetilde{raf}_i)\} \tag{37}$$

From Equations (32) and (37) on the previous page and on the current page, we have:

$$\mathscr{L}_{pr}(\mathcal{RAF}) = \{\mathcal{L}_{gr} \cup \mathcal{L}_{pr}(\widetilde{raf}_1) \cup ... \cup \mathcal{L}_{pr}(\widetilde{raf}_n) | \mathcal{L}_{pr}(\widetilde{raf}_i) \in \mathscr{L}_{pr}^*(\widetilde{raf}_i)\} \tag{38}$$

Finally, from Equations (27) and (38) on the previous page and on the current page we have:

$$\mathscr{L}_{pr}^*(\mathcal{RAF}) = \mathscr{L}_{pr}(\mathcal{RAF}) \tag{39}$$

We prove so that Algorithm 3, when using Algorithm 4 to compute the component labellings, is sound and complete for the *preferred* semantics.

□