Fast Equational Abstraction Refinement for Regular Tree Model Checking

Yohan Boichut, Benoît Boyer, Thomas Genet, Axel Legay

Fast Equational Abstraction Refinement for Regular Tree Model Checking

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

ABSTRACT. *Tree Regular model checking* is the name of a family of techniques for analyzing infinite-state systems in which states are represented by trees and sets of states by tree automata. From the verification point of view, the central problem is to compute the set of reachable states providing a given transition relation. A main obstacle is that this set is in general not computable in a finite time. In this paper, we propose a new CounterExample Guided Abstraction Refinement technique that can be used to check whether a set of state can be reached from the initial set. Contrary to existing techniques, our approach relies on equational abstraction to ease the definition of approximations and on a specific model of tree automata to avoid heavy backward refinement steps.

1 Introduction

At the heart of all the techniques that have been proposed for exploring infinite state spaces, is a symbolic representation that can finitely represent infinite sets of states.

In this paper, we assume that states of the system are represented by trees (terms) and set of states by tree automata. In this context, the transition relation of the system is naturally represented by a set of rewriting rules. It is well-known that this *Tree Regular Model Checking framework* is expressive enough to describe communication protocols [1] as well as a wide range of cryptographic protocols [18, 20, 2] and JAVA applications [5].

In *Tree Regular Model Checking*, the main objective is to compute an automaton representing the set of states of the system. As we are dealing with infinite-state systems, the problem remains undecidable and only partial solutions can be proposed. Among theses solutions, we find the *predicate abstraction methodology* that was promoted by Bouajjani et al. [7, 8]. The idea behind abstract Tree Regular Model Checking consists in computing the automata obtained after successive applications of the rewriting relation and then use techniques coming from the predicate abstraction area in order to over-approximate the set of reachable states. If the property holds on the abstraction, then it also holds on the concrete system. If a counter-example is found on the abstraction, then one has to check if it is indeed a counter-example to the real system. If not, this spurious counter-example must be used to refine the abstraction. Bouajjani's algorithm, which may not terminate, proceeds by successive abstraction/refinement until a decision can be taken.

Independently, Genet et al. [16, 15, 19] proposed *completion* that is another technique to compute an over-approximation of the set of reachable states. The main difference with the work in [7] is that completion techniques use equations to compute the abstraction [19]. Equations gives a simple and formal semantics to abstractions on trees [22]. Contrary to

the work in [7, 8], completion techniques have been applied to very complex case studies such as the verification of (industrial) cryptographic protocols [18, 20, 2] and Java bytecode applications [5]. Unfortunately, these completion techniques do not embed any notion of counter-example based refinement.

The objective of this paper is to overcome the above mentioned problem and propose a *CounterExample Guided Abstraction Refinement procedure* for completion algorithms. Our contribution is in two steps. First, we propose $\mathcal{R}_{/E}$*-automaton*, that is a new extension of tree automata. A $\mathcal{R}_{/E}$-automaton keeps trace of the equations and rewriting rules applied to the initial automaton. The automaton can be used to decide whether a term t (or a set of terms) is reachable from the set of initial states. If the procedure concludes positively, then the term is indeed reachable. If the procedure concludes negatively, then one has to refine the $\mathcal{R}_{/E}$-automaton and restart the process. Our second contribution is to propose a procedure that refines a $\mathcal{R}_{/E}$-automaton in an efficient manner.

Related work. Regular model checking was first apply to compute the set of reachable states of systems whose configurations are represented by words [10, 6]. The approach was then extended to trees and first applied to very simple case studies [1]. In [9], Bouajjani et al. introduced the first Counterexample abstraction techniques for regular model checking and extended it to trees in [7]. One of the main difference with our work is that they have to record all the intermediary automata to decide whether the counter-example is spurious, while we avoid this enumeration by synthetizing the information in a single and hopefully more compact $\mathcal{R}_{/E}$-automaton. Their technique also requires to apply the reverse of the transition relation in a successive manner. This operation, which involves determinization and inclusion checks, may be computationally expensive. Moreover, in their work, they represent the relation with a tree transducer. This automaton, which encodes the full transition relation in a whole, may be huge, while rewriting systems are generally compact. Finally, their abstractions are defined using automata-based predicates which are less declarative than equations. In [4], the authors use rewriting rules instead of transducers, but intermediary steps are still recorded. Moreover, we shall see that our approach is potentially more general than the one in [4]. Finally, our work extends equational abstractions [22, 23] with counterexample detection and refinement.

2 Definitions

In this section, we introduce some definitions and concepts that will be used through the rest of the paper (see also [3, 14, 21]). 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 $\mathcal{V}ar(l) \supseteq \mathcal{V}ar(r)$. A rewrite rule $l \to r$ is *left-linear* if each variable of l occurs only once in l. A TRS $\mathcal{R}$ is left-linear if every rewrite rule $l \to r$ of $\mathcal{R}$ is left-linear. 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 $l \to r \in \mathcal{R}$, $s \to_{\mathcal{R}} t$ denotes that there exists a position $p \in \mathcal{P}os(t)$ and a substitution σ such that $s|_p = l\sigma$ and $r = s[r\sigma]_p$. The reflexive transitive closure of $\to_{\mathcal{R}}$ is denoted by $\to^*_{\mathcal{R}}$ and $s \to^!_{\mathcal{R}} t$ denotes that $s \to^*_{\mathcal{R}} t$ and t is irreducible by $\mathcal{R}$. The set of $\mathcal{R}$-descendants of a set of ground terms I is $\mathcal{R}^*(I) = \{t \in \mathcal{T}(\mathcal{F}) \mid \exists s \in I \text{ s.t. } s \to^*_{\mathcal{R}} t\}$. 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 $l\sigma = r\sigma$. 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 that are used to recognize possibly infinite sets of terms. 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) over the alphabet $\mathcal{F}$ is a tuple $A = \langle \mathcal{F}, Q, Q_F, \Delta \rangle$, where $Q_F \subseteq Q$, Δ is a set of 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 The Tree Regular Model Checking Problem

Our objective is to verify properties of a given system. We will focus on models of systems whose set of reachable states may be, for modeling reasons, infinite [24] – but our solution also works for huge finite-state systems. Our first problem is to provide a symbolic representation to represent and manipulate possibly infinite sets of states. The problem is undecidable and only partial solutions exist. Here, we will use *Tree Regular Model Checking (TRMC)* [1]. In TRMC, a program is a tuple $(\mathcal{F}, I, Rel)$, where

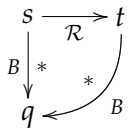
- $\mathcal{F}$ is an alphabet on which a set of terms $\mathcal{T}(\mathcal{F})$ can be defined;
- I is a set of initial configurations represented by a tree automaton A, i.e. $\mathcal{L}(A) = I$;
- Rel is a transition relation represented by a set of left-linear rewriting rules $\mathcal{R}$.

In our setting, a program will thus be represented by the tuple $(\mathcal{F}, A, \mathcal{R})$. It has been shown that the above framework can be used to represent a wide range of applications going from cryptographic protocols to JAVA applications. We consider reachability problems.

DEFINITION 2. *[Reachability problem] Consider a program $(\mathcal{F}, A, \mathcal{R})$ and Bad a set of forbidden terms Bad. The reachability problem consists in checking whether there exists terms of $\mathcal{R}^*(\mathcal{L}(A))$ that belong to Bad.*

For finite-state systems, computing the set of reachable terms ($\mathcal{R}^*(\mathcal{L}(A))$) reduces to enumerate the terms that can be reached from the initial set of configurations. Unfortunately, for infinite-state systems, this enumeration may never terminate. There is thus also a need to "accelerate" the search through the state space in order to reach, in a finite amount of time, states at unbounded depths. Among the existing algorithms used to compute a tree automaton representing the set of reachable terms of a system, one finds *completion algorithm*. A completion algorithm is a semi-algorithm that computes an automaton $A^*_{\mathcal{R}}$ that is possibly an over-approximation of the set of reachable terms. In the rest of this section we introduce the principle of completion and point its current limits.

We say that a tree automaton B is $\mathcal{R}$-closed if for all terms s, t such that $s \to_{\mathcal{R}}$ and s is recognized by B into state q then so is t. The situation is represented with the following graph. It is easy to see that $\mathcal{L}(B) \supseteq \mathcal{R}^*(\mathcal{L}(A))$ if B is $\mathcal{R}$-closed and $\mathcal{L}(B) \supseteq \mathcal{L}(A)$ [12]. From an algorithmic point of view, building a $\mathcal{R}$-closed $A^*_{\mathcal{R}}$ from A consists in *completing* A with new transitions. The completion algorithm computes successive automata $A^1_{\mathcal{R}}, A^2_{\mathcal{R}}, \ldots$ that represent the effect of applying the set of rewriting rules to the initial automaton.



Each application of $\mathcal{R}$ is called a *completion step* and consists in searching for *critical pairs* $\langle t, q\rangle$ where the above diagram is not closed, i.e. $s \to_{\mathcal{R}} t$, $s \to^*_A q$ and $t \not\to^*_A q$. The idea being that the algorithm solves the critical pair by constructing from A, a new tree automaton $A^1_{\mathcal{R}}$ with the additional transitions needed to obtain $t \to^*_{A^1_{\mathcal{R}}} q$, representing the effect of applying $\mathcal{R}$. Then a similar algorithm is applied on $A^1_{\mathcal{R}}$ to obtain $A^2_{\mathcal{R}}$, and so on until a fixpoint $A^*_{\mathcal{R}}$ is reached.

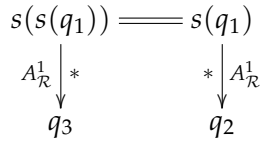
As the language recognized by A may be infinite, it is not possible to find all the critical pairs by enumerating the terms that it recognizes. The solution that was promoted in [16] consists in applying sets of substitutions $\sigma : \mathcal{X} \mapsto Q$ mapping variables of rewrite rules to states representing infinite sets of (recognized) terms. Given a tree automaton $A^i_{\mathcal{R}}$ and a rewrite rule $l \to r \in \mathcal{R}$, to find all the critical pairs of $l \to r$ on $A^i_{\mathcal{R}}$, completion uses a *matching algorithm* [15] that produces the set of substitutions $\sigma : \mathcal{X} \mapsto Q$ and states $q \in Q$ such that $l\sigma \to^*_{A^i_{\mathcal{R}}} q$ and $r\sigma \not\to^*_{A^i_{\mathcal{R}}} q$. Solving critical pairs thus consists in adding new transitions: $r\sigma \to q'$ and $q' \to q$. Those transitions may have to be normalized to respect the definition of transitions of tree automata. As it was shown in [16], this operation may add not only new transitions but also new states to the automaton. In the rest of the paper, the completion-step operation will be represented by $\mathtt{C}$, i.e., the automaton obtained by applying the completion step to $A^i_{\mathcal{R}}$ is denoted $\mathtt{C}(A^i_{\mathcal{R}})$.

The problem is that, except for particular classes [15, 17], the automaton representing the set of reachable terms cannot be obtained from A by applying a finite number of completion steps and the process thus needs to be accelerated. For doing so, one can uses an approximation technique based on a set of equations E and produce an over-approximation of the set of reachable terms, i.e., a tree automaton $A^*_{\mathcal{R},E}$ such that $\mathcal{L}(A^*_{\mathcal{R},E}) \supseteq \mathcal{R}^*(\mathcal{L}(A))$.

To produce such an automaton, each automaton $A^i_{\mathcal{R}}$ obtained by applying i completion steps to A is approximated using a widening function $\mathtt{W}$ parametrized by E. An equation $u = v$ is applied to a tree automaton A as follows: for all substitution $\sigma : \mathcal{X} \mapsto Q$ and distinct states q_1 and q_2 such that $u\sigma \to^*_A q_1$ and $v\sigma \to^*_A q_2$, states q_1 and q_2 are merged.

Completion and widening steps can be linked, i.e. $A^0_{\mathcal{R},E} = A$ and $A^{i+1}_{\mathcal{R},E} = \mathtt{W}(\mathtt{C}(A^i_{\mathcal{R},E}))$, until a $\mathcal{R}$-closed fixpoint $A^*_{\mathcal{R},E}$ is found. In [19], it has been shown that, under some assumptions, the obtained automaton recognizes no more that terms reachable by rewriting with $\mathcal{R}$ modulo E. As a result, the approximation framework and methodology is close to equational abstractions of [22].

EXAMPLE 3. *Let $\mathcal{R} = \{f(x) \to f(s(s(x)))\}$, $E = \{s(s(x)) = s(x)\}$, $A = \langle \mathcal{F}, Q, Q_F, \Delta\rangle$ be a tree automaton such that $Q_F = \{q_0\}$ and $\Delta = \{a \to q_1, f(q_1) \to q_0\}$, i.e. $\mathcal{L}(A) = \{f(a)\}$. The first completion step finds the following critical pair: $f(q_1) \to^*_A q_0$ and $f(s(s(q_1))) \not\to^*_A q_0$. Hence, the completion algorithm produces $A^1_{\mathcal{R}} = \mathtt{C}(A)$ having all transitions of A plus $\{s(q_1) \to q_2, s(q_2) \to q_3, f(q_3) \to q_4, q_4 \to q_0\}$ where q_2, q_3, q_4 are new states produced by normalization of $f(s(s(q_1))) \to q_0$. Applying $\mathtt{W}$ with the equation $s(s(x)) = s(x)$ on $A^1_{\mathcal{R}}$ merges the states q_3 and q_2.*



In [19], $A^1_{\mathcal{R},E} = \mathtt{W}(A^1_{\mathcal{R}})$ is built from $A^1_{\mathcal{R}}$ by renaming q_3 by q_2. The set of transitions of $A^1_{\mathcal{R},E}$ is thus $\Delta \cup \{s(q_1) \to q_2, s(q_2) \to q_2, f(q_2) \to q_4, q_4 \to q_0\}$. Completion stops on $A^1_{\mathcal{R},E}$ because it is $\mathcal{R}$-closed, thus $A^*_{\mathcal{R},E} = A^1_{\mathcal{R},E}$. Now, let us assume that $Bad = \{f(s(a)), f(s(s(a)))\}$. The first term is not in $\mathcal{R}^*(\mathcal{L}(A))$ but the second is. However, those two terms are recognized by $A^*_{\mathcal{R},E}$ and there is no way to distinguish between the two: no way to detect that the second is really *reachable nor to automatically refine the abstraction so as to reject the first one.*

If the intersection between $A^*_{\mathcal{R},E}$ and *Bad* is not empty, then it does not necessarily mean that the system does not satisfy the property. There is thus the need for techniques to decide whether a counter-example is indeed a reachable term that does not satisfy the property or if it is a term added by the abstraction and that cannot be reached from the set of initial states. If the latter case occurs, one has to propose a refinement technique that will remove the false-positive from the abstraction. Studying such techniques for completion automata is the main objective of this paper.

4 $\mathcal{R}_{/E}$-Automaton for refining

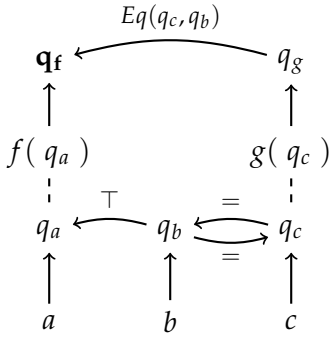
In this section, we propose to extend the completion technique with a counter-example based procedure. Contrary to existing approaches that have to perform a backward propagation from the bad term to the set of initial state, we propose to extend the transition relation of tree automata with information about the rewriting rules and equations that have been applied to the initial automaton.

More precisely, we use the set $\varepsilon_{\mathcal{R}}$ to distinguish a term from its successors that has been obtained by applying one or several rewriting rules. Instead of merging states according to the set of equations, our model link them with transitions that belongs to the set $\varepsilon_=$.

DEFINITION 4. *[$\mathcal{R}_{/E}$-automaton] Given a TRS $\mathcal{R}$ and a set E of equations, a $\mathcal{R}_{/E}$-automaton A is a tuple $\langle \mathcal{F}, Q, Q_F, \Delta \cup \varepsilon_{\mathcal{R}} \cup \varepsilon_=\rangle$. Δ is a set of normalized transitions. $\varepsilon_=$ is a set of ε-transitions. $\varepsilon_{\mathcal{R}}$ is a set of ε-transitions labeled by $\top$ or conjunctions over predicates of the form $Eq(q, q')$ where $q, q' \in Q$, and $q \to q' \in \varepsilon_=$.*

EXAMPLE 5. *Let A be the $\mathcal{R}_{/E}$-automaton recognizing the program where $I = f(a)$, $\mathcal{R} = \{f(c) \to g(c),\ a \to b\}$ and $E = \{b = c\}$.*

In A, the equality $b = c$ is denoted by two transitions $q_c \to q_b$ and $q_b \to q_c$ of $\varepsilon_=$, assuming that b, c are recognized into q_b, q_c, respectively. For the state q_c, the transition $q_b \to q_c$ indicates that the term b is obtained from the term c by equality. Transitions $q_g \to q_f$ and $q_b \to q_a$ denote rewriting steps. Those transitions allow us to deduce $f(a) \to^*_{\mathcal{R}/E} g(c)$ and, $a \to^*_{\mathcal{R}/E} b$. To have $f(a) \to_{\mathcal{R}} f(b) =_E f(c) \to_{\mathcal{R}} g(c)$, which is indeed $f(a) \to^*_{\mathcal{R}/E} g(c)$ unfolded, we use the equality $b = c$ to obtain c from b, relation denoted by the by the transition $q_c \to q_b$. Thus, we label the transition $q_g \to q_f$ with the formula $Eq(q_c, q_b)$ to save this information, whereas the transition $q_b \to q_a$ is labeled with $\top$ which means $a \to^*_{\mathcal{R}} b$.



In what follows, we use $\to^*_\Delta$ to denote the transitive and reflexive closure of Δ. Given a set Δ of normalized transitions, the set of representatives of a state is defined by $Rep(q) = \{t \in \mathcal{T}(\mathcal{F}) | t \to^*_\Delta q\}$. We now formally describe the runs of a $\mathcal{R}_{/E}$-automaton.

DEFINITION 6. *[Run of a $\mathcal{R}_{/E}$-automaton A]*

- $t|_p = f(q_1, \ldots, q_n)$ *and* $f(q_1, \ldots, q_n) \to q \in \Delta$ *then* $t \xrightarrow{\top}_A t[q]_p$
- $t|_p = q$ *and* $q \to q' \in \varepsilon_=$ *then* $t \xrightarrow{Eq(q,q')}_A t[q']_p$
- $t|_p = q$ *and* $q \xrightarrow{\alpha} q' \in \varepsilon_{\mathcal{R}}$ *then* $t \xrightarrow{\alpha}_A t[q']_p$
- $u \xrightarrow{\alpha}_A v$ *and* $v \xrightarrow{\alpha'}_A w$ *then* $u \xrightarrow{\alpha \wedge \alpha'}_A w$

A run $\xrightarrow{\alpha}$ abstracts a rewriting path of $\to_{\mathcal{R}/E}$. If $t \xrightarrow{\alpha} q$, then there exists a term $s \in Rep(q)$ such that $s \to^*_{\mathcal{R}/E} t$. The formula α denotes the subset of transitions of $\varepsilon_=$ needed to recognize t into q.

EXAMPLE 7. *Consider the $\mathcal{R}_{/E}$-automaton A of Example 5 and let $g(b) \xrightarrow{Eq(q_b,q_c) \wedge Eq(q_c,q_b)} q$, we know that there exists a rewriting path of $\to_{\mathcal{R}/E}$ from $f(a)$, the unique term of $Rep(q)$ to $g(b)$. The formula indicates that this rewriting path uses the equivalence relation induced by $b = c$ in both directions (transitions $q_b \to q_c$ and $q_c \to q_b$).*

The relation $\xrightarrow{\alpha}$ corresponds to the standard rewriting relation (see Section 2) of a tree-automaton instrumented with logical formulas.

THEOREM 8. $\forall t \in \mathcal{T}(\mathcal{F} \cup Q),\ q \in Q,\ t \xrightarrow{\alpha}_A q \iff t \to^*_A q$

We now introduce *well-defined* $\mathcal{R}_{/E}$-automata. The well-defined property will be used in the refinement procedure to distinguish between counter-examples and false positives.

DEFINITION 9. *[A well-defined $\mathcal{R}_{/E}$-automaton] A is a* well-defined $\mathcal{R}_{/E}$*-automaton, if :*

- *For all state q of A, and all term v such that $v \xrightarrow{\top}_A q$, there exists u a term representative of q such that $u \to^*_{\mathcal{R}} v$*
- *If $q \xrightarrow{\phi} q'$ is a transition of $\varepsilon_{\mathcal{R}}$, then there exist terms $s, t \in \mathcal{T}(\mathcal{F})$ such that $s \xrightarrow{\phi}_A q$, $t \xrightarrow{\top}_A q'$ and $t \to_{\mathcal{R}} s$.*

The first item in the definition 9 guarantees that every term recognized by using transitions labeled with the formula $\top$ is indeed reachable from the initial set. The second item is used to refine the automaton. A rewriting step of $\rightarrow_{\mathcal{R}/E}$ denoted by $q \xrightarrow{\phi} q'$ holds thanks to some transitions of $\varepsilon_=$ that occurs in ϕ. If we remove transitions in $\varepsilon_=$ in such a way that ϕ does not hold, then the transition $q \xrightarrow{\phi} q'$ should also be removed.

According to the above construction, a term t that is recognized by using at least a transition labeled with a formula different from $\top$ can be removed from the $\mathcal{R}/E$-automaton by removing some transitions in $\varepsilon_=$. This "pruning" operation is illustrated hereafter.

EXAMPLE 10. *We consider the $\mathcal{R}/E$-automaton A of Example 5. This automaton recognizes the term $g(c)$. Indeed, by Definition6, we have $g(c) \xrightarrow{Eq(q_c,q_b)} q_f$. Consider now the rewriting path $f(a) \rightarrow_{\mathcal{R}} f(b) =_E f(c) \rightarrow_{\mathcal{R}} g(c)$. We can see that if the step $f(b) =_E f(c)$ denoted by the transition $q_c \rightarrow q_b$ is removed, then $g(c)$ becomes unreachable. The first step in pruning A consists thus in removing this transition. In a second step, we propagate the information by removing all transition of $\varepsilon_{\mathcal{R}}$ labeled by a formula formed with $Eq(q_c, q_b)$. This is done to remove all terms obtained by rewriting with the equivalence $b =_E c$. After having pruned all the transitions, we observe that the terms recognized by A are given by the following set $\{f(a), f(b)\}$.*

5 On solving the reachability using $\mathcal{R}/E$-automaton

In this section, we extend the completion and widening principles introduced in Section 3 to $\mathcal{R}/E-$automata. We consider an initial set I that can be represented by a tree automaton, and transition relation represented by a set of rewriting rules $\mathcal{R}$. We compute successive approximations $A^i_{\mathcal{R},E}$ from $A^0_{\mathcal{R},E}$ using $A^{i+1}_{\mathcal{R},E} = W(C(A^i_{\mathcal{R},E}))$. We define $A^0_{\mathcal{R},E} = \langle \mathcal{F}, Q^0, Q_F, \Delta^0 \rangle$ that the language of $A^0_{\mathcal{R},E}$ is the terms in I. Observe that $A^0_{\mathcal{R},E}$ is well-defined as the sets $\varepsilon^0_{\mathcal{R}}$ or $\varepsilon^0_=$ are empty. We now detail the completion and widening steps i.e. C and W.

The completion step $\mathtt{C}$. Consider a $\mathcal{R}/E$-automaton $A^i_{\mathcal{R},E} = \langle \mathcal{F}, Q^i, Q_f, \Delta^i \cup \varepsilon^i_{\mathcal{R}} \cup \varepsilon^i_= \rangle$, the completion steps consists in computing an automaton $\mathtt{C}(A^i_{\mathcal{R},E})$ that is obtained from $A^i_{\mathcal{R},E}$ by applying $\mathcal{R}$. As already explained in Section 3, this is done by finding and resolving all critical pairs. A *critical pair* for a $\mathcal{R}/E$-automaton is a triple $\langle r\sigma, \alpha, q \rangle$ such that $l\sigma \rightarrow r\sigma$, $l\sigma \xrightarrow{\alpha}_{A^i_{\mathcal{R},E}} q$ and there is no formula α' such that $r\sigma \xrightarrow{\alpha'}_{A^i_{\mathcal{R},E}} q$. Resolution of such a critical pair consists of adding to $\mathtt{C}(A^i_{\mathcal{R},E})$ the transitions to obtain $r\sigma \xrightarrow{\alpha}_{\mathtt{C}(A^i_{\mathcal{R},E})} q$. This is followed by a normalization step where Q_{new} is a set of new states s.t. $Q_{new} \cap = \emptyset$.

DEFINITION 11. *[Normalization] The normalization is done in two mutually inductive steps parametrized by the configuration c to recognize, and by the set of transitions Δ to extend.*

$$\begin{cases} \mathtt{Norm}(c, \Delta) = \mathtt{Slice}(d, \Delta) & c \rightarrow^!_\Delta d, \text{ and } c,\ d \in \mathcal{T}(\mathcal{F} \cup Q) \\ \mathtt{Slice}(q, \Delta) = \Delta & q \in Q \\ \mathtt{Slice}(f(q_1, \ldots, q_n), \Delta) = \Delta \cup \{f(q_1, \ldots, q_n) \rightarrow q\} & q_i \in Q \text{ and } q \in Q_{new} \\ \mathtt{Slice}(f(t_1, \ldots, t_n), \Delta) = \mathtt{Norm}(f(t_1, \ldots, t_n), \mathtt{Slice}(t_i, \Delta)) & t_i \in \mathcal{T}(\mathcal{F} \cup Q) \setminus Q \end{cases}$$

We thus add transitions such that there exists a state q' with $r\sigma \xrightarrow{\top}_{A^{i+1}_{\mathcal{R},E}} q'$ and $q' \xrightarrow{\alpha} q \in \varepsilon^{i+1}_{\mathcal{R}}$.

We are now ready to define the resolution of a critical pair $p = \langle r\sigma, \alpha, q\rangle$.

DEFINITION 12. *[Resolution of a critical pair] Given a $\mathcal{R}_{/E}$-automaton $A = \langle \mathcal{F}, Q, Q_f, \Delta \cup \varepsilon_{\mathcal{R}} \cup \varepsilon_{=}\rangle$ and a critical pair $p = \langle r\sigma, \alpha, q\rangle$, the* resolution *of p on A is the $\mathcal{R}_{/E}$-automaton $A' = \langle \mathcal{F}, Q', Q_f, \Delta' \cup \varepsilon'_{\mathcal{R}} \cup \varepsilon_{=}\rangle$ where*

- $\Delta' = \Delta \cup \mathtt{Norm}(r\sigma, \Delta \setminus \Delta^0)$
- $\varepsilon'_{\mathcal{R}} = \varepsilon_{\mathcal{R}} \cup \{q' \xrightarrow{\alpha} q\}$ *where q' is the state such that $r\sigma \rightarrow^{!}_{\Delta' \setminus \Delta_0} q'$*
- *Q' is the union of Q and the set of states occurring in Δ'*

Note that Δ_0, the set of transitions of $A^0_{\mathcal{R}}$, is never used for normalization of all $r\sigma$. This is needed to preserve the well-defidness of A'. The $\mathcal{R}_{/E}$-automaton $\mathtt{C}(A^i_{\mathcal{R},E})$ is obtained by recursively applying the above resolution principle to all critical pairs p of the set of critical pairs between $\mathcal{R}$ and $A^i_{\mathcal{R},E}$. The set of all critical pairs is obtained by solving *matching problems* $l \trianglelefteq q$ for all rewrite rule $l \rightarrow r \in \mathcal{R}$ and all state $q \in A^i_{\mathcal{R},E}$. Solving the matching problem $l \trianglelefteq q$ consists of computing S that is the set of all couples (α, σ) such that α is a formula, σ is a substitution of $\mathcal{X} \mapsto Q^i$, and $l\sigma \xrightarrow{\alpha} q$. Each configuration $l\sigma$ corresponds to a subset of terms of $\mathcal{L}(A^i_{\mathcal{R},E}, q)$ that can be rewritten by $l \rightarrow r$. The terms characterized by $l\sigma$ are defined as $l\sigma\sigma'$ where $\sigma' : Q \mapsto \mathcal{T}(\mathcal{F})$ is a substitution, which maps each state q to a term $\sigma'(q) \in \mathcal{L}(A^i_{\mathcal{R},E}, q)$. Definition 13 introduces the matching algorithm to compute the set S, which is denoted by the statement $l \trianglelefteq q \vdash_{A^i_{\mathcal{R},E}} S$. Note that when S is empty, there is no term to rewrite by $l \rightarrow r$.

DEFINITION 13. *[Matching Algorithm]*
Assuming the matching problem $l \trianglelefteq q$ for a $\mathcal{R}_{/E}$-automaton $A^i_{\mathcal{R},E}$. S is the solution of the matching problem, if there exists a derivation of the statement $l \trianglelefteq q \vdash_{A^i_{\mathcal{R},E}} S$ using the rules:

$$\textit{(Var)}\ \frac{}{x \trianglelefteq q \vdash_A \{(\alpha_k, \{x \mapsto q_k\}) \mid q_k \xrightarrow{\alpha_k}_A q\}}(x \in \mathcal{X})$$

$$\textit{(Delta)}\ \frac{t_1 \trianglelefteq q_1 \vdash_A S_1 \quad \dots \quad t_n \trianglelefteq q_n \vdash_A S_n}{f(t_1, \dots, t_n) \trianglelefteq q \vdash_A \bigotimes_1^n S_k}\ (\ f(q_1, \dots, q_n) \rightarrow q \in \Delta\ )^*$$

$$\textit{(Epsilon)}\ \frac{t \trianglelefteq q \vdash_A S_0 \quad t \trianglelefteq q'_1 \vdash_A S_1 \quad \dots \quad t \trianglelefteq q'_n \vdash_A S_n}{t \trianglelefteq q \vdash_A S_0 \cup \bigcup_{k=1}^n \{(\phi \wedge \alpha_k, \sigma) \mid (\phi, \sigma) \in S_k\}}\ \begin{pmatrix} \{(q_k, \alpha_k) \mid q_k \xrightarrow{\alpha_k} q\}_1^n \\ t \notin \mathcal{X} \end{pmatrix}$$

Observe that, by definition, the matching problem considers possibly infinite runs of the form $l\sigma \xrightarrow{\alpha} q$. Indeed, transitions in $\varepsilon^i_{\mathcal{R}} \cup \varepsilon^i_{=}$ can produce loops. In the matching algorithm, we exclude such runs. This is done to keep a finite set of rewriting path, which is computable in a finite amount of time. It is worth mentioning that removing loops does not remove any important information. Consider the automaton A of Example 5. We observe that $f(b) \xrightarrow{Eq(q_b,q_c) \wedge Eq(q_c,q_b)}_A q_f$ uses the loop $f(b) =_E f(c) =_E f(b)$. This loop can be removed as $f(a) \rightarrow^*_{\mathcal{R}} f(b)$ can be obtained by $f(b) \xrightarrow{\top}_A q_f$, which does not contain any loops.

PROPERTY 14. *Let A be a $\mathcal{R}_{/E}-$automaton, q one of its states, $l \in \mathcal{T}(\mathcal{F}, \mathcal{X})$ the linear left member of a rewriting rule and a substitution $\sigma : \mathcal{X} \mapsto Q$ whose domain is range-restricted to $\mathcal{V}(l)$. Assuming that S is the solution of the matching problem $l\sigma \trianglelefteq q$, for all (α, σ) such that $l\sigma \xrightarrow{\alpha}_A q$, a loop free run, then we have $(\alpha, \sigma) \in S$.*

*using $\bigotimes_1^n S_j = \{(\top, id) \oplus (\phi_1, \sigma_1) \oplus \dots \oplus (\phi_n, \sigma_n) \mid (\phi_j, \sigma_j) \in S_j\}$, and $(\phi, \sigma) \oplus (\phi', \sigma') = (\phi \wedge \phi', \sigma \cup \sigma')$.

After computing S for $l \trianglelefteq q$, we identify its elements that correspond to critical pairs. By definition of S, we know that there exists a transition $l\sigma \xrightarrow{\alpha}_{A^i_{\mathcal{R},E}} q$ for $(\alpha, \sigma) \in S$. If there exists a transition $r\sigma \xrightarrow{\alpha'}_{A^i_{\mathcal{R},E}} q$, then $r\sigma$ has already been added to $A^i_{\mathcal{R},E}$. It is thus sufficient to deduce that all terms $l\sigma\sigma'$ of the set represented by the configuration $l\sigma$ are rewritten into terms $r\sigma\sigma'$ represented by the configuration $r\sigma$. In the case where there exists no transition $r\sigma \xrightarrow{\alpha'}_{A^i_{\mathcal{R},E}} q$, then $\langle r\sigma, \alpha', q\rangle$ is a critical pair to solve on $A^i_{\mathcal{R},E}$. The following theorem shows that our methodology is complete.

THEOREM 15. *If $A^i_{\mathcal{R},E}$ is well-defined then so is $\mathtt{C}(A^i_{\mathcal{R},E})$, and $\forall q \in Q^i$, $\forall t \in \mathcal{L}(A^i_{\mathcal{R},E}, q)$, $\forall t \in \mathcal{T}(\mathcal{F})$, $t \to_{\mathcal{R}} t' \implies t' \in \mathcal{L}(\mathtt{C}(A^i_{\mathcal{R},E}), q)$.*

EXAMPLE 16. *Let $\mathcal{R} = \{f(x) \to f(s(s(x)))\}$ and $A^0_{\mathcal{R},E} = \langle \mathcal{F}, Q, Q_F, \Delta^0\rangle$ be a tree automaton such that $Q_F = \{q_0\}$ and $\Delta^0 = \{a \to q_1, f(q_1) \to q_0\}$. Following Definition 13, the solution S of the matching problem $f(x) \trianglelefteq q_0$ is $S = \{(\sigma, \phi)\}$ with $\sigma = \{x \to q_1\}$ and $\phi = \top$. Hence, $\langle f(s(s(q_1))), \top, q_0\rangle$ is the only critical pair to solve, since $f(s(s(q_1))) \overset{\top}{\not\to}_{A^0_{\mathcal{R},E}} q_0$. So, $\mathtt{C}(A^0_{\mathcal{R},E})$ is a $\mathcal{R}_{/E}-$automaton such that $\mathtt{C}(A^0_{\mathcal{R},E}) = \langle \mathcal{F}, Q^1, Q_F, \Delta^1 \cup \varepsilon^1_{\mathcal{R}} \cup \varepsilon^0_{=}\rangle$ with:*
$\Delta^1 = \mathtt{Norm}(f(s(s(q_1))), \emptyset) \cup \Delta^0 = \{s(q_1) \to q_2, s(q_2) \to q_3, f(q_3) \to q_4\} \cup \Delta^0$,
$\varepsilon^1_{\mathcal{R}} = \{q_4 \xrightarrow{\top} q_0\}$, since $f(s(s(q_1))) \to^{!}_{\Delta^1\setminus\Delta^0} q_4$,
$\varepsilon^0_{=} = \emptyset$ and $Q^1 = \{q_0, q_1, q_2, q_3, q_4\}$.

Observe that if $\mathtt{C}(A^i_{\mathcal{R},E}) = A^i_{\mathcal{R},E}$, then we have reached a fixpoint.

The Widening $\mathtt{W}$. Consider a $\mathcal{R}_{/E}$-automaton $A = \langle \mathcal{F}, Q, Q_f, \Delta \cup \varepsilon_{\mathcal{R}} \cup \varepsilon_{=}\rangle$, the widening operation consists in computing a $\mathcal{R}_{/E}$-automaton $\mathtt{W}(A)$ that is obtained from A by using the set of equations E.

For each equation $l = r$ in E, we consider all pair (q, q') of distinct states of Q^i such that there exists a substitution σ to have the following diagram. Observe that $\to^{=}_{A}$ is the transitive and reflexive rewriting relation induced by $\Delta \cup \varepsilon_{=}$.

$$\begin{array}{ccc} l\sigma & \overset{E}{=\!=} & r\sigma \\ {\scriptstyle A}\downarrow & = & \downarrow{\scriptstyle A} \\ q & & q' \end{array}$$

Intuitively, if we have $u \to^{=}_{A} q$, then we know that there exists a term t of $Rep(q)$ such that $t =_E u$. The automaton $\mathtt{W}(A)$ is $\langle \mathcal{F}, Q, Q_f, \Delta \cup \varepsilon_{\mathcal{R}} \cup \varepsilon'_{=}\rangle$, where $\varepsilon'_{=}$ is obtained by adding the transitions $q \to q'$ and $q' \to q$ to $\varepsilon_{=}$, for each pair (q, q'). We also keep $\varepsilon'_{=}$ *closed by transitivity*, but only for pair of distinct states. Roughly, the transitive closure of $\varepsilon'_{=}$ corresponds to propagate explicitly terms that are equivalent by $=_E$. As show in section 6, this aspect is important to refine with accuracy. Note that $\mathtt{W}$ terminates since the number of states of A is finite and the number of transitions to be added to $\varepsilon'_{=}$ is finite.

THEOREM 17. *Assuming that A is well-defined, we have A syntactically included in $\mathtt{W}(A)$, and $\mathtt{W}(A)$ is well-defined.*

EXAMPLE 18. *Consider the $\mathcal{R}_{/E}$-automaton $\mathtt{C}(A^0_{\mathcal{R},E})$ in Example 16. We compute $A^1_{\mathcal{R},E} = \mathtt{W}(\mathtt{C}(A^0_{\mathcal{R},E}))$ using the equation $s(s(x)) = s(x)$. We have $\sigma = \{x \mapsto q_1\}$ and the following diagram. Then, we obtain $A^1_{\mathcal{R},E} = \langle \mathcal{F}, Q^1, Q_f, \Delta^1 \cup \varepsilon^1_{\mathcal{R}} \cup \varepsilon^1_{=}\rangle$, where $\varepsilon^1_{=} = \varepsilon^0_{=} \cup \{q_3 \to q_2,\ q_2 \to q_3\}$ and $\varepsilon^0_{=} = \emptyset$. Observe that $A^1_{\mathcal{R},E}$ is a fixpoint: $\mathtt{C}(A^1_{\mathcal{R},E}) = A^1_{\mathcal{R},E}$.*

$$\begin{array}{ccc} s(s(q_1)) & \overset{E}{=\!=} & s(q_1) \\ {\scriptstyle \mathtt{C}(A^0_{\mathcal{R},E})}\downarrow & = & \downarrow{\scriptstyle \mathtt{C}(A^0_{\mathcal{R},E})} \\ q_3 & & q_2 \end{array}$$

6 Approximation Refinement

Let I be a set of initial terms characterised by the $\mathcal{R}_{/E}-$automaton $A^0_{\mathcal{R},E}$, $\mathcal{R}$ be a TRS, and Bad the set of forbidden terms represented by A_{Bad} a tree automaton. The reachability problem boils down to check $\mathcal{R}^*(I) \cap Bad \stackrel{?}{=} \emptyset$. There are classes of systems for which $\mathcal{R}^*(I)$ is regular and can be computed in a finite amount of time (see appendix F) but, in general, the computation does not terminate. For such cases, our only hope is to work with a *Counterexample-Guided Abstraction Refinement [13]* procedure that computes successive abstractions and successively refine them until the property can be prove correct or not. In the first part of the paper, we have focused on computing the abstraction. We now propose a technique that checks whether a term is indeed reachable from the initial set of terms. If the term is a spurious counterexample, then it has to be eliminated from the approximation. We then generalize the operation to (possibly infinite) sets of terms.

Let $A^k_{\mathcal{R},E} = \langle \mathcal{F}, Q^k, Q_f, \Delta^k \cup \varepsilon^k_{\mathcal{R}} \cup \varepsilon^k_{=} \rangle$ be a $\mathcal{R}_{/E}$-automaton obtained after k steps of completion and widening from $A^0_{\mathcal{R},E}$ and assume that $\mathcal{L}(A^k_{\mathcal{R},E}) \cap Bad \neq \emptyset$. Let t be a term of $\mathcal{L}(A^k_{\mathcal{R},E}) \cap Bad$. Then, we know that there exists a run $t \xrightarrow{\phi}_{A^k_{\mathcal{R},E}} q_f$ with $q_f \in Q_f$. We know that $A^k_{\mathcal{R},E}$ is well-defined by construction. It implies that if $\phi = \top$, we deduce $t \in \mathcal{R}^*(I)$. It means that t is a counter-example, so from a verification point of view, the property is broken as formulated in Section 3. Otherwise, we have that $\phi = \bigwedge_1^n Eq(q_j, q'_j)$, and t is possibly a spurious counterexample. We thus decide to remove it from the approximation. For doing so, we use the *pruning methodology* that was informally introduced in Example 10.

The pruning step $\mathtt{P}$**.** As we have seen in Example 10, we compute the pruned $\mathcal{R}_{/E}$-automaton $\mathtt{P}(A^k_{\mathcal{R},E}, \phi)$ in two steps. The first step consists of removing some transitions of $\varepsilon^k_{=}$ until ϕ does not not hold anymore *i.e.* $\phi = \bot$. Consider the formula ϕ containing the predicate $Eq(q, q')$: we replace this predicate by $\bot$ if we decide to remove the transition $q \to q'$ from $\varepsilon^k_{=}$. The second step consists in propagating the information. Indeed, we also have to remove all transitions $q \xrightarrow{\alpha} q' \in \varepsilon^k_{\mathcal{R}}$, where the conjunction α contains some atoms transitions removed from $\varepsilon^k_{=}$. The procedure is iterated for each possible reduction of t in $A^k_{\mathcal{R},E}$ and thus, we finally get $A^{k+1}_{\mathcal{R},E}$. It is easy to see that, for any ϕ, there exists no run $t \xrightarrow{\phi}_{A^{k+1}_{\mathcal{R},E}} q_f$.

Observe that, when removing t from the abstraction, our procedure may also remove other terms that are generated by ϕ. We now briefly show (see the appendix for details) that the *possibly infinite* set of terms that belong to both the abstraction and the set of bad states can be removed by exploiting the structure of the formula. More precisely, we build a set S containing triples of the form (q, q', ϕ) where q is a state of $A^k_{\mathcal{R},E}$, q' is a state of A_{Bad} and ϕ is a formula on $\varepsilon_=$ transitions of $A^k_{\mathcal{R},E}$. For all triple (q, q', ϕ), the formula ϕ holds if and only if $\mathcal{L}(A^k_{\mathcal{R},E}, q) \cap \mathcal{L}(A_{Bad}, q') \neq \emptyset$. For all triple (q, q', ϕ), where q is final in $A^k_{\mathcal{R},E}$, q' is final in A_{Bad} and $\phi = \top$, then some of the terms recognized by q' in A_{Bad} are reachable. Otherwise, ϕ is the formula to invalidate, i.e. negate some of its atom so that it becomes $\bot$. Starting from ϕ, the refinement process is performed using the technique that presented above.

Example. Consider the $\mathcal{R}_{/E}-$automaton $A^1_{\mathcal{R},E}$ given in Example 18. We define A_{Bad} to be the tree automaton whose final state is q'_0 and whose transitions are $a \to q'_1, s(q'_1) \to$

$q'_2, s(q'_2) \to q'_1$ and $s(q'_2) \to q_0$. The forbidden terms that belong to the language of $\mathcal{L}(A_{Bad})$ are of the form $f(s^{2k+1}(a))$. We observe that $\mathcal{L}(A^1_{\mathcal{R},E}) \cap \mathcal{L}(A_{Bad}) \neq \emptyset$. According to the intersection algorithm sketched above, one can construct a set S of triples (q_0, q'_0, ϕ), where ϕ is the formula used to prune $A^1_{\mathcal{R},E}$ in order to remove those terms that belong to $\mathcal{L}(A^1_{\mathcal{R},E}) \cap \mathcal{L}(A_{Bad})$. Here, $S = \{(q_0, q'_0, Eq(q_2, q_3) \wedge Eq(q_3, q_2)), (q_0, q'_0, Eq(q_2, q_3)), (q_0, q'_0, Eq(q_3, q_2))\}$. We apply the pruning step for each formula ϕ in the set. We compute $\mathtt{P}(A^1_{\mathcal{R},E}, Eq(q_2, q_3) \wedge Eq(q_3, q_2))$. Removing the transition $q_2 \to q_3$ from $\varepsilon^1_=$ is sufficient to invalidate $Eq(q_2, q_3) \wedge Eq(q_3, q_2)$. Moreover, this removing invalidates $(q_0, q'_0, Eq(q_2, q_3))$ too. It remains to prune with $(q_0, q'_0, Eq(q_3, q_2))$. This is done by removing the transition $q_3 \to q_2$ from $\varepsilon^1_=$. At this step, no transition of $\varepsilon^1_{\mathcal{R}}$ can be removed anymore. Indeed, all these transitions are all labeled by $\top$. We thus define $A^2_{\mathcal{R},E} = \mathtt{P}(\mathtt{P}(\mathtt{P}(A^1_{\mathcal{R},E}, Eq(q_2, q_3) \wedge Eq(q_3, q_2)), Eq(q_2, q_3)), Eq(q_3, q_2))$ with $\Delta^2 = \Delta^1$, $\varepsilon^2_{\mathcal{R}} = \varepsilon^1_{\mathcal{R}}$ and $\varepsilon^2_= = \emptyset$. We observe that $A^2_{\mathcal{R},E}$ is not $\mathcal{R}-$closed and should be completed. We thus define $A^3_{\mathcal{R},E} = \mathtt{W}(\mathtt{C}(A^2_{\mathcal{R},E}))$. We found a new critical pair for $f(x) \to f(s(s(x)))$ and we obtain $\Delta^3 = \Delta^2 \cup \{(q_3) \to q_5, s(q_5) \to q_6, f(q_6) \to q_7$, and $\varepsilon^3_{\mathcal{R}} = \varepsilon^2_{\mathcal{R}} \cup \{q_7 \xrightarrow{\top} q_4\}$.

The interesting point concerns the application of $\mathtt{W}$. Observe that the transitions of $\varepsilon^3_=$ directly results from the application of the equation $s(x) = s(s(x))$ *i.e.* transitions $q_2 \to q_3, q_3 \to q_2, q_3 \to q_5, q_5 \to q_3, q_5 \to q_6$, and $q_6 \to q_5$. After the transitive closure of $\varepsilon^3_=$ has been computed, some new transitions are inferred and added to $\varepsilon^3_=$, i.e., $q_2 \to q_5, q_5 \to q_2, q_2 \to q_6, q_6 \to q_2, q_3 \to q_6$ and $q_6 \to q_3$. We again check the emptiness of $\mathcal{L}(A^3_{\mathcal{R},E}) \cap \mathcal{L}(A_{Bad})$. This intersection is still not empty and most of the transitions in $\varepsilon^3_=$ can be removed by the pruning operation. Let $A^4_{\mathcal{R},E}$ be the $\mathcal{R}_{/E}-$automaton obtained by applying $\mathtt{P}$ on $A^3_{\mathcal{R},E}$, it only remains $q_3 \to q_6$ and $q_6 \to q_3$ in $\varepsilon^4_=$. We also observe that no transition of $\varepsilon^3_{\mathcal{R}}$ are removed: $\varepsilon^4_{\mathcal{R}} = \varepsilon^3_{\mathcal{R}}$. In fact, all the transitions in $\varepsilon^4_{\mathcal{R}}$ are labeled by $\top$. Then, we restart the completion process and we observe that $A^4_{\mathcal{R},E} = \mathtt{C}(A^4_{\mathcal{R},E})$. We have thus reached a fixpoint. There, we observe that $\mathcal{L}(A^4_{\mathcal{R},E}) \cap \mathcal{L}(A_{Bad}) = \emptyset$ and conclude that $\mathcal{R}^*(I) \cap Bad = \emptyset$. Observe that our refinement is accurate in this case. Indeed $\mathcal{L}(A^4_{\mathcal{R},E}) = f(s^{2*k}(a))$, that is the exact set of reachable states.

Remarks: The above example cannot be handled with the approach of [4] that also proposes a counterexample guided abstraction technique. Indeed this technique would prove that a property where the bad terms are bounded: $Bad_k = \{f(s^{2*i+1}(a)) \mid i < k\}$ Else the procedure loops, if we consider all the set Bad.

7 Conclusion

We have presented a new CounterExample Guided Abstraction Refinement procedure for the verification of infinite-state systems whose states are represented by trees. Transitions are defined using rewriting rules and abstractions using equations. This work equip the equational abstraction framework [22] with counterexample extraction and automatic refinement. Our next objective is to implement and evaluate our approach. Our technique should be applied to systems that are out of the scope of [4, 7] and should be more efficient as we do avoid intermediary potentially computation intensive determinization and inclusion checks.

A challenge in our implementation will be to develop efficient strategies to refine the abstraction (see the discussion in Section 6). In a second step, we are definitively interested in extending our theory to more complex properties. As an example, one could consider special classes of liveness properties that were introduced in [11] for parametric systems.

References

[1] P. A. Abdulla, A. Legay, A. Rezine, and J. d'Orso. Simulation-based iteration of tree transducers. In *TACAS*, volume 3440 of *LNCS*, pages 30–40. Springer, 2005.

[2] Avispa – a tool for Automated Validation of Internet Security Protocols. `http://www.avispa-project.org`.

[3] F. Baader and T. Nipkow. *Term Rewriting and All That*. Cambridge University Press, 1998.

[4] Y. Boichut, R. Courbis, P.-C. Heam, and O. Kouchnarenko. Finer is better: Abstraction refinement for rewriting approximations. In *RTA*, LNCS 5117, pages 48–62. Springer, 2008.

[5] Y. Boichut, T. Genet, T. Jensen, and L. Leroux. Rewriting Approximations for Fast Prototyping of Static Analyzers. In *RTA*, volume 4533 of *LNCS*, pages 48–62. Springer, 2007.

[6] B. Boigelot, A. Legay, and P. Wolper. Iterating transducers in the large (extended abstract). In *CAV*, LNCS, pages 223–235. Springer, 2003.

[7] A. Bouajjani, P. Habermehl, A. Rogalewicz, and T. Vojnar. Abstract regular tree model checking. *Electronic Notes in Theoretical Computer Science*, 149(1):37–48, 2006.

[8] A. Bouajjani, P. Habermehl, A. Rogalewicz, and T. Vojnar. Abstract regular tree model checking of complex dynamic data structures. In *SAS*, volume 4134 of *LNCS*, pages 52–70. Springer, 2006.

[9] A. Bouajjani, P. Habermehl, and T. Vojnar. Abstract regular model checking. In *CAV*, volume 3114 of *LNCS*, pages 372–386. Springer, 2004.

[10] A. Bouajjani, B. Jonsson, M. Nilsson, and T. Touili. Regular model checking. In *CAV*, volume 1855 of *LNCS*, pages 403–418. Springer-Verlag, 2000.

[11] A. Bouajjani, A. Legay, and P. Wolper. Handling liveness properties in (omega-)regular model checking. In *INFINITY*, volume 138(3) of *ENTCS*. Elsevier, 2004.

[12] B. Boyer, T. Genet, and T. Jensen. Certifying a Tree Automata Completion Checker. In *IJCAR'08*, volume 5195 of *LNCS*. Springer, 2008.

[13] E. M. Clarke. Counterexample-guided abstraction refinement. In *TIME*, page 7. IEEE Computer Society, 2003.

[14] H. Comon, M. Dauchet, R. Gilleron, F. Jacquemard, D. Lugiez, C. Löding, S. Tison, and M. Tommasi. Tree automata techniques and applications. 2008.

[15] G. Feuillade, T. Genet, and V. Viet Triem Tong. Reachability Analysis over Term Rewriting Systems. *Journal of Automated Reasonning*, 33 (3-4):341–383, 2004.

[16] T. Genet. Decidable approximations of sets of descendants and sets of normal forms. In *RTA*, volume 1379 of *LNCS*, pages 151–165. Springer-Verlag, 1998.

[17] T. Genet. Reachability analysis of rewriting for software verification. Université de Rennes 1, 2009. Habilitation.

[18] T. Genet and F. Klay. Rewriting for Cryptographic Protocol Verification. In *Proc. 17th CADE Conf., Pittsburgh (Pen., USA)*, volume 1831 of *LNAI*. Springer-Verlag, 2000.

[19] T. Genet and R. Rusu. Equational tree automata completion. *JSC*, 45:574–597, 2010.

[20] T. Genet, Y.-M. Tang-Talpin, and V. Viet Triem Tong. Verification of Copy Protection Cryptographic Protocol using Approximations of Term Rewriting Systems. In *WITS'2003*, 2003.

[21] R. Gilleron and S. Tison. Regular tree languages and rewrite systems. *Fundamenta Informaticae*, 24:157–175, 1995.

[22] J. Meseguer, M. Palomino, and N. Martí-Oliet. Equational abstractions. *TCS*, 403:239–264, 2008.

[23] T. Takai. A Verification Technique Using Term Rewriting Systems and Abstract Interpretation. In *RTA*, volume 3091 of *LNCS*, pages 119–133. Springer, 2004.

[24] P. Wolper and B. Boigelot. Verifying systems with infinite but regular state spaces. In *CAV*, volume 1427 of *LNCS*, pages 88–97. Springer-Verlag, 1998.

A Proof about semantics

THEOREM 5.

$$\forall t \in \mathcal{T}(\mathcal{F} \cup Q),\ q \in Q,\ t \xrightarrow{\alpha}_A q \iff t \to_A q$$

PROOF. The proof is easily done by induction by arguing that it is enough to forget the formulas manipulated by the definition 6 to have the equivalent step with $\to_A$.

B Proofs about C

PROPERTY 19. *$A^0_{\mathcal{R},E}$ is well-defined.*

PROOF. $A^0_{\mathcal{R},E} = \langle \mathcal{F}, Q^0, Q_f, \Delta^0 \rangle$ fits the definition 9, only if the two items of the definition 9 are verified.

We know that $A^0_{\mathcal{R}}$ has no ε-transitions. All terms are recognized using transitions of Δ^0. It means that for all state q the set of terms is defined as $\{t \in \mathcal{T}(\mathcal{F}) | t \to^*_{\Delta^0} q\}$ which is equal to $Rep(q)$, the set of representatives for q. We also remark that for all term t, $t \to^*_{\Delta_0} q$ implies $t \xrightarrow{\top}_{A^0_{\mathcal{R},E}} q$: the second and third point of the definition 6, are not used, since $\varepsilon^0_{\mathcal{R}}$ and $\varepsilon^0_{=}$ are empty. The first item of definition 9 is ensured: for all state q, and all term $t \xrightarrow{\top} q$, we have $t \in Rep(q)$, and $t \to^*_{\mathcal{R}} t$ by reflexivity.

The second item of 9 holds, since $\varepsilon^0_{\mathcal{R}}$ is empty.

LEMMA 20. *[Solving one critical pair preserves well-definition] Let A and A' be two $\mathcal{R}_{/E}-$automaton such that A' is obtained from A by solving a critical pair $\langle r\sigma, \alpha, q \rangle$ of A. If A is well-defined then so is A'.*

PROOF. Assume that $A = \langle \mathcal{F}, Q, Q_f, \Delta \cup \varepsilon_{\mathcal{R}} \cup \varepsilon_{=} \rangle$ and $A' = \langle \mathcal{F}, Q', Q_f, \Delta' \cup \varepsilon'_{\mathcal{R}} \cup \varepsilon'_{=} \rangle$. According to Definition 12, $\Delta' = \Delta \cup \texttt{Norm}(r\sigma, \Delta \setminus \Delta^0)$, $\varepsilon'_{\mathcal{R}} = \{q' \xrightarrow{\alpha} q\} \cup \varepsilon_{\mathcal{R}}$ and $\varepsilon'_{=} = \varepsilon_{=}$. Following Definition 9, we first show in (1) that for all state q'' of A', and all term v such that $v \xrightarrow{\top}_{A'} q''$, there exists u a term representative of q'' such that $u \to^*_{\mathcal{R}} v$. Then, in (2) we show that if $q_1 \xrightarrow{\phi} q_2$ is a transition of $\varepsilon'_{\mathcal{R}}$, then there exist terms $s, t \in \mathcal{T}(\mathcal{F})$ such that $s \xrightarrow{\phi}_{A'} q_1$, $t \xrightarrow{\top}_{A'} q_2$ and $t \to_{\mathcal{R}} s$.

1. We prove the property by induction on the height of t. Let us assume that for all term t' of height lesser than the height of t and for all $q \in Q_{A'}$, we have $t' \xrightarrow{\top}_{A'} q \implies \exists u \in Rep(q) : u \to^*_{\mathcal{R}} t'$. Now let us prove that this is true for t. We prove it by case on $q \in Q_A$ and $t \xrightarrow{\top}_A q$:
   - If $q \in Q_A$ and $t \xrightarrow{\top}_A q$ then since A is well defined, we get the representative $u \in Rep(q)$ such that $u \to^*_{\mathcal{R}} t$ from well-definition of A.
   - If $q \in Q_A$, $t \not\xrightarrow{\top}_A q$ and $t \xrightarrow{\top}_{A'} q$. We prove the property by induction on the height of t. Now let us consider the term t. Since t is recognized in A' and not in A, this means that the run $t' \xrightarrow{\top}_{A'} q$ needs the transitions added by the resolution of a critical pair. Hence there exists a rewrite rule $l \to r$, a substitution $\sigma : \mathcal{X} \mapsto Q_A$, a formula α and a state q_c such that $l\sigma \xrightarrow{\alpha}_A q_c$ and $\langle r\sigma, \alpha, q_c \rangle$ is the critical pair. Moreover, the resolution of this critical pair creates $\Delta_{A'} = \texttt{Norm}(r\sigma, \Delta_A \setminus \Delta_0)$ and $\varepsilon^{A'}_{\mathcal{R}} = \varepsilon^{A}_{\mathcal{R}} \cup \{q'_c \to q_c\}$ such that $r\sigma \to^!_{\Delta' \setminus \Delta_0} q'_c$. Recall that $t' \xrightarrow{\top}_{A'} q$ needs transitions not occurring in A. However, all the *new* transitions produced by $\texttt{Norm}(r\sigma, \Delta_A \setminus \Delta_0)$ necessarily range on *new* states, i.e. states not occurring in Q_A. As a result, those transitions cannot be used to get $t' \xrightarrow{\top}_{A'} q$ with $q \in Q_A$. This means that the run $t \xrightarrow{\top}_{A'} q$ uses at least once $q'_c \to q_c$ and $\alpha = \top$ since the whole run is labelled by $\top$. To sum up, we know that there exists

a ground context $C[\,]$ such that $t = C[t'] \xrightarrow{\top}_{A'} C[q'_c] \xrightarrow{\top}_{A'} C[q_c] \xrightarrow{\top}_{A} q$. Note that if $q'_c \to q_c$ the same reasonning can be applied. We start to reason on the occurrence of $q'_c \to q_c$ that is the closest to q. Now, our objective is to show that there exists $u \in Rep(q'_c)$ such that $u \to^*_{\mathcal{R}} t'$. If $t' \xrightarrow{\top}_{A} q'_c$ then since A is well defined we directly have the result using definition 9. Otherwise this means that q'_c is new for A (i.e. $q'_c \notin Q_A$) and has been added by the resolution of the critical pair, i.e. $r\sigma \to^{!}_{\Delta'} q'_c$. Because of Theorem 22, we get that there exists a substitution $\sigma' : \mathcal{X} \mapsto \mathcal{T}(\mathcal{F})$ such that $t' = r\sigma'$. Using the same theorem, from $t' = r\sigma' \xrightarrow{\top}_{A'} q'_c$ and $r\sigma \xrightarrow{\top}_{A'} q'_c$, we get that for all variable x of r: $\sigma'(x) \xrightarrow{\top}_{A'} \sigma(x)$. Note that $\sigma(x) \in Q_A$ and that $\sigma'(x)$ are necessarily terms of height lesser to the height of t. Using the induction hypothesis, we get that for all state $\sigma(x)$ there exists a representative u_x such that $u_x \to^*_{\mathcal{R}} \sigma'(x)$. Let σ_{Rep} be the substitution mapping every variable x to u_x. We have $r\sigma_{Rep} \in Rep(q'_c)$. Moreover, $r\sigma_{Rep} \to^*_{\mathcal{R}} r\sigma' = t'$. Now, our objective is to show that $l\sigma_{Rep} \to_{\mathcal{R}} r\sigma_{Rep}$. This is not straightforward since $\mathcal{V}ar(l) \supseteq \mathcal{V}ar(r)$. However, it is possible to extend σ_{Rep} into σ'_{Rep} where every variable y of $\mathcal{V}ar(l)$ not occurring in σ_{Rep} is mapped to a representative of $\sigma(y)$. Hence, $l\sigma'_{Rep} \to_{\mathcal{R}} r\sigma'_{Rep} \to^*_{\mathcal{R}} t'$. From the critical pair we know that $l\sigma \xrightarrow{\alpha}_{A} q_c$ and we found that $\alpha = \top$. Hence $l\sigma'_{Rep} \xrightarrow{\top}_{A} q_c$. Since A is well-defined, we get that there is a representative $v \in Rep(q_c)$ such that $v \to^*_{\mathcal{R}} l\sigma'_{Rep}$. By transitivity of $\to_{\mathcal{R}}$, we get that $v \to^*_{\mathcal{R}} t'$. Above, we found that $t = C[t'] \xrightarrow{\top}_{A'} C[q'_c] \xrightarrow{\top}_{A'} C[q_c] \xrightarrow{\top}_{A} q$. From this and $v \in Rep(q_c)$, we get that $C[v] \xrightarrow{\top}_{A} q$. Since A is well defined, we know that there exists a representative $w \in Rep(q)$ such that $w \to^*_{\mathcal{R}} C[v]$. To conclude, we found $w \in Rep(q)$ and $w \to^*_{\mathcal{R}} C[v] \to C[t'] = t$.

- If $q \notin Q_A$ ($q \in Q'_A \setminus Q_A$), $t \not\xrightarrow{\top}_{A} q$ and $t \xrightarrow{\top}_{A'} q$. Since $q \in Q'_A \setminus Q_A$, we know that q has been added by the resolution of a critical pair. As above, we can deduce that there exists a rewrite rule $l \to r$, a substitution $\sigma : \mathcal{X} \mapsto Q_A$, a formula α and a state q_c such that $l\sigma \xrightarrow{\alpha}_{A} q_c$ and $\langle r\sigma, \alpha, q_c \rangle$ is the critical pair. Moreover, the resolution of this critical pair creates $\Delta_{A'} = \texttt{Norm}(r\sigma, \Delta_A \setminus \Delta_0)$ and $\varepsilon^{A'}_{\mathcal{R}} = \varepsilon^{A}_{\mathcal{R}} \cup \{q'_c \to q_c\}$ such that $r\sigma \to^{!}_{\Delta' \setminus \Delta_0} q'_c$. Since q is a new state of A', q has been necessarily used for the normalization of a subterm of $r\sigma$. More precisely, we know that there exists a term $s \in \mathcal{T}(\mathcal{F}, \mathcal{X})$ and a context $C[\,]$ (possibly empty) such that $r\sigma = C[s]$, $C[s]\sigma \to^*_{\Delta'} q'_c$ and $s\sigma \to^*_{\Delta'} q$. Similarly, we know that there exists a substitution $\sigma' : \mathcal{X} \mapsto \mathcal{T}(\mathcal{F})$ such that $s\sigma' = t$. We get that for every variable x of r: $\sigma'(x) \xrightarrow{\top}_{A'} \sigma(x)$. Note that $\sigma(x) \in Q_A$ and that $\sigma'(x)$ are necessarily terms of height lesser to the height of t. Using the induction hypothesis, we get that for every state $\sigma(x)$ there exists a representative u_x such that $u_x \to^*_{\mathcal{R}} \sigma'(x)$. Let σ_{Rep} be the substitution mapping every variable x to u_x. We have $s\sigma_{Rep} \in Rep(q)$ and $s\sigma_{Rep} \to^*_{\mathcal{R}} s\sigma' = t$.

2. Easily, for any transitions $q_1 \xrightarrow{\phi} q_2 \in \varepsilon_{\mathcal{R}}$, the property still holds. Let us focus now on the transition $q' \xrightarrow{\alpha} q$ resulting from the resolving of the critical pair $\langle r\sigma, \alpha, q \rangle$. By definition, the critical pair $\langle r\sigma, \alpha, q \rangle$ results from the application of the matching algorithm of Definition 13. So there exists a rule $l \to r \in \mathcal{R}$ such that $(\alpha, \sigma) \in S$ with $l \trianglelefteq q \vdash_A S$. Moreover, since the critical pair has to be solved: $l\sigma \xrightarrow{\alpha} q$ and there is no formula α' such that $r\sigma \xrightarrow{\alpha'}_{A} q$. Since $\mathcal{R}$ is left-linear, for each variable $x \in \mathcal{V}ar(l)$, one can define the substitution $\sigma' : \mathcal{X} \to \mathcal{T}(\mathcal{F})$ as follows: Assuming q_s being the state of A such that $\sigma(x) = q_s$, let $\sigma'(x) = Rep(q_s)$. By definition of Rep, $Rep(q_s) \xrightarrow{\top} q_s$. So, there exists a derivation such that $l\sigma' \xrightarrow{\top} l\sigma$ and $l\sigma \xrightarrow{\alpha} q$. One can deduce that $r\sigma' \xrightarrow{\top} r\sigma$. According to Lemma 22, one can deduce that there exists a unique q'

such that $r\sigma \to^*_{\mathtt{Norm}(r\sigma,\Delta\setminus\Delta^0)} q'$. If $\mathtt{Norm}(r\sigma, \Delta \setminus \Delta^0) \neq \emptyset$ then each transition composing it is of the form $f(q'_1, \ldots, q'_n) \to q'_{n+1}$. Consequently, $r\sigma \xrightarrow{\top} q'$. Considering the transition $q' \xrightarrow{\alpha} q$, one has $r\sigma' \xrightarrow{\top} r\sigma \xrightarrow{\top} q' \xrightarrow{\alpha} q$. Finally, assuming $s = l\sigma'$ and $t = r\sigma'$, there exists $s, t \in \mathcal{T}(\mathcal{F})$ such that one has $s \xrightarrow{\alpha} q$, $t \xrightarrow{\alpha} q'$ and $s \to_{\mathcal{R}} t$.

To conclude, A' is also well-defined.

THEOREM 10. *If $A^i_{\mathcal{R},E}$ is well-defined then so is $\mathtt{C}(A^i_{\mathcal{R},E})$, and $\forall q \in Q^i$, $\forall t \in \mathcal{L}(A^i_{\mathcal{R},E}, q)$, $\forall t \in \mathcal{T}(\mathcal{F})$, $t \to_{\mathcal{R}} t' \implies t' \in \mathcal{L}(\mathtt{C}(A^i_{\mathcal{R},E}), q)$.*

PROOF.

Let CP be the finite set of critical pairs computed from $A^i_{\mathcal{R},E}$ to solve. Assume that $CP = \{\langle r1\sigma_1, \alpha_1, q_1\rangle, \ldots, \langle r1\sigma_m, \alpha_1, q_m\rangle\}$. By definition, considering $A_0 = A^i_{\mathcal{R},E}$ there exists a sequence of $\mathcal{R}_{/E}-$automata $A_1, \ldots, A_m$ where A_j is obtained from A_{j-1} by solving the critical pair $\langle r_j\sigma_j, \alpha_j, q_j\rangle$ according Definition 12. Thus, $\mathtt{C}(A^i_{\mathcal{R},E}) = A_n$. For a question of readability and in order to prevent any confusion between notations, each $\mathcal{R}_{/E}-$automaton A_j is defined as follows: $A_j = \langle \mathcal{F}, Q^{n+1}, Q_f, \Delta'^j \cup \varepsilon'^j_{\mathcal{R}} \cup \varepsilon'^j_{=}\rangle$.

First, let us show that $\mathtt{C}(A^i_{\mathcal{R},E})$ is well-defined if $A^i_{\mathcal{R},E}$ is well-defined.

$\mathtt{C}(A^i_{\mathcal{R},E})$ is well-defined: Let P_n be the following proposition: A_n is well-defined.

- P_0: Trivial since $A_0 = A^i_{\mathcal{R},E}$ and $A^O_{\mathcal{R},E}$ is well-defined by hypothesis.
- $P_n \Rightarrow P_{n+1}$: By hypothesis, A_{n+1} is obtained from A_n by solving the critical pair $\langle r_{n+1}\sigma_{n+1}, \alpha_{n+1}, q_{n+1}\rangle$. Applying Lemma 20, one obtains automatically that A_{n+1} is well-defined.

So, one can deduce that $\mathtt{C}(A^i_{\mathcal{R},E})$ is well-defined.

$\mathtt{C}(A^i_{\mathcal{R},E})$ covers terms accessible in one rewrite step from terms of $A^i_{\mathcal{R},E}$: Let q be a state of $A^i_{\mathcal{R},E}$ and t be a term of $\mathcal{L}(A^i_{\mathcal{R},E}, q)$. Suppose there exist a position $p \in \mathcal{P}os(t)$, a rule $l \to r \in \mathcal{R}$ and a substitution $\sigma' : \mathcal{X} \to \mathcal{T}(\mathcal{F})$ such that $t|_p = l\sigma'$. Let t' be the term such that $t' = t[r\sigma']_p$. Since $t \in \mathcal{L}(A^i_{\mathcal{R},E}, q)$, there exists a state q' of $A^i_{\mathcal{R},E}$ such that $t|_p = l\sigma' \to^*_{A^i_{\mathcal{R},E}} q'$ and $t[q']_p \to^*_{A^i_{\mathcal{R},E}} q$. Following Property 14, there exists $(\alpha, \sigma) \in S$ with $l \trianglelefteq q' \vdash_{A^i_{\mathcal{R},E}} S$ such that $l\sigma' \to^*_{A^i_{\mathcal{R},E}} l\sigma$ and $l\sigma \to^*_{A^i_{\mathcal{R},E}} q'$. If $\langle r\sigma, \alpha, q'\rangle$ is already solved then $r\sigma \to^*_{A^i_{\mathcal{R},E}} q'$. Consequently, $r\sigma'$ can also be reduced to q' in $A^i_{\mathcal{R},E}$. Since $t' = t[r\sigma']_p \to^*_{A^i_{\mathcal{R},E}} q$, $t' \in \mathcal{L}(\mathtt{C}(A^i_{\mathcal{R},E}), q)$. Suppose now that $r\sigma \not\to^*_{A^i_{\mathcal{R},E}} q'$. So, there exists $\langle r_i\sigma_i, \alpha_i, q_i\rangle \in CP$ such that $\langle r_i\sigma_i, \alpha_i, q_i\rangle = \langle r\sigma, \alpha, q'\rangle$. By construction, $r\sigma \to^*_{A_i} q'$. Consequently, $r\sigma'$ can also be reduced to q' in A_i. Since A_i is syntactically included in $\mathtt{C}(A^i_{\mathcal{R},E})$, one can deduce that $t' = t[r\sigma']_p \to^*_{C(A^i_{\mathcal{R},E})} q$. Concluding the proof.

C Proofs about W

LEMMA 19. *[W preserves well-definition] Let A be a $\mathcal{R}_{/E}$-automaton. If A is well-defined, then so is $\mathtt{W}(A)$.*

PROOF. Assume that $A = \langle \mathcal{F}, Q, Q_f, \Delta \cup \varepsilon_{\mathcal{R}} \cup \varepsilon_{=}\rangle$ is well-defined. We have $\mathtt{W}(A) = \langle \mathcal{F}, Q, Q_f, \Delta \cup \varepsilon_{\mathcal{R}} \cup \varepsilon'_{=}\rangle$, where $\varepsilon_{\mathcal{R}} \supseteq \varepsilon'_{\mathcal{R}}$, since W only adds transitions to the $\varepsilon_{\mathcal{R}}$. We have to prove the two items of definition 9.

- The transitions of $\varepsilon'_{=}$ are never used for a run $\xrightarrow{\alpha}$ where $\alpha = \top$, thanks to the second point of the definition 6. It means that for all term t and all state q, $t \xrightarrow{\top}_{W(A)} q$ is equivalent to $t \xrightarrow{\top}_{A} q$. Since A is well-defined, we know that there exists $u \in Rep(q)$ such that $u \to^*_{\mathcal{R}} t$. u is also a representative of $\mathtt{W}(A)$, and we deduce that first point of the definition 9 holds for $\mathtt{W}(A)$.
- Function $\mathtt{W}$ only adds transitions to $\varepsilon'_{=}$ and do not remove transitions of A. For all transitions $q \xrightarrow{\alpha} q' \in \varepsilon'_{\mathcal{R}}$ we have $q \xrightarrow{\alpha} q' \in Drw$. Since A is well-defined, we know that there exist terms $s, t \in \mathcal{T}(\mathcal{F})$ such that $s \xrightarrow{\phi}_{A} q$, $t \xrightarrow{\top}_{A} q'$ and $t \to_{\mathcal{R}} s$. We also have $s \xrightarrow{\phi}_{\mathtt{W}(A)} q$, $t \xrightarrow{\top}_{\mathtt{W}(A)} q'$ and $t \to_{\mathcal{R}} s$.

LEMMA 20. *For all $\mathcal{R}_{/E}$-automaton A, $\mathcal{L}(\mathtt{W}(A)) \supseteq \mathcal{L}(A)$.*

PROOF. This is easy to see since widening only adds transitions, and thus, does not restrict the recognized language.

D Proofs about matching

THEOREM 21. *[Matching Algorithm is complete] Let A be a $\mathcal{R}_{/E}-$automaton, q one of its states, $l \in \mathcal{T}(\mathcal{F}, \mathcal{X})$ the linear left member of a rewriting rule and σ a Q-substitution with a domain range-restricted to $\mathcal{V}(l)$. If the set S is solution of the matching problem $l\sigma \trianglelefteq q$, then we have $\forall(\alpha, \sigma)$, $l\sigma \xrightarrow{\alpha}_{A} q \iff (\alpha, \sigma) \in S$*

PROOF. Assuming $\mathcal{F}$ a set of symbols, $\mathcal{X}$ a set of variable and Q a set of states. We define $A = \langle \mathcal{F}, Q, Q_f, \Delta \cup \varepsilon_{\mathcal{R}} \cup \varepsilon_{=} \rangle$; $l \in \mathcal{T}(\mathcal{F}, \mathcal{X})$ 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 $f(t_1, \ldots, t_n)$.

We are going to decompose $f(t_1, \ldots, t_n)\sigma \xrightarrow{\alpha}_{A} q$ into sequences of transitions. First observe that, by splitting σ into $\sigma_1 \ldots \sigma_n$, we have that $f(t_1, \ldots, t_n)\sigma$ is equal to $f(t_1\sigma_1, \ldots, t_n\sigma_n)$. Assume $\sigma = \sigma_1 \sqcup \cdots \sqcup \sigma_n$ 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 $f(t_1\sigma_1, \ldots, t_n\sigma_n) \xrightarrow{\alpha}_{A} q$ to show that transitions of A used to recognized the term $f(t_1\sigma_1, \ldots, t_n\sigma_n)$ are considered by the corresponding steps of the matching algorithm.

We observe that the term $f(t_1\sigma_1, \ldots, t_n\sigma_n)$ is recognized in state q. Indeed, we have $f(q_1, \ldots, q_n) \to q' \in \Delta$, 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:

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

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

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

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

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

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 $f(q_1, \ldots, q_n) \to q' \in \Delta$. From this, we obtain a set $S' = \bigotimes_1^n S_i$. By unfolding the definition of $\otimes$, 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.

E Proofs about normalization

To normalize, we assume that Δ, the second argument of `Norm`, is determinist. It means that if Δ contains two normalized transitions of the form $f(q_1, \ldots, q_n) \to q$ and $f(q_1, \ldots, q_n) \to q'$, then we have $q = q'$. It ensures that there exists a normal form for any term which is rewritten by Δ. It is required by the first step of `Norm`.

Note that all proofs about `Norm` are done using the measure $\mu : \mathcal{T}(\mathcal{F} \cup Q) \to \mathbb{N}$ that counts the number of occurences of symbols in $\mathcal{F}$ of 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 $\mu(f(t_1, \ldots, t_n)) = 1 + \sum_1^n \mu(t_i)$.

LEMMA 22. *[Existence of a representative] Assume that $A_{\mathcal{R},E}$ is a $\mathcal{R}_{/E}$-automaton obtained after k steps of completion from $A^0_{\mathcal{R},E}$. Let c be a configuration. If $\Delta' = \mathtt{Norm}(c, \Delta \setminus \Delta^0)$, then there exists a state q such that $c \to^!_{\Delta'} q$.*

PROOF.
Assuming $\mathcal{F}$ a set of symbols, and Q a set of states. We define $A_{\mathcal{R},E} = \langle \mathcal{F}, Q, Q_f, \Delta \cup \varepsilon_{\mathcal{R}} \cup \varepsilon_= \rangle$; $c \in \mathcal{T}(\mathcal{F} \cup Q)$. Assume that $\Delta^1 = \Delta \setminus \Delta^0$ is determinist.

The first step $\mathtt{Norm}(t, \Delta^1)$ consists in rewriting c by Δ^1 in its normal form d

The second step $\mathtt{Slice}(d, \Delta^1)$ returns Δ^2 such that there exists a unique state q such that $d \to^!_{\Delta^2} q$.

The proof is one by induction on the decreasing of $\mu(d)$. We consider the 3 cases of $\mathtt{Slice}(d, \Delta^1)$

1. $\mathtt{Slice}(q, \Delta^1) = \Delta^1$. It means that d is the state q. There exists a unique state ,which is q, such that $d \to^!_{\Delta^1} q$.
2. $\mathtt{Slice}(f(q_1, \ldots, q_n), \Delta^1) = \Delta^1 \cup \{f(q_1, \ldots, q_n) \to q \mid q \in Q_{new}\}$. 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^1 preserves determinism. We know that it is impossible to rewrite more $d = f(q_1, \ldots, q_n)$ using transitions of Δ^1 : the new transition $f(q_1, \ldots, q_n) \to q$ is the unique way to rewrite d. We deduce that $\Delta^2 = \Delta^1 \cup \{f(q_1, \ldots, q_n) \to q \mid q \in Q_{new}\}$ is deterministic, and $d \to^!_{\Delta^2} q$.

3. $\texttt{Slice}(f(t_1,\ldots,t_n),\Delta^1) = \texttt{Norm}(f(t_1,\ldots,t_n),\texttt{Slice}(t_i,\Delta^1)\ )$, $t_i \in \mathcal{T}(\mathcal{F}\cup Q)\setminus Q$. 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 $\texttt{Slice}(t_i,\Delta^1)$ to obtain Δ^2 for which there exists a state q such that $t_i \to^!_{\Delta^2} q$. Using this new set Δ^2, we unfold $\texttt{Norm}(f(t_1,\ldots,t_n),\Delta^2)$ which consists in rewriting $f(t_1,\ldots,t_n)$ using Δ^2. 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^2. Note that if some subterms of t_i are also subterms of some other t_j, it will also be rewritten by Δ^2 in t'_j until we reach the normal form. Each step of rewriting by Δ^2 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 $\mu(t_j) \geq \mu(t'_j)$ from $t_j \to^!_{\Delta^2} t'_j$ using Δ^2. 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^2. Then, we use again the induction hypothesis to deduce that $\Delta' = \texttt{Slice}(f(t'_1,\ldots,t'_n),\Delta^2)$ extends Δ^2 in order to have a unique state q such that $f(t'_1,\ldots,t'_n) \to^!_{\Delta^3} q$. By transivity, we have $d \to^!_{\Delta'} q$ using the deterministic set Δ' for d which is equal to $f(t_1,\ldots,t_n)$.

Finally, we proved that $\Delta' = \texttt{Slice}(d,\Delta^1)$ extends Δ^1 preserving its determinism such that there exists a state q for which $d \to^!_{\Delta'} q$. We also know that $c \to^!_{\Delta'} d$. We can conclude that $\Delta' = \texttt{Norm}(c,\Delta^1)$ is determinist, and there exists a state q such that $c \to^!_{\Delta'} q$.

Let $A = \langle \mathcal{F}, Q, Q_f, \Delta\cup\varepsilon_{\mathcal{R}}\cup\varepsilon_{=}\rangle$ be a $\mathcal{R}_{/E}$-automaton, and $c \in \mathcal{T}(\mathcal{F}\cup Q)$ a configuration of A. We prove that the $\Delta^1 = \Delta\setminus\Delta^0$ is injective *i.e.* for all $c,\ d \in \mathcal{T}(\mathcal{F}\cup Q)$, if we have $c \to^*_{\Delta^1} q$ and $c \to^*_{\Delta^1} q$, then $c = d$. From this property, we deduce that if we have $\Delta^2 = \texttt{Norm}(r\sigma,\Delta^1)$ all term t such that $t \to^*_{\Delta^2\cup\Delta} q$, then we deduce that $t = r\sigma'$ where $\sigma' : \mathcal{X} \to \mathcal{T}(\mathcal{F})$. It is important to ensure that there is no more term added than terms defined as t, when a critical pair is solved. added

LEMMA 23. *[Normalisation and injectivity] If Δ^1 is injective, then so is $\texttt{Norm}(c \mid \Delta^1)$.*

PROOF. Assuming $\mathcal{F}$ a set of symbols, and Q a set of states. We define: $A = \langle \mathcal{F}, Q, Q_f, \Delta\cup\varepsilon_{\mathcal{R}}\cup\varepsilon_{=}\rangle$; $c \in \mathcal{T}(\mathcal{F}\cup Q)$, and $\Delta^1 = \Delta\setminus\Delta^0$.

The injectivity of $\Delta^2 = \texttt{Norm}(c,\Delta^1)$ holds, if there there is only one transition per state in Δ^1. When function `Slice` creates new transition, it uses a new state. Assuming that Δ^1 has only one transition per state, so has Δ^2. This is enougth to ensure the injectivity of Δ^2. Note that all initial $\mathcal{R}_{/E}$-automaton $A^0_{\mathcal{R},E}$ ensures this property, since the set of transitions used by function `Norm` is empty : $\Delta^0\setminus\Delta^0$.

F Using $\mathcal{R}_{/E}$-automata completion for exact computation of reachable terms

The $\mathcal{R}_{/E}$-completion can be used for regular model-checking for most of the known classes of $\mathcal{R}$ for which $\mathcal{R}^*(\mathcal{L}(A))$ is regular. On those classes, completion always stops on a $\mathcal{R}_{/E}$-automaton $A^*_{\mathcal{R}}$ with an empty set $\varepsilon_{=}$. As a result, all the terms recognized by $A^*_{\mathcal{R}}$ are reachable and all triples $(q,q',\phi) \in S$ are such that $\phi = \top$.

THEOREM 24. *[Exact computation with completion] If $E = \emptyset$ and $\mathcal{R}$ is ground [27, 25], right-linear and monadic [31], linear and semi-monadic [26], linear and inversely growing [29], constructor [30] or linear generalized finite path overlapping [23], 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 $\varepsilon_{=}$ set and, thus, every transition of $\varepsilon_{\mathcal{R}}$ is labelled by $\top$. In other words, in this case, $\mathcal{R}_{/E}$-completion produces a usual tree automaton instead of a $\mathcal{R}_{/E}$-automaton. As a result, when $E = \emptyset$, the algorithm

proposed in this paper totally coincides with the one of [19] dealing with tree automata. In [17], it has been shown that the algorithm of [19] terminates with $E = \emptyset$ for the above classes (Theorem 114). Furthermore, Theorem 45 and Theorem 49 of [19] guarantee that, in this case, $A_{\mathcal{R},\emptyset}$ is such that $\mathcal{L}(A^*_{\mathcal{R},\emptyset}) = \mathcal{R}^*(\mathcal{L}(A))$.

Additional References

[25] W. S. Brainerd. Tree generating regular systems. *Information and Control*, 14:217–231, 1969.

[26] J. Coquidé, M. Dauchet, R. Gilleron, and S. Vágvölgyi. Bottom-up tree pushdown automata and rewrite systems. In R. V. Book, editor, *Proc. 4th RTA Conf., Como (Italy)*, volume 488 of *LNCS*, pages 287–298. Springer-Verlag, 1991.

[27] M. Dauchet and S. Tison. The theory of ground rewrite systems is decidable. In *Proc. 5th LICS Symp., Philadelphia (Pa., USA)*, pages 242–248, June 1990.

[28] R. Gilleron and S. Tison. Regular tree languages and rewrite systems. *Fundamenta Informaticae*, 24:157–175, 1995.

[29] F. Jacquemard. Decidable approximations of term rewriting systems. In H. Ganzinger, editor, *Proc. 7th RTA Conf., New Brunswick (New Jersey, USA)*, pages 362–376. Springer-Verlag, 1996.

[30] P. Réty. Regular Sets of Descendants for Constructor-based Rewrite Systems. In *Proc. 6th LPAR Conf., Tbilisi (Georgia)*, volume 1705 of *LNAI*. Springer-Verlag, 1999.

[31] K. Salomaa. Deterministic Tree Pushdown Automata and Monadic Tree Rewriting Systems. *J. of Computer and System Sciences*, 37:367–394, 1988.

G Algorithms and proofs for emptiness decision of intersection

We define a specific algorithm building the set S of reachable states for the product automaton for $\mathcal{R}_{/E}$-automaton A and automaton B where each product state is labelled by a formula on states of A. We first define an order $>$ on formulas.

DEFINITION 25. *Given ϕ_1 and ϕ_2 two formulas, $\phi_1 > \phi_2$ iff $\phi_2 \models \phi_1$ and $\phi_1 \not\models \phi_2$.*

DEFINITION 26. *[Reachable states of the product of a $\mathcal{R}_{/E}$-automaton and a tree automaton] Let $A = \langle \mathcal{F}, Q^A, Q^A_f, \Delta^A, \varepsilon_{\mathcal{R}}, \varepsilon_= \rangle$ be a $\mathcal{R}_{/E}$-automaton and $B = \langle \mathcal{F}, Q^B, Q^B_f, \Delta^B \rangle$ be an epsilon-free tree automaton. The set S of reachable states of $A \times B$ is the set of triples (q, q', ϕ) where $q \in Q^A, q' \in Q^B$ and ϕ is a formula. Starting from the set $Q^A \times Q^B \times \{\bot\}$, the value of S can be computed using the following two deduction rules :*

$$\frac{\{(q_1, q'_1, \phi_1), \ldots, (q_n, q'_n, \phi_n)\} \cup \{(q, q', \phi)\} \cup P}{\{(q_1, q'_1, \phi_1), \ldots, (q_n, q'_n, \phi_n)\} \cup \{(q, q', \phi \vee \bigwedge_{i=1}^n \phi_i)\} \cup P}$$

if $f(q_1, \ldots, q_n) \to q \in \Delta^A$ *and* $f(q'_1, \ldots, q'_n) \to q' \in \Delta^B$ *and* $(\phi \vee \bigwedge_{i=1}^n \phi_i) > \phi$

$$\frac{\{(q_1, q, \phi_1), (q_2, q, \phi_2)\} \cup P}{\{(q_1, q, \phi_1), (q_2, q, (\phi_1 \wedge \phi) \vee \phi_2)\} \cup P}$$

if $q_1 \xrightarrow{\phi} q_2 \in \varepsilon_{\mathcal{R}}$ *and* $((\phi_1 \wedge \phi) \vee \phi_2) > \phi_2$ *or* *if* $q_1 \to q_2 \in \varepsilon_=$ *and* $\phi = Eq(q_1, q_2)$ *and* $((\phi_1 \wedge \phi) \vee \phi_2) > \phi_2$

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 triple $(q, q', \phi) \in S$ with q final in A and q' final in B, if $\phi \models \top$ then some of the terms recognized by q' in B are reachable. Otherwise, ϕ is the formula to invalidate, i.e. negate some of its atom so that it becomes $\bot$.

LEMMA 27.*[Emptiness decision of the product of a $\mathcal{R}_{/E}$-automaton and a tree automaton] Let A be a $\mathcal{R}_{/E}$-automaton and B a tree automaton. Let S be the set of reachable states of $A \times B$ defined according to definition 26. For all final state q of A, all final state q' of B, all formulas $\phi_S \neq \bot$, $\phi \neq \bot$ and all term $t \in \mathcal{T}(\mathcal{F})$, we have $t \xrightarrow{\phi}{}^*_A q$ and $t \to^*_B q'$ (i.e. $\mathcal{L}(A) \cap \mathcal{L}(B) \neq \emptyset$) if and only if there exists a triple $(q, q', \phi_S) \in S$ such that $\phi \models \phi_S$.*

PROOF. Let $A = \langle \mathcal{F}, Q^A, Q^A_f, \Delta^A, \varepsilon_{\mathcal{R}}, \varepsilon_= \rangle$ be the $\mathcal{R}_{/E}$-automaton and $B = \langle \mathcal{F}, Q^B, Q^B_f, \Delta^B \rangle$ be the tree automaton. We prove a stronger property on all states q of A and q' of B (and not only for final states). First, we prove the 'only if' part. Let us assume that there exists a term $t \in \mathcal{T}(\mathcal{F})$ such that $t \xrightarrow{\phi}{}^*_A q, t \to^*_B q'$. By induction on the height of t we have:

- If t is a constant, since B is an epsilon-free tree automaton, the only way to have $t \to^*_B q'$ is to have $t \to q' \in B$. With regards to A, by definition 6, $t \xrightarrow{\phi}{}^*_A q$ means that there exists states $q_0, q_1, \ldots, q_n$ and formulas $\phi_1, \ldots, \phi_n$ such that $t \to_{\Delta_A} q_0 \xrightarrow{\phi_1} q_1 \xrightarrow{\phi_2} \ldots q_n$ with $q = q_n$ and $\phi = \phi_1 \wedge \ldots \wedge \phi_n$.Transitions $q_i \xrightarrow{\phi_i} q_{i+1}$ are either transitions of $\varepsilon_{\mathcal{R}}$ or transitions of $\varepsilon_=$ with $\phi_i = \top$. Because of transitions $t \to q_0 \in \Delta_A$ and $t \to q' \in \Delta_B$, using the first case of definition 26, we get that $(q_0, q', \top) \in S$. Similarly, using the second case of the definition, we obtain that there exists formulas ϕ'_i with $i = 1 \ldots n$ such that $(q_1, q', \phi_1 \vee \phi'_1), (q_2, q', (\phi_1 \wedge \phi_2) \vee \phi'_2), \ldots (q_n, q', (\phi_1 \wedge \ldots \wedge \phi_n) \vee \phi'_n)$ belong to S. Finally, since $q_n = q$ and $\phi = \phi_1 \wedge \ldots \wedge \phi_n$, we that $(q, q', \phi \vee \phi'_n) \in S$. Furthermore, we trivially have that $\phi_S = \phi \vee \phi'_n$ and $\phi \models \phi_S$.
- Assume that for all term of height lesser or equal to $n \in \mathbb{N}$, the property is true. Let us prove that it is also true for a term $f(t_1, \ldots, t_n)$ with $t_1, \ldots, t_n$ of height lesser or equal to n. Since $f(t_1, \ldots, t_n) \to^*_B q'$ and B is an epsilon free tree automaton, we obtain that $\exists q'_1, \ldots, q'_n \in Q^B$ such that $\forall i = 1 \ldots n : t_i \to^*_B q'_i$ and $f(q'_1, \ldots, q'_n) \to q' \in \Delta_B$. With regards to A, by definition 6, $f(t_1, \ldots, t_n) \xrightarrow{\phi}{}^*_A q$ means that there exists states $q_0, q_1, \ldots, q_m, q''_1, \ldots, q''_n$ and formulas $\phi_1, \ldots, \phi_m, \phi'_1, \ldots, \phi'_n$ such that $\forall i = 1 \ldots n : t_i \xrightarrow{\phi'_i}{}^*_A q''_i$, $f(q''_1, \ldots, q''_n) \to_{\Delta_A} q_0$ and $q_0 \xrightarrow{\phi_1} q_1 \xrightarrow{\phi_2} \ldots q_n$, $q = q_n$. Furthermore, we obtain that $\phi = \bigwedge_{i=1}^n \phi'_i \wedge \bigwedge_{i=1}^m \phi_i$. Since terms t_i are of height lesser or equal to n, $\forall i = 1 \ldots n : t_i \to^*_B q_i$ and $\forall i = 1 \ldots n : t_i \xrightarrow{\phi'_i}{}^*_A q''_i$, we can apply the induction hypothesis and obtain that $\forall i = 1 \ldots n : (q_i, q''_i, \phi''_i) \in S$ with $\phi'_i \models \phi''_i$. Besides to this, using case 1 of definition 6 on $f(q_1, \ldots, q_n) \to q' \in \Delta_B$, $f(q''_1, \ldots, q''_n) \to q_0 \in \Delta_A$, and $\forall i = 1 \ldots n : (q_i, q''_i, \phi''_i) \in S$, we obtain that there exists a formula ϕ' such that $(q_0, q', (\bigwedge_{i=1}^n \phi''_i) \vee \phi') \in S$. Then, like in the base case, since $q_0 \xrightarrow{\phi_1} q_1 \xrightarrow{\phi_2} \ldots q_n$, $q = q_n$, we can deduce that there exists a formula ϕ'' such that $(q, q', (\bigwedge_{i=1}^n \phi''_i \wedge \bigwedge_{i=1}^m \phi_i) \vee \phi'') \in S$. Let $\phi_S = (\bigwedge_{i=1}^n \phi''_i \wedge \bigwedge_{i=1}^m \phi_i) \vee \phi''$. Since we know from above that $\phi = \bigwedge_{i=1}^n \phi'_i \wedge \bigwedge_{i=1}^m \phi_i$ and $\forall i = 1 \ldots n : \phi'_i \models \phi''_i$, we obtain that $\phi \models \phi_S$.

Second, we prove the 'if' part: if $(q, q', \phi_S) \in S$ and $\phi_S \neq \bot$ then there exists a term t and a formula $\phi \neq \bot$ such that $\phi \models \phi_S$, $t \xrightarrow{\phi}{}^*_A q$ and $t \to^*_B q'$. We make a proof by induction on the number of applications of the two rules of definition 26, necessary to prove that (q, q', ϕ_S) in S.

- If the number of steps is 0 then, since the computation of S starts from the set $Q^A \times Q^B \times \bot$, then all (q, q', ϕ_S) are such that $\phi_S = \bot$, which is a contradiction.
- We assume that the property is true for any triple (q, q', ϕ_S) which can be deducted by n or less applications of the rules of definition 26. Now, we consider the case of a triple (q, q', ϕ_S) that is deduced at the $n+1$-th step of application of the deduction rules.
  - If the first rule is concerned, this means that there exists triples $(q_1, q'_1, \phi_1), \ldots, (q_n, q'_n, \phi_n)$ and (q, q', ϕ) in S deduced before $n+1$-th step, as well as transitions $f(q_1, \ldots, q_n) \to q \in \Delta_A$ and $f(q'_1, \ldots, q'_n) \to q' \in \Delta_B$. Furthermore, we know that $\phi_S = \phi \vee \bigwedge_{i=1}^n \phi_i$. If $\phi \neq \bot$ then, since (q, q', ϕ) was shown to belong to S before $n+1$-th step, we can apply the

induction hypothesis and directly obtain that there exists a term t and a formula ϕ' such that $\phi' \models \phi$, $t \xrightarrow{\phi'}{}^*_A q$ and $t \to^*_B q'$. Note that $\phi' \models \phi$ implies $\phi' \models \phi_S$. Otherwise, if $\phi = \bot$, then we can apply the induction hypothesis on triples (q_i, q'_i, ϕ_i), $i = 1 \dots n$ and obtain that $\forall i = 1 \dots n : \exists \phi'_i : \exists t_i \in \mathcal{T}(\mathcal{F}) : \phi'_i \models \phi_i$, $t_i \xrightarrow{\phi'_i}{}^*_A q_i$ and $t_i \to^*_B q'_i$. Finally, because of the two transitions $f(q_1, \dots, q_n) \to q \in \Delta_A$ and $f(q'_1, \dots, q'_n) \to q' \in \Delta_B$, we get that $f(t_1, \dots, t_n) \xrightarrow{\phi'}{}^*_A f(q_1, \dots, q_n) \to^*_A q$ with $\phi' = \bigwedge_{i=1}^n \phi'_i$ on one side and $f(t_1, \dots, t_n) \to_B f(q'_1, \dots, q'_n) \to^*_B q$ on the other side. Furthermore, since $\forall i = 1 \dots n : \phi'_i \models \phi_i$, we have $\bigwedge_{i=1}^n \phi'_i \models \bigwedge_{i=1}^n \phi_i$. Recall that $\phi' = \bigwedge_{i=1}^n \phi'_i$ and $\phi_S = \phi \vee \bigwedge_{i=1}^n \phi_i$. Hence, $\phi' \models \phi_S$.

– If the second rule is concerned, this means that there exists triples (q_1, q', ϕ_1) and (q, q', ϕ_2) in S deduced before the $n+1$-th step. Furthermore, we know that $\phi_S = (\phi_1 \wedge \phi) \vee \phi_2$. Like above, if $\phi_2 \neq \bot$ then we can apply induction hypothesis on (q, q', ϕ_2) and trivially get the result. Otherwise, if $\phi_2 = \bot$ then we can use induction hypothesis on the triple (q_1, q', ϕ_1) and obtain that there exists a formula ϕ'_1 and a term t_1 such that $t_1 \xrightarrow{\phi'_1}{}^*_A q_1$, $t_1 \to^*_B q'$ and $\phi'_1 \models \phi_1$. Then, by case on the epsilon transition used for the deduction on S, we prove that $t_1 \xrightarrow{\phi'_1 \wedge \phi}{}^*_A q$:

  * Assume that $q_1 \xrightarrow{\phi} q \in \varepsilon_{\mathcal{R}}$. Then, by definition 6, we obtain that $t_1 \xrightarrow{\phi'_1 \wedge \phi}{}^*_A q$. Furthermore, since $\phi'_1 \models \phi_1$, we have that $\phi'_1 \wedge \phi \models \phi_1 \wedge \phi$ and, finally, that $\phi'_1 \wedge \phi \models \phi_S$.
  * Assume that $q_1 \to q \in \varepsilon_=$. By definition 6, we obtain that $t \xrightarrow{\phi_1 \vee Eq(q_1, q)}{}^*_A q$. Finally, like above, we can deduce that $\phi'_1 \wedge Eq(q_1, q) \models \phi_1 \wedge Eq(q_1, q)$ and thus $\phi'_1 \wedge Eq(q_1, q) \models \phi_S$.