Confluence of right ground term rewriting systems is decidable

Lukasz Kaiser

Mathematische Grundlagen der Informatik, RWTH Aachen

Abstract. Term rewriting systems provide a versatile model of computation. An important property which allows to abstract from potential nondeterminism of parallel execution of the modelled program is confluence. In this paper we prove that confluence of a fairly large class of systems, namely right ground term rewriting systems, is decidable. We introduce a labelling of variables with colours and constrain substitutions according to these colours. We show how right ground rewriting systems can be reduced to simple systems with coloured variables. Such systems can be analysed using reduction-automata techniques which leads to an interesting decision procedure for confluence.

Introduction

Term rewriting systems (TRS) were developed from mathematical logic and are used in many contexts in computer science. They serve as models for computer programs, abstract mathematical structures and are used in equational reasoning. Such systems consist of sets of rewriting rules that can be applied to transform one term into another. There are many interesting properties of TRS and algorithms working on them used in different fields including functional programming languages, where properties like confluence and termination of TRS are investigated.

Confluence, also called the Church-Rosser property, is a very important property of TRS and programs that contain some kind of nondeterminism, for example parallel or probabilistic programs. It states that after any possible rewritings of a term or after a number of steps of program execution on different execution paths there is always a way to rewrite to a common term or follow the program execution to the same result, which can eliminate the problem of nondeterminism.

Confluence is known to be undecidable for general TRS. Oyamaguchi studied confluence of a simple class of ground TRS already in 1987 and showed it to be decidable [11]. Dauchet et. al. gave a decision procedure for the first order theory of ground rewrite systems in 1990 [5] using methods related to tree automata and tree transducers. In 2001, Comon, Godoy and Nieuwenhuis showed that confluence of ground TRS can be decided in polynomial time [1] and they were the first to use new methods like analysing top stable symbols to attack the problem. This line of research was continued by Tiwari [13].

Ordered term rewriting systems were also analysed and Comon, Narendran, Nieuwenhuis and Rusinowitch proved the decidability of confluence of such systems for wide classes of orderings [3, 4].

In recent years, there was an active development in the theory of wider classes of TRS, right ground systems and linear shallow systems. Godoy, Tiwari, and Verma showed that the confluence of linear shallow term rewrite systems can be decided in polynomial time [7]. Their article not only extended the methods of [1] but also simplified and clarified the proofs. Finally the proofs of [7] were again redone and presented in a clarified form in [6].

When we go outside linear systems, things become undecidable quite fast. Marcinkowski proved in 1997 that the first order theory of right ground rewriting is undecidable even for one step rewriting [10]. Also in 2003 Jacquemard proved that reachability and confluence are undecidable for general flat term rewriting systems [8].

When we consider the natural syntactic division of rewriting systems based on whether the rules are ground, linear of flat and we want to analyse reachability, joinability, confluence and first order theory of such systems then the results mentioned before, together with the reductions in [15] answer all decidability questions except for the one we want to investigate here, the confluence of right ground systems. This was a long standing open problem [16] solved in [9] and also recently in an independent work by Tiwari, Godoy and Verma in [14], where authors further developed stability and rewrite closure methods used in [1, 7].

We extend the right ground rewriting system to a system with constraints, analyse the constrained system and look for constrained substitutions. This allows us to see the methods used before in a different context and use reduction automata techniques (see [2]) to complete the proof. Combining automata techniques and analysis of rewriting properties has already proved successful many times and goes back to [5, 11], conditional rewriting systems are also well known and widely used. Moreover, methods using automata techniques and constrained rewriting have often been used in different contexts, so we hope that the presented methods not only give the decision procedure for confluence but can also be extended to other problems and used in program analysis.

The organisation of this article follows the outline of the proof that confluence of RGTRS is decidable and the reductions done to the system. First we define the basic notions and tools that will be used for right ground systems and reduce the rewriting system by naming all ground terms in the rules by new constants and then by taking a limited rewrite closure. This reduction has already become a standard starting point when analysing right ground rewriting systems. Then we prove a technical lemma and reduce the non-confluence problem to the problem of deep non-joinability of constants and semi non-confluence, which is also a variation of a well known method.

Later we introduce colour constraints and coloured substitutions and show how standard unification can be extended to the coloured case. We analyse stability of terms and reduce semi non-confluence to the existence of stable terms fulfilling some constraints. We then show how to decide the existence of such terms by reducing it to emptiness of reduction automata which is known to be decidable. Also the deep joinability of constants is reduced to emptiness of reduction automata, which completes the proof.

Acknowledgement. The result proved in the paper was first proved as a part of the authors masters thesis [9] written under the direction of *Leszek Pacholski*. We also want to thank *Jerzy Marcinkowski* for helpful comments and revision. Later *Dietmar Berwanger* and *Erich Grädel* helped to prepare the current version. We also want to thank *Christof Löding* for references about tree automata that made it possible to clarify and simplify the proofs a lot.

1 Basic notions

1.1 Terms and positions in terms

Let us assume that we are given a finite set of symbols Σ called the *signature* and a function *arity* : $\Sigma \to \mathbb{N}$. Symbols with arity 0 will be called constants ($\Gamma = \{c \in \Sigma : \text{arity}(c) = 0\}$) and denoted by letters a, b, c. The other symbols will be called function symbols and denoted by letters f, g, h. We also assume that there is an infinite set of variables V which will be denoted by letters x, y, z. Throughout the paper the signature will be assumed to be constant, also in all algorithmic problems the maximal arity of function symbols is assumed to be a constant and not an input parameter.

Terms over Σ are defined inductively as the smallest set $\mathcal{T}$ such that:

- $\mathcal{T} \supseteq V$,
- if $f \in \Sigma$ with arity n and $t_1, \dots, t_n \in \mathcal{T}$ then $f(t_1, \dots, t_n) \in \mathcal{T}$.

The set $\text{Var}(t)$ of variables occurring in a term t is also defined inductively by $\text{Var}(c) := \emptyset$, $\text{Var}(x) := \{x\}$ and $\text{Var}(f(t_1, \dots, t_n)) = \text{Var}(t_1) \cup \dots \cup \text{Var}(t_n)$. When $\text{Var}(t) = \emptyset$ then the term t is called *ground*.

The usual intuition behind terms is to view them as labelled trees, therefore we introduce the notion of positions in terms. The set P of *positions* in terms is the set of sequences of positive natural numbers. By $\lambda \in P$ we will denote the empty sequence or the top (root) position in the term.

For a given term t and position p we either say that p does not exist in t or define the term at position p in t (denoted by $t|_p$) in the following inductive way:

- λ exists in each term and $t|_\lambda = t$,
- $p = (n, q)$ exists in $t = f(t_1, \dots, t_m)$ if $m \geq n$ and q exists in t_n and in such case $t|_p = t_n|_q$.

A position p is *above* some position q if there exists a sequence r of numbers such that $q = (p, r)$. In this case we also say that q is *below* p. The height of a position is its length. The height of a term is the maximal height of a position existing in this term.

For example in the term $f(a, f(b, c))$ position $2, 1$ exists and $f(a, f(b, c))|_{2,1} = b$, but neither the position 3 nor the position $1, 2$ exists. The height of $f(a, f(b, c))$ is 2, the height of $f(b, c)$ is 1 and the height of a constant is 0.

1.2 Substitutions and rewritings

Substituting term s in term t at position p yields the term $r = t[s]_p$ such that for all positions q not below p that exist in t, it holds that $r|_q = t|_q$ and $r|_p = s$. Less formally r is just t with the subtree at position p replaced by s, for example substituting $f(a, b)$ at position 1 in $f(a, f(b, c))$ yields the term $f(f(a, b), f(b, c))$.

Substituting term s in term t *for a variable* x is defined as substituting s in t at all positions p where $t|_p = x$. A *substitution* (usually denoted with letters σ, τ, ρ) is a set of pairs, each consisting of a variable and a term (such pairs are denoted by $x \leftarrow t$). Applying a substitution $\sigma = \{x_1 \leftarrow t_1; \ldots; x_n \leftarrow t_n\}$ to a term t, we obtain a term $r = t\sigma$ which is the result of substituting each x_i by t_i in t. As an example, let us take the term $t = f(x, y)$ and the substitution $\sigma = \{x \leftarrow a, y \leftarrow f(b, c)\}$. Then $t\sigma = f(x, y)\sigma = f(a, f(b, c))$.

A *rewriting rule* is a pair of terms t and s denoted by $t \to s$ such that $\mathrm{Var}(t) \supseteq \mathrm{Var}(s)$. The rule is called *ground* if both t and s are ground and *right ground* if s is ground.

A rewriting rule $l \to r$ can be *applied* to a term t at position p, if there exists a substitution σ of variables in l such that $t|_p = l\sigma$. The result of applying the rule is $t[r\sigma]_p$ - term t rewritten at position p. You should note that there is only one possible result of applying a rule to a term at a given position and that since $\mathrm{Var}(l) \supseteq \mathrm{Var}(r)$, a ground term remains ground after applying a rule to it at any position. For example we can apply a right ground rule $f(x, x) \to c$ to the term $f(c, f(a, a))$ at position 2 and obtain the term $f(c, c)$.

A *term rewriting system* (TRS) is a set of term rewriting rules and throughout this article we consider only systems with finitely many rules. The system is ground (GTRS) or right ground (RGTRS), if all rules in the system are ground or respectively right ground. We say that a term t rewrites to a term r with respect to a given TRS T, if there is a rule in T and a position p in t such that r is the result of applying the rule to t at p and we denote it by $t \to_T r$. The relation $\xrightarrow{*}_T$ is the transitive and reflexive closure of the relation $\to_T$, where $t \xrightarrow{*}_T s$ means that t rewrites to s in a finite number of steps. We will often talk about successive rewriting steps $t \to_T t_1 \to_T t_2 \to_T \ldots \to_T t_n \to_T s$ forming a rewriting path $t \xrightarrow{*} s$. When the system is clear from the context we will omit the index T.

Continuing our previous example, if we take a RGTRS with only one rule $T = \{f(x, x) \to c\}$ then $f(c, f(a, a)) \to_T f(c, c)$ and since $f(c, c) \to_T c$, we can say that $f(c, f(a, a)) \xrightarrow{*}_T c$ on the rewriting path $f(c, f(a, a)) \to f(c, c) \to c$.

Given a term rewriting system T and two terms s and t we will say that s is *reachable* from t if $t \xrightarrow{*}_T s$ and that s is *joinable* with t if there exists a term u such that both $s \xrightarrow{*}_T u$ and $t \xrightarrow{*}_T u$. Any such term u that both $s \xrightarrow{*}_T u$ and $t \xrightarrow{*}_T u$ will be called a *joinability witness* for s and t. In our example $f(c, f(a, a))$ and $f(b, b)$ are joinable, since both can be rewritten to c, and c is the only joinability witness of these two terms.

We say that t and s are *deeply joinable*, if all pairs of terms to which these two respectively rewrite are joinable. More formally when $t \xrightarrow{*}_T t_1$ and $s \xrightarrow{*}_T s_1$ then t_1 and s_1 have to be joinable. If the two terms are not deeply joinable then there exist two non-joinable terms t_1 and s_1 such that $t \xrightarrow{*}_T t_1$ and $s \xrightarrow{*}_T s_1$ which will be called *witnesses of deep non-joinability.* A term t is *confluent* with respect to T, if it is deeply joinable with itself and the witnesses of deep non-joinability of t with t will then be called the *witnesses of non-confluence.* A TRS T is confluent if all terms are confluent with respect to T.

Example 1. Let us take a right ground rewriting system

$$R = \{c \to f(c, c), c \to g(c, c), f(x, f(x, x)) \to c\}.$$

Let us now look at the term $t = f(c, c)$. We can rewrite it at position 2 to $s = f(c, g(c, c))$ and it is easy to see that s can not be rewritten to c, since the g symbol at position 2 will not be reduced by any of the rewriting rules as it is too near to the root position to be destroyed inside the variable in the third rule.

Also please note that $t = f(c, c)$ can be rewritten also as position 2 but with a different rewrite rule obtaining $f(c, f(c, c))$, which can be further reduced to c. So t is not confluent with respect to R and one possible pair of witnesses of non-confluence is $f(c, g(c, c))$ and c.

All mentioned properties (reachability, joinability, deep-joinability, confluence of a term and of a TRS) can also be analysed as algorithmic decision problems: given the TRS and possibly the terms as arguments, decide if the property holds or not.

2 Basic tools for right ground TRS

It is a well known (see [12]) fact that reachability and joinability are decidable for right ground TRS.

Fact 1. *Reachability and joinability problems are decidable for RGTRS.*

2.1 Naming ground terms with constants

We consider an arbitrary RGTRS

$$R = \{l_1 \to r_1, l_2 \to r_2, \ldots, l_n \to r_n\}.$$

Let us now take any ground term of height one $f(c_1, \ldots, c_n)$ appearing as a sub-term of any right side r_i and introduce a constant to name it. So for the term $f(c_1, \ldots, c_n)$ we add a new constant $c_{f(c_1,\ldots,c_n)}$ and two new rewrite rules

$$c_{f(c_1,\ldots,c_n)} \to f(c_1, \ldots, c_n),$$
$$f(c_1, \ldots, c_n) \to c_{f(c_1,\ldots,c_n)}.$$

Then we replace each occurrence of $f(c_1, \ldots, c_n)$ in R with $c_{f(c_1,\ldots,c_n)}$.

Let us notice that the new term rewriting system R_1 obtained in this way is confluent if, and only if, R is confluent, since the relations $\xrightarrow{*}_R$ and $\xrightarrow{*}_{R_1}$ are identical on terms without the constant $c_{f(c_1,\ldots,c_n)}$ and this constant can always be replaced with $f(c_1, \ldots, c_n)$. Therefore we can repeat this procedure until the resulting RGTRS R' has only the following types of rules:

$>$: rules in the form $c \to f(c_1, \dots, c_n)$,
$\leq$: rules $t \to c$, where t is any term.

Of course, here f stands for different function symbols and c for different constants. Further, we will call the rules of type $>$ *increasing*, those of type $\leq$ *non-increasing*, since the first ones increase the height of the term and the second ones do not. This extension allows us to restrict our attention to RGTRS that have only the two types of rules given above, and for a given RGTRS T with such rules we will denote by $T^>$ the rules of the first kind in T and by $T^\leq$ the rules of the second kind. More detailed description of this method and the proof that it preserves confluence can be found in [1].

Since we know that reachability for right ground systems is decidable, we can extend R' to a new system R'' in the following way: for each constants c and c' and each term $f(c_1, \dots, c_n)$ of height one, we have

$$c \to c' \in R'' \text{ if } c \xrightarrow{*}_{R'} c',$$

$$c \to f(c_1, \dots, c_n) \in R'' \text{ if } c \xrightarrow{*}_{R'} f(c_1, \dots, c_n),$$
$$f(c_1, \dots, c_n) \to c \in R'' \text{ if } f(c_1, \dots, c_n) \xrightarrow{*}_{R'} c.$$

Therefore, if a constant rewrites to a term of height one or a term of height one rewrites to a constant, or constant rewrites to another constant, then the rewriting can be done in one step. If RGTRS is in this form, we will call it *reduced.*

The following simple lemma will be used very often.

Lemma 1. *For ground terms $t_1, t_2, \dots, t_n, s$ and reduced RGTRS T we have*

$$t := f(t_1, \dots, t_n) \xrightarrow{*}_T s$$

if and only if one of the following conditions holds:

(1) $s = f(s_1, \dots, s_n)$ and for each i we have $t_i \xrightarrow{*}_T s_i$,
(2) there is a constant c such that $t \xrightarrow{*}_T c$ and $c \xrightarrow{*}_{T^>} s$.

Proof. In any TRS, if any of these two conditions hold, then obviously $t \xrightarrow{*}_T s$. The converse is true in any reduced RGTRS since if there is a rewriting at the root position in t somewhere on the path $t \xrightarrow{*}_T s$, then it has to go through a constant because all non-increasing rules rewrite to a constant. Also, when rewriting from a constant we do not need to use the decreasing rules any more since, if a constant rewrites to a term of height one, then the rewriting can be done in one step without decreasing rules. ∎

Definition 1. *A ground term t is* stable *with respect to a rewriting system T if no sub-term of t that is not a constant rewrites (is in $\xrightarrow{*}_T$ relation) to a constant.*

Stability is a very important property in connection with Lemma 1, since intuitively in stable terms the rewriting needs to be done only at leaf positions. One can think of stability as a normal form with respect only to non-increasing rules. Stability is also useful when analysing joinability, which is expressed by the following lemma.

Lemma 2. *A stable term $f(t_1,\ldots,t_n)$ is not joinable with a constant c with respect to a reduced RGTRS T if and only if for any term $f(c_1,\ldots,c_n)$ such that $c \xrightarrow{*}_T f(c_1,\ldots,c_n)$ there is some sub-term t_i not joinable with c_i.*

Proof. Indeed, the term $f(t_1,\ldots,t_n)$ is stable, so it does not rewrite to any constant and it can be joined with c only if $c \xrightarrow{*}_T f(c_1,\ldots,c_n)$ and $f(c_1,\ldots,c_n)$ will be joined with $f(t_1,\ldots,t_n)$ without rewriting to a constant, so each constant c_i must be joined with the appropriate sub-term t_i. ∎

2.2 Reduction of the confluence problem

Let us now reduce the problem of confluence to a more tractable problem. First we have to define when a RGTRS T is *semi non-confluent*.

Definition 2. *Rewriting system T is semi non-confluent if there exists a term s and a constant c such that s is an instance of the left hand side of some rule $l \to c \in R$ and on the other hand s can be rewritten to a term r and r is not joinable with c.*

Please note that if T is semi non-confluent then it clearly is not confluent, but there can also be other reasons for a system not to be confluent. The following lemma reduces the general confluence case for reduced RGTRS to semi non-confluence and the confluence of constants.

Lemma 3. *If a reduced right ground term rewriting system T is not confluent then either there exists a constant that is not confluent or T is semi non-confluent.*

The prove this lemma, we look at the smallest term that is not confluent with respect to T and analyse possible rewriting paths to the witnesses of non-confluence relying on Lemma 1. The proof is given in detail in appendix A.

3 Coloured terms

Let us now define a set of constraints that we will call colours and show some basic properties of coloured terms and coloured rewritings. This can be interpreted as a simple form of conditional rewriting systems, but we will not introduce the general definitions of conditional systems and only concentrate on our simple case.

The colour constraints are defined in a very simple way, a *colour* K is a set of constants $K = \{c_1,\ldots,c_m\}$. We say that a ground term t has colour K with respect to a TRS T if each $c_i \xrightarrow{*}_T t$. We will omit the TRS T if it is fixed in the context. Please note that with this definition each term t has a number of colours, actually one biggest colour

$$K(t) := \{c \ : \ c \xrightarrow{*}_T t\}$$

and all its sub-colours. Each term has $\emptyset$ as its colour.

Definition 3. *A* coloured term *is a term t with each variable $x \in \mathrm{Var}(t)$ labelled with a colour C_x. A* correct *ground substitution for a coloured term with respect to a TRS T is a substitution σ such that only ground terms are substituted for variables and a ground term s is substituted for a given variable x only if C_x is a colour of s w.r.t T, i.e. $C_x \subseteq K(s)$.*

Definition 4. *A* coloured (right ground) rewrite rule *is a pair consisting of a coloured term and a constant. A coloured rewrite rule $l \to c$ can be applied to a ground term t at position p if there exists a correct ground substitution σ for l such that $t|_p = l\sigma$.*

We will now fix a reduced RGTRS with respect to which the colourings are defined and extend it with a set of coloured rewrite rules so that on any rewriting path of a ground term the increasing rewritings can take place only at the end.

Example 2. Let us continue our example for

$$R = \{c \to f(c,c), c \to g(c,c), f(x, f(x,x)) \to c\}$$

and the colour $K = \{c\}$. Let us take any term t such that $c \xrightarrow{*} t$ and look at the rewriting path

$$f(t,c) \to f(t, f(c,c)) \xrightarrow{*} f(t, f(t,t)) \to c. \tag{1}$$

Please note that using the second rewrite rule in the last step was possible because $c \xrightarrow{*} t$, e.g. for $t = f(c,c)$. Also please note that such rewriting could be done for each term t with colour K.

This suggests a new coloured rewriting rule

$$f(x : K, f(c,c)) \to c,$$

where $x : K$ denotes that the variable x is coloured with K. Looking at the rewriting (1) it is also evident that the coloured rule

$$f(x : K, c) \to c \tag{2}$$

can also be added to the system without changing the semantics or rewriting.

What we will do next is to show how using coloured rules we can eliminate the need to change increasing and non-increasing rules on a rewriting path with respect to a reduced RGTRS.

Please look at the rewriting (1) and follow it again for $t = f(c,c)$, so

$$f(f(c,c),c) \xrightarrow{*}_{R^>} f(f(c,c), f(f(c,c), f(c,c))) \to_{R^\leq} c.$$

As you can see we have to interchange rewriting with $R^>$ and with $R^\leq$ to rewrite the term to c. But if we add the rule (2) to the non-increasing rules ($R^\leq$) then we do not have to use the increasing rules any more.

We will generalise this example to an arbitrary reduced RGTRS T by taking all possible positions in the left sides of rewriting rules in T and substituting there all possible constants and looking if appropriate colouring for the remaining variables can be found. First let us introduce a notation and define what an appropriate colouring is.

We will say that a term s *grows from* a term t if $t \xrightarrow{*}_{T^>} s$. Please note that in such case all rewritings on the rewriting path take place in the leafs of the term (viewed as a tree).

Let us now take a term l (possibly a left side of a rewriting rule) and a sequence of different positions $P = p_1, \ldots, p_n$ existing in l and a sequence of constants $A = c_1, \ldots, c_n$. We will be interested in the term l with each constant c_i substituted at the corresponding position p_i and we will use the notation

$$l(A, P) := (((l[c_1]_{p_1})[c_2]_{p_2})\ldots)[c_n]_{p_n}.$$

Definition 5. *Given a term l a sequence P of positions in l and a sequence A of constants with the same length as P we will say that a colouring*

$$\{x_1 : K_1, \ldots, x_n : K_n\}$$

of variables in l is appropriate *w.r.t. A and P if there exists a term s that fulfils the following properties. The term s grows from $l(A, P)$ and contains exactly the same positions as l and at all positions where there is no variable in l it has the same symbols as l. Then the colouring is appropriate if for each variable x_i the assigned colour K_i is equal to the set of constants that appear in s at the positions at which x_i appears in l.*

Please note that in this definition we assume that the positions P are incomparable with the prefix ordering of positions, so all constants can be put in parallel and the order of positions in P does not matter.

Let us analyse this definition looking at the example presented before. We can take the term $l = f(x, f(x, x))$ and choose to insert the constant c at position 2, so $A = c$ and $P = 2$ and $l(A, P) = f(x, c)$. Although $f(x, c)$ can grow either to $f(x, g(c, c))$ or to $f(x, f(c, c))$, according to the definition we will consider only the second case, as the first one has g at position 1, which is different from f at position 1 in l. We can see that $x : \{c\}$ is the appropriate colouring in this case.

Let us now take all possible rules $l \to c \in T^{\leq}$, all possible sequences of different positions P in l and for each P take all sequences of constants A with the same length.

Let us now colour each rewrite rule

$$l(A, P) \to c.$$

Let us take all possible appropriate colourings of the variables from l with respect to A and P. To obtain colourings of variables from $l(A, P)$ we can just cast each colouring of variables of l, but we will exclude some of them. Namely, if in a colouring of variables of l there are coloured variables that does not appear in $l(A, P)$ and they are coloured with $K_1, \ldots, K_m$ then we will allow the cast of this colouring only if each colour K_i is satisfiable, i.e. there exists a term u such that all constants in K_i rewrite to u. Please note that it is decidable whether a colour is satisfiable as it is a simple extension of joinability (see [12]) and we will call u the satisfiability witness for K_i.

Let us denote the set of all coloured rewrite rules obtained in this way with respect to T coloured with all allowed colourings by T^c. Since we have defined correct ground substitutions for coloured rewrite rules we define the relation $\rightarrow_{T^c}$ and $\xrightarrow{*}_{T^c}$ on ground terms in the same way as we did for uncoloured rewrite rules, only using correct ground substitutions.

Lemma 4. *For any reduced RGTRS T with T^c defined as above and for any two terms t and s if $t \xrightarrow{*}_{T^c} s$ then also $t \xrightarrow{*}_T s$.*

The proof of this lemma follows the construction presented above and is given in detail in appendix B. As we see from the above lemma the extension of T with coloured rules is correct in the sense that it does not change the semantic of rewriting. Moreover, we do not need any more to grow constants in order to match a sub-term in a rewriting rule, since a coloured rule can be used instead, as stated in the following lemma, which is proved in similar way in appendix B.

Definition 6. *Term s grows from a term t* in bounds *of a term l with respect to a reduced RGTRS T if $t \xrightarrow{*}_{T^>} s$ and all rewritings either take place on the positions that exist in l or at (new) positions that do not exist in t.*

Lemma 5. *Given a reduced RGTRS T let us take a rule $l \rightarrow c \in T$ and two ground terms u and w such that w grows from u in bounds of l and w is an instance of l. Then any rewriting path in T in the form*

$$u \xrightarrow{*}_{T^>} w \rightarrow_{\{l \rightarrow c\}} c$$

can be reduced to one step rewriting in the system T^c defined above, so $u \rightarrow_{T^c} c$.

The construction of such coloured closure of the rewriting system will be later used to show that stability of a term with respect to a reduced RGTRS t can be replaced by a property analogous to being a normal form with respect to T^c and therefore that stable terms can be recognised by a reduction automaton.

Before we proceed to analyse confluence we need one more tool to handle unification in the coloured case. Let us assume that we are given a coloured term t and a coloured rewrite rule $l \rightarrow c$ and we want to describe the set of substitutions σ for variables of t such that $t\sigma$ is an instance of l, i.e. there is a correct substitution τ for l such that $t\sigma = l\tau$.

If we forget about colours then we can take the most general unifier α of t and l and denote $u = t\alpha = l\alpha$. As the colours are only constrains on the non-coloured case then obviously all substitutions σ we are looking for will just constrain the most general unifier α. It can also be noted that the right substitutions σ impose exactly such constraints, that guarantee, that on positions where coloured variables appeared in t and l, there will only appear ground terms with the right colour in u. Unluckily, to propagate the constraints from positions in u where there were coloured variables in t and l down to the variables in u we will have to increase the number of unifiers with colour constraints. Let us fix a reduced RGTRS T and state the following lemma.

Lemma 6. *For two coloured terms t and s with disjoint variables there exists a set $u_1, \ldots, u_l$ of terms such that for correct ground substitutions σ, ρ it holds $t\sigma = s\rho$ if, and only if, there exists an i and a correct ground substitution τ for which*

$$t\sigma = u_i\tau = s\rho.$$

Moreover, for each i there exists a coloured substitution μ_i (substituting coloured terms for variables) such that $u_i = t\mu_i = s\mu_i$. The set $\{\mu_1, \ldots, \mu_l\}$ is called the most general unifier of t and s and is computable.

Proof. Let α be the most general unifier of t and s forgetting about the colour constraints and let $u = t\alpha = s\alpha$. It should be noted that there are correct ground substitution σ and ρ such that $t\sigma = s\rho$ exactly then, when there is a ground substitution β for variables in u for which

$$u\beta = t\sigma = s\rho$$

and if there was a variable coloured with colour K at position p in t or in s, then the term substituted at this position has the colour K.

As we see we can describe all the substitutions we are looking for by giving the term u and the set of constraints consisting of a position and a colour. Such constraints can be propagated to lower positions and finally be checked for constants and set as new colours for variables, but for the price of creating multiple copies of u with different constraint sets. The details of how the constraints are propagated are given in appendix B.

4 Stability of coloured terms

According to Lemma 3 we know that we only need to decide deep non-joinability of constants and the semi non-confluence property. We will reduce semi non-confluence to a set of instances of the coloured stability problem. We assume that a reduced RGTRS T is fixed.

Definition 7. *The* coloured stability problem *asks given a coloured term t and a constant c to decide if there exists a correct substitution σ such that $t\sigma$ is stable and not joinable with c.*

Lemma 7. *The problem to decide for a given term s and a constant c if there exists a substitution σ and a stable term t such that $s\sigma \xrightarrow{*} t$ and t is not joinable with c, can be reduced to a finite set of instances of the coloured stability problem.*

Please note that if there exists any such term t then there also exists a stable one. Hence, we can assume that t is stable.

Proof. Let us analyse the reduction path $s\sigma \xrightarrow{*} t$. We can restrict our attention to substitutions σ such that there are no rewritings in the substituted variables, since if there is a need to rewrite, we could have substituted already the rewritten form. Therefore we can also assume that the rewritings are done in the appropriate bounds and use Lemma 5 to describe the rewriting path. First let us divide the rewritings on the path into segments of increasing and non-increasing rewritings (the increasing segments may have length 0)

$$s\sigma = s_1 \xrightarrow{*}_{T>} s'_2 \rightarrow_{T\leq} s_2 \xrightarrow{*}_{T>} s'_3 \rightarrow_{T\leq} s_3 \ldots \rightarrow_{T\leq} s_n \xrightarrow{*}_{T>} s'_{n+1} = t.$$

Then using Lemma 5 we can describe this path with coloured rewritings in the following way:

$$s\sigma = s_1 \rightarrow_{T^c} s_2 \rightarrow_{T^c} \ldots \rightarrow_{T^c} s_n \xrightarrow{*}_{T>} t.$$

Since s is given and the number of positions in s is bounded, we can enumerate all positions in s at which these non-increasing rewritings take place together with the rules applied there. Let us denote these positions by $p_1, \ldots, p_n$ and the coloured rules used at these positions by $l_1 \rightarrow c_1, \ldots, l_n \rightarrow c_n$. For given positions and rules we will enumerate all coloured terms $t_1, \ldots, t_m$ such that if there exists a ground substitution σ satisfying

$$s\sigma = s_1 \rightarrow_{\{l_1 \rightarrow c_1\}} s_2 \rightarrow_{\{l_2 \rightarrow c_2\}} \cdots \rightarrow_{\{l_n \rightarrow c_n\}} s_n$$

then there exists a correct ground substitution ρ for some t_i such that $s_n = t_i\rho$.

If we find such terms t_i then we can substitute for each constant a in t_i a new variable coloured with $\{a\}$ obtaining a terms t'_i and then we will know that $t = t'_i\rho$ for some correct ground substitution ρ and in this way the problem will be reduced.

We will now show how to enumerate the requested coloured terms t_i using the unifiers we defined before. We will proceed inductively with respect to n (the length of the rewriting path $s_1 \xrightarrow{*} s_n$) starting with s and we will show how to proceed one step, generating for one coloured term the appropriate set of coloured terms.

In an intermediate step let us consider the coloured term u such that $s_i = u\rho$ for some correct ground substitution ρ and let s_i be rewritten to s_{i+1} by the coloured non-increasing rule $l_i \rightarrow c_i$ used at position p in u. It is now enough to enumerate the terms $v_1, \ldots, v_m$ such that if for some correct ground substitution σ the term $u\sigma$ can be rewritten with $l_i \rightarrow c_i$ at position p, then $v = v_j\rho$ for some $1 \leq j \leq m$ and some correct substitution ρ. In such case $u|_p\sigma = l_i\tau$ for some correct τ and from Lemma 6 we know that there exists the set

$$\{\mu_1, \ldots, \mu_m\} = \mathrm{mgu}(u|_p, l_i).$$

Then it is sufficient to take $v_i = u\mu_i[c_i]_p$ to get the desired terms. ∎

5 Reduction automata

We have reduced the confluence problem to the coloured stability problem and to the problem of confluence of constants. We will now show how to solve these problems using reduction automata. The definitions, facts and theorems presented here can be found in [2] in the chapter about automata with equality and disequality constraints. Since we are using exactly the same objects as presented in that chapter, we do not present all the terminology with the same level of detail as presented there.

Reduction automata are a special kind of automata with equality and disequality constraints (AWEDC). An *equality (disequality) constraint* is an expression $p_1 = p_2$ ($p_1 \neq p_2$), where p_1 and p_2 are positions and is satisfied by a term t if $t|_{p_1} = t|_{p_2}$ ($t|_{p_1} \neq t|_{p_2}$). An *automaton with equality and disequality constraints* is a tuple

$$(Q, \Sigma, Q_f, \Delta),$$

where Σ is the signature, Q is a finite set of states, $Q_f \subseteq Q$ and Δ is a set of rewrite rules in the form

$$f(q_1, \ldots, q_n) \rightarrow^{\alpha} q,$$

where $q_1, \ldots, q_n, q \in Q$ and α is a boolean combination of equality and disequality constraints.

The language accepted by an automaton and the run of an automaton on a term is defined in an analogous way to the standard automata, only by each application of a rule the corresponding constraint must hold. The automaton is *deterministic* if for every term t there is at most one state q such that there exists a run of the automaton on t ending in the state q, and it is *complete* if there is at least one such state.

A *reduction automata* is a member of AWEDC such that there is a ordering on Q such that for each rule $f(q_1, \ldots, q_n) \rightarrow^{\alpha} q$, where α is not trivial (empty) the state q is strictly smaller than each state q_i. The most important facts about reduction automata (see [2]) that we will use are the following.

Fact 2. *The class of reduction automata is closed under union and intersection. There is a construction for the union that preserves determinism.*

Fact 3. *With each reduction automaton we can associate a complete reduction automaton that accepts the same language. This construction preserves determinism. The class of complete deterministic reduction automata is closed under complement.*

Fact 4. *The emptiness of a language accepted by a reduction automata is decidable.*

Fact 5. *It is possible to construct a deterministic complete reduction automaton accepting the set of terms that are correct ground substitutions of a given term with coloured variables. It is also possible to construct a deterministic complete reduction automaton encompassing such correct ground substitutions.*

From these facts only Fact 5 is not a literal copy of facts from [2], since there the construction is presented for uncoloured terms. But since colour constraints can be expressed as tree automata, deterministic and without constraints, we can use the same construction as presented in [2] for uncoloured terms only adding the states of automata recognising coloured constraints and substituting accepting states of these automata for $q_\top$ used in the uncoloured construction to denote all non-special terms.

Using these facts and the relation between stability with respect to T and being a normal form with respect to T^c that is proved in Lemma 5 we can prove the following lemma (see appendix C for details).

Lemma 8. *The coloured stability problem for a term t and constant c with respect to a reduced RGTRS T is decidable.*

The analysis of deep joinability of constants relies on a technical lemma similar to Lemma 2 that concerns joinability. To use reduction automata for deep joinability of constants we have to analyse pairs and construct the automaton for terms with signature extended to cope with pairs. The technical details are given in appendix C together with the proof of the following lemma.

Lemma 9. *Deep joinability of constants with respect to a RGTRS is decidable.*

From the results proved in lemmas 3, 7, and 8 and 9 follows our main theorem.

Theorem 1. *Confluence of right ground term rewriting systems is decidable.*

6 Conclusions and remarks

We showed how to analyse confluence of right ground term rewriting systems. Our results provide a method to reduce confluence to satisfiability of a constrained stability of terms. Although the presented techniques rely heavily on the fact that the analysed TRS is right ground, it could be interesting to try to extend them to other classes of TRS. The use of reduction automata for solving constrained stability and its extension to deep joinability of constants might be transferred to other cases. It might also be used to prove more refined results concerning right ground or non-increasing systems.

These methods might also be used to analyse special classes of RGTRS in order to get complexity results. Finding an optimised algorithm for coloured stability for linear TRS would open the way to show that left linear right ground TRS are in coNP. If there is no such algorithm then due to the tight integration with automata methods there is a chance that the strict complexity bounds for automata might be translated to show that this problem is not in coNP.

The presented technique of colouring variables with automatic constraints and using more powerful automata to analyse the resulting constrained programs can certainly be used also in other contexts for program analysis.

References

1. H. Comon, G. Godoy, R. Nieuwenhuis, *The Confluence of Ground Term Rewrite Systems is Decidable in Polynomial Time*, 42nd Annual IEEE Symposium on Foundations of Computer Science, Las Vegas, NV, USA, 2001.

2. H. Comon, F. Jacquemard, M. Dauchet, D. Lugiez, R. Gilleron, S. Tison, M. Tomassi, *Tree Automata Techniques and Applications*, available on internet under `http://www.grappa.univ-lille3.fr/tata/`

3. H. Comon, P. Narendran, R. Nieuwenhuis, M. Rusinowitch, *Decision Problems in Ordered Rewriting*, IEEE Symposium on Logic in Computer Science, Indianapolis, IN, USA, 1998.

4. H. Comon, P. Narendran, R. Nieuwenhuis, M. Rusinowitch, *Deciding the Confluence of Ordered Term Rewrite Systems*, ACM Transactions on Computational Logic 33 - 55, ACM Press, New York, NY, USA, 2003.

5. M. Dauchet, S. Tison, *The Theory of Ground Rewrite Systems is Decidable*, IEEE Symposium on Logic in Computer Science, Philadelphia, PA, 1990.

6. G. Godoy, R. Nieuwenhuis, A. Tiwari, *Classes of Term Rewrite Systems with Polynomial Confluence Problems*, ACM Transactions on Computational Logic Vol. 5 No. 2, pp. 321-331, 2004.

7. G. Godoy, A. Tiwari, R. Verma, *On the Confluence of Linear Shallow Term Rewrite Systems*, Symposium on Theoretical Aspects of Computer Science, Berlin, Germany, 2003.

8. F. Jacquemard, *Reachability and Confluence are Undecidable for Flat Term Rewriting Systems*, Information Processing Letters, 87 (5) pp.265-270, 2003.

9. L. Kaiser *Confluence of Right Ground Term Rewriting Systems is Decidable*, Masters Thesis, University of Wroclaw, June 2003.

10. J. Marcinkowski *Undecidability of the First Order Theory of One-Step Right Ground Rewriting*, Intl. Conference on Rewriting Techniques and Applications, Sitges, Spain, 1997.

11. M. Oyamaguchi, *The Church-Rosser Property for Ground Term Rewriting Systems is Decidable*, Theoretical Computer Science, 49 (1) pp.43-79, 1987.

12. M. Oyamaguchi, *The Reachability and Joinability Problems for Right-Ground Term-Rewriting Systems*, Journal of Information Processing, 13 (3) pp.347-354, 1990.

13. A. Tiwari, *Polynomial time Algorithms for Deciding Confluence of Certain Term Rewrite Systems*, IEEE Symposium on Logic in Computer Science, Copenhagen, Denmark, 2002.

14. A. Tiwari, G. Godoy, R. Verma, *Confluence Characterization Using Rewrite Closure with Application to Right Ground Systems*, Applicable Algebra in Engeneering, Communication and Computing Vol. 15 No. 1, pp. 13-36, 2004.

15. R. Verma, M. Rusinowitch, D. Lugiez *Algorithms and Reductions for Rewriting Problems*, Intl. Conference on Rewriting Techniques and Applications, Tsukuba, Japan, 1998.

16. RTA list of open problems, available on internet under `http://www.lsv.ens-cachan.fr/~treinen/rtaloop/problems/`

A Proof of Reduction of Confluence

Lemma 10. *If a right ground term rewriting system R is not confluent then either there exists a constant that is not confluent or the following semi non-confluence property is fulfilled. A RGTRS R is* semi non-confluent *if there exists a term s and a constant c such that s is an instance of a left hand side of a rule $l \to c \in R$ and on the other hand s can be rewritten to a term r and r is not joinable with c.*

Proof. Let us assume that R is not confluent, so there exists a lowest term t that is not confluent. If there exist a few such lowest witnesses of non-confluence with equal height, we can take any of them. If t is a constant then the proof is complete. Assume $t = f(t_1, \ldots, t_n)$. Since t is not confluent, we know that there exist witnesses u, v of non-confluence, so $t \xrightarrow{*} u$, $t \xrightarrow{*} v$ and u and v are not joinable. We can assume that u is the first term on the rewriting path $t \xrightarrow{*} u$ that is not joinable with v and v is the first on the path $t \xrightarrow{*} v$ not joinable with u, otherwise we could just take the terms appearing before on the paths.

Let us now show that there has to be a constant on the rewriting path $t \xrightarrow{*} u$ or $t \xrightarrow{*} v$. Indeed, if there was no constant on these paths then we know by Lemma 1 that $u = f(u_1, \ldots, u_n)$ and $v = f(v_1, \ldots, v_n)$ and for each i $t_i \xrightarrow{*} u_i$ and $t_i \xrightarrow{*} v_i$. But since u and v are not joinable so there exists an i such that u_i and v_i are not joinable, and for this i the term t_i would not be confluent itself, which contradicts the assumption that t was the lowest not confluent term. We can now assume without loss of generality, that there is a constant c on the rewriting path $t \xrightarrow{*} u$. Even more, we can assume that this is the first constant on this path and that each term on the path before c is joinable with v, since any term on the path before u was joinable with v.

We know that

$$t \xrightarrow{*} s \to c \xrightarrow{*} u$$

and that $t \xrightarrow{*} v$ and u is not joinable with v. Let us assume that v and c are joinable and let v_1 be a joinability witness for v and c. Then $c \xrightarrow{*} v_1$, $c \xrightarrow{*} u$ and v_1 is not joinable with u hence c is not confluent, which contradicts the assumption that all constants are confluent. Therefore we know that not v is not joinable with c. Also since s is on the rewriting path before c we know that s is joinable with v and we can denote a witness of their joinability by r. Then we have all the terms required in our assertion, since $s \xrightarrow{*} r$ and r is not joinable with c because v is not joinable with c and $v \xrightarrow{*} r$. ∎

B Proofs of Properties of Coloured Closure

Lemma 11. *For any reduced RGTRS T with T^c defined before and for any two terms t and s if $t \xrightarrow{*}_{T^c} s$ then also $t \xrightarrow{*}_T s$.*

Proof. Let a rule $l(A, P) \to c \in T^c$ be applied to some ground term w at position p with $w|_p = u$. So there is a correct substitution σ such that $u = l(A, P)\sigma$.

Since $l(A,P)$ in the rule is appropriately coloured so there exists the term s that witnesses that the colouring is appropriate and s grows with respect to T from $l(A,P)$ and differs from l only at positions with variables. We can rewrite u in the same way as $l(A,P)$ grows since u is an instance of $l(A,P)$. Therefore we obtain a term v such that $u \xrightarrow{*}_{T>} v$ and at all positions p in l where there are no variables $v|_p = l|_p$.

Let us now take a variable x appearing in l and consider all terms appearing in v at positions where x appears in l. At positions that also appear in $l(A,P)$ there is the term $x\sigma$ and at the other we have some constants $c_1, c_2, \dots, c_n$. But since the colouring is appropriate, then x is coloured with $K = \{c_1, \dots, c_n\}$ and since σ is correct then for each c_i we have $c_i \xrightarrow{*}_T x\sigma$. If we do this rewriting for each variable in l, it becomes clear that $v \xrightarrow{*}_T l\sigma$ and therefore $u \xrightarrow{*}_T c$. You should note that if the variable x does not appear in $l(A,P)$ then we have to rewrite each c_i to the satisfiability witness for K instead of rewriting to $x\sigma$. ∎

Lemma 12. *Given a reduced RGTRS T let us take a rule $l \to c \in T$ and two ground terms u and w such that w grows from u in bounds of l and w is an instance of l. Then any rewriting path in T in the form*

$$u \xrightarrow{*}_{T>} w \to_{\{l \to c\}} c$$

can be reduced to one step rewriting in the system T^c defined before, so $u \to_{T^c} c$.

Proof. Since u grows to an instance of l, there is a sequence of positions in u where there are constants and these positions can grow first to a term s that is identical to l except for the positions where l has variables and later to w being an instance of l. Let us denote the sequence of positions in u mentioned above by P and the sequence constant appearing at respective positions in u by A.

Let us then consider the rule $l(A,P) \to c \in T^c$ with the appropriate colouring for variables of l that comes from s. Please note that since s grows to an instance of l then in the appropriate colouring all colours of variables of l that are not variables of $l(A,P)$ must be satisfiable as the witnesses appear in w, so the mentioned rule indeed is in T^c with the casted colouring. Then it is clear that u rewrites with this rule to c, since the colour constraints are fulfilled in u as they were in w. ∎

We will now repeat literally a part of the proof presented in the paper to be sure that the notation is consistent.

Lemma 13. *For two coloured terms t and s with disjoint variables there exists a set $u_1, \dots, u_l$ of terms such that for correct ground substitutions σ, ρ it holds $t\sigma = s\rho$ if, and only if, there exists an i and a correct ground substitution τ for which*

$$t\sigma = u_i\tau = s\rho.$$

Moreover, for each i there exists a coloured substitution μ_i (substituting coloured terms for variables) such that $u_i = t\mu_i = s\mu_i$. The set $\{\mu_1, \dots, \mu_l\}$ is called the most general unifier of t and s and is computable.

Proof. Let α be the most general unifier of t and s forgetting about the colour constraints and let $u = t\alpha = s\alpha$. It should be noted that there are correct ground substitution σ and ρ such that $t\sigma = s\rho$ exactly then, when there is a ground substitution β for variables in u for which

$$u\beta = t\sigma = s\rho$$

and if there was a variable coloured with colour K at position p in t or in s, then the term substituted at this position has the colour K.

As we see we can describe all the substitutions we are looking for by giving the term u and the set of constraints consisting of a position and a colour. We will now show how such constraint can be propagated to lower positions but for the price of creating multiple copies of u with different constraint sets.

If we have a colour

$$K = \{c_1, \dots, c_m\}$$

at a position in u where the sub-term at this position is $f(w_1, \dots, w_n)$ then the constraint can be satisfied only if for each $c_i \in K$ there is at least one rule in the form

$$c_i \to f(a^i_1, \dots, a^i_n) \in T.$$

Let us now take all possible ways to choose one such rule for each $c_i \in K$. Then for each w_j we have a new colour constraint defined by

$$K_j = \{a^1_j, a^2_j, \dots, a^m_j\}.$$

In this way we reduced a colour constraint to lower positions, but for each way of choosing the rules from the system we had to create a separate instance of the term u with coloured positions. Since for all constants we took into account all possible ways to satisfy the colour constraint, all possible correct substitutions will be taken into account.

If we repeat the above procedure then all colours will be propagated to constants, where they can be checked for satisfiability and either accepted or rejected, and to variables. Taking into account only the cases where the colour constraints were accepted at positions with constants we are left with a set of coloured terms $u_1, \dots u_l$ that we were looking for and since these are just differently coloured copies of u then we can define μ_i to be the most general unifier β with the same colours as the variables in u_i.

C Reduction Automata Constructions and Proofs

Let us first concentrate on the coloured stability problem and start with a simple fact about possible automata construction.

Fact 6. *There is a reduction automata accepting all the normal forms with respect to a given set of coloured rewrite rules.*

Proof. Construct the sum of the automata encompassing the coloured rewrite rules which have a deterministic reduction automata by Fact 5. This construction can be done so that the resulting automata is deterministic (see Fact 2 or [2]) and according to Fact 3 it can also be made complete. Therefore we can construct it's complement using Fact 3. ∎

Lemma 14. *The coloured stability problem for a term t and constant c with respect to a reduced RGTRS T is decidable.*

Proof. According to Lemma 5 we can check the stability of a given term t by creating a reduction automata accepting all normal forms with respect to the coloured rewrite system T^c. When we know that the term is stable we can use Lemma 2 to construct a tree automaton without constraints that will accept only terms that are not joinable with the constant c. This automata works in the described way only on stable terms, but stability is assured by intersecting it with the reduction automata recognising stable terms. Intersecting it again with the automata that accepts only correct ground substitutions of t and checking the emptiness yields a decision procedure according to Fact 4. ∎

We will start analysing deep joinability of constants by exhaustively checking if any two constants have deep non-joinability witnesses of depth zero (other constants). For other cases we will observe the following lemma.

Lemma 15. *Two constants a, b are deeply non-joinable if, and only if, they have witnesses of deep non-joinability of height zero or one of the following holds:*

(1) There exists a term t for which $b \xrightarrow{*} t$ and t is not joinable with a or a constant c such such that $a \to c$, or vice versa (swapping a and b).
(2) There exist terms of height one $f(c_1, \dots, c_n)$ and $g(d_1, \dots, d_m)$ with $f \neq g$ for which $a \to f(c_1, \dots, c_n)$ and $b \to g(d_1, \dots, d_m)$. Moreover, there exist stable terms $f(u_1, \dots, u_n)$ and $g(v_1, \dots, v_m)$ with each u_i having colour $\{c_i\}$ and each v_j having colour $\{d_j\}$.
(3) There exist terms $f(a_1, \dots, a_n)$ and $f(b_1, \dots, b_n$ for which $a \xrightarrow{*} f(a_1, \dots, a_n)$ and $b \xrightarrow{*} f(b_1, \dots, b_n)$. Moreover, stable terms $u = f(u_1, \dots, u_n)$ and $v = f(v_1, \dots, v_n)$ exist with each u_i having colour $\{c_i\}$ and each v_j having colour $\{d_i\}$ and for some $1 \leq i \leq n$ the terms u_i and v_i are witnesses of deep non-joinability of the constants a_i and b_i.

Proof. It is evident that if any of these conditions holds then the constants are deeply non-joinable.

For the converse we need to look at the paths from constants to the witnesses of deep non-joinability of which at least one is of height at least one. If one of the witnesses is of height one then it is covered by the first case taking into account the the fact that the considered RGTRS is reduced

In the other case you note that there exist stable witnesses of deep non-joinability. If these have different function symbols at the root position then stability is enough for them to be witnesses of deep non-joinability. If they have the same function symbol in the head then since they are stable and not joinable then according to lemma 2 they have to have some non-joinable children, which are then witnesses of deep non-joinability for other constants. ∎

Since we are now analysing pairs of terms let us extend our signature by new function symbols P, P_l, P_r with arity two. We will later say that $P(t, s)$ *denotes* t and s, $P_l(t, s)$ denotes the left term t and $P_r(t, s)$ the right term s. Let us also extend our set of coloured rewrite rules so that for each rule $l \to c$ and each position p in l we add the rules

(1) $l[P(l|_p, x)]_p \to c$,
(2) $l[P(x, l|_p]_p \to c$,
(3) $l[P_r(x, l|_p)]_p \to c$,
(4) $l[P_l(l|_p, x)]_p \to c$,

where x is a new variable $x \notin \mathrm{Var}(l)$. We repeat this process as long as possible without having two $P's$ one after another on any path in the term l considered as a tree. Please note that a term t with a P symbol is stable with respect to the new set of rules if all terms that it denotes are stable.

Fact 7. *For each pair of constants a and b there exists a tree automaton $A_{[a,b]}$ that accepts a stable term if it denotes the pair of witnesses of deep non-joinability of a and b.*

Proof. For constants a, b we will denote by q_a the state for all terms with the extended signature for which the denoted term is reachable from a, and by $q_{a,b}$ the state when the denoted term is reachable from a and not joinable with b.

We will denote the state which is reached by a stable term if the term denotes a pair of deep non-joinability witnesses of a and b by $q_{[a,b]}$ and we will also use $q_{l[a,b]}$ and $q_{r[a,b]}$ for the left and right witness. This defines our set of states and by Lemma 15 we can construct $A_{[a,b]}$ with the following rules:

(1) $P(q_a, q_{b,a}) \to q_{[a,b]}$ and $P(q_{a,b}, q_b) \to q_{[a,b]}$,
(2)
$$P(f(q_{a_1}, \ldots, q_{a_n}), g(q_{b_1}, \ldots, q_{b_m})) \to q_{[a,b]}$$
for each $f(a_1, \ldots, a_n) \xleftarrow{*} a$ and $g(b_1, \ldots, b_m) \xleftarrow{*} b$ with $f \neq g$,
(3)
$$P(f(q_{a_1}, \ldots, q_{l[a_j,b_j]}, \ldots, q_{a_n}), f(q_{b_1}, \ldots, q_{r[a_j,b_j]}, \ldots, q_{b_n})) \to q_{[a,b]}$$
for each $f(a_1, \ldots, a_n) \xleftarrow{*} a$ and $f(b_1, \ldots, b_n) \xleftarrow{*} b$,
(4) all above items repeated with P_l or P_r instead of the first P on the left side and $q_{l[]}$ or $q_{r[]}$ on the right side accordingly,
(5) ϵ-transitions from $q_{a,b}$ to q_a.

The correctness of the construction follows from Lemma 15. ∎

Lemma 16. *Deep joinability of constants with respect to a RGTRS is decidable.*

Proof. We showed that we can construct an automaton accepting the witnesses of deep non-joinability of two constants when the terms are stable and we showed before that we can construct an reduction automaton accepting only stable terms (only now we use an extended signature and other set of coloured rewrite rules). Then we can use Fact 4 to decide the emptiness of intersection of these automata.