A new sexy title ???

Y. Boichut[1], B. Boyer[2], T. Genet[2]and A. Legay[3]
LIFO - Université Orléans, France
Université Rennes 1, France
INRIA - Rennes, France

ABSTRACT.

Commentaire : Qui refait l'Abstract ?

1 Introduction

2 Definitions

Commentaire : Toutes les definitions à propos de Réécriture, substitutions (Q-subst inclues) et Automates d'arbres.

Attention :

pas de def independante de normalized transitions et epsilon-transition uniquement une def globale pour les automates suivie du langage d'un automate d'arbres sans format def.

Comprehensive surveys can be found in [**?**, **?**, **?**] for rewriting and tree automata.

Let $\mathcal{F}$ be a finite set of symbols, each associated with an arity function, and let $\mathcal{X}$ be a countable set of variables. $\mathcal{T}(\mathcal{F},\mathcal{X})$ denotes the set of terms, and $\mathcal{T}(\mathcal{F})$ denotes the set of ground terms (terms without variables). The set of variables of a term t is denoted by $\mathcal{V}ar(t)$. A substitution is a function σ from $\mathcal{X}$ into $\mathcal{T}(\mathcal{F},\mathcal{X})$, which can be uniquely extended to an endomorphism of $\mathcal{T}(\mathcal{F},\mathcal{X})$. A position p for a term t is a word over $\mathbb{N}$. The empty sequence λ denotes the top-most position. The set $\mathcal{P}os(t)$ of positions of a term t is inductively defined by $\mathcal{P}os(t) = \{\lambda\}$ if $t \in \mathcal{X}$ and $\mathcal{P}os(f(t_1,\ldots,t_n)) = \{\lambda\} \cup \{i.p \mid 1 \leq i \leq n \text{ and } p \in \mathcal{P}os(t_i)\}$ otherwise. If $p \in \mathcal{P}os(t)$, then $t|_p$ denotes the subterm of t at position p and $t[s]_p$ denotes the term obtained by replacement of the subterm $t|_p$ at position p by the term s. A term rewriting system (TRS) $\mathcal{R}$ is a set of *rewrite rules* $l \to r$, where $l, r \in \mathcal{T}(\mathcal{F},\mathcal{X})$, $l \notin \mathcal{X}$, and $\mathcal{V}ar(l) \supseteq \mathcal{V}ar(r)$. The TRS $\mathcal{R}$ induces a rewriting relation $\to_{\mathcal{R}}$ on terms as follows. Let $s, t \in \mathcal{T}(\mathcal{F},\mathcal{X})$ and $l \to r \in \mathcal{R}$, $s \to^p_{\mathcal{R}} t$ denotes that there exists a position $p \in \mathcal{P}os(t)$ and a substitution σ such that $s|_p = l\sigma$ and $r = s[r\sigma]_p$. Note that the rewriting position p can generally be omitted, i.e. we generally write $s \to_{\mathcal{R}} t$. The reflexive transitive closure of $\to_{\mathcal{R}}$ is denoted by $\to^*_{\mathcal{R}}$. The set of $\mathcal{R}$-descendants of a set of ground terms I is $\mathcal{R}^*(I) = \{t \in \mathcal{T}(\mathcal{F}) \mid \exists s \in I \text{ s.t. } s \to^*_{\mathcal{R}} t\}$. This set must be bound by the number of rewriting step used : for example $\mathcal{R}(I) = \{t \in \mathcal{T}(\mathcal{F}) \mid \exists s \in I \text{ s.t. } s \to_{\mathcal{R}} t\}$. Similarly, an equation set E is a set of *equations* $l = r$, where $l, r \in \mathcal{T}(\mathcal{F},\mathcal{X})$. For all equation $l = r \in \mathcal{R}$ and all substitution σ we have $l\sigma =_E r\sigma$. The relation $=_E$ is the smallest congruence such that for all substitution σ we have $l\sigma = r\sigma$.By mixing rewriting and equations, given a TRS $\mathcal{R}$ and a set of equations E, a term $s \in \mathcal{T}(\mathcal{F})$ is rewritten modulo E into $t \in \mathcal{T}(\mathcal{F})$, denoted $s \to_{\mathcal{R}/E} t$, if there exist $s \in \mathcal{T}(\mathcal{F})'$ and $t' \in \mathcal{T}(\mathcal{F})$ such that $s =_E s' \to_{\mathcal{R}} t' =_E t$. Thus, the set of $\mathcal{R}$-descendants modulo E of a set of ground terms I is ${\mathcal{R}/E}^*(I) = \{t \in \mathcal{T}(\mathcal{F}) \mid \exists s \in I \text{ s.t. } s \to^*_{\mathcal{R}/E} t\}$.

We now define tree automata. Let Q be a finite set of symbols, with arity 0, called *states* such that $Q \cap \mathcal{F} = \emptyset$. $\mathcal{T}(\mathcal{F} \cup Q)$ is called the set of *configurations*. A *transition* is a rewrite rule $c \to q$, where c is a configuration and q is state. A transition is *normalized* when $c = f(q_1,\ldots,q_n)$, $f \in \mathcal{F}$ whose arity is n, and $q_1,\ldots,q_n \in Q$. A *ε-transition* is a transition of the form $q \to q'$ where q and q' are states.

DEFINITION 1.*[Bottom-up nondeterministic finite tree automaton] A bottom-up nondeterministic finite tree automaton (tree automaton for short) is a quadruple $A = \langle \mathcal{F}, Q, Q_F, \Delta\rangle$ and where $Q_F \subseteq Q$, Δ is a set normalized transitions and ε-transitions.*

The transitive and reflexive *rewriting relation* on $\mathcal{T}(\mathcal{F} \cup Q)$ induced by all the transitions of A is denoted by $\to_A$. The tree language recognized by A in a state q is $\mathcal{L}(A,q) = \{t \in \mathcal{T}(\mathcal{F}) \mid t \to_A q\}$. The language recognized by A is $\mathcal{L}(A) = \bigcup_{q \in Q_F} \mathcal{L}(A,q)$.

3 Tree automata completion

Ici une présentation du principe de la technique de vérif basée sur la complétion d'automate d'arbre à la Genet... schéma ? et parler des différentes étapes de calcul pour une étape de complétion. filtrage, normalisation, application des équations... présentation de l'exemple fil conducteur du papier et montrer les limitations de la technique en cas d'échec => ce qui motive la section suivante...

Lors de la complétion, dans quel cas un $q_1 \xrightarrow{\phi} q_2$ devient un $q_1 \xrightarrow{\phi\vee\phi'} q_2$? J'ai l'impression que notre ordre **strict** sur les formules est l'implication **stricte**, i.e $A > B$ ssi $A \not\Longleftarrow B$ et $A \Longleftarrow B$ (et on aura $A = B$ ssi $A \Leftrightarrow B$.). Par ex, une branche du treilli des formules ($\top$ est en haut et $\bot$ est en bas): $\top > A \vee B > A > \bot$ car $\top \Longleftarrow A \vee B \Longleftarrow A \Longleftarrow \bot$ et $\top \not\Longrightarrow A \vee B \not\Longrightarrow A \not\Longrightarrow \bot$.

Par complétion, on va toujours de la droite vers la gauche, on est strictement croissant dans l'ordre $>$. Logique, car on augmente le nombre "chances" d'avoir la transitions $q_1 \to q_2$. Initialement ça pourrait être $\bot$ même si on ne le fait pas en pratique. Puis on augmente le nombre de chemins, jusqu'à atteindre éventuellement $\top$: on est sûr de l'avoir. Par complétion on ne fait qu'affablir la formule, croitre par rapport à $>$.

Moralité pour remplacer $q_1 \xrightarrow{\phi} q_2$ par $q_1 \xrightarrow{\phi\vee\phi'} q_2$, il faut que $\phi \vee \phi' > \phi$, c'est à dire:

- $\phi \vee \phi' \Longleftarrow \phi$, ça c'est toujours vrai.
- $\phi \vee \phi' \not\Longrightarrow \phi$, ça c'est à vérifier à chaque fois.

4 $\mathcal{R}_{/E}$-Automaton + contruction

Commentaire : trouver un vrai titre
Introduire l'approche choisie : une nouvelle race d'automate permettant de distinguer un terme dont on est sûr qu'il est atteignable d'un terme qui ne l'est pas, juste en reconnaissant le terme... On se place dans le cadre de la vérification d'une propriété d'atteignabilité où le problème est formé par $\mathcal{R}$, I^0 et E.

- Def de $\mathcal{R}_{/E}$-automate
- Exemple (reprise de l'exemple ???) peut-être pas Donner seulement l'intuition qui est derrière juste avec quelques transitions jouet
- Prop 1 (cf. rta10)
- Nouvelle sémantique + Equivalence avec la sémantique standard ! (implique de rester dans la même classe de complexité des automates meilleures que les automates à contraintes, TAGGED automatas ...)

Ensuite on arrive dans la partie construction d'un RE-automate en utilisant un schéma similaire à complétion Genet.

- definition de $A^0_{\mathcal{R}}$: limitation dûe à Prop 1 : I doit être finie mais expliquer qu'il sera facilement possible de contourner la limitation
- construction $A^{i+1}_{\mathcal{R}}$ à partir de $A^i_{\mathcal{R}}$
  - matching + lemme
  - réutilisation de la normalisation definie precedemment + nouveau lemme Prop 1.
  - abstraction liée aux équations

– Théroèmes qui assurent la correction de l'approche ! clôture + contre-exemple

DEFINITION 2.*[Reachable states of a $\mathcal{R}_{/E}$-automaton] Let $A = \langle \mathcal{F}, Q, Q_f, \Delta, \epsilon_R, ?? \rangle$ be a $\mathcal{R}_{/E}$-automaton. The set S of reachable states of A is a set of pairs (q, ϕ) where $q \in Q$ and ϕ is a formula. Starting from the set $Q \times \{\bot\}$, the value of S can be computed using the following two deduction rules :*

$$\frac{\{(q_1,\phi_1),\ldots,(q_n,\phi_n)\} \cup \{(q,\phi)\} \cup P}{\{(q_1,\phi_1),\ldots,(q_n,\phi_n)\} \cup \{(q, \phi \vee \bigwedge_{i=1}^{n} \phi_i)\} \cup P} \qquad \frac{\{(q_1,\phi_1),(q_2,\phi_2)\} \cup P}{\{(q_1,\phi_1),(q_2,(\phi_1 \wedge \phi) \vee \phi_2)\} \cup P}$$

$$\begin{array}{c} \text{if } f(q_1,\ldots,q_n) \to q \in \Delta \\ \text{and } (\phi \vee \bigwedge_{i=1}^{n} \phi_i) > \phi \\ \text{(i.e. } (\phi \vee \bigwedge_{i=1}^{n} \phi_i) \Longrightarrow \not\Longleftarrow \ \phi) \end{array} \qquad \begin{array}{c} \text{if } q_1 \xrightarrow{\phi} q_2 \in \epsilon_{\mathcal{R}} \\ \text{and } (\phi_1 \wedge \phi) \vee \phi_2 > \phi_2 \\ \text{and (i.e. } (\phi_1 \wedge \phi) \vee \phi_2 \Longrightarrow \not\Longleftarrow \ \phi_2 \end{array}$$

DEFINITION 3.*[$\mathcal{R}_{/E}$-automaton emptiness decision] Let $A = \langle \mathcal{F}, Q, Q_f, \Delta, \epsilon_R, ?? \rangle$ be a $\mathcal{R}_{/E}$-automaton. Let S be the set of reachable states of A defined according to definition 2. The language recognized by A is empty if and only if there exists a pair $(q, \phi) \in S$ such that $q \in Q_f$ and ϕ is satisfiable.*

With regards to the reachability problem, this definition, provides a way to distinguish between real counterexamples and terms which can be rejected using abstraction refinement. Indeed, for all final state q and all pair $(q, \phi) \in S$, if ϕ is a tautology then this means that any term recognized by q is reachable. Otherwise, ϕ is the formula to invalidate, i.e. negate some of its atom so that it becomes unsatisfiable.

DEFINITION 4.*[$\mathcal{R}_{/E}$-automaton intersection algorithm]*

Let $A = \langle \mathcal{F}, Q^A, Q_f^A, \Delta^A, \epsilon_R^A, ?? \rangle$ and $B = \langle \mathcal{F}, Q^B, Q_f^B, \Delta^B, \epsilon_R^B, ?? \rangle$ be two $\mathcal{R}_{/E}$-automata. The product $\mathcal{R}_{/E}$-automaton $A \cap B = \langle \mathcal{F}, Q^{A\cap B}, Q_f^{A\cap B}, \Delta^{A\cap B}, \epsilon_R^{A\cap B}, ??? \rangle$ where $Q^{A\cap B}$, $Q_f^{A\cap B}$ and $\Delta^{A\cap B}$ are defined in the usual way: $Q^{A\cap B} = Q^A \times Q^B$, $Q_f^{A\cap B} = Q_f^A \times Q_f^B$ and

- $\Delta^{A\cap B} = \{f((q_1,q_1'),\ldots,(q_n,q_n')) \to (q,q') \mid f(q_1,\ldots,q_n) \to q \in \Delta^A \text{ and } f(q_1',\ldots,q_n') \to q' \in \Delta^B\}$

but $\epsilon_{\mathcal{R}}^{A\cap B}$ is defined by:

- $\epsilon_{\mathcal{R}}^{A\cap B} = \{(q_1,q_1') \xrightarrow{\phi^A \wedge \phi^B} (q_2,q_2') \mid q_1 \xrightarrow{\phi^A} q_2 \in \epsilon_{\mathcal{R}}^A \text{ and } q_1' \xrightarrow{\phi^B} q_2' \in \epsilon_{\mathcal{R}}^B\} \cup$
  $\{(q_1,q_1') \xrightarrow{\phi^A} (q_2,q_1') \mid q_1 \xrightarrow{\phi^A} q_2 \in \epsilon_{\mathcal{R}}^A \text{ and } q_1' \in Q^B \text{ and } \forall q_2' \in Q^B . q_1' \to q_2' \notin \epsilon_{\mathcal{R}}^B\} \cup$
  $\{(q_1,q_1') \xrightarrow{\phi^B} (q_1,q_2') \mid q_1 \in Q^A \text{ and } \forall q_2 \in Q^A . q_1 \to q_2 \notin \epsilon_{\mathcal{R}}^A \text{ and } q_1' \xrightarrow{\phi^B} q_2' \in \epsilon_{\mathcal{R}}^B\}$

DEFINITION 5.*[Deciding the emptiness of the intersection between a $\mathcal{R}_{/E}$-automaton and a tree automaton Bad]*

1. *From Bad build an equivalent $\mathcal{R}_{/E}$-automaton Bad′ where every epsilon transition $q_1 \to q_2$ of Bad is replaced by a transition $q_1 \xrightarrow{\top} q_2$ in Bad′*
2. *Build the intersection between A and Bad′*
3. *Check emptiness of $A \cap$ Bad′*

THEOREM 6. *If $E = \emptyset$ and $\mathcal{R}$ is ground [?, ?], right-linear and monadic [?], linear and semi-monadic [?], linear and inversely growing [?] or linear generalized finite path overlapping [?], then completion of a tree automaton A terminates on $A^*_{\mathcal{R},\emptyset}$ and $\mathcal{L}(A^*_{\mathcal{R},\emptyset}) = \mathcal{R}^*(\mathcal{L}(A))$.*

PROOF. When $E = \emptyset$, the completion algorithm does not produce any transitions for the ϵ_E set and, thus, every transition of $\epsilon_{\mathcal{R}}$ is labelled by $\top$. As a result, this algorithm totally coincides with the one of [?]. In [?], it has been shown that the algorithm of [?] terminates with $E = \emptyset$ for the above classes (Theorem 114). Furthermore, Theorem 45 and Theorem 49 of [?] guarantee that, in this case, $A_{\mathcal{R},\emptyset}$ is such that $\mathcal{L}(A^*_{\mathcal{R},\emptyset}) = \mathcal{R}^*(\mathcal{L}(A))$.

5 Approximation Refinement

Schéma "à la Bouajjani" qui explique le principe de l'approche à la CEGAR:

- Completion + Etape eventuelle de raffinement ...
- Dans quel cas l'étape de raffinement est-elle déclenchée???
  - On a calculé $A^{i+1}_{\mathcal{R}}$
  - Un terme de "bad" est reconnu par l'automate : on calule en utilisant le filtrage toutes les formules ϕ: soit c'est un contre-exemple car il existe $\phi = \top$ (preuve de correction pour le matching alors...) donc le système viole la propriété sinon c'est un contre-exemple sinon on raffine
  - Expliquer que 3 étapes sont nécessaires pour raffiner :
    - supprimer toutes les transitions de $\Delta_{\mathcal{R}}$ telles que toutes les formules deviennent fausses puisque $Eq(q, q') \equiv q \to q' \in \Delta_{\mathcal{R}}$ si toutes les formules ϕ sont fausses alors le terme n'est plus reconnus
    - exliquer le mécanisme pour garder l'information que l'on pouvait inférer par clôture!!! Reprendre l'exemple $f(s(\dots s(a)\dots))$
    - il faut supprimer tous les termes obetnus par réécriture de termes contenus par dans la partie de l'approx que l'on vient de raffiner... on ne sait plus si ils sont atteignables pour l'instant, tant que l'on a pas tenter de compléter l'automate obtenu par l'élagage.

6 Conclusion

Commentaire : à discuter...

A Lemmas, Propositions and Theorems

THEOREM 7. *[Matching Algorithm is complete] Let A be a $\mathcal{R}_{/E}$-automaton, and q one of its states. Assume also l, the linear left member of a rewriting rule. If by using matching algorithm, we deduce that $l\sigma \trianglelefteq q \vdash S$, then we have $\forall(\alpha,\sigma),\ l\sigma \xrightarrow{\alpha}_A q \implies (\alpha,\sigma) \in S$*

PROOF. Assuming $\mathcal{F}$ a set of symbols, $\mathcal{X}$ a set of variable and Q a set of states. We define $A = \langle \mathcal{F}, Q, Q_f, \Delta \cup \Delta_{\mathcal{R}} \cup \Delta_{=} \rangle$; $l \in \mathcal{T}(\mathcal{F},\mathcal{X})$ and $q \in Q$; $\sigma : \mathcal{V}ar(l) \to Q$ and $\alpha = \bigwedge_1^n Eq(q_k, q'_k)$ such that $l\sigma \xrightarrow{\alpha}_A q$.

The proof is done by induction on the term l.

Base case: l is a variable.

In this case, σ must be a Q-substitution of the form $\sigma = \{l \mapsto q'\}$. Using this observation and the hypothesis, we have $q' \xrightarrow{\alpha}_A q$. The matching problem $l \trianglelefteq q$ is solved using Rule (Var). This means that $S = \{(\alpha_k, \{l \mapsto q_k\}) \mid q_k \xrightarrow{\alpha_k}_A q\}$. By definition of S we see that S contains (α,σ).

Induction : Assume now l is a linear term of the form $f(t_1,\dots,t_n)$.

We are going to decompose $f(t_1,\dots,t_n)\sigma \xrightarrow{\alpha}_A q$ into sequences of transitions. First observe that, by splitting σ into $\sigma_1 \dots \sigma_n$, we have that $f(t_1,\dots,t_n)\sigma$ is equal to $f(t_1\sigma_1,\dots,t_n\sigma_n)$. Assume $\sigma = \sigma_1 \sqcup \dots \sqcup \sigma_n$ with $dom(\sigma_i) = \mathcal{V}(t_i)$ and $\forall x \in dom(\sigma_i),\ \sigma_i(x) = \sigma(x)$. Since l is linear, each variable in X occurs at most one time in l. This means that the sets $\mathcal{V}(t_i)$ are disjoints and so are the domains of the σ_i. This ensures that σ is well-defined.

Now, we study the decomposition of $f(t_1\sigma_1,\dots,t_n\sigma_n) \xrightarrow{\alpha}_A q$ to show that transitions of A used to recognized the term $f(t_1\sigma_1,\dots,t_n\sigma_n)$ are considered by the corresponding steps of the matching algorithm.

We observe that the term $f(t_1\sigma_1,\dots,t_n\sigma_n)$ is recognized in state q. Indeed, we have $f(q_1,\dots,q_n) \to q' \in \Delta$, and each subterm $t_i\sigma_i$ is recognized in state q_i such that $t_i\sigma_i \xrightarrow{\alpha_i} q_i$. Composing recognizing of each subterm, we obtain the following sequence:

$$f(t_1,\dots,t_n) \xrightarrow{\alpha_1} f(q_1,t_2,\dots,t_n) \xrightarrow{\bigwedge_1^2 \alpha_i} f(q_1,q_2,t_3,\dots,t_n) \xrightarrow{\bigwedge_1^3 \alpha_i} \dots \xrightarrow{\bigwedge_1^n \alpha_i} f(q_1,\dots,q_n) \xrightarrow{\bigwedge_1^n \alpha_i \wedge \top} q'$$

There are two cases to consider : (1) $q = q'$ and (2) $q \neq q'$. (1) If $q = q'$, the decomposition is complete and $f(t_1\sigma_1,\dots,t_n\sigma_n) \xrightarrow{\alpha}_A q$ with $\alpha = \bigwedge_1^n \alpha_i$.

$$f(t_1\sigma_1,\dots,t_n\sigma_n) \xrightarrow{\bigwedge_{i=1}^n \alpha_i} f(q_1,\dots,q_n) \xrightarrow{\bigwedge_{i=1}^n \alpha_i} q$$

(2) $q \neq q'$: $f(t_1\sigma_1,\dots,t_n\sigma_n) \xrightarrow{\alpha}_A q$ holds only if we have a transition $q' \xrightarrow{\alpha'} q$ such that $\alpha = \bigwedge_1^n \alpha_i \wedge \alpha'$.

$$f(t_1\sigma_1,\dots,t_n\sigma_n) \xrightarrow{\bigwedge_{i=1}^n \alpha_i} f(q_1,\dots,q_n) \xrightarrow{\top} q' \xrightarrow{\alpha'} q$$

By induction, we know that for each sequence $t_i\sigma_i \xrightarrow{\alpha_i} q_i$, the matching problem is solved *i.e.* $t_i \trianglelefteq q_i \vdash S_i$ with S_i contains (α_i,σ_i). Rule (Delta) is applied to all premises $t_i \trianglelefteq q_i \vdash_A S_i$ for the transition $f(q_1,\dots,q_n) \to q' \in \Delta$. From this, we obtain a set $S' = \bigotimes_1^n S_i$. By unfolding

the definition of $\bigotimes$, we have $S = \{(\top, id) \oplus (a^1, s^1) \oplus \ldots (a^n, s^n) \mid (a^i, s^i) \in S_i\}$. Since each S_i contains (α_i, σ_i), S' contains $(\top, id) \oplus (\alpha_1, \sigma_1) \oplus \ldots (\alpha_n, \sigma_n)$ which is, by definition of $\oplus$ equal to $(\bigwedge_1^n \alpha_i,\ \sigma)$. Thus, we obtain a intermediate statement $f(t_1, \ldots, t_n) \lhd q' \vdash_A S'$ such that $f(t_1, \ldots, t_n)\sigma \xrightarrow{\bigwedge_1^n \alpha_i} q'$, where $(\bigwedge_1^n \alpha_i, \sigma) \in S'$.

This statement must correspond to one of the premises of Rule (Epsilon) to produce the expected statement $f(t_1, \ldots, t_n) \trianglelefteq q \vdash_A S$. There are two cases to consider :$q = q'$ and $q \neq q'$.

If $f(q_1, \ldots, q_n) \to q' \in \Delta$ is the last transition used to have $f(t_1, \ldots, t_n)\sigma \xrightarrow{\alpha}_A q$ then we have $\alpha = \bigwedge_1^n \alpha_i$ and we are in the case $q = q'$: this case corresponds to the premiss 0 of Rule (Epsilon) and $S' = S_0$. By definition of Rule (Epsilon), S' is included in S. This means that $(\alpha, \sigma) \in S$.

If we have $q \neq q'$, then it remains a sequence of transitions $q' \xrightarrow{\alpha'} q$ to have $f(t_1, \ldots, t_n)\sigma \xrightarrow{\alpha}_A q$. The couple (α', q') is in the set $\{(q_k, \alpha_k) \mid q_k \xrightarrow{\alpha_k} q\}$.This means that the statement $f(t_1, \ldots, t_n) \trianglelefteq q \vdash_A S'$ is one the remaining premisses. By definition of Rule (Epsilon), S contains all couple $(a \wedge \alpha', s)$ where $(a, s) \in S'$. In particular, S contains $(\bigwedge_1^n \alpha_i \wedge \alpha', \sigma)$ which concludes the proof.

DEFINITION 8. *Normalization Let $A = \langle \mathcal{F}, Q, Q_f, \Delta \cup \Delta_{\mathcal{R}} \cup \Delta_= \rangle$ be a $\mathcal{R}_{/E}$-automaton, and $s \in \mathcal{T}(\mathcal{F} \cup Q)$ a configuration. The normalization $\downarrow (s \mid \Delta)$ extends, by adding normalized transitions, Δ to have a state q such that $s \to^{\not e} q$. Q may also be extended when it is necessary. Assuming Q_{new} a set of new states i.e. $Q \cap Q_{new} = \emptyset$.*

The normalization is done in two mutually inductive steps. $\downarrow (c \mid \Delta)$ It consists to rewrite c by Δ until rewriting is impossible to obtain $d \in \mathcal{T}(\mathcal{F} \cup Q)$: $c \to^{\not e} d$ and $d \not\to^{\not e}$. Then, it computes $\downarrow' (d \mid \Delta)$ which is inductively defined by:

$\downarrow' (q \mid \Delta) =\ \Delta, q \in Q$

$\downarrow' (f(q_1, \ldots, q_n) \mid \Delta) =\ \Delta \cup \{f(q_1, \ldots, q_n) \to q\}, q_i \in Q$ *and* $q \in Q_{new}$

$\downarrow' (f(t_1, \ldots, t_n) \mid \Delta) =\ \downarrow (f(t_1, \ldots, t_n) \mid \downarrow' (t_i \mid \Delta)\ ), t_i \in \mathcal{T}(\mathcal{F} \cup Q) \setminus Q.$

LEMMA 9. *Normalization is done correctly Let A be a $\mathcal{R}_{/E}$-automaton, and c a configuration. After growing transition set and state set of A using $\downarrow (t \mid \Delta)$, there exists a unique state q such that $c \to^{\not e} q$.*

PROOF. Assuming $\mathcal{F}$ a set of symbols, and Q a set of states. We define $A = \langle \mathcal{F}, Q, Q_f, \Delta \cup \Delta_{\mathcal{R}} \cup \Delta_= \rangle$; $c \in \mathcal{T}(\mathcal{F} \cup Q)$.

The proof is done using a measure $\mu : \mathcal{T}(\mathcal{F} \cup Q) \to \mathbb{N}$ that counts the number of occurences of symbols in $\mathcal{F}$ used in a configuration. Example : $\mu(f(q_1, g(q_2), a)) = 3$. We define it inductively by $\mu(q) = 0$ if $q \in Q$, and $\mu(f(t_1, \ldots, t_n)) = 1 + \sum_1^n \mu(t_i)$.

The first step of $\downarrow (t \mid \Delta)$ consists in rewriting the configuration c by transitions of Δ until rewriting becomes impossible. This means that we obtain a configuration d such that $c \to^{\not e} d$. d is unique: rewritting it more by Δ is impossible, and Δ is deterministic by the definition of $\mathcal{R}_{/E}$-automaton.

The second step $\downarrow' (d \mid \Delta)$ with Δ is deterministic, and it is impossible to rewrite again d by Δ. We are proving that there exists a unique state q such that $d \to^{\not e} q$ by induction on the decreasing of $\mu(d)$.

We consider the 3 cases of $\downarrow' (d \mid \Delta)$

1. $\downarrow'(q \mid \Delta) = \Delta$. It means that d is the state q. There exists a unique state ,which is q, such that $d \to^{\not g} q$ obtained by rewriting with Δ deterministic.
2. $\downarrow'(f(q_1,\ldots,q_n) \mid \Delta) = \Delta \cup \{f(q_1,\ldots,q_n) \to q \mid q \in Q_{new}\}$. Each q_i is a state. The configuration $f(q_1,\ldots,q_n)$ can be used as the left-member of a normalised ground transition. We build the new transition $f(q_1,\ldots,q_n) \to q$ using a new state q. Adding a such transition to Δ preserves determinism. We know that it is impossible to rewrite $d = f(q_1,\ldots,q_n)$ using transitions of Δ : the new transition $f(q_1,\ldots,q_n) \to q$ is the unique way to rewrite d. We deduce that $\Delta \cup \{f(q_1,\ldots,q_n) \to q \mid q \in Q_{new}\}$ is deterministic, and $d \to^{\not g} q$.
3. $\downarrow'(f(t_1,\ldots,t_n) \mid \Delta) = \downarrow(f(t_1,\ldots,t_n) \mid \downarrow'(t_i \mid \Delta))$, $t_i \in \mathcal{T}(\mathcal{F} \cup Q) \setminus Q$. Here, we have the direct subterm t_i of d which is not a state. We deduce $\mu(t_i) < \mu(d)$ from the definition of μ. By induction, Δ is extended by $\mid \downarrow'(t_i \mid \Delta)$ to obtain Δ' for which there exists a unique state q such that $t_i \to^{\not g} q$. Using this new set Δ', we unfold $\downarrow(f(t_1,\ldots,t_n) \mid \Delta')$ which consists in rewriting $f(t_1,\ldots,t_n)$ using Δ'. We obtain a new configuration $f(t'_1,\ldots,t'_n)$ where we know at less t'_i is equal to q since the direct subterm t_i can be rewritten in q using Δ'. Note that if some subterms of t_i are also subterms of some other t_j, it will also be rewritten by Δ' in t'_j until rewritting becomes impossible. Each step of rewritting by Δ necessarly replaces a symbol of $\mathcal{F}$ by a state of Q by definition of a normalised transition. This remark allows to prove that $\mu(f(t_1,\ldots,t_n) > \mu(f(t'_1,\ldots,t'_n)$. For the direct subterm t_i, we know $\mu(t_i) > 0$ (t_i is not a state), and $\mu(t'_i) = 0$ (t'_i is the state q). For all other direct subterm t_j with $j <> i$ we deduce $\mu(t_j) >= \mu(t'_j)$ from $t_j \to^{\not g} t'_j$ using Δ'. We have $\mu(f(t_1,\ldots,t_n) > \mu(f(t'_1,\ldots,t'_n)$ by definition of μ, and $f(t'_1,\ldots,t'_n)$ is rewritten as most as possible by the deterministic Δ'. Then, we use again the induction hypothesis to deduce that $\downarrow'(f(t'_1,\ldots,t'_n) \mid \Delta')$ extends $/Delta'$ with normalised transitions in order to have a unique state q such that $f(t'_1,\ldots,t'_n) \to q$. Finally by transivity, we have $d \to^{\not g} q$ using the deterministic set Δ' for d which is equal to $f(t_1,\ldots,t_n)$

Finally, we proved that $\downarrow'(d \mid \Delta)$ extends Δ preserving its determinism such that there exists a state q for which $d \to^{\not g} q$. We also know that $c \to^{\not g} d$ using Δ. We can conclude that $\downarrow(c, \Delta)$ extends Δ preserving its determinism such that there exists a state q for which $c \to^{\not g} q$.

LEMMA 10. *Normalisation and injectivity*

Let $A = \langle \mathcal{F}, Q, Q_f, \Delta \cup \Delta_{\mathcal{R}} \cup \Delta_{=} \rangle$ be a $\mathcal{R}_{/E}$-automaton, and $c \in \mathcal{T}(\mathcal{F} \cup Q)$ a configuration of A. If Δ is injective, then so is $\downarrow(c \mid \Delta)$.

PROOF. Assuming $\mathcal{F}$ a set of symbols, and Q a set of states. We define: $A = \langle \mathcal{F}, Q, Q_f, \Delta \cup \Delta_{\mathcal{R}} \cup \Delta_{=} \rangle$; $c \in \mathcal{T}(\mathcal{F} \cup Q)$.

The property Prop1. holds only if the relation $\to^{\not g}$ induced by Δ is (1) *d*eterministic and (2) *i*njective. We prove each indepently each property is preserved.

(2) The injectivity of $\to^{\not g}$ is preserved by normalization.

By definition of property Prop1., we know that the injectivty of $\to^{\not g}$ is ensured if