HAL
archives-ouvertes.fr

Closure of Hedge-Automata Languages by Hedge Rewriting

Florent Jacquemard, Michael Rusinowitch

▶ **To cite this version:**

Florent Jacquemard, Michael Rusinowitch. Closure of Hedge-Automata Languages by Hedge Rewriting. A. Voronkov. 19th International Conference on Rewriting Techniques and Applications - RTA 2008, 2008, Hagenberg, Austria. Springer Berlin / Heidelberg, 5117, pp.157-171, 2008, Lecture Notes in Computer Science. <10.1007/978-3-540-70590-1_11>. <inria-00329803>

HAL Id: inria-00329803
https://hal.inria.fr/inria-00329803

Submitted on 13 Oct 2008

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

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

Closure of Hedge-Automata Languages by Hedge Rewriting

Florent Jacquemard[1] and Michael Rusinowitch[2]

[1] INRIA Futurs & LSV, UMR CNRS, ENS Cachan, France
florent.jacquemard@lsv.ens-cachan.fr
[2] LORIA & INRIA Lorraine, UMR 7503, rusi@loria.fr

Abstract. We consider rewriting systems for unranked ordered terms, i.e. trees where the number of successors of a node is not determined by its label, and is not a priori bounded. The rewriting systems are defined such that variables in the rewrite rules can be substituted by hedges (sequences of terms) instead of just terms. Consequently, this notion of rewriting subsumes both standard term rewriting and word rewriting.
We investigate some preservation properties for two classes of languages of unranked ordered terms under this generalization of term rewriting. The considered classes include languages of hedge automata (HA) and some extension (called CF-HA) with context-free languages in transitions, instead of regular languages.
In particular, we show that the set of unranked terms reachable from a given HA language, using a so called inverse context-free rewrite system, is a HA language. The proof, based on a HA completion procedure, reuses and combines known techniques with non-trivial adaptations. Moreover, we prove, with different techniques, that the closure of CF-HA languages with respect to restricted context-free rewrite systems, the symmetric case of the above rewrite systems, is a CF-HA language. As a consequence, the problems of ground reachability and regular hedge model checking are decidable in both cases. We give several counter examples showing that we cannot relax the restrictions.

1 Introduction

In many applications the system states can be modeled by words or trees, sets of configurations by word or tree languages and the transitions of the system can be represented by rewrite rules. In this setting verifying whether a system can enter a set of unsafe states can be expressed as a reachability problem. This approach to the analysis of infinite-state systems requires the computation of the closure of languages under rewrite rules or at least an over-approximation of this closure. Since the usually considered languages are regular the approach is called *regular model checking* [2, 1]. Regular model checking has been quite successful in protocol and hardware verification. For increasing the scope of regular model checking it is therefore important to be able to derive new classes of languages and rewrite systems such that the rewrite closure is computable.

Unranked trees as well as ordered sequences of unranked trees called *hedges* [14, 15, 5] are flexible structures that are quite appealing to represent XML documents where the number of nodes can be modified, for instance when these nodes correspond to database records. Unranked trees have also been employed to model multithreaded recursive program configurations where the number of parallel processes is unbounded [3, 19]. Hedge-automata (HA) are considered now as the natural model of automata for unranked trees. A hedge automaton is a variation of tree automata for hedges. Given a hedge, a hedge automaton assigns some state to a node whenever the *sequence* of states of the siblings belong to some specified word language (sometimes called horizontal language).

Although regular model checking with languages for words and *ranked* trees (where function symbols have fixed arity) has been widely investigated, very few results are available for *unranked* trees and almost none exists on the *computation of exact reachability sets* for HA languages.

In this paper we tackle the problem above by proving (Theorem 1) that we can compute a HA for recognizing the rewrite closure of a language defined by a given HA, for the class of rewrite systems with inverse context-free rules, which are rules whose right-hand side is of type $f(x)$ where x is a variable. Hence in that case we can compute the exact reachability set from the initial one. The rewriting notion that we consider here for unranked terms generalizes ranked term rewriting and is close to the one that has been introduced by [23]. The idea is that the variables in the rewrite rules can be substituted by hedges (sequences of terms) instead of just terms. Moreover our results cannot be derived from related ones on ranked terms (*e.g.* [16]) using encodings of unranked terms into ranked ones (such as the *First-Child-Next-Sibling* encoding or the encoding used in stepwise automata [4]). Relaxing the condition in the definition in the above class of rewrite systems leads to counterexamples (Propositions 3–6).

We have also considered a more general class of automata for unranked ordered trees, called CF-HA, where word context-free languages are used instead of regular ones at the horizontal level. We show (Theorem 2) that CF-HA are preserved by rewrite closure using context-free rewrite rules. Context-free rewrite rules are the symmetric case of inverse context-free rules, *i.e.* rules with left-hand-side of the form $f(x)$. Some additional restrictions are assumed for this result, they cannot be relaxed as shown by the counter examples in Proposition 7–10.

Related works. Whether the rewrite closure of regular ranked trees languages is regular too is a problem that has been addressed in [20, 8, 10, 16, 22, 21, 6]. An important breakthrough of the proof in [16] (against former results) is that it works for TRS which are not left-linear. H. Ohsaki introduces equational tree automata for associative and commutative theories in [17] and study their closure properties for Boolean operations. T. Touili has studied the regular model checking problem for HA [23]. She shows how to compute the image of a HA language in one step of rewriting by a right-linear rewrite system. She also gives a procedure to compute an over-approximation of the rewrite closure of a HA. We rather compute exactly this closure for a class of non-linear rewrite systems.

Our first main result (Theorem 1) can be viewed as a non trivial generalization of both [16] and [23], with proof techniques extending both former constructions.

C. Löding and A. Spelten [12] compute exact rewrite closure of HA for extensions of ground term rewriting and prefix word rewriting. These results cannot be compared to ours since in our case variables (that can be substituted by arbitrarily large hedges) allow non local hedge transformations.

There exists other rewriting notions like the top-down XML transformations [13] or the relabeling transducers of [19] but they do not cover our notion since either they use specific hedge traversal strategies or they are structure-preserving.

Layout of the paper. In Section 2 we introduce terms, hedges and the related rewriting concepts. In particular we define hedge rewriting systems (HRS) and context-free rewrite rules. In Section 3 we recall the hedge-automata classes HA and CF-HA that we shall investigate. In Section 4 we show that the class of HA languages, (*i.e.* recognized by HA) is preserved by rewrite closure for rewriting systems containing rules that are inverse context-free. In Section 5 we show that a class of context-free hedge rewrite systems preserves CF-HA languages. In both Sections 4 and 5, we also exhibit some counter-examples obtained when trying to relax the conditions on rules.

2 Hedge Rewriting

We consider a finite alphabet Σ and an infinite set of variables $\mathcal{X}$. The set of *terms* over Σ and $\mathcal{X}$ is $\mathcal{T}(\Sigma,\mathcal{X}) := \mathcal{X} \cup \{f(h) \mid f \in \Sigma, h \in \mathcal{H}(\Sigma,\mathcal{X})\}$ and the set $\mathcal{H}(\Sigma,\mathcal{X})$ of *hedges* over Σ and $\mathcal{X}$ is the set of finite (possibly empty) sequences of terms of $\mathcal{T}(\Sigma,\mathcal{X})$. When h is empty, $f()$ will be simply written f. We will sometimes consider a term as a hedge of length one, *i.e.* consider that $\mathcal{T}(\Sigma,\mathcal{X}) \subset \mathcal{H}(\Sigma,\mathcal{X})$. The sets of ground terms (terms without variables) and ground hedges are respectively denoted $\mathcal{T}(\Sigma)$ and $\mathcal{H}(\Sigma)$. A hedge $h \in \mathcal{H}(\Sigma,\mathcal{X})$ is called *linear* if every variable of $\mathcal{X}$ occurs at most once in h.

The set of variables occurring in a term $t \in \mathcal{T}(\Sigma,\mathcal{X})$ is denoted $var(t)$. A *substitution* σ is a mapping from $\mathcal{X}$ to $\mathcal{H}(\Sigma,\mathcal{X})$ of finite domain. The application of a substitution σ to a hedge $h \in \mathcal{H}(\Sigma,\mathcal{X})$, denoted $h\sigma$, is the homomorphic extension of σ to $\mathcal{H}(\Sigma,\mathcal{X})$, defined, for $t_1,\ldots,t_n \in \mathcal{T}(\Sigma,\mathcal{X})$, with $n \geq 0$, by $(t_1 \ldots t_n)\sigma := t_1\sigma \ldots t_n\sigma$ and $f(h)\sigma := f(h\sigma)$.

The set of *positions* $\mathcal{P}os(t)$ of a term $t \in \mathcal{T}(\Sigma,\mathcal{X})$ is a set of sequences of positive integers. The empty sequence, denoted ε, is the root position of a term. The subterm of t at position p, denoted $t|_p$, is defined by $f(t_1 \ldots t_n)|_{ip} := t_i|_p$ if $i \leq n$ and, $f(h)|_\varepsilon := f(h)$. The replacement in $t \in \mathcal{T}(\Sigma,\mathcal{X})$ of the subterm at position p by $t' \in \mathcal{T}(\Sigma,\mathcal{X})$ is denoted $t[t']_p$. The *depth* of a term is the maximal length of one of its positions.

A *context* is a linear hedge of $\mathcal{H}\big(\Sigma,\{x\}\big)$, denoted $C[x]$. The application of a context $C[x]$ to a hedge h is defined by $C[h] := C\{x \mapsto h\}$.

A hedge rewriting system (HRS) is a set of rewrite rules of the form $\ell \to r$ where $\ell \in \mathcal{T}(\Sigma,\mathcal{X}) \setminus \mathcal{X}$ and $r \in \mathcal{T}(\Sigma,\mathcal{X})$ (ℓ and r are respectively called *lhs* and

rhs of the rule). The rewrite relation $\xrightarrow[\mathcal{R}]{}$ of an HRS $\mathcal{R}$ is the binary relation on $\mathcal{H}(\Sigma, \mathcal{X})$ defined by $h \xrightarrow[\mathcal{R}]{} h'$ iff $h = (t_1 \dots t_n)$, there exists $i \leq n$, a position $p \in \mathcal{P}os(t_i)$, a rule $\ell \to r \in \mathcal{R}$ and a substitution σ such that $t_i|_p = \ell\sigma$ and $h' = t_1 \dots t_{i-1} t_i[r\sigma] t_{i+1} \dots t_n$. The reflexive and transitive closure of $\xrightarrow[\mathcal{R}]{}$ is denoted $\xrightarrow[\mathcal{R}]{*}$.

Example 1. With $\mathcal{R} = \{g(x) \to x\}$, $\xrightarrow[\mathcal{R}]{}$ associates to a term $g(h)$ the hedge h of its arguments. With $\mathcal{R} = \{g(x) \to g(axb)\}$, $g(c) \xrightarrow[\mathcal{R}]{*} g(a^n c b^n)$ for every $n \geq 0$.

Given a set of terms $L \subseteq \mathcal{T}(\Sigma)$ and an HRS $\mathcal{R}$, we note $\mathcal{R}^*(L)$ the set $\{t \in \mathcal{T}(\Sigma) \mid \exists s \in L, s \xrightarrow[\mathcal{R}]{*} t\}$. We restrict to terms (instead of hedges) because we are mainly interested in term languages below.

A rewrite rule $\ell \to r$ is called *left-linear* (resp. *right-linear*, *linear*) if ℓ (resp. r, both) is linear, *left-ground* (resp. *right-ground*) if $\ell \in \mathcal{T}(\Sigma)$ (resp. $r \in \mathcal{T}(\Sigma)$), *collapsing* if $r \in var(\ell)$, it is called *context-free* if $\ell = f(x)$ with $x \in \mathcal{X}$ (it is not required that $x \in var(r)$ however) and *inverse context-free* if $r \to \ell$ is context-free, *prefix* (resp. *postfix*) if $r = g(t_0 \dots t_n\, x)$ (resp. $r = g(x\, t_0 \dots t_n)$) with $x \in var(\ell)$ and no variable of ℓ occurs in the terms $t_0, \dots, t_n$. A rewrite system is said to have one of the above properties if all its rules have this property.

Example 2. We give a few applications of our rewrite rules in the vein of [23]. A context-free rule $\mathsf{doc}(x) \to \mathsf{doc}(\mathsf{a}x\bar{\mathsf{a}})$ can be employed to introduce tags in an XML document. An inverse context-free rule can be used to eliminate comments $\mathsf{doc}(x\ \mathsf{comment}\ y\ \overline{\mathsf{comment}}) \to \mathsf{doc}(x)$. Non left-linear inverse context-free rules are quite useful for processing list of items as in: $\mathsf{doc}(\mathsf{todo}\ x\ \overline{\mathsf{todo}}\ y\ \mathsf{done}\ x\ \overline{\mathsf{done}}) \to \mathsf{doc}(y)$.

Note that hedge rewriting cannot be reduced to term rewriting through encoding of unranked trees into ranked trees like the First-Child/Next-Sibling encoding, or the encoding used in stepwise automata (see details in the companion report [11]).

3 Hedge-Automata, Context-Free Hedge-Automata

We recall now the definition of hedge-automata [14] (denoted HA) and the less known class of context-free hedge automata (denoted CF-HA) introduced in [18] and where they are shown to recognize the closure of regular (ranked) tree languages modulo associativity.

A *hedge automaton* (resp. *context-free hedge automaton*) is a tuple $\mathcal{A} = (Q, \Sigma, Q^{\mathsf{f}}, \Delta)$ where Q is a finite set of states, Σ is an unranked alphabet, $Q^{\mathsf{f}} \subseteq Q$ is a set of final states, and Δ is a set of transitions of the form $f(L) \to q$ where $f \in \Sigma, q \in Q$ and $L \subseteq Q^*$ is a regular word language (resp. a context-free word language). When Σ is clear from the context it is omitted in the tuple specifying $\mathcal{A}$.

We define the move relation between ground hedges in $\mathcal{T}(\Sigma \cup Q)$ as follows: for every terms t, t' we have $t \xrightarrow[\mathcal{A}]{} t'$ if there exists a context $C[x]$ and a transition $f(L) \to q$ in Δ such that $t = C[f(q_1 \dots q_n)]$, $q_1 \dots q_n \in L$ and $t' = C[q]$. The

relation $\xrightarrow[\mathcal{A}]{*}$ is the transitive closure of $\xrightarrow[\mathcal{A}]{}$. Following [23], we extend $\xrightarrow[\mathcal{A}]{}$ to terms of $\mathcal{T}(\Sigma \cup 2^{Q^*})$ as follows: $C\big[f(L_1 \dots L_n)\big] \xrightarrow[\mathcal{A}]{} C[q]$ if there exists a rule $f(L) \to q$ in $\mathcal{A}$ such that $L_1 \dots L_n \subseteq L$ (in this definition, a lone state q is considered as a singleton set $\{q\}$).

The language denoted by $L(\mathcal{A}, q)$ is the set of ground terms $t \in \mathcal{T}(\Sigma)$ such that $t \xrightarrow[\mathcal{A}]{*} q$. A term is accepted by $\mathcal{A}$ if there is $q \in Q^{\mathsf{f}}$ such that $t \in L(\mathcal{A}, q)$. The language denoted by $L(\mathcal{A})$ is the set of terms accepted by $\mathcal{A}$.

It is know that for both classes of automata [14, 18] membership and emptiness problems are decidable. Moreover HA are closed under Boolean operations.

We call a HA or CF-HA $\mathcal{A} = (Q, Q^{\mathsf{f}}, \Delta)$ *normalized* if for every $f \in \Sigma$ and every $q \in Q$, there is at most one transition rule $f(L_{f,q}) \to q$ in Δ. Every HA (resp. CF-HA) can be transformed into a normalized HA (resp. CF-HA) in polynomial time by replacing every two rules $f(L_1) \to q$ and $f(L_2) \to q$ by $f(L_1 \cup L_2) \to q$.

A HA $\mathcal{A} = (Q, Q^{\mathsf{f}}, \Delta)$ is called *deterministic* iff for all two transitions rules $f(L_1) \to q_1$ and $f(L_2) \to q_2$ in Δ, either $L_1 \cap L_2 = \emptyset$ or $q_1 = q_2$. It is called *complete* if for all $f \in \Sigma$ and and $w \in Q^*$, there exists at least one rule $f(L) \to q \in \Delta$ such that $w \in L$. When $\mathcal{A}$ is deterministic (resp. complete), for all $t \in \mathcal{T}(\Sigma)$, there exists at most (resp. at least) one state $q \in Q$ such that $t \in L(\mathcal{A}, q)$.

Every HA can be completed by adding a sink state (and using the closure properties of regular languages). A determinization procedure (with a subset construction) which preserves completeness is described in Section 4.1 (see also [4]).

3.1 Epsilon- and Collapsing Transitions

We can extend HA and CF-HA with *ε-transitions* of the form $q \to q'$, where q and q' are states, without augmenting the respective expressiveness of these classes. We also consider the extensions of HA (resp. CF-HA), with *collapsing transitions* of the form $L \to q$ where L is a regular (resp. CF) language and q is a state. The move relation for the extended set of transitions is defined as for HA and CF-HA for standard transition and by $C[q_1 \dots q_k] \xrightarrow[\mathcal{A}]{} C[q]$ if $L \to q$ is a collapsing transition of $\mathcal{A}$ and $q_1 \dots q_k \in L$. Note that the collapsing transition $L \to q$ is never applied at the root position (i.e. the above context C cannot be a variable) because HA and CF-HA are limited to the recognition of terms only (and not hedges).

Unlike ε-transitions, collapsing transitions strictly extend HA in expressiveness. However, we show that they can be eliminated for CF-HA.

Proposition 1. *For every extended HA or CF-HA with collapsing transitions $\mathcal{A}$, there exists a CF-HA $\mathcal{A}'$ (without collapsing transitions) such that $L(\mathcal{A}') = L(\mathcal{A})$.*

Proof. Assume that $L \to q$ is a collapsing transition of $\mathcal{A}$. Then we get a CF-HA $\mathcal{A}'$ such that $L(\mathcal{A}') = L(\mathcal{A})$ by replacing every transition $f(L_1) \to q_1$ by the transition $f(L_2) \to q_1$ where L_2 is the context-free word language generated by the grammar G_2 as follows. We consider a context-free grammar G for L (resp.

G_1 for L_1) with axiom X (resp. X_1). The axiom of G_2 is X_1 and the set of productions in G_2 contains *i)* $G[q \leftarrow X_q] \cup G_1[q \leftarrow X_q]$ *i.e.* the terminal q is replaced by a non terminal X_q and *ii)* we add to these rules the production: $X_q := q \mid X$. We can iterate this construction to eliminate all collapsing transitions. □

Proposition 2. *There exists an extended HA with collapsing transitions whose language is not a HA language.*

Proof. Consider the extended HA $\mathcal{A} = (\{q, q_a, q_b, q_{\mathsf{f}}\}, \{g, a, b, c\}, \{q_{\mathsf{f}}\}, \Delta)$ where

$$\Delta = \{c \to q,\ a \to q_a,\ b \to q_b,\ g(q) \to q_{\mathsf{f}},\ q_a q q_b \to q\}$$

Its recognized language is $\{g(a^n c b^n) \mid n \geq 0\}$ and this is not a HA language. □

3.2 Decision Problems

The problem of *ground reachability* and *ground joinability* are to decide that, given two ground terms $s, t \in \mathcal{T}(\Sigma)$ and a HRS $\mathcal{R}$, whether, $s \xrightarrow[\mathcal{R}]{*} t$, respectively, $s \xrightarrow[\mathcal{R}]{*} \circ \xleftarrow[\mathcal{R}]{*} t$.

Regular hedge model checking is the problem to decide, given two HA languages L_{init} and L_{err} and a HRS $\mathcal{R}$ whether $\mathcal{R}^*(L_{\mathsf{init}})$ contains a term of L_{err}. Ground reachability is reducible to regular hedge model-checking. Indeed, given s, t and $\mathcal{R}$, $s \xrightarrow[\mathcal{R}]{*} t$ iff $\mathcal{R}^*\big(\{s\}\big) \cap \{t\} \neq \emptyset$. Note also that if ground-reachability (hence regular hedge model-checking) is undecidable for a class of HRS, then $\mathcal{R}^*(L)$ is not recursive in general when $\mathcal{R}$ is in this class and L is a HA or CF-HA. Indeed, by definition $s \xrightarrow[\mathcal{R}]{*} t$ iff $t \in \mathcal{R}^*\big(\{s\}\big)$.

4 Closure of Regular Hedge Automata Languages

In this section, we prove one result of preservation of HA language for a class of HRS, and give several counter example showing that the restrictions defining this class of HRS are necessary.

4.1 Inverse Context-Free Rewrite Rules

Theorem 1. *The closure $\mathcal{R}^*(L)$ of a HA language $L \subseteq \mathcal{T}(\Sigma)$ under rewriting by an inverse context-free HRS $\mathcal{R}$ is a HA language.*

Proof. Let $\mathcal{A} = (Q, Q^{\mathsf{f}}, \Delta)$ be a complete and normalized HA recognizing L. We shall construct below a finite sequence of HA $\big(\mathcal{A}_i\big)_{i \geq 0}$ whose last element recognizes $\mathcal{R}^*(L)$. Our construction uses elements of [16] and [23], but it is not a simple combination of both. Indeed, on one side we generalize [23] to an unbounded number of rewriting steps, and on the other side we generalize [16] to unranked tree languages. Both generalizations are non-trivial and require new constructions and new conditions.

For each $f \in \Sigma$, $q \in Q$, we note $L_{f,q}$ the language in the transition (assumed unique) $f(L_{f,q}) \to q \in \Delta$. We construct first from $\mathcal{A}$ a deterministic HA $\mathcal{A}_\mathsf{d} = (Q_\mathsf{d}, Q^\mathsf{f}_\mathsf{d}, \Delta_\mathsf{d})$ recognizing L. The HA $\mathcal{A}_\mathsf{d}$ is obtained by a subset construction, see *e.g.* [4], with $Q_\mathsf{d} := 2^Q$, $Q^\mathsf{f}_\mathsf{d} := \{s \in Q_\mathsf{d} \mid s \cap Q^\mathsf{f} \neq \emptyset\}$ and $\Delta_\mathsf{d} := \big\{f(L_{f,s}) \to s \;\big|\; f \in \Sigma, s \subseteq Q\big\}$ where $L_{f,s} := \big(\bigcap_{q\in s} S_{f,q}\big) \setminus \big(\bigcup_{q\notin s} S_{f,q}\big)$ and $S_{f,q} = \big\{s_1 \dots s_n \in Q^*_\mathsf{d} \;\big|\; \exists q_1 \in s_1, \dots, q_n \in s_n, q_1 \dots q_n \in L_{f,q}\big\}$[1].

Next, following the approach of [23], we define first the set of languages of Q^*_d that will be used in the transitions of the $\mathcal{A}_i$'s constructed below. However, we must consider here a bigger set than [23] in order to deal with non linear variables in *lhs* of rules. Let $\mathcal{L}$ be the smallest set of subsets of Q^*_d such that

i. all $L_{f,s}$ (for $f \in \Sigma$ and $s \in Q_\mathsf{d}$) and Q^*_d are in $\mathcal{L}$,
ii. if $L \in \mathcal{L}$ and $u, v \in Q^*_\mathsf{d}$, then $u^{-1}\, L\, v^{-1} \in \mathcal{L}$, where

$$u^{-1}\, L\, v^{-1} := \{w \in Q^*_\mathsf{d} \mid uwv \in L\},$$

iii. if $L_1, L_2 \in \mathcal{L}$ then $L_1 \cap L_2 \in \mathcal{L}$,
iv. if $L_1, L_2 \in \mathcal{L}$ then $L_1 \setminus L_2 \in \mathcal{L}$.

Note that the condition $Q^*_\mathsf{d} \in \mathcal{L}$ in *i* together with *iii* and *iv* imply that $\mathcal{L}$ is also closed under union (if $L_1, L_2 \in \mathcal{L}$ then $L_1 \cup L_2 \in \mathcal{L}$), by De Morgan's Law.

Let us show that $\mathcal{L}$ is finite and that all its members are regular languages. First, let us note that $\mathcal{L}_1$, the smallest set satisfying *i* and *ii* above, is a finite set of regular languages of Q^*_d, since every $L_{f,q}$ is regular by hypothesis. The closure $\mathcal{L}_2$ of $\mathcal{L}_1$ under *iii* and then *iv* is also a finite set of regular languages. The following lemma shows that $\mathcal{L}_2$ fulfills *ii*, *i.e.* that $\mathcal{L}_2 = \mathcal{L}$.

Lemma 1. *For all $L_1, L_2 \subseteq Q^*_\mathsf{d}$, $u_1, u_2, v_1, v_2, u, v \in Q^*$, we have*
$u^{-1}\big(u_1^{-1}\, L_1\, v_1^{-1} \cap u_2^{-1}\, L_2\, v_2^{-1}\big)\, v^{-1} = (u_1 u)^{-1}\, L_1\, (vv_1)^{-1} \cap (u_2 u)^{-1}\, L_2\, (vv_2)^{-1}$,
$u^{-1}\big(u_1^{-1}\, L_1\, v_1^{-1} \setminus u_2^{-1}\, L_2\, v_2^{-1}\big)\, v^{-1} = (u_1 u)^{-1}\, L_1\, (vv_1)^{-1} \setminus (u_2 u)^{-1}\, L_2\, (vv_2)^{-1}$.

Proof. The set in the left-hand-side of the first identity in Lemma 1 is $A = \big\{\ell \mid u\ell v \in \{\ell' \mid u_1 \ell' v_1 \in L_1 \text{ and } u_2 \ell' v_2 \in L_2\}\big\}$, and the set in its right hand side is $B = \big\{\ell \mid u_1 u \ell v v_1 \in L_1 \text{ and } u_2 u \ell v v_2 \in L_2\big\}$. If $\ell \in A$, then $u_1 u \ell v v_1 \in L_1$ and $u_2 u \ell v v_2 \in L_2$, hence $\ell \in B$. Conversely, if $\ell \in B$, then $u\ell v \in u_1^{-1}\, L_1\, v_1^{-1} \cap u_2^{-1}\, L_2\, v_2^{-1}$, hence $\ell \in A$. The proof is very similar for the identity with the complementation. □

Let us now construct the HA $\mathcal{A}_0, \mathcal{A}_1, ...$ as announced. The set of states and final states of each of these HA are respectively Q_d and Q^f_d. We give below an iterative construction of the respective transition sets Δ_i, $i \geq 0$.
Let $\Delta_0 = \Delta_\mathsf{d}$. Assume that Δ_i has been constructed and contains one transition $f(L^i_{f,s}) \to s$ for every $f \in \Sigma$ and $s \in Q_\mathsf{d}$; Δ_{i+1} is obtained from Δ_i as follows: choose (non deterministically) an inverse context-free rewrite rule $\ell \to g(x) \in \mathcal{R}$, and a substitution $\tau : var(\ell) \cup \{x\} \to \{L' \in \mathcal{L} | \forall s_1 \dots s_k \in L', \forall j \leq k, L(\mathcal{A}_i, s_j) \neq \emptyset\}$, such that $\ell\tau \xrightarrow[\mathcal{A}_i]{*} s' \in Q_\mathsf{d}$. Let $L' = x\tau$ (note that if the variable x does not occur in ℓ, then L' is an arbitrary language of $\mathcal{L}$ of sequences of states reachable by $\mathcal{A}_i$); Δ_{i+1} is obtained as follows: for each $s \in Q_\mathsf{d}$,

[1] Note that $S_{f,q}$ and $L_{f,s}$ are indeed regular languages, see [4].

1. replace the rule $g\big(L^i_{g,s}\big) \to s$ by $g\big(L^i_{g,s} \cap L'\big) \to s \cup s'$ and $g\big(L^i_{g,s} \setminus L'\big) \to s$
2. after this operation, normalize the set of transition rules with the operation described in Section 3 (page 5). (Note that if $s' \subseteq s$ then the normalization merges the 2 rules and regenerate $g\big(L^i_{g,s}\big) \to s$.)

The idea behind this construction is that if s' is reachable from a lhs $\ell\tau$ of rewrite rule, then the states in s' must also be reachable from the corresponding rhs $g(x\tau)$. Note that for all transitions $g\big(L\big) \to s$ produced by the algorithm, we have $L \in \mathcal{L}$ (even after normalization), according to the closure properties of this set. Since $\mathcal{L}$ and the set of states s is finite (no new state is added) this shows that the construction terminates say with a HA $\mathcal{A}_j$ that will be denoted $\mathcal{A}^*$.

We can also show the following invariant: *every $\mathcal{A}_i$ constructed in the algorithm is deterministic, complete and normalized*. Indeed, assume that $\mathcal{A}_i$ has these properties. If $s' \subseteq s$ no transition is added and the invariant is trivially preserved; hence we can assume now $s' \not\subseteq s$. If another rule $g\big(L^i_{g,s\cup s'}\big) \to s \cup s'$ was in Δ_i it is merged with $g\big(L^i_{g,s} \cap L'\big) \to s \cup s'$ by normalization producing the rule $g\big((L^i_{g,s} \cap L') \cup L^i_{g,s\cup s'}\big) \to s \cup s'$. Hence there is at most one $L^{i+1}_{g,s\cup s'} = (L^i_{g,s} \cap L') \cup L^i_{g,s\cup s'}$ such that $g\big(L^{i+1}_{g,s\cup s'}\big) \to s \cup s' \in \Delta_{i+1}$. Note also that there is at most one $L^{i+1}_{g,s} = L^i_{g,s} \setminus L'$ such that $g\big(L^{i+1}_{g,s}\big) \to s \in \Delta_{i+1}$. It is easy to see (from the fact that $\mathcal{A}_i$ is deterministic and normalized) that $L^{i+1}_{g,s\cup s'}$, $L^{i+1}_{g,s}$, and $L^{i+1}_{g,s''}$, for all $s'' \notin \{s, s'\}$, are pairwise disjoint, hence $\mathcal{A}_{i+1}$ is deterministic. From the facts that $\mathcal{A}_i$ is complete and that $L^i_{g,s} \cap L'$ and $L^i_{g,s} \setminus L'$ form a partition of $L^i_{g,s}$, we deduce that $\mathcal{A}_{i+1}$ is also complete.

We show in [11] that $L(\mathcal{A}^*) = \mathcal{R}^*(L)$. Let us simply sketch the proof here for space reasons.
The proof of the direction $L(\mathcal{A}^*) \subseteq \mathcal{R}^*(L)$ relies on the following lifting lemma.

Lemma 2. *For all $i \geq 0$, $t \in \mathcal{T}(\Sigma, \mathcal{X})$, $\sigma : var(t) \to \mathcal{H}(\Sigma)$ and $\theta : var(t) \to Q^*_{\mathsf{d}}$ such that for all $x \in var(t)$, $x\sigma$ and $x\theta$ have the same length, if $t\theta \xrightarrow[\mathcal{A}_i]{*} s_0 \in Q_{\mathsf{d}}$, and for all $x \in var(t)$, all components $(x\theta)|_j$ of $x\theta$ (state of Q_{d}) and $q \in (x\theta)|_j$, there exists $u \in L(\mathcal{A}_i, q)$ such that $u \xrightarrow[\mathcal{R}]{*} (x\sigma)|_j$, then for all $q' \in s_0$, there exists $v \in L(\mathcal{A}_i, q')$ s. t. $v \xrightarrow[\mathcal{R}]{*} t\sigma$.*

Lemma 2 is proved (see [11]) by induction on i, and, for the induction step, by a second induction on the number of applications of a rule of $\Delta_{i+1} \setminus \Delta_i$ in the reduction $t\theta \xrightarrow[\mathcal{A}_{i+1}]{*} s_0$. Intuitively, every such application corresponds to a rewrite step in $v \xrightarrow[\mathcal{R}]{*} t\sigma$. Now, for the particular case of Lemma 2 where $t \in \mathcal{T}(\Sigma)$, we have that if $t \xrightarrow[\mathcal{A}_i]{*} s_0$, for some i and $s_0 \in Q^{\mathsf{f}}_{\mathsf{d}}$, for all $q^{\mathsf{f}} \in s_0$, where q^{f} is a final state of $\mathcal{A}$, there exists $u \in L(\mathcal{A}, q^{\mathsf{f}}) \subseteq L(\mathcal{A})$ such that $u \xrightarrow[\mathcal{R}]{*} t$. This terminates the proof of the direction $L(\mathcal{A}^*) \subseteq \mathcal{R}^*(L)$.

For the direction $L(\mathcal{A}^*) \supseteq \mathcal{R}^*(L)$, assume that $t \in L(\mathcal{A})$ and that $t \xrightarrow[\mathcal{R}]{*} t'$. We show in [11] that $t' \in L(\mathcal{A}_i)$ for some i by induction on the length of the rewrite sequence. □

Corollary 1. *Ground reachability, ground joinability and regular hedge model-checking are decidable for inverse context-free HRS.*

We present in the next subsections (4.2–4.4) some counter examples showing that relaxing the assumption on $\mathcal{R}$ in Theorem 1 invalidate the result.

4.2 Collapsing Rewrite Rules

Collapsing rules preserve regularity of term languages [16] when the function symbols are ranked. Indeed, in this case, if $\mathcal{R}$ is left-linear and collapsing, a tree automaton (TA) recognizing L can be completed into a TA recognizing $\mathcal{R}^*(L)$ just by the iterated addition of ε-transitions of the form $x\tau \to q$ when there is $\ell \to x \in \mathcal{R}$ and a substitution $\tau : var(\ell) \to Q$ such that $\ell\tau \xrightarrow[\mathcal{A}]{*} q$. When $\mathcal{R}$ is just collapsing (not left-linear), the construction requires determinism and hence is more complicated but the idea is the same [16].

In the case of unranked terms and HA, if we want to follow the principles of the construction of Section 4.1, we need to add *collapsing transitions* and not just ε-transitions. But the addition of collapsing transitions does not preserve HA languages (Proposition 1). The following proposition shows that the above construction is actually not possible for collapsing rewrite rules.

Proposition 3. *$\mathcal{R}^*(L)$ is not a HA language in general when L is a HA language and $\mathcal{R}$ is a linear collapsing HRS.*

Proof. We use the principle of the construction in the proof of Proposition 1. Let $\Sigma = \{f, g, a, b, c\}$, let L be the language of the HA

$$\mathcal{A} = \big(\{q, q_a, q_b, q_{\mathsf{f}}\}, \{q_{\mathsf{f}}\}, \{c \to q, a \to q_a, b \to q_b, g(q_a q q_b) \to q, f(q) \to q_{\mathsf{f}}\}\big)$$

and let $\mathcal{R} = \{g(x) \to x\}$. Assume that $\mathcal{R}^*(L)$ is a HA language. Its intersection with the HA language $\{f(a^*cb^*)\}$ is $\{f(a^ncb^n) \,|\, n \geq 0\}$. It is not a HA language. This contradicts the fact that HA languages are closed under intersection. □

Note that the completion of the above $\mathcal{A}$, following the procedure in the proof of Theorem 1, would add the collapsing transition $q_a q q_b \to q$.

4.3 Flat Linear Rewrite Rules

In the case of ranked terms, it is known [16] that regularity of tree languages is preserved under rewriting with systems with right-linear rules of the form $\ell \to f(u_1, \ldots, u_n)$ where f has arity n and each u_i $(i \leq n)$ is either a ground term or a variable of $var(\ell)$. We call such a rule *flat* if its *lhs* and *rhs* both have depth one. Note that this class of TRS is not captured by the HRS of Theorem 1 (when restricted to ranked terms). The above regularity preservation result is no longer true for unranked terms.

Proposition 4. *$\mathcal{R}^*(L)$ is not a HA language in general when L is a a HA language and $\mathcal{R}$ is a context-free, linear and flat HRS. Moreover, it can be assumed that all the rules of $\mathcal{R}$ are prefix or postfix.*

Proof. Let us consider the context-free HRS $\mathcal{R} = \{g(x) \to g(axb)\}$ of Example 1, and the HA language $L = \{g(c)\}$. The language $\mathcal{R}^*(L) = \{g(a^n c b^n) \mid n \geq 0\}$ is not HA. We can transform the above $\mathcal{R}$ into $\mathcal{R}' = \{g(x) \to g'(ax), g'(y) \to g(yb)\}$ whose rules are prefix or postfix (and linear) and which is such that $\mathcal{R}'^*(L) \cap \mathcal{T}(\{g, a, b\}) = \mathcal{R}^*(L)$. □

Note that the language in the above proof is recognized by a CF-HA. We shall show below (Theorem 2 in Section 5) that context-free HRS like the $\mathcal{R}$ above preserve CF-HA languages.

We show now the stronger result that the closure of a HA language under rewriting with a flat HRS, even linear, is neither HA, nor CF-HA and actually not even recursive.

Proposition 5. *$\mathcal{R}^*(L)$ is not recursive in general when L is a HA language and $\mathcal{R}$ is a linear and flat HRS whose rules contain at most two variables.*

Proof. We reduce the blank accepting problem for TM to ground reachability for an HRS. Let $\mathcal{M}$ be a TM with a tape alphabet Γ and a state set S and let $\Sigma = \Gamma \cup S \cup \{g\}$. A configuration of $\mathcal{M}$ is represented by a term $g(w)$ where w is a word of $\Gamma^* S \Gamma^*$ (the position of the state symbol indicates the position of the head of $\mathcal{M}$ and the rest represents the contents of the tape). We assume, *wlog* unique blank initial and final configurations, respectively c_i and c_f. We consider a HRS $\mathcal{R}$ containing one rule for each transition of $\mathcal{M}$. For instance, $\mathcal{R}$ contains a rule $f(xasy) \to f(xs'a'y)$ corresponding to a transition $s, a \to L, s', a'$ (with $s, s' \in S$ and $a, a' \in \Gamma$) and $f(xasby) \to f(xa'bs'y)$ to the transition $s, a \to R, s'$. The blank tape is accepted by $\mathcal{M}$ iff $c_\mathsf{i} \xrightarrow[\mathcal{R}]{*} c_\mathsf{f}$. □

As a consequence, regular hedge model checking is undecidable for the HRS of Proposition 5, according to the remarks in Section 3.2.

4.4 Rewrite Rules with Flat and One-Variable or Ground Right-Hand-Sides

If we relax the inverse context-free condition, with only one variable allowed in the *rhs* of rules, but possibly with two occurrences, both at depth 1, then the result of Theorem 1, again, is not valid anymore.

Proposition 6. *$\mathcal{R}^*(L)$ is not recursive in general when L is a HA language and $\mathcal{R}$ is a HRS whose rhs of rules are ground or of the form $d(xx)$.*

We reduce in [11], the blank accepting problem for a TM to ground reachability for a HRS with right-ground (but not left-linear) rules and a rule $d(xx) \to d'(xx)$.

5 Closure of Context-Free Hedge Automata Languages

It has been observed [9] that in several cases, one class of word rewrite system preserves regularity and its symmetric class preserves context-free languages. In

this section, we prove a similar result by showing that a restricted case of context-free HRS, *i.e.* of the symmetric version of the systems considered in Section 4, preserve CF-HA languages. We give next some counterexamples showing that the restrictions are necessary for this result.

5.1 Linear Restricted Context-Free Rewrite Rules

We call a HRS $\mathcal{R}$ *restricted context-free* if it is context-free, and moreover, for all rule $f(x) \to r \in \mathcal{R}$, x can occurs in r only at depth at most 1. Note that this definition includes the case of collapsing rules $f(x) \to x$.

Theorem 2. *The closure $\mathcal{R}^*(L)$ of a CF-HA language L under rewriting by a linear restricted context-free HRS $\mathcal{R}$ is a CF-HA language.*

Proof. Let $\mathcal{A}_L = (Q_L, Q_L^{\mathsf{f}}, \Delta_L)$ be a normalized CF-HA recognizing L. We shall construct an extended CF-HA $\mathcal{A}'$ with collapsing transitions (see Section 3.1 for the definition) recognizing $\mathcal{R}^*(L)$. The result follows then from Proposition 1.

First, let us construct for each rule $f(x) \to g(r_1 \dots r_n) \in \mathcal{R}$ and every subterm $r \neq x$ amongst $r_1, \dots, r_n$ (let us denote $rhs(\mathcal{R})$ the set of such subterms) a CF-HA (with collapsing transitions) $\mathcal{A}_r = (Q_r, \emptyset, \Delta_r)$ characterizing the set of ground instances of r. We have in $\mathcal{A}_r$ one state $q_u \in Q_r$ for each non-variable subterm u of r, and a universal state $q_\forall \in Q_r$. Below, for every subterm u of r, we shall write q_u to denote either the state q_u if u is not a variable or $q_\forall$ otherwise. The set of final states of $\mathcal{A}_r$ is left unspecified. It is indeed not relevant to our purpose since $\mathcal{A}_r$ is only used as a part of the CF-HA $\mathcal{A}'$ constructed below. The transition set Δ_r contains one rule $f(q_{u_1} \dots q_{u_n}) \to q_{f(u_1 \dots u_n)}$ for each subterm $f(u_1 \dots u_n)$ of r (as specified above, q_i is $q_\forall$ if u_i is a variable and q_i is a state q_{u_i} otherwise). It contains moreover one collapsing transition $q_\forall^* \to q_\forall$ and one transition rule $f(q_\forall^*) \to q_\forall$ for each $f \in \Sigma$. The states sets Q_r and Q_L are assumed pairwise disjoint. Let $\mathcal{A} := (Q, Q_L^{\mathsf{f}}, \Delta)$ with

$$Q := Q_L \uplus \biguplus_{r \in rhs(\mathcal{R})} Q_r \text{ and } \Delta := \Delta_L \uplus \biguplus_{r \in rhs(\mathcal{R})} \Delta_r.$$

For each $f \in \Sigma$, $q \in Q$, let $L_{f,q}$ be the context-free language in the transition (assumed unique) $f(L_{f,q}) \to q \in \Delta$, and let $\mathcal{G}_{f,q} = (Q, N_{f,q}, I_{f,q}, P_{f,q})$ be a CF grammar generating $L_{f,q}$, with alphabet (set of terminal symbols) Q, set of non terminal symbols $N_{f,q}$, axiom $I_{f,q} \in N_{f,q}$, and set of production rules $P_{f,q}$. The sets of non-terminals $N_{f,q}$ are assumed pairwise disjoint.

We complete the grammars $\mathcal{G}_{f,q}$ with new non-terminals $I'_{f,q}$ and some sets $P'_{f,q}$ of new production rules containing:

i. $I'_{f,q} := I_{f,q}$ for all $f \in \Sigma$, $q \in Q$,
ii. $I'_{g,q} := q_{r_1} \dots q_{r_n} I'_{f,q}\, q_{s_1} \dots q_{s_m}$ for each rule $f(x) \to g(r_1 \dots r_n x s_1 \dots s_m) \in \mathcal{R}$, with $n, m \geq 0$, and $x \notin var(r_1, \dots, r_n, s_1, \dots, s_m)$, and
iii. $I'_{g,q} := q_{r_1} \dots q_{r_n}$ (with $n > 0$), or $I'_{g,q} := \varepsilon$ (with $n = 0$), for each rule $f(x) \to g(r_1 \dots r_n) \in \mathcal{R}$ with $x \notin var(r_1, \dots, r_n)$, if $L(\mathcal{A}, q) \cap f\big(\mathcal{H}(\Sigma)\big) \neq \emptyset$.

Note that in the cases *ii* and *iii* cover all the cases of linear restricted context-free rewrite rules, except the collapsing rules.

Let $N = \bigcup_{f\in\Sigma, q\in Q} \big(N_{f,q} \cup \{I'_{f,q}\}\big)$ and $P = \bigcup_{f\in\Sigma, q\in Q} (P_{f,q} \cup P'_{f,q})$.

Let us clean up these sets: if the language generated by a CF grammar $\big(Q, N, I'_{f,q}, P\big)$ is empty then we remove $I'_{f,q}$ from N and all the productions of P which contain $I'_{f,q}$. We iterate this operation, until there is no remaining non-terminals generating an empty language in N (note that the construction stops since we only remove non-terminals and productions). Let us note N' and P' the sets of non-terminals and productions obtained. For each $f \in \Sigma$, $q \in Q$, let $\mathcal{G}'_{f,q} = \big(Q, N', I'_{f,q}, P'\big)$, and let $L'_{f,q}$ be its language.

Finally, $\mathcal{A}' = (Q, Q^{\mathrm{f}}_L, \Delta')$ is obtained by the addition of collapsing transitions corresponding to the collapsing rewrite rules in $\mathcal{R}$

$$\Delta' = \big\{ f(L'_{f,q}) \to q \;\big|\; f \in \Sigma, q \in Q, L'_{f,q} \neq \emptyset \big\} \cup \{ L'_{f,q} \to q \mid f(x) \to x \in \mathcal{R} \}$$

We show in [11] that $L(\mathcal{A}') = \mathcal{R}^*\big(L(\mathcal{A})\big)$.

The proof of the direction $\subseteq$ is by induction on the number of application of collapsing transitions other than $q^*_\forall \to q_\forall$ in a reduction by $\mathcal{A}'$. For the base case, we need to consider the occurrences of non-terminals $I'_{g,q}$ in the derivations with the grammars $\mathcal{G}'_{f,q}$. Intuitively every occurrence of such $I'_{g,q}$ corresponds to a rewrite step with a context-free rule of $\mathcal{R}$.

The proof of the direction $\supseteq$ is by induction on the length of a rewrite sequence $u \xrightarrow[\mathcal{R}]{*} t$ for $u \in L(\mathcal{A})$. □

Corollary 2. *Reachability and regular hedge model-checking are decidable for linear restricted context-free HRS.*

Proof. The intersection of an CF-HA language and a HA languages is a CF-HA language, and emptiness of CF-HA is decidable. □

It is shown in [18] that the languages of CF-HA are closures of regular tree languages modulo associativity of one or several binary function symbols. Therefore, the above results are also valid for these languages.

5.2 Linear Context-Free Rewrite Rules

Context-free HRS are named after context-free tree grammars, whose production rules have the form $N(x_1, \ldots, x_n) \to r$ where N is a non-terminal of arity n (from a finite set $\mathcal{N}$), $x_1, \ldots, x_n \in \mathcal{X}$ and $r \in \mathcal{T}\big(\Sigma \cup \mathcal{N}, \{x_1, \ldots, x_n\}\big)$. Note that our definition of context-free HRS is restricted to unary non-terminals. However, even for this case of unary non-terminals and right-linear rewrite rules, the result of Theorem 2 cannot be generalized to context-free HRS.

Proposition 7. *$\mathcal{R}^*(L)$ is not a CF-HA language in general when L is a CF-HA language and $\mathcal{R}$ is a linear context-free HRS.*

Proof. Let us consider the context-free HRS: $\mathcal{R} = \{f(x) \to g(f(ax))\}$ and let $L = \{f(c)\}$. The set $\mathcal{R}^*(L)$ is $\{\underbrace{g(g(\ldots g(}_{n} f(a^n c)))) \mid n \in \mathbb{N}\}$.

Using a pumping argument, we can show that it is not a CF-HA language. □

The above counter-example shows the importance for Theorem 2 of the condition, in the definition of restricted context-free HRS, that the variable x in a *lhs* of rule occurs at a *shallow* position in the corresponding *rhs*.

5.3 Restricted Context-Free Rewrite Rules

If we keep the restricted context-free condition (the variable x in the *lhs* of a rule occurs at a shallow position in the corresponding *rhs*) but we drop the linearity condition, we also lose the CF-HA preservation result of Theorem 2.

Proposition 8. *$\mathcal{R}^*(L)$ is not a CF-HA language in general when L is a CF-HA language and $\mathcal{R}$ is a restricted context-free HRS.*

Proof. Let $\mathcal{R} = \{f(x) \to f(xx)\}$ and $L = \{f(a)\}$. We have that $\mathcal{R}^*(L) = \{f(a^n) \mid n = 2^k, k \geq 0\}$ which is not a CF-HA language. Assume indeed that this language is recognized by a CF-HA $(Q, Q^{\mathsf{f}}, \Delta)$. It means that Δ contains a transition $f(L) \to q$ where L is a context-free language of words of Q^* of length 2^k, $k \geq 0$. The image of L under the strictly alphabetic homomorphism which translates every state $q \in Q$ into a is context-free. As it is a one letter language, it is also regular. But it is well known that this language $\{a^n \mid n = 2^k, k \geq 0\}$ is actually not regular. □

5.4 Mixing Inverse CF and Restricted CF Rewrite Rules

We show now that the results of Theorems 1 and 2 cannot be combined. In other terms, for some HRS containing both linear inverse context-free and restricted context-free rules, the set of descendants of a HA language is not a HA language, neither a CF-HA language and even not recursive.

Proposition 9. *$\mathcal{R}^*(L)$ is not recursive in general when L is a HA language and $\mathcal{R}$ is a HRS whose rules are either inverse context-free or restricted context-free and contain only one variable.*

Proof. We reduce the Post Correspondence Problem (PCP). Let us consider an instance $\mathcal{P} = \{\langle u_i, v_i \rangle \mid i \leq n, u_i, v_i \in \Gamma^*\}$ of PCP on an finite alphabet Γ. The problem is to find a sequence $i_1, \ldots, i_k \leq n$ such that $u_{i_1} \ldots u_{i_k} = v_{i_1} \ldots v_{i_k}$

Let $\mathcal{R}$ be an HRS containing a rule $f_0(x) \to f_0(\widetilde{u_i} x v_i)$ for each pair $\langle u_i, v_i \rangle \in \mathcal{P}$ ($\widetilde{u_i}$ is the mirror image of u_i), and two rules $f_0(axa) \to f_1(x)$ and $f_1(axa) \to f_1(x)$ for each $a \in \Gamma$. We assume that f_0, f_1, and c are symbols not in Γ. We have that $f_0(c) \xrightarrow[\mathcal{R}]{*} f_1(c)$ iff $\mathcal{P}$ has a solution. □

Moreover, as we have shown that context-free HRS do not preserve HA languages (Proposition 4), the symmetric also holds for inverse-context-free HRS and CF-HA languages.

Proposition 10. *$\mathcal{R}^*(L)$ is not recursive in general when L is a CF-HA language and $\mathcal{R}$ is an inverse context-free HRS.*

Proof. Let $\mathcal{R}_1$ be the subset of the context-free rewrite rules of the HRS of the above proof of Proposition 9, and $\mathcal{R}_2$ be the subset of the other rules. Note that $\mathcal{R}_2$ is an inverse context-free HRS.

By Theorem 2, $L = \mathcal{R}_1^*\big(\{f_0(c)\}\big)$ is a CF-HA language. Like in the proof of Proposition 9, we have that $f_1(c) \in \mathcal{R}_2^*(L)$ iff the PCP has a solution. Hence, because of the decidability of the membership problem for CF-HA, $\mathcal{R}_2^*(L)$ cannot be a CF-HA language. □

6 Conclusion

We have shown that HA and CF-HA languages are preserved by rewrite closure for interesting classes of non ground hedge rewriting rules. These rules allow us for instance to modify the structure of XML documents when processing them. We plan to extend our results to non ordered unranked trees by considering sheaves automata as in [5] or commutative hedge automata (see [3] for application to process rewrite systems).

Regularity preservation has been studied in the case of ranked terms for transducing term rewriting system, *i.e.* rewrite rules corresponding to transducers rules [21]. A generalization of such classes of TRS to hedge rewriting seems conceptually close to XML transformations [13] and we plan to study the preservation of HA or CF-HA languages w.r.t. to such HRS.

References

1. P. A. Abdulla, B. Jonsson, M. Nilsson, and M. Saksena. A survey of regular model checking. In *Proc. of the 15th Int. Conf. on Concurrency Theory (CONCUR'04)*, vol. 3170 of *LNCS*, pages 35–48. Springer, 2004.
2. A. Bouajjani, B. Jonsson, M. Nilsson, and T. Touili. Regular model checking. In *Proc. of the 12th Int. Conf. on Computer Aided Verification (CAV'00)*, vol. 1855 of *LNCS*, pages 403–418, 2000.
3. A. Bouajjani and T. Touili. On computing reachability sets of process rewrite systems. In *Proc. 16th Int. Conf. Term Rewriting and Applications (RTA'05)*, vol. 3467 of *LNCS*, pages 484–499. Springer, 2005.
4. H. Comon, M. Dauchet, R. Gilleron, F. Jacquemard, D. Lugiez, S. Tison, and M. Tommasi. Tree automata techniques and applications. Available on: `http://www.grappa.univ-lille3.fr/tata`. Last release October, 12th 2007.
5. S. Dal-Zilio and D. Lugiez. XML schema, tree logic and sheaves automata. In *Proc. 14th Int. Conf. Rewriting Techniques and Applications (RTA'03)*, vol. 2706 of *LNCS*, pages 246–263. Springer, 2003.
6. I. Durand and G. Sénizergues. Bottom-up rewriting is inverse recognizability preserving. In *Proc. 18th Int. Conf. Term Rewriting and Applications (RTA'07)*, vol. 4533 of *LNCS*, pages 107-121. Springer, 2007.
7. J. d'Orso and T. Touili. Regular hedge model checking. In *In Proc. of the 4th IFIP Int. Conf. on Theoretical Computer Science (TCS'06)*. IFIP, 2006.

8. R. Gilleron and S. Tison. Regular tree languages and rewrite systems. *Fundamenta Informaticae*, 24(1/2):157–176, 1995.
9. D. Hofbauer and J. Waldmann. Deleting string rewriting systems preserve regularity. *Theor. Comput. Sci.*, 327(3):301–317, 2004.
10. F. Jacquemard. Decidable approximations of term rewriting systems. In *Proc. of the 7th Int. Conf. on Rewriting Techniques and Applications (RTA'96)*, vol. 1103 of *LNCS*, pages 362–376. Springer Verlag, 1996.
11. F. Jacquemard and M. Rusinowitch. Rewrite closure of hedge-automata languages. Research Report LSV-08-05, Laboratoire Spécification et Vérification, ENS Cachan, France, 2007. Available on: `http://www.lsv.ens-cachan.fr/Publis`
12. C. Löding and A. Spelten. Transition graphs of rewriting systems over unranked trees. In *Proc. 32nd Int. Symposium on Mathematical Foundations of Computer Science (MFCS'07)*, vol. 4708 of *LNCS*, pages 67–77, 2007.
13. W. Martens and F. Neven. On the complexity of typechecking top-down XML transformations. Theor. Comput. Sci. Vol 336, N. 1, 2005, pages 153–180.
14. M. Murata. "Hedge Automata: a Formal Model for XML Schemata". `http://www.horobi.com/Projects/RELAX/Archive/hedge_nice.html`, 2000.
15. M. Murata, D. Lee, and M. Mani. Taxonomy of xml schema languages using formal language theory. In *In Extreme Markup Languages*, 2001.
16. T. Nagaya and Y. Toyama. Decidability for left-linear growing term rewriting systems. In *Proc. 10th Int. Conf. on Rewriting Techniques and Applications (RTA'99)*, vol. 1631 of *LNCS*, pages 256–270. Springer Verlag, 1999.
17. H. Ohsaki. Beyond the regularity: Equational tree automata for associative and commutative theories. In *Proc. of CSL'01*, vol. 2142 of *LNCS*. Springer, 2001.
18. H. Ohsaki, H. Seki, and T. Takai. Recognizing boolean closed A-tree languages with membership conditional rewriting mechanism. In *Proc. of the 14th Int. Conf. on Rewriting Techniques and Applications (RTA'03)*, vol. 2706 of *LNCS*, pages 483–498. Springer Verlag, 2003.
19. J. d'Orso and T. Touili. Regular Hedge Model Checking. In Proc. of the 4th IFIP Int. Conf. on Theoretical Computer Science (TCS'06). 2006, IFIP.
20. K. Salomaa. Deterministic Tree Pushdown Automata and Monadic Tree Rewriting Systems. *J. of Comp. and System Sci.*, vol. 37, pages 367–394, 1988.
21. H. Seki, T. Takai, Y. Fujinaka and Y. Kaji. Layered Transducing Term Rewriting System and Its Recognizability Preserving Property. In Proc. of 13th Int. Conf. on Rewriting Techniques and Applications (RTA'02), vol. 2378 of *LNCS*, pages 98-113, 2002.
22. T. Takai, Y. Kaji, and H. Seki. Right-linear finite path overlapping term rewriting systems effectively preserve recognizability. In *Proc. of 11th Int. Conf. on Rewriting Techniques and Applications (RTA'00)*, vol. 1833 of *LNCS*, pages 246–260, 2000.
23. T. Touili. Computing transitive closures of hedge transformations. In *In Proc. 1st Int. Workshop on Verification and Evaluation of Computer and Communication Systems (VECOS'07)*, eWIC Series. British Computer Society, 2007.