# Model-Checking Infinite Systems Generated by Ground Tree Rewriting

Christof Löding

RWTH Aachen, Lehrstuhl Informatik VII, 52056 Aachen, Germany
loeding@informatik.rwth-aachen.de

**Abstract.** We consider infinite graphs that are generated by ground tree (or term) rewriting systems. The vertices of these graphs are trees. Thus, with a finite tree automaton one can represent a regular set of vertices. It is shown that for a regular set $T$ of vertices the set of vertices from where one can reach (respectively, infinitely often reach) the set $T$ is again regular. Furthermore it is shown that the problems, given a tree $t$ and a regular set $T$, whether all paths starting in $t$ eventually (respectively, infinitely often) reach $T$, are undecidable. We then define a logic which is in some sense a maximal fragment of temporal logic with a decidable model-checking problem for the class of ground tree rewriting graphs.

## 1 Introduction

Graphs play an important role in the behavioral description of programs or processes. The use of theoretically unbounded data structures (such as stacks, queues, etc.) in these programs requires the use of infinite (transition-)graphs. In order to allow algorithmic applications like verification of infinite-state systems, these systems have to be given effectively, i.e., by some finite object. In recent years many classes of finitely representable infinite graphs have been studied. A starting point for this research on infinite graphs was an analysis of the configuration graphs of pushdown automata by Muller and Schupp in [13]. The vertices of pushdown graphs are words (the control state followed by the stack content) and the edges are defined via the transitions which are prefix rewriting rules on these words. In [13] it is shown that pushdown graphs have a decidable monadic second-order (MSO) theory. Later, more efficient algorithms have been developed for solving reachability problems, model-checking temporal logics [8], and synthezising strategies for games [10, 14] on pushdown graphs.

Using rewriting as the basic process for generation of infinite graphs, a variety of possibilities arises to define classes of graphs different from pushdown graphs. Let us briefly review some of the approaches. The result on the decidability of the MSO theory generalizes from pushdown graphs to prefix recognizable graphs [3], which are also generated by prefix rewriting on words; but the rewriting rules refer to to regular languages instead of single words. Even more general classes of graphs can be obtained when using finite word transducers to define the edge relations of graphs. This leads to automatic graphs in case of synchronous transducers (see e.g. [1]) and to rational graphs [12] in case of asynchronous transducers. The formalisms to generate automatic and rational graphs are very strong and even simple reachability problems on these graphs are undecidable. In [11] model-checking problems for process rewriting graphs are studied (in process rewriting parallel composition of words is allowed in addition to the usual sequential composition).

In the present paper we use another generalization of word rewriting, namely tree rewriting, as already considered in [2]. In tree (or term) rewriting the basic objects in the rewriting systems are trees. For ground tree rewrite systems (GTRS) confluence [6], the first-order theory [7], and several reachability problems [5] have been shown to be decidable. As the example of the infinite grid (see Section 2 below) shows, the MSO theory of a GTRS graph can be undecidable. Our goal is to develop an expressive fragment of temporal logic with a decidable model-checking problem on GTRS graphs. Since reachability analysis of systems constitutes an important part of verification, we analyze different reachability problems. As a means of specification we will use finite tree automata to represent sets of vertices. Given such a regular set $T$ represented by a tree automaton we address the following problems for GTRS graphs.

(1) Compute the set of all vertices from where one can reach $T$.
(2) Compute the set of all vertices from where one can infinitely often visit $T$.
(3) Given a single tree $t$, do all paths starting in $t$ eventually (respectively, infinitely often) visit $T$?

We give algorithms to solve the Problems (1) and (2). The first problem was already solved in the context of term rewriting (see e.g. [4]), so the solution of the second problem is the main contribution of the present paper. The problems in (3) are shown to be undecidable and hence mark a boundary in the design of a fragment of temporal logic with decidable model-checking problem for GTRS graphs.

The paper is organized as follows. In Section 2 we introduce GTRS and tree automata. In Section 3 we give an algorithm to solve the reachability problem (1), in Section 4 we solve the recurrence problem (2), and in Section 5 we show the undecidability of the problems of universal reachability and universal recurrence from (3). Finally, in Section 6, we use the results of the former sections and present a logic with a decidable model-checking problem for GTRS graphs.

I would like to thank Wolfgang Thomas and the anonymous referees for many helpful comments.

## 2 Ground Tree Rewriting Systems and Tree Automata

A ranked alphabet $A$ is a family of sets $(A_i)_{i\in[k]}$, where $[k] = \{0,\ldots,k\}$. For simplicity we identify $A$ with the set $\bigcup_{i=0}^{k} A_i$. A ranked tree $t$ over $A$ is a mapping $t : D_t \to A$ with $D_t \subseteq [k-1]^*$ such that $D_t \neq \emptyset$ is prefix closed, for each $ui \in D_t$ we have $uj \in D_t$ for all $j \leq i$, and if $u0,\ldots,u(i-1) \in D_t$, $ui \notin D_t$, then

$t(u) \in A_i$. $D_t$ is called the domain of $t$. We will mainly be interested in finite trees, except for Section 4 where we also use infinite trees. The set of all finite trees over $A$ is called $T_A$. By $\sqsubseteq$ we denote the prefix ordering on $\mathbb{N}^*$.

For $u \in \mathbb{N}^*$ we define $uD_t = \{uv \in \mathbb{N}^* \mid v \in D_t\}$ and $u^{-1}D_t = \{v \in \mathbb{N}^* \mid uv \in D_t\}$. For $u \in D_t$ the subtree $t^u$ of $t$ at $u$ is the tree with domain $D_{t^u} = u^{-1}D_t$ and $t^u(v) = t(uv)$. For trees $t, t' \in T_A$ and $u \in D_t$ we define $t(u \leftarrow t')$ to be the tree $s$ with domain $D_s = uD_{t'} \cup (D_t \setminus u(u^{-1}D_t))$ and $s(v) = t'(u^{-1}v)$ for $v \in uD_{t'}$ and $s(v) = t(v)$ for $v \in D_t \setminus u(u^{-1}D_t)$. This means we replace the subtree $t^u$ in $t$ by $t'$.

To denote trees we will use the usual graphical notation (as e.g. in Figure 1) and the term notation, where $a(t_0, \ldots, t_{i-1})$ denotes the tree with $a$ at the root and $t_0, \ldots, t_{i-1}$ as subtrees.

A ground tree rewriting system (GTRS) is a tuple $S = (A, \Sigma, P, t_I)$, where $A = (A_i)_{i \in [k]}$ is a ranked alphabet, $\Sigma$ is an alphabet, $P$ is a finite set of rules $t_1 \to^\sigma t_2$ with $t_1, t_2 \in T_A$, $\sigma \in \Sigma$, and $t_I \in T_A$ is the initial tree. For two trees $t, t' \in T_A$ we write $t \to_S^\sigma t'$ iff there exists a rule $t_1 \to^\sigma t_2$ in $P$ and $u \in D_t$ such that $t^u = t_1$ and $t' = t(u \leftarrow t_2)$. We write $t \to_S t'$ iff there is a $\sigma \in \Sigma$ with $t \to_S^\sigma t'$. By $\to_S^+$ we denote the transitive closure of $\to_S$ and by $\to_S^*$ we denote the transitive and reflexive closure of $\to_S$.

The tree language generated by $S$ is $T(S) = \{t \in T_A \mid t_I \to_S^* t\}$. The edge labeled graph $G_S = (V_S, E_S, \Sigma)$ generated by $S$ is defined by $V_S = T(S)$, and $(t, \sigma, t') \in E_S$ iff $t \to_S^\sigma t'$.

*Example 1.* The GTRS $S = (A, \Sigma, P, t_I)$ given by $\Sigma = \{0, 1\}$, $A = (A_i)_{i \in [2]}$ with $A_0 = \{a, b\}$, $A_1 = \{c\}$, and $A_2 = \{d\}$, $P = \{b \to^0 c(b), a \to^1 c(a)\}$, and $t_I = d(a, b)$ generates the infinite grid as shown in Figure 1. $\square$

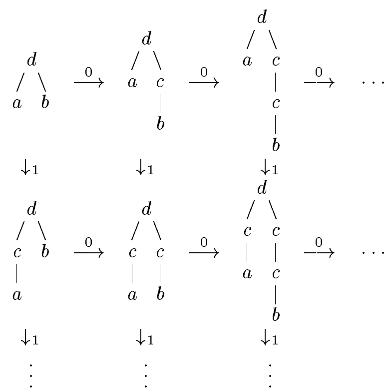

**Fig. 1.** The infinite grid generated by a GTRS.

As usual a path $\pi$ through a graph $G = (V, E)$ is a (possibly infinite) sequence of vertices such that two successive vertices on this path form an edge.

If we consider a path $\pi$ through a graph $G_S$ for a GTRS $S$, then the edges are generated by rewriting rules. We will sometimes refer to these rewritings as the rewritings on $\pi$. Given a path $\pi$ and $i \in \mathbb{N}$ we denote the $i$th element on $\pi$ by $\pi(i)$ and the suffix of $\pi$ starting at $\pi(i)$ by $\pi^i$.

If $T \subseteq T_A$, then $t \to_S^\omega T$ means that there is a path through $G_S$ starting in $t$ that infinitely often visits the set $T$ and if $\pi$ is a path through $G_S$, then $\pi : t \to_S^\omega T$ means that $\pi$ is a path starting in $t$ that infinitely often visits $T$. Furthermore we use similar notations as, e.g., $t \to_S^* T_1 \to_S^* T_2$ if there is a path that starts in $t$ and first visits $T_1$ and then $T_2$.

For a set $T \subseteq T_A$ of trees let $\mathrm{sub}(T) = \{t' \in T_A \mid \exists t \in T \text{ and } u \in D_t \text{ with } t^u = t'\}$ be the set of all subtrees of trees in $T$. For later use we define the sets $P_L = \{t \in T_A \mid \exists t' \in T_A \text{ with } t \to t' \in P\}$, $P_R = \{t \in T_A \mid \exists t' \in T_A \text{ with } t \to t' \in P\}$, and $\mathrm{sub}(P) = \mathrm{sub}(P_L) \cup \mathrm{sub}(P_R)$.

A nondeterministic tree automaton (NTA) is a tuple $\mathcal{A} = (Q, A, \Delta, F)$, where $Q$ is a finite set of states, $A = (A_i)_{i \in [k]}$ is a ranked alphabet, $F \subseteq Q$ is a set of final states and $\Delta \subseteq (\bigcup_{i=0}^k Q^i \times A_i) \times Q$ is the transition relation. Given a tree $t \in T_A$, a tree $\rho \in T_Q$ is a run of $\mathcal{A}$ on $t$ iff $D_\rho = D_t$, and for each $u \in D_t$ with $t(u) \in A_i$ we have $(\rho(u0), \ldots, \rho(u(i-1)), t(u), \rho(u)) \in \Delta$. We extend $\Delta$ to trees in the natural way: $(t, q) \in \Delta$ iff $\rho(\varepsilon) = q$ for a run $\rho$ of $\mathcal{A}$ on $t$. A tree $t$ is accepted by $\mathcal{A}$ iff $(t, q) \in \Delta$ for some $q \in F$ and the language $T(\mathcal{A})$ accepted by $\mathcal{A}$ is $T(\mathcal{A}) = \{t \in T_A \mid (t, q) \in \Delta(t) \text{ for some } q \in F\}$. A language of trees is called regular iff it can be recognized by an NTA.

For $t \in T_A$, a run $\rho$ of an NTA on $t$, and a vertex $u \in D_t$ it is clear that the subtree $\rho^u$ of $\rho$ is a run of the NTA on $t^u$.

The notion of a run can also be applied to infinite trees. This will be used in Section 4.

In the next section we also consider a more general type of automata, namely nondeterministic tree automata with $\varepsilon$-transitions ($\varepsilon$-NTA). An $\varepsilon$-NTA is a tuple $\mathcal{A} = (Q, A, \Delta, F)$, where $Q$, $A$, and $F$ are as in NTA and $\Delta \subseteq ((\bigcup_{i=0}^k Q^i \times A_i) \cup Q) \times Q$ is the transition relation. The transitions of $\Delta$ from the set $Q \times Q$ are called $\varepsilon$-transitions because in a run these $\varepsilon$-transitions do not depend on the input tree. For an $\varepsilon$-NTA $\mathcal{A} = (Q, A, \Delta, F)$ one can easily define an equivalent $\varepsilon$-free NTA $\mathcal{A}^- = (Q, A, \Delta^-, F)$ by $(q_1, \ldots, q_i, a, q) \in \Delta^-$ if there are $p_0, \ldots, p_j \in Q$ with $p_j = q$, $(q_1, \ldots, q_i, a, p_0) \in \Delta$, and $(p_l, p_{l+1}) \in \Delta$ for all $l \in \{0, \ldots, j-1\}$. Then we can define $(t, q) \in \Delta$ iff $(t, q) \in \Delta^-$.

## 3 Reachability

In this section we give an algorithm to solve the following reachability problem:

**(Reachability):** Given a GTRS $S = (A, \Sigma, P, t_I)$ and a regular set of trees $T \subseteq T_A$, compute the set of all trees in $T_A$ from which one can reach a tree of $T$ in the graph $G_S$ associated to $S$.

This problem was already solved in [5] and an algorithm similar to the one presented here can be found in [4] for more general rewrite systems. Thus, we only provide the algorithm for the sake of completeness and without correctness proof.

The main idea of the algorithm is to simulate the rewriting rules within the tree automaton $\mathcal{A} = (Q, A, \Delta, F)$ that accepts the regular set $T = T(\mathcal{A})$. If the automaton contains for each rewriting rule $t_1 \to t_2$ a unique state $q_{t_1}$ that identifies $t_1$, i.e., $(t, q_{t_1}) \in \Delta$ iff $t = t_1$, then such the rule $t_1 \to t_2$ can be simulated by adding an $\varepsilon$-transition $(q_{t_1}, q)$ to $\Delta$, where $q$ is a state with $(t_2, q) \in \Delta$. So the algorithm adds a new part to $\mathcal{A}$ such that subtrees from $P_L$ can be uniquely identified and then starts adding $\varepsilon$-transitions as described above.

First we define the new part that is added to $\mathcal{A}$. Given a set $P$ of rewriting rules over a ranked alphabet $A = (A_i)_{i \in [k]}$, let

- $Q^P = \{q_t \mid t \in \mathrm{sub}(P_L)\}$, and
- $\Delta^P = \{(q_{t_1}, \ldots, q_{t_i}, a, q_{a(t_1,\ldots,t_i)} \mid a(t_1, \ldots, t_i) \in \mathrm{sub}(P_L)\}$.

If these new states and transitions are added to $\mathcal{A}$, then the accepted language does not change and for each $t \in P_L$ the state $q_t$ can only be reached via $t$.

Figure 2 shows an algorithm solving the reachability problem. Note that the algorithm always terminates since only finitely many transitions can be added to the automaton $\mathcal{A}_0$.

INPUT: GTRS $S = (A, \Sigma, P, t_I)$, $\varepsilon$-NTA $\mathcal{A} = (Q, A, \Delta, F)$
$Q_0 = Q \dot\cup Q^P$, $\Delta_0 = \Delta \dot\cup \Delta^P$, $F_0 = F$
$\mathcal{A}_0 := (Q_0, A, \Delta_0, F_0)$
$i := 0$
**while** $\exists t_1 \to t_2 \in P, q \in Q_0$ with $(t_2, q) \in \Delta_i$ and $(t_1, q) \notin \Delta_i$ **do**
  $i := i + 1$
  $\Delta_i := \Delta_{i-1} \cup \{(q_{t_1}, q)\}$
  $\mathcal{A}_i := (Q_0, A, \Delta_i, F_0)$
**end**
$m := i$
OUTPUT: $\varepsilon$-NTA $\mathcal{A}_m$

**Fig. 2.** Algorithm to solve the reachability problem.

For a proof of the following theorem see e.g. [4] (or [8] for the special case of pushdown systems).

**Theorem 1.** *Given a GTRS $S = (A, \Sigma, P, t_I)$ and an $\varepsilon$-NTA $\mathcal{A}$, the algorithm from Figure 2 computes an $\varepsilon$-NTA with $|Q| + |\mathrm{sub}(P_L)|$ states accepting the set* $\{t \in T_A \mid t \to_S^* T\}$.

## 4 Recurrence

After having solved the reachability problem from the former section the next step is to deal with repeated reachability or recurrence.

**(Recurrence):** Given a GTRS $S = (A, \Sigma, P, t_I)$ and a regular set of trees $T \subseteq T_A$, compute the set $\{t \in T_A \mid t \rightarrow_S^\omega T\}$.

For this section we fix a GTRS $S = (A, \Sigma, P, t_I)$ and an NTA $\mathcal{A} = (Q, A, \Delta, F)$. By $\mathcal{A}_q$ we denote the automaton $\mathcal{A}_q = (Q, A, \Delta, \{q\})$.

There are two main steps in the construction of an NTA for the set $\{t \in T_A \mid t \rightarrow_S^\omega T(\mathcal{A})\}$.

**Step 1.** We reduce the recurrence problem for $T(\mathcal{A})$ to the reachability problem for a set $R(\mathcal{A})$ (defined below) by showing $\{t \in T_A \mid t \rightarrow_S^\omega T(\mathcal{A})\} = \{t \in T_A \mid t \rightarrow_S^* R(\mathcal{A})\}$. The set $R(\mathcal{A})$ is regular and can be constructed if we can decide for all trees $t \in P_L$ and all states $q \in Q$ whether $t \rightarrow_S^\omega T(\mathcal{A}_q)$.
**Step 2.** We give a procedure for deciding for $t \in P_L$ and $q \in Q$ if $t \rightarrow_S^\omega T(\mathcal{A}_q)$.

*Step 1.* Informally speaking, the set $R(\mathcal{A})$ (for a fixed GTRS $S$) contains all the trees $t$ such that there is a state $q \in Q$ and vertex $u \in D_t$ with $t^u \in P_L$, $t^u \rightarrow_S^\omega T(\mathcal{A}_q)$, and if we assume that there is a run of $\mathcal{A}$ on $t$ that labels $u$ with $q$, then there is an accepting run of $\mathcal{A}$ on $t$. The last condition ensures that $t(u \leftarrow t') \in T(\mathcal{A})$ for each $t' \in T(\mathcal{A}_q)$. Since the second requirement says that from $t^u$ we can infinitely often reach a tree from $T(\mathcal{A}_q)$, it is clear that from $t$ we can infinitely often reach $T(\mathcal{A})$. On the other hand, if we are given a tree $t$ with $t \rightarrow_S^\omega T(\mathcal{A})$, then we can also reach $R(\mathcal{A})$ from $t$. Before we prove this in Lemma 1 we give a formal definition of $R(\mathcal{A})$.

Given $t \in T_A$, $u \in D_t$, and $q \in Q$, we say that $t(u \leftarrow q)$ is in $T(\mathcal{A})$ if there is a tree $t_1 \in T_A$ such that there is an accepting run $\rho$ of $\mathcal{A}$ on $t(u \leftarrow t_1)$ with $\rho(u) = q$. For this definition we assume that all states of $\mathcal{A}$ are reachable, i.e., for each $q \in Q$ there is a $t \in T_A$ with $(t, q) \in \Delta$. We define the set

$$R(\mathcal{A}) = \left\{ t \in T_A \mid \exists u \in D_t, q \in Q : \begin{array}{c} t^u \in P_L, t^u \rightarrow_S^\omega T(\mathcal{A}_q), \\ \text{and } t(u \leftarrow q) \in T(\mathcal{A}) \end{array} \right\}.$$

**Lemma 1.** *Let $t \in T_A$. Then $t \rightarrow_S^\omega T(\mathcal{A})$ iff $t \rightarrow_S^* R(\mathcal{A})$.*

*Proof.* "$\Rightarrow$": Assume that $t \rightarrow_S^\omega T(\mathcal{A})$ and let $\pi = t_0 \rightarrow_S t_1 \rightarrow_S t_2 \rightarrow_S t_3 \rightarrow_S \cdots$ be a path starting at $t = t_0$ that infinitely often visits $T(\mathcal{A})$. For $i \in \mathbb{N}$ let $u_i \in D_{t_i}$ and $t_i' \in T_A$ such that $t_{i+1} = t_i(u_i \leftarrow t_i')$ (i.e. the rewriting rules are applied to the $t_i^{u_i}$). There is a $j \in \mathbb{N}$ such that no $u_i$ is a proper prefix of $u_j$ for all $i > j$, and there is an infinite number of $i > j$ such that $u_j$ is a prefix of $u_i$. In particular, this means $u_j \in D_{t_i}$ for all $i > j$. Let $t_L = t_j^{u_j}$. We know that $t_L \in P_L$. Since $\pi$ infinitely often visits $T(\mathcal{A})$, there must be $q \in Q$ that is infinitely often at $u_j$ in accepting runs on trees from $T(\mathcal{A})$ on $\pi$. We get $t_L \rightarrow_S^\omega T(\mathcal{A}_q)$ since $u_i$ is not a proper prefix of $u_j$ for $i > j$ and because there are infinitely many substitutions "below" $u_j$ ($u_j$ is a prefix of $u_i$ for infinitely many $i > j$).

Let $k > j$ be such that there is an accepting run of $\mathcal{A}$ on $t_k$ that labels $u_j$ with $q$. Define $t' = t_k(u_j \leftarrow t_L)$ and $u = u_j$. Then $(t')^u = t_L$. From the accepting run on $t_k$ that labels $u$ with $q$ we get $t'(u \leftarrow q) \in T(\mathcal{A})$. Thus, $t' \in R(\mathcal{A})$.

For all $i > j$ we have that $u_i$ is not a proper prefix of $u_j$. Therefore we get $t_j \rightarrow_S^* t'$ and as a consequence $t \rightarrow_S^* t'$.

"$\Leftarrow$": Let $t' \in R(\mathcal{A})$ such that $t \rightarrow_S^* t'$. It is sufficient to show that $t' \rightarrow_S^\omega T(\mathcal{A})$. Let $q \in Q$ and $u \in D_{t'}$ be such that$(t')^u \in P_L$ and $(t')^u \rightarrow_S^\omega T(\mathcal{A}_q)$. Let $t_0 = (t')^u$ and let $t_0 \rightarrow_S t_1 \rightarrow_S t_2 \rightarrow_S t_3 \rightarrow_S \cdots$ be a path starting at $t_0$ that infinitely often visits $T(\mathcal{A}_q)$. Then $t \rightarrow_S t(u \leftarrow t_1) \rightarrow_S t(u \leftarrow t_2) \rightarrow_S t(u \leftarrow t_3) \rightarrow_S \cdots$ is a path starting in $t$ that infinitely often visits $T(\mathcal{A})$. □

Since the set $P_L$ is finite, it is not difficult to verify that the defining properties of trees in $R(\mathcal{A})$ can be checked by a finite tree automaton. But to construct such an automaton we have to decide for each $t \in P_L$ and $q \in Q$ whether $t \rightarrow_S^\omega T(\mathcal{A}_q)$. This is the second step mentioned above and will be done in the remainder of this section.

*Step 2.* There are two possibilities on how to visit the set $T(\mathcal{A})$ infinitely often along a path $\pi$.

(a) There is a single tree $t \in T(\mathcal{A})$ that is visited infinitely often along $\pi$.
(b) There are infinitely many different trees from $T(\mathcal{A})$ on $\pi$.

To distinguish the two cases we analyze how the trees evolve along a path $\pi$ by defining the limit of $\pi$. This is the tree consisting of all the vertices of trees on $\pi$ that are eventually not involved in the rewriting steps any more (i.e. the vertices that are fixed from a certain point onwards). In case (a) this limit will be a finite tree. If the limit of $\pi$ is infinite, then we have to deal with case (b). In both cases we can find a normal form for paths infinitely often visiting $T(\mathcal{A})$. That is, if there is a path $\pi' : t \rightarrow_S^\omega T(\mathcal{A})$ for $t \in P_L$, then there is also a path $\pi$ in normal form with $\pi : t \rightarrow_S^\omega T(\mathcal{A})$. Then we describe a procedure to decide for $t \in P_L$ and $q \in Q$ if there exists a path $\pi$ in normal form with $\pi : t \rightarrow_S^\omega T(\mathcal{A}_q)$.

The normal form for case (a) is based on the same ideas as in Step 1 from above and is stated without proof in Lemma 2. The normal form for case (b) (Lemma 3) is more complicated and will be described in more detail after the formal definition of the limit of a path.

A branch $\beta$ of a tree $t$ is a maximal prefix closed subset of $D_t$ such that for each $u \in \beta$ there is at most one $i \in \mathbb{N}$ with $ui \in \beta$ (in other words: a maximal subset of $D_t$ that is linearly ordered by $\sqsubseteq$). Given a branch $\beta$ of a tree $t$ and a vertex $v \in D_t$ we define $\beta(v)$ to be the maximal prefix $u$ of $v$ with $u \in \beta$.

Let $\pi$ be an infinite path through $G_S$. We say that $u \in \mathbb{N}^*$ is stable on $\pi$ if $u \in D_{\pi(i)}$ for each $i \in \mathbb{N}$ and none of the vertices used in the rewritings on $\pi$ is a prefix of $u$. This simply means that $u$ is never involved in any of the substitutions on the path $\pi$ (note that $u$ is a prefix of itself). The limit $\lim(\pi)$ of $\pi$ is a (possibly infinite) tree with domain

$$D_{\lim(\pi)} = \{u \in \mathbb{N}^* \mid \exists j \in \mathbb{N} \, : \, u \text{ stable on } \pi^j\}.$$

If a vertex $u$ is eventually stable on $\pi$ then it eventually has a fixed label $a$. We take this label to be the label of $u$ in $\lim(\pi)$, i.e., $\lim(\pi)(u) = a$.

The path $\pi$ is called stable iff for all $u \notin D_{\lim(\pi)}$ there is a $j \in \mathbb{N}$ such that $u \notin D_{\pi(i)}$ for all $i \geq j$.

If $\lim(\pi)$ is finite for a path $\pi : t \to T(\mathcal{A})$, then it is not difficult to see that it is possible to visit $T(\mathcal{A})$ infinitely often in the following way. From $t$ one can reach a tree $t''$ which has a subtree $t' = (t'')^u \in P_L$. From $t'$ we can reach a tree $r$ and from $r$ one can reach $t'$ again such that $t''(u \leftarrow r) \in T(\mathcal{A})$. This is formalized in the next lemma. The proof is similar to the construction in the proof of Lemma 1.

**Lemma 2.** *If there exists a path $\pi : t \to_S^\omega T(\mathcal{A})$ with $\lim(\pi)$ finite, then there are $t' \in P_L$, $t'' \in T_A$, $u \in D_{t''}$, and $q \in Q$ such that $t \to_S^* t''$, $(t'')^u = t'$, $t' \to_S^* T(\mathcal{A}_q) \to_S^+ t'$, and $t''(u \leftarrow q) \in T(\mathcal{A})$.* □

The normal form for paths $\pi'$ with $\lim(\pi')$ infinite is obtained by removing unnecessary rewritings. By unnecessary we mean that these rewritings are not essential for visiting $T(\mathcal{A})$ infinitely often. This results in a new path $\pi$ that is stable such that $\lim(\pi)$ only has one infinite branch. Furthermore we can ensure that there are infinitely many trees on $\pi$ that are accepted by $\mathcal{A}$ such that the accepting runs agree on growing initial segments of the infinite branch of $\lim(\pi)$.

**Lemma 3.** *If there exists a path $\pi' : t \to_S^\omega T(\mathcal{A})$ with $\lim(\pi')$ infinite, then there is a path $\pi : t \to_S^\omega T(\mathcal{A})$ with*

*(i)* *$\pi$ is stable,*
*(ii)* *$\lim(\pi)$ has exactly one infinite branch $\beta$, and*
*(iii)* *there is a run $\rho_{\lim}$ of $\mathcal{A}$ on $\lim(\pi)$ with the following property. For each $u \in \beta$ there are infinitely many $i \in \mathbb{N}$ such that there is an accepting run $\rho_i$ of $\mathcal{A}$ on $\pi(i)$ that agrees with $\rho_{\lim}$ on $\{v \mid \beta(v) \sqsubseteq u\}$.*

*Proof.* For simplicity we will only speak of runs instead of runs of $\mathcal{A}$.

Let $\beta$ be an infinite branch of $\lim(\pi')$. We first define a mapping $\rho_\beta$ on $\beta$ that will be extended to the run $\rho_{\lim}$. In the definition of $\rho_\beta$ we make use of König's Lemma to be able to satisfy property (iii).

For $u \in \beta$ let $\beta_u = \{v \in \beta \mid v \sqsubseteq u\}$ be the initial segment of $\beta$ up to $u$ and $V_u = \{\rho : \beta_u \to Q \mid \exists^\omega i \;:\; \pi'(i)$ is accepted by a run that agrees with $\rho$ on $\beta_u\}$ ($\exists^\omega$ means "there are infinitely many"). Define a graph $G = (V, E)$ with $V = (\bigcup_{u \in \beta} V_u) \cup \{\emptyset\}$. The edge relation $E$ is defined as follows.

- $(\rho, \rho') \in E$ if there are $u, u' \in \beta$ with $u' = ui$ for some $i \in \mathbb{N}$ such that $\rho : \beta_u \to Q$, $\rho' : \beta_{u'} \to Q$, and $\rho, \rho'$ agree on $\beta_u$.
- $(\emptyset, \rho) \in E$ if $\rho : \beta_\varepsilon \to Q$.

Then $G$ is an infinite tree (in the graph theoretic sense) of finite degree. Therefore, by König's Lemma, there is an infinite path in $G$. On this path growing initial segments of $\beta$ are labelled consistently. This gives us our mapping $\rho_\beta$.

Now we will modify the path $\pi'$ to obtain the path $\pi$. For each $u \in \beta$ let $i_u \in \mathbb{N}$ be minimal such that the successor $s(u)$ of $u$ on $\beta$ is stable on $(\pi')^{i_u}$ and

$\pi'(i_u)$ is accepted by a run $\rho_u$ that agrees with $\rho_\beta$ on $\beta_{s(u)}$. To obtain $\pi$ remove for each $u \in \beta$ all substitutions on $(\pi')^{i_u}$ at a vertex $v$ with $\beta(v) = u$. Then $\pi$ is still an infinite path and $\pi$ is stable. Furthermore $\lim(\pi)$ only contains the infinite branch $\beta$. It remains to define $\rho_{\lim}$ and verify (iii). For each $u \in D_{\lim(\pi)}$ let $\rho_{\lim}(u) = \rho_{\beta(u)}(u)$. Then one can verify that $\rho_{\lim}$ is indeed a run of $\mathcal{A}$ on $\lim(\pi)$. Since $\pi$ is obtained from $\pi'$ by removing rewritings we get a natural mapping $\varphi : \mathbb{N} \to \mathbb{N}$ associating $\pi'(i)$ with $\pi(\varphi(i))$. This mapping is inductively defined as $\varphi(0) = 0$ and

$$\varphi(i+1) = \begin{cases} \varphi(i) & \text{if the rewriting generating the edge from } \pi'(i) \text{ to} \\ & \pi'(i+1) \text{ was removed from } \pi', \\ \varphi(i)+1 & \text{otherwise.} \end{cases}$$

By the definition of $\pi$ the tree $\pi(\varphi(i_u))$ is accepted for each $u \in \beta$ by a run that agrees with $\rho_{\lim}$ on $\{v \mid \beta(v) \sqsubseteq u\}$. $\square$

Lemmas 2 and 3 provide us with normal forms for paths infinitely often visiting $T(\mathcal{A})$ (or any other regular set of trees). To decide for $t \in P_L$ and $q \in Q$ whether $t \to_S^\omega T(\mathcal{A}_q)$ we construct a finite graph $G$ with vertex set $\mathrm{sub}(P) \times Q \times Q$ and edges labelled with 0 or 1 such that there is a path $\pi$ in normal form with $\pi : t \to_S^\omega T(\mathcal{A}_q)$ iff there is a path through $G$ with infinitely many edges labelled 1. We will motivate the construction of $G$ by the normal form obtained in Lemma 3 but it is not difficult to see that we can also use $G$ to find paths with finite limit.

In the first component of the vertices of $G$ the rewriting steps at vertices on the infinite branch $\beta$ (according to Lemma 3) will be simulated (usually more than one at a time). The second component will keep track of the state labelling $\rho_{\lim}$ of $\beta$. As we have seen in Lemma 3, there are accepting runs for trees on the infinite path $\pi$ that agree with $\rho_{\lim}$ on growing initial segments of $\lim(\pi)$. But it is not guaranteed that these runs completely agree with $\rho_{\lim}$. So, in the third component of the vertices we will simulate the actual accepting runs on trees that are on $\pi$. The property that we will find runs that agree with $\rho_{\lim}$ on growing initial segments allow us to reset the third component to the second one whenever we find an accepting run.

Traversing one edge in $G$ corresponds to going one vertex along the branch $\beta$. If we are at vertex $u$ on $\beta$ and $ui$ is the successor of $u$ on $\beta$, then going from $(t, q, p)$ to $(t', q', p')$ in $G$ means that $t$ is the tree at $u$ before the first substitution takes place at $u$ and $t'$ is the tree at $ui$ after the last substitution at $u$. Thus, one edge of $G$ corresponds to the sequence of substitutions starting at the first substitution at a certain vertex on $\beta$ and ending after the last substitution at this vertex. The edge of $G$ is labelled with 1 if during this sequence of substitutions a tree from $T(\mathcal{A}_q)$ occurs. This is checked using the states in the second and third component of the vertices of $G$.

Formally we define $G = (V, E)$, where $V = \mathrm{sub}(P) \times Q \times Q$ and $E$ contains the following edges.

(1) $(t, q, p) \xrightarrow{0} (t', q', p') \in E$ if there exist $t_1, \ldots, t_l \in T_A$, $q_1, \ldots, q_l, p_1, \ldots, p_l \in Q$, $i \in \{1, \ldots, l\}$ such that

- $a(t_1, \ldots, t_l) \in \mathrm{sub}(P)$ and $t \to_S^* a(t_1, \ldots, t_l)$,
- $t' = t_i$, $q' = q_i$, $p' = p_i$,
- $(q_1, \ldots, q_l, a, q), (p_1, \ldots, p_l, a, p) \in \Delta$, and
- $t_j \to_S^* T(\mathcal{A}_{p_j}) \to_S^* T(\mathcal{A}_{q_j})$ for all $j \in \{1, \ldots, l\}$ with $j \neq i$.

(2) $(t, q, p) \xrightarrow{1} (t', q', p') \in E$ if there exist $t_1, \ldots, t_l \in T_A$, $q_1, \ldots, q_l, p_1, \ldots, p_l \in Q$, $i \in \{1, \ldots, l\}$ such that
- $a(t_1, \ldots, t_l) \in \mathrm{sub}(P)$ and $t \to_S^* T(\mathcal{A}_p) \to_S^* a(t_1, \ldots, t_l)$,
- $t' = t_i$, $q' = q_i$, $p' = p_i$,
- $(q_1, \ldots, q_l, a, q), (p_1, \ldots, p_l, a, q) \in \Delta$, and
- $t_j \to_S^* T(\mathcal{A}_{p_j}) \to_S^* T(\mathcal{A}_{q_j})$ for all $j \in \{1, \ldots, l\}$ with $j \neq i$.

The only differences between (1) and (2) are that in (2) we require $a(t_1, \ldots, t_l)$ to be reachable from $t$ via a tree from $T(\mathcal{A}_p)$ and that we reset the simulation of a run in the third component.

**Lemma 4.** *Let $t \in P_L$ and $q \in Q$.*

*(i) There is a path $\pi$ in $G_S$ with $\pi : t \to_S^\omega T(\mathcal{A}_q)$ and $\lim(\pi)$ infinite iff there is an infinite path in $G$ with infinitely many edges labelled 1 starting from $(t, q, q)$.*

*(ii) There is a path $\pi$ in $G_S$ with $\pi : t \to_S^\omega T(\mathcal{A}_q)$ and $\lim(\pi)$ finite iff there is a path in $G$ from $(t, q, q)$ to $(t', q', q')$ such that $t' \to_S^* T(\mathcal{A}_{q'}) \to_S^+ t'$.*

*Proof.* The proof for (ii) is rather simple with the use of Lemma 2. The idea for the proof of (i) is as follows.
"$\Leftarrow$": By induction on $n$ one can show that if there is a path from $(t, q, p)$ to $(t', q', p')$ in $G$ starting with $n$ 0-edges and ending with one 1-edge, then there is a tree $t''$ and $u \in D_{t''}$ such that $t \to_S^* T(\mathcal{A}_p) \to_S^* t''$, $(t'')^u = t'$, and $t''(u \leftarrow q') \in T(\mathcal{A}_q)$. So, from an infinite path in $G$ starting in $(t, q, q)$ and having infinitely many 1-edges one can construct a path in $G_S$ that infinitely often visits $T(\mathcal{A}_q)$.
"$\Rightarrow$": For this direction the normal form of Lemma 3 is used. The first component in the vertices of $G$ is used to simulate the rewritings on the branch $\beta$ at the point where the vertices on $\beta$ become stable. The second component guesses the labelling of $\beta$ by $\rho_{\lim}$ and the third component is used to simulate the labelling of $\beta$ by accepting runs on trees that are on $\pi$. Property (iii) from Lemma 3 ensures that there are accepting runs that agree with $\rho_{\lim}$ on growing initial segments of $\lim(\pi)$. $\square$

The graph $G$ can be constructed effectively since the only nontrivial conditions in the construction of the edges are instances of the reachability problem. Furthermore condition (i) of the previous lemma is decidable by standard algorithms on finite graphs. Condition (ii) of the previous lemma is decidable because the condition $t' \to_S^* T(\mathcal{A}_{q'}) \to_S^+ t'$ can also be formulated as an instance of the reachability problem. Therefore we get the following lemma.

**Lemma 5.** *For $t \in P_L$ and $q \in Q$ it is decidable whether $t \to_S^\omega T(\mathcal{A}_q)$.*

Summarizing the results from this section we get the following theorem.

**Theorem 2.** *Given a GTRS $S = (A, \Sigma, P, t_I)$ and an NTA $\mathcal{A} = (Q, A, \Delta, F)$, one can construct an $\varepsilon$-NTA with $\mathcal{O}(|Q| + |\mathrm{sub}(P_L)|)$ states accepting the set $\{t \in T_A \mid t \rightarrow^{\omega}_{S} T(\mathcal{A})\}$.*

*Proof.* An automaton for $R(\mathcal{A})$ on an input $t \in T_A$ has to guess a subtree $t^u \in P_L$ and $q \in Q$ with $t^u \rightarrow^{\omega}_{S} T(\mathcal{A}_q)$ and then verify that $t(u \leftarrow q)$ is in $T(\mathcal{A})$. This can be done by an automaton with $\mathcal{O}(|Q| + |\mathrm{sub}(P_L)|)$ states (and by Lemma 5 we can also construct this automaton). Then we apply Theorem 1 and obtain an automaton for $\{t \in T_A \mid t \rightarrow^{\omega}_{S} T(\mathcal{A})\}$ with $\mathcal{O}(|Q| + |\mathrm{sub}(P_L)|)$ states. $\square$

## 5 Universal Reachability and Universal Recurrence

In the previous sections we asked for the existence of a path with a certain property. In this section we address the dual problems.

**(Universal Reachability):** Given a GTRS $S = (A, \Sigma, P, t_I)$ and a regular set of trees $T \subseteq T_A$, does every maximal path in $G_S$ starting in $t_I$ visit $T$?

**(Universal Recurrence):** Given a GTRS $S = (A, \Sigma, P, t_I)$ and a regular set of trees $T \subseteq T_A$, does every infinite path in $G_S$ starting in $t_I$ visit $T$ infinitely often?

We will show the undecidability of these two problems using the same ideas as in [9] where the undecidability of a similar property for Basic Parallel Processes is shown. The general idea is to use reductions from undecidable problems for Turing machines (TM) by defining for a given TM $M$ a GTRS $S_M$ that can simulate computations of $M$. A TM configuration $a_1 \cdots a_k q b_l \cdots b_1$, where the word $a_1 \cdots a_k b_l \cdots b_1$ is on the tape and the TM is in state $q$ with the reading head on $b_l$, will be represented by a tree with two branches with $X, a_1, \ldots, a_k$ on the left branch and $X, b_1, \ldots, b_l, q$ on the right branch. The $X$ symbols are used to model the infinite blank parts of the tape to the left and to the right. For example the first tree in Figure 3 represents the configuration $qa$. For a configuration $\kappa$ of $M$ we will denote the corresponding tree as $t(\kappa)$.

With this coding of configurations it is not possible to exactly simulate the transitions of $M$ by $S_M$. The GTRS $S_M$ can simulate the correct behavior of $M$ as well as incorrect behavior (i.e., in $S_M$ one can reach $t(\kappa')$ from $t(\kappa)$ although $M$ cannot reach $\kappa'$ when started in $\kappa$). But $S_M$ will be constructed in such a way that for every path $\pi$ through $G_{S_M}$ that leads from $t(\kappa)$ to $t(\kappa')$ there is a tree from a regular set $T_{\mathrm{err}}$ on $\pi$ iff $M$ cannot reach $\kappa'$ when started $\kappa$. This property is obtained by adding auxiliary symbols to the tree alphabet and simulating a transition of $M$ in more than one rewriting step with intermediate trees that do not code configurations.

Let $M = (Q, B, \Gamma, q_0, q_s, \delta)$ be a deterministic TM with state set $Q$, input alphabet $B$, tape alphabet $\Gamma$, starting state $q_0$, stop state $q_s$, and transition function $\delta : Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R\}$. Furthermore let $Q \cap \Gamma = \emptyset$, and $\sqcup$ be the blank symbol. If $\kappa$ and $\kappa'$ are configurations of $M$, then we write as usual $\kappa \vdash_M \kappa'$

if $\kappa'$ is the successor configuration of $\kappa$, and $\kappa \vdash_M^* \kappa'$ if $\kappa'$ is a configuration that is reachable from $\kappa$. The initial configuration of $M$ on the empty tape is denoted by $\kappa_\varepsilon$.

The GTRS $S_M = (A, \Sigma, P, t_I)$ is defined as follows. The tree alphabet is $A = A_0 \cup A_1 \cup A_2$ with $A_2 = \{\bullet\}$, $A_1 = Q \cup \Gamma \cup \{X\}$, and $A_0 = A_1 \cup \{add_1, add_2, add_3, rem_1, rem_2, err\}$. We do not need the transition labels $\Sigma$, thus we assume that $\Sigma$ contains one symbol and omit this symbol in the rest of the proof.

The initial tree is $t_I = t(\kappa_\varepsilon) = \begin{array}{ccc} & \bullet & \\ / & & \backslash \\ X & & X \\ & & | \\ & & q_0 \end{array}.$

The set $P$ contains the following rewriting rules.

1. For $\delta(q,a) = (p,b,L)$, $c \in \Gamma$: $\begin{array}{c} a \\ | \\ q \end{array} \to \begin{array}{c} b \\ | \\ p \\ | \\ c \\ | \\ add_1 \end{array}$ and if $a = \sqcup$, then $\begin{array}{c} X \\ | \\ q \end{array} \to \begin{array}{c} X \\ | \\ b \\ | \\ p \\ | \\ c \\ | \\ add_1 \end{array}$.
2. For $\delta(q,a) = (p,b,R)$: $\begin{array}{c} a \\ | \\ q \end{array} \to \begin{array}{c} p \\ | \\ b \\ | \\ rem_1 \end{array}$ and if $a = \sqcup$, then $\begin{array}{c} X \\ | \\ q \end{array} \to \begin{array}{c} X \\ | \\ p \\ | \\ b \\ | \\ rem_1 \end{array}$.
3. For all $a \in \Gamma \cup \{X\}$, $b \in \Gamma$: $a \to \begin{array}{c} a \\ | \\ b \\ | \\ add_1 \end{array}$, $b \to \begin{array}{c} b \\ | \\ rem_1 \end{array}$, $X \to \begin{array}{c} X \\ | \\ \sqcup \\ | \\ rem_1 \end{array}$.
4. $add_1 \to add_2$, $add_2 \to add_3$, $rem_1 \to rem_2$.
5. For all $a \in \Gamma \cup \{X\}$, $b \in \Gamma$, $q \in Q$: $\begin{array}{c} q \\ | \\ b \\ | \\ add_3 \end{array} \to \begin{array}{c} b \\ | \\ q \end{array}$, $\begin{array}{c} a \\ | \\ b \\ | \\ add_3 \end{array} \to \begin{array}{c} a \\ | \\ b \end{array}$.
6. For all $a \in \Gamma \cup \{X\}$, $b \in \Gamma$, $q \in Q$: $\begin{array}{c} a \\ | \\ b \\ | \\ rem_2 \end{array} \to a$, $\begin{array}{c} q \\ | \\ b \\ | \\ rem_2 \end{array} \to q$.
7. For all $a \in \Gamma \cup \{X\}$, $b \in \Gamma$, $p, q \in Q$: $q \to \begin{array}{c} a \\ | \\ p \\ | \\ err \end{array}$, $\begin{array}{c} q \\ | \\ err \end{array} \to q$.

Figure 3 shows the correct simulation of the transition $\delta(q,a) = (p,b,L)$ (the TM is in the configuration $qa$). The symbols $a$ and $q$ are replaced by $b$ and $p$. Since the TM moves the head to the left, it has to guess which symbol is on the left hand side of the head. Here it is the blank symbol. The symbol $add_1$ indicates that the blank symbol should be added to the right branch of the tree. Now the left branch has to confirm with $rem_1$. Then the two branches alternatingly increase their $add$ and $rem$ symbols until they reach $add_3$ and $rem_2$. Now the blank symbol can be removed from the left branch and then it is added to the right branch.

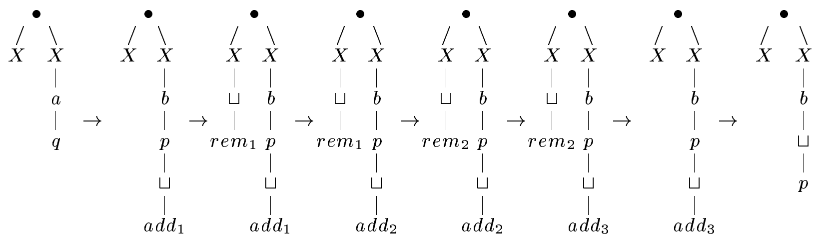

**Fig. 3.** Example for the simulation of a TM-transition by a GTRS.

The set $T_{\text{err}}$ contains all trees that result from a violation of the protocol and all the trees that encode a stop configuration of the TM. From the description below it is easy to see that $T_{\text{err}}$ can be defined by an NTA. A tree $t$ is in $T_{\text{err}}$ iff

1. $t$ contains the *err* symbol,
2. $t$ contains more than one *add* or more than one *rem* symbol,
3. $t$ contains a $rem_i$ symbol and no $add_i$ or $add_{i+1}$ symbol,
4. $t$ contains an $add_2$ symbol and no *rem* symbol, or
5. $t$ contains subtrees of the form $\overset{a}{\underset{rem_i}{|}}$ and $\overset{b}{\underset{add_j}{|}}$ with $a \neq b$.

By a tedious distinction of cases one can check the following lemma.

**Lemma 6.** *(i) For each configuration $\kappa$ of $M$: $t(\kappa_\varepsilon) \to_S^* t(\kappa)$.*
*(ii) For all configurations $\kappa, \kappa'$ of $M$ there is a path from $t(\kappa)$ to $t(\kappa')$ in $G_{S_M}$ not visiting $T_{\text{err}}$ iff $\kappa \vdash_M^* \kappa'$.*

**Theorem 3.** *The problem (Universal Reachability) is undecidable.*

*Proof.* Let $T_{\text{stop}}$ be the set of all trees encoding a stop or deadlocking configuration. Then, by Lemma 6 (ii), it is clear that a TM $M$ does not stop on the empty tape iff there is a path through $G_{S_M}$ starting in $t_I$ that never visits $T_{\text{err}} \cup T_{\text{stop}}$. □

**Theorem 4.** *The problem (Universal Recurrence) is undecidable.*

*Proof.* We use a reduction from a problem similar to the halting problem which can easily be shown to be undecidable:

Given a Turing machine $M$, does there exist a configuration $\kappa$ of $M$ such that $M$ does not stop when started in $\kappa$?

Suppose that such a configuration of $M$ exists. then $t(\kappa_\varepsilon) \to_S^* t(\kappa)$ by Lemma 6 (i), and by Lemma 6 (ii) there is an infinite path starting in $t(\kappa)$ not visiting $T_{\text{err}}$.

If there is an infinite path $\pi$ that visits $T_{\text{err}}$ only finitely often, then one can pick a tree of the form $t(\kappa)$ on $\pi$ such that there is no tree from $T_{\text{err}}$ on the suffix of $\pi$ starting in $t(\kappa)$ (it is not difficult to see that such a tree exists). Then the suffix of $\pi$ starting in $t(\kappa)$ corresponds to an infinite computation of $M$ starting in $\kappa$, again by Lemma 6 (ii). □

## 6 A Logic for Model-Checking over GTRS Graphs

For a fixed ranked alphabet $A$ and an alphabet $\Sigma$ formulas of our logic are defined by the grammar (in CTL-like syntax)

$$\phi := \top \mid \bot \mid T \mid \neg\phi \mid \phi \vee \phi \mid \langle\sigma\rangle\, \phi \mid EF\phi \mid EGF\phi$$

where $\sigma \in \Sigma$ and $T \subseteq T_A$ regular. Given a GTRS $S = (A, \Sigma, P, t_I)$ define the semantics $\|\phi\|_S$ (or simply $\|\phi\|$) of a formula $\phi$ as follows.

- $\|\top\| = T_A$, $\|\bot\| = \emptyset$, $\|T\| = T$, $\|\neg\phi\| = T_A \setminus \|\phi\|$, $\|\phi_1 \vee \phi_2\| = \|\phi_1\| \cup \|\phi_2\|$,
- $\|\langle\sigma\rangle\, \phi\| = \{t \in T_A \mid t \to_S^\sigma \|\phi\|\}$,
- $\|EF\phi\| = \{t \in T_A \mid t \to_S^* \|\phi\|\}$, and
- $\|EGF\phi\| = \{t \in T_A \mid t \to_S^\omega \|\phi\|\}$.

Then $S$ is a model of $\phi$, denoted by $S \models \phi$, iff $t_I \in \|\phi\|$. With the results from the previous sections we get the following theorem.

**Theorem 5.** *For a GTRS $S = (A, \Sigma, P, t_I)$ and a formula $\phi$ one can decide whether $S \models \phi$.*

*Proof.* We can construct automata for $\|\top\|$, $\|\bot\|$, $\|T\|$, $\|\neg\phi\|$, $\|\phi_1 \vee \phi\|$ with standard tree automaton constructions. Given an automaton for $\|\phi\|$, the automata for $\|EF\phi\|$ and $\|EGF\phi\|$ can be constructed according to Theorems 1 and 2. An automaton for $\|\langle\sigma\rangle\, \phi\|$ has to guess a subtree and a rewriting rule such that rewriting this subtree with this rule yields a tree from $\|\phi\|$. We do not give the details of this straightforward construction here. Thus, to decide if $S \models \phi$ we construct the automaton for $\|\phi\|$ and then check if $t_I \in \|\phi\|$. □

The fragment of temporal logic presented here is maximal in the sense that including the operators $EG$ or $EFG$ leads to an undecidable model-checking problem as shown by the results from the previous section. Furthermore, until-operators (as known from temporal logic) also lead to undecidability, because in the GTRS constructed in the proof of Theorem 3 the fact that the TM stops on the empty tape can easily expressed by the until formula $E(\neg T_{\text{err}})UT_{\text{stop}}$ because the existence of a path starting in the initial tree and remaining in the complement of $T_{\text{err}}$ until it eventually reaches $T_{\text{stop}}$ is equivalent to the existence of a halting computation of the TM on the empty tape.

## Conclusion

In the present paper we have analyzed various decision problems for infinite graphs generated by ground tree rewriting. The problems of reachability and recurrence as stated in Sections 3 and 4 were solved whereas the problems of universal reachability and universal recurrence from Section 5 were shown to be undecidable. These considerations lead to a rather expressive and in some sense maximal fragment of temporal logic with a decidable model-checking problem for GTRS graphs.

A drawback of the decision algorithm is that the size of the constructed automaton is non-elementary in the number of nested negations in the formula since complementing NTA gives an exponential blow-up. Note that we cannot simply push the negations inwards to the atomic formulas because this would lead to universal path quantifiers ($A$ instead of $E$) which are difficult to handle with nondeterministic tree automata. The use of alternating tree automata may help to obtain a better complexity.

## References


1. Achim Blumensath and Erich Grädel. Automatic structures. In *Proceedings of LICS '00*, pages 51–62. IEEE Computer Society Press, 2000.
2. Walter S. Brainerd. Tree generating regular systems. *Information and Control*, 14:217–231, 1969.
3. Didier Caucal. On infinite transition graphs having a decidable monadic theory. In *Proceedings ICALP '96*, volume 1099 of *LNCS*. Springer-Verlag, 1996.
4. J.L. Coquidé, M. Dauchet, R. Gilleron, and S. Vágvölgyi. Bottom-up tree pushdown automata: Classification and connection with rewrite systems. *Theoretical Computer Science*, 127(1):69–98, 1994.
5. J.L. Coquidé and R. Gilleron. Proofs and reachability problem for ground rewrite systems. In *Aspects and Prospects of Theoretical Computer Science*, volume 464 of *LNCS*, pages 120–129. Springer, 1990.
6. Max Dauchet, Thierry Heuillard, Pierre Lescanne, and Sophie Tison. Decidability of the confluence of finite ground term rewrite systems and of other related term rewrite systems. *Information and Computation*, 88(2):187–201, October 1990.
7. Max Dauchet and Sophie Tison. The theory of ground rewrite systems is decidable. In *Proceedings LICS '90*, pages 242–248. IEEE Computer Society Press, 1990.
8. Javier Esparza, David Hansel, Peter Rossmanith, and Stefan Schwoon. Efficient algorithms for model checking pushdown systems. In *Proceedings of CAV 2000*, volume 1855 of *LNCS*, pages 232–247. Springer-Verlag, 2000.
9. Javier Esparza and Astrid Kiehn. On the model checking problem for branching time logics and Basic Parallel Processes. In *Proceedings of CAV '95*, volume 939 of *LNCS*, pages 353–366, 1995.
10. Orna Kupferman and Moshe Y. Vardi. An automata-theoretic approach to reasoning about infinite-state systems. In E. A. Emerson and A. P. Sistla, editors, *Proceedings of CAV 2000*, volume 1855 of *LNCS*. Springer-Verlag, 2000.
11. Richard Mayr. Process rewrite systems. *Information and Computation*, 156(1–2):264–286, 2000.
12. Christophe Morvan. On rational graphs. In *Proceedings of FoSSaCS '99*, volume 1784 of *LNCS*, pages 252–266. Springer, 1999.
13. David E. Muller and Paul E. Schupp. The theory of ends, pushdown automata, and second-order logic. *Theoretical Computer Science*, 37:51–75, 1985.
14. Igor Walukiewicz. Pushdown processes: Games and model checking. In Rajeev Alur and Thomas A. Henzinger, editors, *Proceedings of CAV '96*, volume 1102 of *LNCS*, pages 62–74. Springer-Verlag, July-August 1996.