Rewrite Closure and CF Hedge Automata

Florent Jacquemard, Michael Rusinowitch

Rewrite Closure and CF Hedge Automata*

Florent Jacquemard[2] and Michael Rusinowitch[1]

[1] INRIA Nancy–Grand Est & LORIA UMR – rusi@loria.fr
615 rue du Jardin Botanique, 54602 Villers-les-Nancy, France.
[2] INRIA Paris–Rocquencourt & Ircam UMR – florent.jacquemard@inria.fr
1 place Igor Stravinsky, 75004 Paris, France.

Abstract. We introduce an extension of hedge automata called bidimensional context-free hedge automata. The class of unranked ordered tree languages they recognize is shown to be preserved by rewrite closure with inverse-monadic rules. We also extend the parameterized rewriting rules used for modeling the W3C XQuery Update Facility in previous works, by the possibility to insert a new parent node above a given node. We show that the rewrite closure of hedge automata languages with these extended rewriting systems are context-free hedge languages.

Introduction

Hedge Automata (HA) are extensions of tree automata to manipulate unranked ordered trees. They appeared as a natural tool to support document validation since the number of children of a node is not fixed in XML documents and the structural information (type) of an XML document can be specified by an HA.

A central problem in XML document processing is *static typechecking*. This problem amounts to verifying at compile time that every output XML document which is the result of a specified query or transformation applied to an input document with a valid input type has a valid output type. However for transformation languages such as the one provided by XQuery Update Facility (XQUF), the output type of (iterated) applications of update primitives are not easy to predict. Another important issue for XML data processing is the specification and enforcement of access policies. A large amount of work has been devoted to secure XML querying. But most of the work focuses on read-only rights, and very few have considered update rights for a model based on XQUF operations [7, 3, 9]. These works have considered the sensitive problem of access control policy inconsistency, that is, *whether a forbidden operation can be simulated through a sequence of allowed operations*. For instance [9] presents a hospital database example where it is forbidden to rename a patient name in a medical file but the same effect can be obtained by deleting this file and inserting a new one. This example illustrates a so-called *local inconsistency* problem and its detection can be reduced to checking the emptiness of a HA language.

In formal verification of infinite state systems several regular model checking approaches represent sets of configurations by regular languages, transitions by

* This work has been partly supported by the ANR ContINT grant INEDIT (2012-15).

rewrite rules and (approximations of) reachable configurations as rewrite closure of regular languages see e.g. [6, 2]. Regular model checking [1] is extended from tree to hedge rewriting and hedge automata in [15], which gives a procedure to compute reachability sets *approximations*. Here we compute exact reachability sets when the configuration sets are represented by context-free hedge automata, hence beyond the regular (HA) ones. These results are interesting for automated verification where reachability sets are not always regular.

To summarize, several XML validation or infinite-state verification problems would benefit from procedures to compute rewrite-closure of hedge languages. We also need decidable formalisms beyond regular tree languages to capture rewrite closures.

Contributions. In [9] we have proposed a model for XML update primitives of XQUF as parameterized rewriting rules of the form: "insert an unranked tree from a regular tree language L as the first child of a node labeled by a". For these rules, we give type inference algorithms, considering types defined by several classes of unranked tree automata. In particular we have considered context-free hedge automata (CFHA, e.g. [8]), a more general class than regular hedge automata and obtained by requiring that the sequences of sibling states under a node to be in a context-free language. In this submission we first introduce a non-trivial extension of context-free hedge languages defined by what we call bidimensional context-free hedge automata (Section 2). This class is more expressive as shown by examples. The class is also shown to be preserved by rewrite closure when applying inverse-monadic rules that are more general than the rules that were considered in [8](Section 3).

Then we extend the parameterized rewriting rules used for modeling XQUF in [9] by the possibility to insert a new parent node above a given node. We show in Section 4 how to compute the rewrite closure of HA languages with these extended rewriting systems. Although the obtained results are more general than [9] the proofs are somewhat simpler thanks to a new uniform representation of vertical and horizontal steps of CFHA. A full version is available at [10].

Related work. [14] presents a static analysis of XML document adaptations, expressed as sequences of XQUF primitives. The authors also use an automatic inference method for deriving the type, expressed as a HA, of a sequence of document updates. The type is computed starting from the original schema and from the XQuery Updates formulated as rewriting rules as in [9]. However differently from our case the updates are applied in parallel in one shot.

1 Preliminaries

We consider a finite alphabet Σ and an infinite set of variables $\mathcal{X}$. The symbols of Σ are generally denoted $a, b, c \ldots$ and the variables x, $y \ldots$ The sets of *hedges* and *trees* over Σ and $\mathcal{X}$, respectively denoted $\mathcal{H}(\Sigma, \mathcal{X})$ and $\mathcal{T}(\Sigma, \mathcal{X})$, are defined recursively as the smallest sets such that: every $x \in \mathcal{X}$ is a tree, if $t_1, \ldots, t_n$ is a finite sequence of trees (possibly empty), then $t_1 \ldots t_n$ is a hedge and if h is a hedge and $a \in \Sigma$, then $a(h)$ is a tree. The empty hedge (case $n \geq 0$ above) is

denoted ε and the tree $a(\varepsilon)$ will be simply denoted by a. We use the operator . to denote the concatenation of hedges. A root (resp. leaf) of a hedge $h = (t_1 \dots t_n)$ is a root node (resp. leaf node, i.e. node without child) of one of the trees $t_1, ..., t_n$. The root node of $a(h)$ is called the *parent* of every root of h and every root of h is called a *child* of the root of $a(h)$.

We will sometimes consider a tree 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 trees (trees without variables) and ground hedges are respectively denoted $\mathcal{T}(\Sigma)$ and $\mathcal{H}(\Sigma)$. The set of variables occurring in a hedge $h \in \mathcal{H}(\Sigma, \mathcal{X})$ is denoted $var(h)$. A hedge $h \in \mathcal{H}(\Sigma, \mathcal{X})$ is called *linear* if every variable of $var(h)$ occurs once in h. A *substitution* σ is a mapping of finite domain from $\mathcal{X}$ into $\mathcal{H}(\Sigma, \mathcal{X})$, whose application (written with postfix notation) is extended homomorphically to $\mathcal{H}(\Sigma, \mathcal{X})$. The set $\mathcal{C}(\Sigma)$ of *contexts* over Σ contains the linear hedges of $\mathcal{H}\big(\Sigma, \{x\}\big)$. The application of a context $C \in \mathcal{C}(\Sigma)$ to a hedge $h \in \mathcal{H}(\Sigma, \mathcal{X})$ is defined by $C[h] := C\{x \mapsto h\}$.

A *hedge rewriting system* (**HRS**) $\mathcal{R}$ over a finite unranked alphabet Σ is a set of *rewrite rules* of the form $\ell \to r$ where $\ell \in \mathcal{H}(\Sigma, \mathcal{X}) \setminus \mathcal{X}$ and $r \in \mathcal{H}(\Sigma, \mathcal{X})$; ℓ and r are respectively called left- and right-hand-side (*lhs* and *rhs*) of the rule. Note that we do not assume the cardinality of $\mathcal{R}$ to be finite. A HRS is called ground, resp. linear, if all its *lhs* and *rhs* of rules are ground, resp. linear.

The rewrite relation $\xrightarrow[\mathcal{R}]{}$ of a HRS $\mathcal{R}$ is the smallest binary relation on $\mathcal{H}(\Sigma, \mathcal{X})$ containing $\mathcal{R}$ and closed by application of substitutions and contexts. In other words, $h \xrightarrow[\mathcal{R}]{} h'$, iff there exists a context C, a rule $\ell \to r$ in $\mathcal{R}$ and a substitution σ such that $h = C[\ell\sigma]$ and $h' = C[r\sigma]$. The reflexive and transitive closure of $\xrightarrow[\mathcal{R}]{}$ is denoted $\xrightarrow[\mathcal{R}]{*}$. Given $L \subseteq \mathcal{H}(\Sigma, \mathcal{X})$ and a HRS $\mathcal{R}$, we define the *rewrite closure* of L under $\mathcal{R}$ as $post^*_{\mathcal{R}}(L) := \{h' \in \mathcal{H}(\Sigma, \mathcal{X}) \mid \exists h \in L, h \xrightarrow[\mathcal{R}]{*} h'\}$.

Example 1. Let us consider the following rewrite rules

$$\mathcal{R} = \{p_0(x) \to a.p_1(x), p_1(x) \to p_2(x).c, p_2(x) \to p_0(b(x)), p_2(x) \to b(x)\}.$$

Starting from $p_0 = p_0(\varepsilon)$, we have the following rewrite sequence $p_0 \to a.p_1 \to a.p_2.c \to a.p_0(b).c \to a.a.p_1(b).c \to a.a.p_2(b).c.c \to a.a.p_0(b(b)).c.c \to \dots$ The trees of the rewrite closure of p_0 under $\mathcal{R}$ which do not contain the symbols p_0, p_1, p_2 is the set of T-patterns of the form $a \dots a.b(\dots b(b)).c \dots c$ with the same number of a, b and c.

2 Bidimensional Context-Free Hedge Automata

A *bidimensional context-free hedge automaton* (**CF^{2}HA**) is a tuple $\mathcal{A} = \langle \Sigma, Q, Q^{\mathsf{f}}, \Delta \rangle$ where Σ is a finite unranked alphabet, Q is a finite set of states disjoint from Σ, $Q^{\mathsf{f}} \subseteq Q$ is a set of final states, and Δ is a set of rewrite rules of one of the following form, where $p_1, \dots, p_n \in Q \cup \Sigma$, $q \in Q$ and $n \geq 0$

$$\begin{array}{rl} p_1(x_1) \dots p_n(x_n) \to q(x_1 \dots x_n) & \text{called } \textit{horizontal} \text{ transitions,} \\ p_1\big(p_2(x)\big) \to q(x) & \text{called } \textit{vertical} \text{ transitions.} \end{array}$$

The move relation $\xrightarrow[\mathcal{A}]{}$ between ground hedges of $\mathcal{H}(\Sigma \cup Q)$ is defined as the rewrite relation defined by Δ. The language of a $\mathsf{CF}^2\mathsf{HA}$ $\mathcal{A}$ in one of its states q, denoted by $L(\mathcal{A}, q)$, is the set of ground hedges $h \in \mathcal{H}(\Sigma)$ such that $h \xrightarrow[\mathcal{A}]{*} q$ (we recall that q stands for $q(\varepsilon)$). A hedge is accepted by $\mathcal{A}$ if there exists $q \in Q^{\mathsf{f}}$ such that $h \in L(\mathcal{A}, q)$. The language of $\mathcal{A}$, denoted by $L(\mathcal{A})$ is the set of hedges accepted by $\mathcal{A}$. We shall also consider below the following kind of transitions, which have the same expressiveness as $\mathsf{CF}^2\mathsf{HA}$.

$$\begin{array}{ll} p_1(\delta_1) \dots p_n(\delta_n) \to q(\delta_1 \dots \delta_n) & n > 0 \\ p_1(p_2(\delta_1)) \to q(\delta_1) & \text{every } \delta_i \text{ is either a variable } x_i \text{ or } \varepsilon \end{array}$$

Example 2. The language of T-patterns over $\Sigma = \{a, b, c\}$, see Example 1, is recognized by $\langle \Sigma, \{q_0, q_1, q_2\}, \{p_0\}, \Delta \rangle$ with $\Delta = \{b(x_1) \to q_0(x_1),\ a.q_0(x_2) \to q_1(x_2),\ q_1(x_1).c \to q_2(x_1),\ q_2(b(x)) \to q_0(x)\}$.

2.1 Related Models

The $\mathsf{CF}^2\mathsf{HA}$ capture the expressiveness of two models of automata on unranked trees: the hedge automaton [11] and the lesser known extension of [12] that we call CFHA. A *hedge automaton* (**HA**), resp. *context-free hedge automaton* (**CFHA**) is a tuple $\mathcal{A} = \langle \Sigma, Q, Q^{\mathsf{f}}, \Delta \rangle$ where Σ, Q and Q^{f} are as above, and the transitions of Δ have the form $a(L) \to q$ where $a \in \Sigma$, $q \in Q$ and $L \subseteq Q^*$ is a regular word language (resp. a context-free word language). The language of hedges accepted is defined as for $\mathsf{CF}^2\mathsf{HA}$, using the rewrite relation of Δ.

The CFHA languages form a strict subclass of $\mathsf{CF}^2\mathsf{HA}$ languages. Indeed every CFHA can be presented as a $\mathsf{CF}^2\mathsf{HA}$ with variable-free transitions of the form

$$p_1 \dots p_n \to q \quad a(q_1) \to q_2 \quad \text{where } a \in \Sigma \text{ and } q_1, q_2 \text{ are states.}$$

It can be shown that the set of T-patterns of Example 2 is not a CFHA language, using a pumping argument on the paths labeled by b.

The HA languages, also called *regular* languages, also form a strict subclass of $\mathsf{CF}^2\mathsf{HA}$ languages. Every HA can indeed be presented as a $\mathsf{CF}^2\mathsf{HA}$ $\mathcal{A} = (\Sigma, Q, Q^{\mathsf{f}}, \Delta)$ with variable-free transitions constrained with a type discipline: $Q = Q_{\mathsf{h}} \uplus Q_{\mathsf{v}}$ and every transition of Δ has one of the forms

$$\varepsilon \to q_{\mathsf{h}} \quad q_{\mathsf{h}}.q_{\mathsf{v}} \to q'_{\mathsf{h}} \quad a(q_{\mathsf{h}}) \to q_{\mathsf{v}} \quad \text{where } q_{\mathsf{h}}, q'_{\mathsf{h}} \in Q_{\mathsf{h}}, q_{\mathsf{v}} \in Q_{\mathsf{v}}, a \in \Sigma.$$

From now on, we shall always consider HA and CFHA presented as $\mathsf{CF}^2\mathsf{HA}$.
The following example shows that $\mathsf{CF}^2\mathsf{HA}$ can capture some CF ranked tree languages. Capturing the whole class of CF RTL would require however a further generalization where permutations of variables are possible in the horizontal transitions of $\mathsf{CF}^2\mathsf{HA}$. Such a generalization is out of the scope of this paper.

Example 3. The language $\{h^n(g(a^n(0), b^n(0))) \mid n \geq 1\}$ is generated by the CF ranked tree grammar [4] with non-terminals A and S (S is the axiom) and productions $A(x_1, x_2) \to h\big(A(a(x_1), b(x_2))\big)$, $A(x_1, x_2) \to g(x_1, x_2)$ and $S \to A(0, 0)$. It is also recognized by the $\mathsf{CF}^2\mathsf{HA}$ with transition rules $a(x_1).b(x_2) \to q(x_1.x_2)$, $g(x_1) \to q_0(x_1)$, $q_0(q(x)) \to q_1(x)$, $h(q_1(x)) \to q_0(x)$ (q_0 is final).

2.2 Properties

The class of $\mathsf{CF}^2\mathsf{HA}$ language is closed under union (direct construction by disjoint union of automata) and not closed under intersection or complementation (because CF word languages are defined by $\mathsf{CF}^2\mathsf{HA}$ without vertical transitions).

Property 4. The membership problem is decidable for $\mathsf{CF}^2\mathsf{HA}$.

Proof. Let $h \in \mathcal{H}(\Sigma)$ be a given hedge and $\mathcal{A}$ be a given $\mathsf{CF}^2\mathsf{HA}$. We assume *wlog* that $\mathcal{A}$ is presented as a set Δ of transitions in the above alternative form $p_1(\delta_1) \dots p_n(\delta_n) \to q(\delta_1 \dots \delta_n)$, with $n > 0$, and $p_1(p_2(\delta_1)) \to q(\delta_1)$.

Moreover, we assume that every transition of the form $q_1(x_1) \to q_2(x_1)$, where q_1 and q_2 are states, has been removed, replacing arbitrarily q_1 by q_2 in the *rhs* of the other transitions. Similarly, we remove $q_1 \to q_2$, replacing arbitrarily *rhs*'s of the form q_1 by q_2. All these transformations increase the size of $\mathcal{A}$ polynomialy.

Then all the horizontal transitions with $n = 1$ have the form $a(\delta_1) \to q(\delta_1)$, with $a \in \Sigma$. It follows that the application of every rule of Δ strictly reduces the measure on hedges defined as pair (# of occurrences of symbols of Σ, # of occurrences of state symbols), ordered lexicographically. During a reduction of h by Δ, each of the two components of the above measure is bounded by the size of h. It follows that the membership $h \in L(\mathcal{A})$ can be tested in PSPACE. □

Property 5. The emptiness problem is decidable in PTIME for $\mathsf{CF}^2\mathsf{HA}$.

Proof. Let $\mathcal{A} = \langle \Sigma, Q, Q^{\mathsf{f}}, \Delta \rangle$. We use a marking algorithm with two marks: h and v. First, for technical convenience, we mark every symbol in Σ with v. Then we iterate the following operations until no marking is possible (note that the marking is not exclusive: some states may have 2 marks h and v).
For all transition $p_1(x_1) \dots p_n(x_n) \to q(x_1 \dots x_n)$ in Δ such that every p_i is marked, if at least one p_i is marked with v, then mark q with v, otherwise mark q with h.
For all transition $p_1\big(p_2(x)\big) \to q(x)$ in Δ such that p_1 is marked v, if p_2 is marked with v, then mark q with v, otherwise, if p_2 is marked with h, then mark q with h.

The number of iterations is at most $2.|Q|$ and the cost of each iteration is linear in the size of $\mathcal{A}$. Then $q \in Q$ is marked with h only iff there exists $h \in \mathcal{H}(\Sigma)$ such that $h \xrightarrow[\Delta]{*} q$, and it is marked with v iff there exists $C[\,] \in \mathcal{C}(\Sigma)$ such that for all $h \in \mathcal{H}(\Sigma)$, $C[h] \xrightarrow[\Delta]{*} q(h)$. Hence $L(\mathcal{A}) = \emptyset$ iff no state of Q^{f} is marked. □

For comparison, for both classes of HA and CFHA, the membership and emptiness problems are decidable in PTIME, the class of HA languages is closed under Boolean operations and the class of CFHA languages is closed under union but not closed under intersection and complementation, see [11, 12, 4].

3 Inverse Monadic Hedge Rewriting Systems

A rewrite rule $\ell \to r$ over Σ is called *monadic* (following [13, 5]) if $r = a(x)$ with $a \in \Sigma$, $x \in \mathcal{X}$, *inverse-monadic* if $r \to \ell$ is monadic and $r \notin \mathcal{X} \cup \{\varepsilon\}$, and

1-childvar if it contains at most one variable and this variable has no siblings in ℓ and r. Intuitively, every finite, linear, inverse-monadic, 1-childvar HRS can be transformed into a HRS equivalent *wrt* reachability whose rules are inverse of transitions of $\mathsf{CF}^2\mathsf{HA}$. It follows that such HRS preserve $\mathsf{CF}^2\mathsf{HA}$ languages.

Example 6. The HRS of Example 1 is linear, inverse-monadic, and 1-childvar. The closure of the language $\{p_0\}$ is the $\mathsf{CF}^2\mathsf{HA}$ language of T-patterns.

Theorem 7. *Let L be the language of $\mathcal{A}_L \in \mathsf{CF}^2\mathsf{HA}$, and $\mathcal{R}$ be a finite, linear, inverse-monadic, 1-childvar* HRS*. There exists an effectively computable $\mathsf{CF}^2\mathsf{HA}$ recognizing $post^*_{\mathcal{R}}(L)$, of size polynomial in the size of $\mathcal{R}$ and $\mathcal{A}_L$.*

Proof. Let $\mathcal{A}_L = \langle \Sigma, Q_L, Q_L^{\mathsf{f}}, \Delta_L \rangle$, we construct a $\mathsf{CF}^2\mathsf{HA}$ $\mathcal{A} = \langle \Sigma, Q, Q^{\mathsf{f}}, \Delta \rangle$. The state set Q contains all the states of Q_L, one state $\underline{h}$ for every non-variable sub-hedge of a *rhs* of rule of $\mathcal{R}$, one state $\underline{a}$ for each $a \in \Sigma$ and one new state $q \notin Q_L$. For each $p \in Q_L \cup \Sigma$, we note $\underline{p} = \underline{a}$ if $p = a \in \Sigma$ and $\underline{p} = p$ otherwise. Let $Q^{\mathsf{f}} = Q_L^{\mathsf{f}}$ and let Δ_0 contain the following transition rules, where $a \in \Sigma$, $t \in \mathcal{T}(\Sigma, \{x\})$ and $h \in \mathcal{H}(\Sigma, \{x\}) \setminus \{\varepsilon\}$.

$$
\begin{array}{llll}
\underline{p_1}(x_1) \dots \underline{p_n}(x_n) \to q(x_1 \dots x_n) & \text{if } p_1(x_1) \dots p_n(x_n) \to q(x_1 \dots x_n) \in \Delta_L & & \\
\underline{p_1}\big(\underline{p_2}(x)\big) \to q(x) & \text{if } p_1\big(p_2(x)\big) \to q(x) \in \Delta_L & & \\
\underline{t}(x).\underline{h} \to \underline{t.h}(x) & \text{if } x \in var(t), \underline{t.h} \in Q & a(x) \to \underline{a}(x) & \\
\underline{t}(x).\underline{h} \to q(x) & \text{if } x \in var(t), \underline{t.h} \notin Q & a(\underline{h}(x)) \to \underline{a(h)}(x) & \text{if } \underline{a(h)} \in Q \\
\underline{t.h}(x) \to \underline{t.h}(x) & \text{if } x \notin var(t), \underline{t.h} \in Q & a(\underline{h}(x)) \to \underline{a}(x) & \text{if } \underline{a(h)} \notin Q \\
\underline{t.h}(x) \to q(x) & \text{if } x \notin var(t), \underline{t.h} \notin Q & a(q(x)) \to \underline{a}(x) &
\end{array}
$$

Finally let $\Delta = \Delta_0 \cup \{\underline{h}(x) \to \underline{a}(x) \mid a(x) \to h \in \mathcal{R}\}$. Let $\ell \in \mathcal{H}(\Sigma)$ be such that $\ell \xrightarrow[\Delta]{*} s(u)$ $(\star)$, with $s \in Q$ and $u \in \mathcal{H}(Q \cup \Sigma)$. We show by induction on the number N of applications of rules of $\Delta \setminus \Delta_0$ in $(\star)$ that there exists $\ell' \in \mathcal{H}(\Sigma)$ such that $\ell' \xrightarrow[\mathcal{R}]{*} \ell$ and moreover, if $s = \underline{h}$, then h matches ℓ', if $s = q$ then ℓ' is not matched by a non-variable subhedge of *rhs* of rule of $\mathcal{R}$ and if $s \in Q_L$, then $\ell' \in L(\mathcal{A}_L, s)$.

If $N = 0$, then the property holds with $\ell' = \ell$ (this can be shown by induction on the length of $(\star)$). If $N > 0$, we can assume that $(\star)$ has the following form.

$$\ell = C[k] \xrightarrow[\Delta_0]{*} C[\underline{h}(v)] \xrightarrow[\Delta \setminus \Delta_0]{} C[\underline{a}(v)] \xrightarrow[\Delta]{} s(u)$$

It follows that h matches k, i.e. there exists w such that $k = h[w]$, and $w \xrightarrow[\Delta_0]{*} v$. Hence $\ell' = C[a(w)] \xrightarrow[\mathcal{R}]{} \ell$, and $\ell' \xrightarrow[\Delta_0]{*} C[a(v)] \xrightarrow[\Delta_0]{} C[\underline{a}(v)] \xrightarrow[\Delta]{} s(u)$. We can then apply the induction hypothesis to ℓ', and immediately conclude for ℓ. □

The following Example 8 illustrates the importance of the 1-childvar and condition in Theorem 7.

Example 8. With the following rewrite rule $a(x) \to c\,a(e\,x\,g)\,d$ we generate from $\{a\}$ the language $\{c^n a(e^n g^n)\, d^n \mid n \geq 1\}$, seemingly not $\mathsf{CF}^2\mathsf{HA}$.

In [8] it is shown that the closure of a HA language under rewriting with a monadic HRS is a HA language. It follows that the backward rewrite closure of a HA language under an inverse-monadic HRS is HA.

4 Update Hedge Rewriting Systems

In this section, we turn to our motivation of studying XQuery Update Facility primitives modeled as parameterized rewriting rules.

Let $\mathcal{A} = \langle \Sigma, Q, Q^{\mathsf{f}}, \Delta \rangle$ be a HA. A hedge rewriting system over Σ parametrized by $\mathcal{A}$ (**PHRS**) is given by a finite set, denoted $\mathcal{R}/\mathcal{A}$, of rewrite rules $\ell \to r$ where $\ell \in \mathcal{H}(\Sigma, \mathcal{X})$ and $r \in \mathcal{H}(\Sigma \uplus Q, \mathcal{X})$ and symbols of Q can only label leaves of r ($\uplus$ stands disjoint union, hence we implicitly assume that Σ and Q are disjoint sets). In this notation, $\mathcal{A}$ may be omitted when it is clear from context or not necessary. The rewrite relation $\xrightarrow[\mathcal{R}/\mathcal{A}]{}$ associated to a PHRS $\mathcal{R}/\mathcal{A}$ is defined as the rewrite relation $\xrightarrow[\mathcal{R}[\mathcal{A}]]{}$ where the HRS $\mathcal{R}[\mathcal{A}]$ is the (possibly infinite) set of all rewrite rules obtained from rules $\ell \to r$ in $\mathcal{R}/\mathcal{A}$ by replacing in r every state $p \in Q$ by a ground hedge of $L(\mathcal{A}, p)$. Note that when there are multiple occurrences of a state p in a rule, each occurrence of p is independently replaced with a hedge in $L(\mathcal{A}, p)$, which can generally be different from one another. Given a set $L \subseteq \mathcal{H}(\Sigma, \mathcal{X})$, we define $post^*_{\mathcal{R}/\mathcal{A}}(L)$ to be $post^*_{\mathcal{R}[\mathcal{A}]}(L)$. We call *updates* parametrized rewrite rules of the following form

$a(x) \to b(x)$		node renaming	(ren)
$a(x) \to a(u_1\, x\, u_2)$	$u_1, u_2 \in Q^*$	addition of child nodes	(ac)
$a(x) \to v_1\, a(x)\, v_2$	$v_1, v_2 \in Q^*$	addition of sibling nodes	(as)
$a(x) \to b\big(a(x)\big)$		addition of parent node	(ap)
$a(x) \to u$	$u \in Q^*$	node replacement/recursive deletion	(rpl)
$a(x) \to x$		single node deletion	(del)

Note that the particular case of (rpl) of rpl with $u = \varepsilon$ corresponds to the deletion of the whole subtree $a(x)$. In the rest of the paper, a PHRS containing only updates will be called update PHRS (**uPHRS**).

4.1 Loop-free uPHRS

In order to simplify the proofs we can reduce to the case where there exists no looping sequence of renaming. This motivates the following definition:

Definition 9. *An* uPHRS $\mathcal{R}/\mathcal{A}$ *is* loopfree *if there exists no sequence* $a_1, \ldots, a_n$ *($n > 1$) such that for all* $1 \leq i < n$, $a_i(x) \to a_{i+1}(x) \in \mathcal{R}$ *and* $a_1 = a_n$.

Given a uPHRS $\mathcal{R}/\mathcal{A}$, we consider the directed graph G whose set of nodes is Σ and containing an edge $\langle a, b \rangle$ iff $a(x) \to b(x)$ is in $\mathcal{R}$. For every strongly connected component in G we select a representative. We denote by $\hat{a}$ the representative of a in its component and more generally by $\hat{h}$ the hedge obtained from $h \in \mathcal{H}(\Sigma)$ by replacing every function symbol a by its representative $\hat{a}$. We define $\hat{\mathcal{R}}$ to be $\mathcal{R}$ where every rule $\ell \to r$ is replaced by $\hat{\ell} \to \hat{r}$ (if the two members get equal we can remove the rule). We define $\hat{\mathcal{A}}$ analogously.

Lemma 10. *Given an* uPHRS $\mathcal{R}/\mathcal{A}$ *the* uPHRS $\hat{R}/\hat{A}$ *is loopfree and for all* $h, h' \in \mathcal{H}(\Sigma)$ *we have* $h \xrightarrow[\mathcal{R}/\mathcal{A}]{*} h'$ *iff* $\hat{h} \xrightarrow[\hat{R}/\hat{A}]{*} \hat{h}'$.

Proof. By induction on the length of derivations. □

4.2 Rewrite Closure

The rest of the section is devoted to the proof of the following theorem of construction of $\mathsf{CF}^2\mathsf{HA}$ for the forward closure by updates.

Theorem 11. *Let $\mathcal{A}$ be a HA over Σ, and L be the language of $\mathcal{A}_L \in$ CFHA, and $\mathcal{R}/\mathcal{A}$ be a loop-free* uPHRS. *There exists an effectively computable CFHA recognizing $post^*_{\mathcal{R}/\mathcal{A}}(L)$, of size polynomial in the size of $\mathcal{R}/\mathcal{A}$ and $\mathcal{A}_L$ and exponential in the size of the alphabet Σ.*

The construction of the CFHA works in 2 steps: construction of an initial automaton and completion loop. We shall use the following notion in order to simplify the proof: a CFHA $\langle \Sigma, Q, Q^{\mathsf{f}}, \Delta \rangle$ is called *normalized* if for all $a \in \Sigma$ and $q \in Q$, there exists one unique state of Q denoted q^a such that $a(q^a) \to q \in \Delta$, and moreover, q^a does neither occur in a left hand side of an horizontal transition of Δ nor in a right hand side of a vertical transition of Δ. With some state renaming, every CFHA $\mathcal{A}$ can be transformed in PTIME into a normalized CFHA $\mathcal{A}'$, of size linear in the size of $\mathcal{A}$, and such that $L(\mathcal{A}') = L(\mathcal{A})$.

Initial automaton. Let $\mathcal{A} = \langle \Sigma, Q_{\mathcal{A}}, Q^{\mathsf{f}}_{\mathcal{A}}, \Delta_{\mathcal{A}} \rangle$ and $\mathcal{A}_L = \langle \Sigma, Q_L, Q^{\mathsf{f}}_L, \Delta_L \rangle$. We assume that the state sets $Q_{\mathcal{A}}$ and Q_L are disjoint.

First, let us merge $\mathcal{A}$ and $\mathcal{A}_L$ into a CFHA $\mathcal{B} = \langle \Sigma, P, P^{\mathsf{f}}, \Gamma \rangle$ obtained by the normalization of $\langle \Sigma, Q_{\mathcal{A}} \uplus Q_L, Q^{\mathsf{f}}_L, \Delta_{\mathcal{A}} \uplus \Delta_L \rangle$. Below, the states of P will be denoted by the letters p or q. Let P_{in} be the subset of states of P of the form q^a (remember that q^a is a state of P uniquely characterized by $a \in \Sigma$, $q \in P$, since $\mathcal{B}$ is normalized). We assume *wlog* that P_{in} and P^{f} are disjoint and that $\mathcal{B}$ is *clean*, *i.e.* for all $p \in P$, $L(\mathcal{B}, p) \neq \emptyset$.

Next, in a preliminary construction step, we transform the initial automaton $\mathcal{B}$ into a CFHA $\mathcal{A}_0 = \langle \Sigma, Q, Q^{\mathsf{f}}, \Delta_0 \rangle$. Let us call *renaming chain* a sequence $a_1, \ldots, a_n$ of symbols of Σ such that $n \geq 1$ for all $1 \leq i < n$, $a_i(x) \to a_{i+1}(x) \in \mathcal{R}$. Since $\mathcal{R}$ is loop-free, the length of every renaming chains is bounded by $|\Sigma|$. The fresh state symbols of Q are defined as extensions of the symbols of $P \setminus P_{\mathsf{in}}$ with renaming chains. We consider two modes for such states: the *push* and *pop* modes, characterized by a chain respectively in superscript or subscript.

$$Q = P \cup \{q_a \mid q^a \in P_{\mathsf{in}}\} \cup \left\{ \begin{array}{l|l} q^{a_1 \ldots a_n} & q \in P \setminus P_{\mathsf{in}}, n \geq 2, \\ q_{a_1 \ldots a_n} & a_1, \ldots, a_n \text{ is a renaming chain} \end{array} \right\}$$

Let $Q^{\mathsf{f}} = P^{\mathsf{f}}$ be the subset of final states. Intuitively, in the state $q^{a_1 \ldots a_n}$, the chain of Σ^+ represents a sequence of renamings, with $\mathcal{R}/\mathcal{A}$, of the parent of the current symbol, starting with a_1 and ending with a_n. Note that the states of P_{in} are particular cases of such states, with a chain of length one. A state $q_{a_1 \ldots a_n}$ will be used below to represent the tree $a_n(q^{a_1 \ldots a_n})$.
The initial set of transitions Δ_0 is defined as follows

$$\begin{aligned} \Delta_0 = \Gamma_h \cup \{q_{a_1} \to q \mid q_{a_1} \in Q\} \\ \cup \{a_n(q^{a_1 \ldots a_n}) \to q_{a_1 \ldots a_n} \mid q^{a_1 \ldots a_n}, q_{a_1 \ldots a_n} \in Q, n \geq 1\} \end{aligned}$$

where Γ_h is the subset of horizontal transitions of Γ. Note that $\mathcal{A}_0$ is not normalized. The following lemma is immediate by construction of Γ and $\mathcal{A}_0$.

Lemma 12. *For all $q \in Q_{\mathcal{A}}$ (resp. $q \in Q_L$) $L(\mathcal{A}_0, q) = L(\mathcal{A}, q)$ (resp. $L(\mathcal{A}_L, q)$).*

Proof. Every vertical transition in Γ has the form $a(q^a) \to q$ and can be simulated by the 2 steps $a(q^a) \to q_a \to q$. Moreover, all the states $q^{a_1 \dots a_n}$ and $q_{a_1 \dots a_n}$ with $n \geq 2$ are empty for $\mathcal{A}_0$. □

For the construction of $\mathcal{A}'$, we shall complete incrementally Δ_0 into Δ_1, Δ_2,... by adding some transition rules, according to a case analysis of the rules of $\mathcal{R}/\mathcal{A}$. For each construction step $i \geq 0$, we let $\mathcal{A}_i = \langle \Sigma, Q, Q^{\mathsf{f}}, \Delta_i \rangle$.

Automata completion. The construction of the sequence (Δ_i) works by iteration of a case analysis of the rewrite rules of $\mathcal{R}/\mathcal{A}$, presented in Table 1. Assuming that Δ_i is the last set built, we define its extension Δ_{i+1} by application of the first case in Table 1 such that $\Delta_{i+1} \neq \Delta_i$. In the rules of Table 1, $a_1, \dots, a_n, b$ are symbols of Σ, and u, v are sequences of $Q_{\mathcal{A}}^*$.

	$\mathcal{R}/\mathcal{A}$ contains	$\Delta_{i+1} = \Delta_i \cup$
(ren)	$a_n(x) \to b(x)$	$\{q^{a_1 \dots a_n} \to q^{a_1 \dots a_n b} \mid q^{a_1 \dots a_n b} \in Q\}$ $\cup \{q_{a_1 \dots a_n b} \to q_{a_1 \dots a_n} \mid q_{a_1 \dots a_n b} \in Q\}$
(ac)	$a_n(x) \to a_n(u\, x\, v)$	$\{u\, q^{a_1 \dots a_n}\, v \to q^{a_1 \dots a_n} \mid q^{a_1 \dots a_n} \in Q\}$
(as)	$a_n(x) \to u\, a_n(x)\, v$	$\{u\, q_{a_1 \dots a_n}\, v \to q_{a_1 \dots a_n} \mid q_{a_1 \dots a_n} \in Q\}$
(ap)	$a_n(x) \to b\big(a_n(x)\big)$	$\{b\big(q_{a_1 \dots a_n}\big) \to q_{a_1 \dots a_n} \mid q_{a_1 \dots a_n} \in Q\}$
(rpl)	$a_n(x) \to u$	$\{u \to q_{a_1 \dots a_n} \mid q_{a_1 \dots a_n} \in Q\}$
(del)	$a_n(x) \to x$	$\{q^{a_1 \dots a_n} \to q_{a_1 \dots a_n} \mid q^{a_1 \dots a_n} \in Q\}$

Table 1. CFHA Completion

Only a bounded number of rules can be added to the Δ_i's, hence eventually, a fixpoint Δ_k is reached, that we will denote Δ'. We also write $\mathcal{A}'$ for $\mathcal{A}_k$.

The following Lemma 13 shows that the automata computations simulate the rewrite steps, *i.e.* that $L(\mathcal{A}') \subseteq post^*_{\mathcal{R}/\mathcal{A}}(L)$. Let us abbreviate $\mathcal{R}/\mathcal{A}$ by $\mathcal{R}$. We use the notation $h \xrightarrow[\mathcal{R}]{a_1 \dots a_n} h'$, for a renaming chain $a_1, \dots, a_n$ ($n \geq 1$), if there exists $h_1, \dots h_n \in \mathcal{H}(\Sigma)$ such that

$$h = a_1(h_1) \xrightarrow[\mathcal{R}]{*} a_1(h_2) \xrightarrow[\mathsf{ren}]{} a_2(h_2) \xrightarrow[\mathcal{R}]{*} \dots \xrightarrow[\mathcal{R}]{*} a_{n-1}(h_n) \xrightarrow[\mathsf{ren}]{} a_n(h_n) \xrightarrow[\mathcal{R}]{*} h'$$

where the reductions denoted $\xrightarrow[\mathsf{ren}]{}$ are rewrite steps with rules of $\mathcal{R}/\mathcal{A}$ of type (ren), applied at the positions of a_1,..., a_n, and all the other rewrite steps (denoted $\xrightarrow[\mathcal{R}]{*}$) involve no rule of type (ren).

Lemma 13 (Correctness). *For all $h \in \mathcal{H}(\Sigma)$,*

i. if $h \xrightarrow[\mathcal{A}']{*} q_{a_1 \dots a_n}$, with $n \geq 1$, then there exists $h_1 \in \mathcal{H}(\Sigma)$ such that $a_1(h_1) \xrightarrow[\mathcal{B}]{*} q$ and $a_1(h_1) \xrightarrow[\mathcal{R}]{a_1 \dots a_n} h$,

ii. *if* $h \xrightarrow[\mathcal{A}']{*} q^{a_1 \dots a_n}$, *with* $n \geq 1$, *then there exists* $h_1 \in \mathcal{H}(\Sigma)$ *such that* $h_1 \xrightarrow[\mathcal{B}]{*} q^{a_1}$, *and* $a_1(h_1) \xrightarrow[\mathcal{R}]{a_1 \dots a_n} a_n(h)$,
iii. *if* $h \xrightarrow[\mathcal{A}']{*} q \in P \setminus P_{\mathsf{in}}$, *then there exists* $h' \in \mathcal{H}(\Sigma)$ *such that* $h' \xrightarrow[\mathcal{B}]{*} q$ *and* $h' \xrightarrow[\mathcal{R}]{*} h$.

Proof. (sketch) Let $s \in Q$ be such that $h \xrightarrow[\mathcal{A}']{*} s$ and let us call ρ this reduction. With a commutation of transitions, we can assume that ρ has the following form,

$$\rho : h = t_1 \dots t_m \xrightarrow[\mathcal{A}']{*} \underbrace{s_1 \dots s_m \xrightarrow[\mathcal{A}']{*} s}_{\rho_0}$$

where $t_1, \dots, t_m \in \mathcal{T}(\Sigma)$, $s_1, \dots, s_m \in Q$, and for all $1 \leq i \leq m$, $t_i \xrightarrow[\mathcal{A}']{*} s_i$, and the last step of this reduction involves a vertical transition $a(q^{a_1 \dots a_n}) \to s_i$ or $b(q_{a_1 \dots a_n}) \to s_i$. The proof is by induction on the length of ρ.

The shortest possible ρ has 2 steps: $h = t_1 = a(\varepsilon) \xrightarrow[\mathcal{A}_0]{} a(q^a) \xrightarrow[\mathcal{A}_0]{} q = s$ and (iii) holds immediately with $h' = h$, by Lemma 12.

For the induction step, we consider the length of ρ_0. If $|\rho_0| = 0$, we have necessarily $m = 1$, and the reduction ρ has one of the two following forms ($\boldsymbol{v} \in Q^*$).

$$h = t_1 = b(h') \xrightarrow[\mathcal{A}']{*} b(\boldsymbol{v}) \xrightarrow[\mathcal{A}']{*} b(q_{a_1 \dots a_n}) \xrightarrow[\mathcal{A}']{} q_{a_1 \dots a_n} = s_1 = s \quad (1)$$

$$h = t_1 = a_n(h') \xrightarrow[\mathcal{A}']{*} a_n(\boldsymbol{v}) \xrightarrow[\mathcal{A}']{*} a_n(q^{a_1 \dots a_n}) \xrightarrow[\mathcal{A}_0]{} q_{a_1 \dots a_n} = s_1 = s \quad (2)$$

In the case (1), assume that the vertical transition $b(q_{a_1 \dots a_n}) \to q_{a_1 \dots a_n}$ has been added to $\mathcal{A}'$ because $\mathcal{R}/\mathcal{A}$ contains a rule $a_n(x) \to b\big(a_n(x)\big)$. By induction hypothesis (i) applied to the sub-reduction $h' \xrightarrow[\mathcal{A}']{*} q_{a_1 \dots a_n}$, there exists $h_1 \in \mathcal{H}(\Sigma)$ such that $a_1(h_1) \xrightarrow[\mathcal{B}]{*} q$, and $a_1(h_1) \xrightarrow[\mathcal{R}]{a_1 \dots a_n} h'$. It follows in particular that there exists h_n such that $a_n(h_n) \xrightarrow[\mathcal{R}]{*} h'$, and using the above (ap) rewrite rule, $a_n(h_n) \xrightarrow[\mathcal{R}]{} b\big(a_n(h_n)\big) \xrightarrow[\mathcal{R}]{*} b(h') = h$. Therefore, $a_1(h_1) \xrightarrow[\mathcal{R}]{a_1 \dots a_n} h$ and (i) holds for h and s.

In the case (2), by induction hypothesis (ii) applied to the sub-reduction $h' \xrightarrow[\mathcal{A}']{*} q^{a_1 \dots a_n}$, there exists $h_1 \in \mathcal{H}(\Sigma)$ such that $h_1 \xrightarrow[\mathcal{B}]{*} q^{a_1}$, hence $a_1(h_1) \xrightarrow[\mathcal{B}]{*} q$, and $a_1(h_1) \xrightarrow[\mathcal{R}]{a_1 \dots a_n} a_n(h') = h$. Therefore (i) holds for h and s.

Assume now that $|\rho_0| > 0$, and let us analyze the horizontal transition rule used in the last step of ρ_0. In order to comply with spaces restrictions, we will present only one significant case in this extended abstract (see [10] for the other cases).

Case (ac). The last step of ρ_0 uses $u\, q^{a_1 \dots a_n}\, v \to q^{a_1 \dots a_n}$ and this transition has been added to Δ' because $\mathcal{R}/\mathcal{A}$ contains a rule $a_n(x) \to a_n(u\, x\, v)$, with $u, v \in Q^*_{\mathcal{A}}$. In this case, the reduction ρ has the following form,

$$h = \ell\, h'\, r \xrightarrow[\mathcal{A}']{*} u\, q^{a_1 \dots a_n}\, v \xrightarrow[\mathcal{A}']{} q^{a_1 \dots a_n} = s \quad (3)$$

where $\ell \xrightarrow[\mathcal{A}']{*} u$, $h' \xrightarrow[\mathcal{A}']{*} q^{a_1 \dots a_n}$, and $r \xrightarrow[\mathcal{A}']{*} v$. By induction hypothesis (ii) applied to $h' \xrightarrow[\mathcal{A}']{*} q^{a_1 \dots a_n}$, there exists h_1 such that $h_1 \xrightarrow[\mathcal{B}]{*} q^{a_1}$ and $a_1(h_1) \xrightarrow[\mathcal{R}]{a_1 \dots a_n}$

$a_n(h')$, and by induction hypothesis (iii) applied to $\ell \xrightarrow[\mathcal{A}']{*} u$ (resp. $r \xrightarrow[\mathcal{A}']{*} v$), and by Lemma 12, there exists $\ell' \in \mathcal{H}(\Sigma)$ (resp. $r' \in \mathcal{H}(\Sigma)$) such that $\ell' \xrightarrow[\mathcal{A}]{*} u$ (resp. $r' \xrightarrow[\mathcal{A}]{*} v$) and $\ell' \xrightarrow[\mathcal{R}]{*} \ell$ (resp. $r' \xrightarrow[\mathcal{R}]{*} r$). It follows that $a_n(h') \xrightarrow[\mathcal{R}]{} a_n(\ell'\, h'\, r') \xrightarrow[\mathcal{R}]{*} a_n(\ell\, h'\, r) = a_n(h)$. Hence $a_1(h_1) \xrightarrow[\mathcal{R}]{a_1 \ldots a_n} a_n(h)$ and (ii) holds for h and s. □

Corollary 14. $L(\mathcal{A}') \subseteq post^*_{\mathcal{R}/\mathcal{A}}(L)$

Proof. By definition of Q^{f}, $h \in L(\mathcal{A}')$ iff $h \xrightarrow[\mathcal{A}']{*} q \in P^{\mathsf{f}} = Q^{\mathsf{f}}_L$, and $P^{\mathsf{f}} \subseteq P \setminus P_{\mathsf{in}}$. By Lemma 13, case (iii), it follows that $h \in post^*_{\mathcal{R}/\mathcal{A}}(L(\mathcal{B}, q)) \subseteq post^*_{\mathcal{R}/\mathcal{A}}(L)$. □

Lemma 15 (Completeness). *For all $h \in \mathcal{H}(\Sigma)$ and $s \in Q$, if $h \xrightarrow[\mathcal{A}_0]{*} s$ and $h \xrightarrow[\mathcal{R}]{*} h'$, then $h' \xrightarrow[\mathcal{A}']{*} s$.*

The proof is by induction on the length of the rewrite sequence $h \xrightarrow[\mathcal{R}]{*} h'$ (see [10]). As another consequence of the result of [8] on the rewrite closure of HA languages under monadic HRS, the backward closure of a HA language under an uPHRS is HA.

The rules of type (ren), (as), (ap) and (rpl) can be easily simulated by the HRS of Theorem 11. In particular, the parameters' semantics can be simulated using ground rewrite rules (with such rules, a symbol can generate a HA language). The rules (ac) are not 1-childvar and the rules (del) is not inverse-monadic.

Example 8 shows the problems that can arise when combining in one single rewrite rule two rules of the form (as) and (ac), forcing synchronization of two updates. Note that the rule $a(x) \to c\, a(e\, x\, g)\, d$ of this example can be simulated by the 2 rules $a(x) \to c\, a'(x)\, d$ and $a'(x) \to a(e\, x\, g)$. The former rule is of the type of Theorem 11 (it combines types (as) and (ren)). The latter (which is not 1-varchild) combines types (ac) and (ren). This shows that such combinations can also lead to the behavior exposed in Example 8.

Future Works

As for future works on $\mathsf{CF}^2\mathsf{HA}$ languages several directions deserve to be followed. A first direction might be to derive pumping properties for these classes of languages.

A second direction would be to look for an analogous of Parikh characterization for the number of different symbols occurring in the hedges of given $\mathsf{CF}^2\mathsf{HA}$ languages. One may define and study HRS with counting constraints on horizontal and vertical paths.

Finally, it would be is worth investigating the parallel rewriting of [14], on all a-positions, since it is closer to the semantics of XQUF, and get an analogous of Theorem 11 for the parallel rewrite closure.

References

1. Bouajjani, A., Jonsson, B., Nilsson, M., Touili, T.: Regular model checking. In: Proc. of the 12th Int. Conf. on Computer Aided Verification. LNCS, vol. 1855, pp. 403–418 (2000)
2. Bouajjani, A., Touili, T.: On computing reachability sets of process rewrite systems. In: Proceedings 16th Int. Conf. Term Rewriting and Applications (RTA). Lecture Notes in Computer Science, vol. 3467, pp. 484–499. Springer (2005)
3. Bravo, L., Cheney, J., Fundulaki, I.: ACCOn: checking consistency of XML write-access control policies. In: Proc. 11th Int. Conf. on Extending Database Technology (EDBT). ACM Int. Conf. Proc. Series, vol. 261, pp. 715–719. ACM (2008)
4. Comon, H., Dauchet, M., Gilleron, R., Löding, C., Jacquemard, F., Lugiez, D., Tison, S., Tommasi, M.: Tree automata techniques and applications. Available on: `http://tata.gforge.inria.fr/` (2007), release October, 12th 2007
5. Coquide, J.L., Dauchet, M., Gilleron, R., Vagvolgyi, S.: Bottom-up tree pushdown automata : Classification and connection with rewrite systems. Theoretical Computer Science 127, 69–98 (1994)
6. Feuillade, G., Genet, T., Viet Triem Tong, V.: Reachability Analysis over Term Rewriting Systems. Journal of Automated Reasoning 33 (3-4), 341–383 (2004)
7. Fundulaki, I., Maneth, S.: Formalizing XML access control for update operations. In: Proc. of the 12th ACM symposium on Access control models and technologies (SACMAT). pp. 169–174. ACM (2007)
8. Jacquemard, F., Rusinowitch, M.: Closure of Hedge-automata languages by Hedge rewriting. In: Proc. of the 19th RTA. LNCS, vol. 5117, pp. 157–171. Springer (2008)
9. Jacquemard, F., Rusinowitch, M.: Rewrite-based verification of XML updates. In: Proc. of the 12th ACM SIGPLAN Int. Symp. on Principles and Practice of Declarative Programming (PPDP). pp. 119–130. ACM (2010)
10. Jacquemard, F., Rusinowitch, M.: Rewrite Closure and CF Hedge Automata. `http://hal.inria.fr/hal-00752496` (2012), (long version)
11. Murata, M.: Hedge automata: a formal model for XML schemata. `http://www.xml.gr.jp/relax/hedge_nice.html` (2000)
12. Ohsaki, H., Seki, H., Takai, T.: Recognizing boolean closed a-tree languages with membership conditional rewriting mechanism. In: Proc. of the 14th RTA. Lecture Notes in Computer Science, vol. 2706, pp. 483–498. Springer Verlag (2003)
13. Salomaa, K.: Deterministic tree pushdown automata and monadic tree rewriting systems. J. Comput. Syst. Sci. 37(3), 367–394 (1988)
14. Solimando, A., Delzanno, G., Guerrini, G.: Automata-based Static analysis of XML Document Adaptation. In: Proceedings 3d International Symposium on Games, Automata, Logics and Formal Verification. vol. 96, pp. 85–98 (2012)
15. Touili, T.: Computing transitive closures of hedge transformations. In: Proc. 1st International Workshop on Verification and Evaluation of Computer and Communication Systems (VECOS). eWIC Series, British Computer Society (2007)