Unranked tree rewriting and effective closures of languages

Florent Jacquemard\textsuperscript{1} \quad Michael Rusinowitch\textsuperscript{2}

\textsuperscript{1}Ircam, INRIA Paris-Rocquencourt
\textsuperscript{2}LORIA, INRIA Nancy

June 27, 2013

IFIP WG 1.6 meeting \quad – \quad Eindhoven
### Term and Unranked Tree Rewriting

<table>
<thead>
<tr>
<th>Ranked</th>
<th>Unranked</th>
</tr>
</thead>
<tbody>
<tr>
<td>finite ranked signature</td>
<td>finite alphabet</td>
</tr>
<tr>
<td>$\Sigma = { \bot, \top : 0, \neg : 1, \vee, \wedge : 2 }$</td>
<td>$\Sigma = { \bot, \top, \neg, \vee, \wedge }$</td>
</tr>
<tr>
<td>term $: = a(\text{term}_1, \ldots, \text{term}_n)$ $a \in \Sigma_n$</td>
<td>tree $: = a(\text{hedge})$ $a \in \Sigma$</td>
</tr>
<tr>
<td>$\wedge(t_1, \wedge(t_2, t_3))$</td>
<td>hedge $: = \text{tree}^*$ $\wedge(t_1 t_2 t_3)$</td>
</tr>
</tbody>
</table>

### Rewrite Rules

- terms $\times$ terms
- hedges $\times$ hedges

### Substitutions

- variables $\rightarrow$ terms
- variables $\rightarrow$ hedges

---

Unranked Tree Rewriting:
[Löding Spelten 07 MFCS], [Touili 07 VECOS]
Unranked Tree Rewriting

Node renaming

\[ \text{entree}(x) \rightarrow \text{entry}(x) \]

- the rule can be applied to any node labeled by \text{entree}
- the variable \( x \) is instantiated by a finite sequence of trees (hedge)
Unranked Tree Rewriting

Insert first

$\text{book}(x) \rightarrow \text{book} (\text{entry } x)$
Unranked Tree Rewriting

Insert into

\[ \text{book}(x_1 \ x_2) \rightarrow \text{book}(x_1 \ \text{entry} \ x_2) \]
Unranked Tree Rewriting

Insert after

\[ n(x) \rightarrow n(x) \text{ph}(1) \]
Unranked Tree Rewriting

\[
\text{favorite}(x) \rightarrow x
\]
Collapsing Unranked Tree Rewriting Rule

\[ \text{favorite}(x) \rightarrow x \]

"delete a single node labeled by favorite"
"move the trees in the sequence of children \( x \) up to the position of the deleted node."

- useful for constructing security views of documents
Motivations and Rewrite Closure
Motivations

Analysis of programs and protocols
  ▶ Tree Regular Tree Model Checking

XML processing and verification
  ▶ transformations (XSLT), static type checking
  ▶ update primitives (XQuery UF), reachability
  ▶ consistency of R/W access control policies

Rhythm trees
  ▶ tree structured representation of music notation
  ▶ simplification of rhythms, decision of equivalences.
Verification of Infinite State Systems

Tree Regular Tree Model Checking [Abdulla et al 2002 CAV]:

- configurations are represented by trees,
- transitions by rewrite rules / tree transducers,
- verification by reachability analysis.

\[ \mathcal{R}^* (L_{init}) \cap L_{error} = \emptyset \]

higher-order functional programs : [Jones, Andersen 2007], [Kochems Ong 2011 RTA] (collecting semantics)

multithreaded recursive programs:

- term model: [Seidl 2009 IIA] [Bouajjani et al 2000], [Genet, Tong 2001], [Genet, Rusu 2010].
- unranked tree model: [Bouajjani Touili 2005 RTA].
Static Typechecking

\[ \mathcal{T} : \text{tree transducer rewrite system (tree transformation model)} \]

Typechecking:

\( \mathcal{T} \) always converts valid input data from a tree set \( L_{\text{in}} \)
into valid output data from a tree set \( L_{\text{out}} \)

\[ \mathcal{T}(L_{\text{in}}) \subseteq L_{\text{out}} \]

\[ L_{\text{in}} \cap \mathcal{T}^{-1}(L_{\text{out}}) = \emptyset \]

Composition (Boolean closure)

Forward rewrite closure

Backward rewrite closure

Decision procedures
Consistency of R/W Access Control Policies

[Fundulaki Maneth], [Bravo et al, ACCOn] atomic r/w access (updates) modeled by rewrite rules

An ACP is defined by two rewrite systems:

- $\mathcal{R}_+$: authorized operations,
- $\mathcal{R}_-$: forbidden operations.

It is

- inconsistent if one rule of $\mathcal{R}_-$ can be simulated through a sequence of rules of $\mathcal{R}_+$.
- locally inconsistent for a tree $t$ if there exists $u$ such that $t \xrightarrow{\mathcal{R}_-} u$ and $t \xrightarrow{\mathcal{R}_+} u$, i.e. $\mathcal{R}_-(t) \cap \mathcal{R}_+(t) \neq \emptyset$. 
When $R^*(L)$ or $(R^{-1})^*(L)$ is effectively regular (for $L$ regular)
- RTMC, typechecking, local inconsistency reduce to tree automata decision problems
  [Milo Suciu Vianu 03 JCSS], [Tosawa 2001]

Otherwise
- approximate
  [Touili Bouajani RTA 05], [Genet, Rusu 2010]...
- extend the tree automata model
Term Rewriting

Ranked Tree Automata
Ranked Tree Automata

\[ \mathcal{A} = \langle \Sigma, Q, F, \Delta \rangle \]

- \( \Sigma \) ranked alphabet, every symbol has a fixed arity
- \( Q \) finite state set,
- \( F \subseteq Q \) final states,
- \( \Delta \) set of transitions \( a(q_1, \ldots, q_n) \rightarrow q \)
  - \( a \in \Sigma \), \( a \) of arity \( n \)
  - \( q_1, \ldots, q_n, q \in Q \)

Consider \( \Delta \) as a TRS over \( \Sigma \cup Q \).

**Language**

\[
L(\mathcal{A}, q) = \{ t \mid t \xrightarrow{\Delta}^* q \}
\]

\[
L(\mathcal{A}) = \bigcup_{q \in F} L(\mathcal{A}, q)
\]

Regular sets of terms = ranked tree automata languages
Regularity Preservation

[Salomaa 1988]

Linear and right-flat rewrite rules preserve regular languages.

I.e. for all linear and right-flat TRS $R$, the forward closure $R^*(L)$ of a regular language $L$ is regular.

Ranked tree automata completion:

- Given $A = \langle \Sigma, Q, F, \Delta_0 \rangle$ and $R$ over $\Sigma$,
- Compute $A^*$ such that $L(A^*) = R^*(L(A))$.

By superposition of $A$’s transitions into $R$’s rules

For $a(b(x_1), c(x_2)) \rightarrow c(x_2, x_1) \in R$

Also for right-linear and right-flat TRS [Nagaya, Toyama 2002].
Unranked Tree Rewriting

Hedge Automata
Hedge Automata

\[ \mathcal{A} = (\Sigma, Q, F, \Delta) \] with

- \( \Sigma \) ranked alphabet, every symbol has a fixed arity
- \( Q \) finite state set,
- \( F \subseteq Q \) final states,
- \( \Delta \) set of transitions \( a(L) \rightarrow q \)
  - \( a \in \Sigma, q \in Q, \)
  - \( L \) regular language over \( Q^* \).

\( \Delta \) represents the (possible infinite) rewrite system

\[ \Delta_\infty = \{ a(q_1 \ldots q_n) \rightarrow q \mid a(L) \rightarrow q \in \Delta, q_1 \ldots q_n \in L \} \]

Language
\[ L(\mathcal{A}, q) = \{ t \mid t \xrightarrow{\Delta_\infty}^* q \} \]
\[ L(\mathcal{A}) = \bigcup_{q \in F} L(\mathcal{A}, q) \]

Regular sets of unranked tree = HA languages

\( \equiv \) regular term set via binary encodings
Hedge Automata and XML Typing

DTD

```
book
  | entry*
  |   
  | name
  |   | phone*
  |   |   | email*
  |   | first
  |   |   | char*
  |   | last
  |   |   | char*
  |   | char*
```

corresponding HA

```
book(q_e*) → q_b
entry(q_n q_h* q_m*) → q_e
name(q_f q_l) → q_n
first(p*) → q_f
last(p*) → q_l
phone(p*) → q_h
email(q_u q_d) → q_m
user(p*) → q_u
dom(p*) → q_d
a → q
b → q
...```

---

**DTD**

- **book**
  - **entry**
    - **name**
    - **phone**
      - **email**
        - **first**
          - **char**
        - **last**
          - **char**
    - **char**
    - **char**

**corresponding HA**

- **book(q_e*) → q_b**
- **entry(q_n q_h* q_m*) → q_e**
- **name(q_f q_l) → q_n**
- **first(p*) → q_f**
- **last(p*) → q_l**
- **phone(p*) → q_h**
- **email(q_u q_d) → q_m**
- **user(p*) → q_u**
- **dom(p*) → q_d**
- **a → q**
- **b → q**
- **...**
Hedge Automata: Main Properties

- Boolean closures of recognized languages
- Membership $t \in L(\mathcal{A})$ is decidable
  - PTIME when horizontal languages are presented by NFAs
  - NP-complete when horizontal languages are presented by alternating automata
- Emptiness: $L(\mathcal{A}) = \emptyset$ is decidable
  - PTIME when horizontal languages are presented by NFAs
  - PSPACE-complete when horizontal languages are presented by alternating automata
HA Preservation

[JR 2008 RTA]
inverse CF rewrite rules $\ell \rightarrow a(x), x \in \text{vars}(\ell)$ preserve HA.

\[
\text{book}\left(\text{entry}(\text{name}(y) y_1) \text{ entry}(\text{name}(y) y_2) x\right) \rightarrow \text{book}(x)
\]

exponential construction (needs determinization)
non-trivial combination

- [Touili 07] (unranked trees)
  HA completion, for linear HRS, approximative.
- [Nagaya Toyama 99] (terms)
  TA completion, non-left-linear TRS.
HA Preservation

[JR 2008 RTA]

inverse CF rewrite rules \( \ell \rightarrow a(x), x \in \text{vars}(\ell) \) preserve HA.

exponential construction (needs determinization)

- \( A_0 := \) determination of the given HA (subset constr.)
- complete according to this schema

\[
\begin{align*}
\text{if } \ell \sigma & \xrightarrow{\star} S \\
\text{replace } a(L') & \rightarrow S' \quad \in A_i \\
\text{by } a(L' \cap x\sigma) & \rightarrow S' \cup S \quad \in A_{i+1} \\
\text{and } a(L' \setminus x\sigma) & \rightarrow S' \quad \in A_{i+1}
\end{align*}
\]

with substitution \( \sigma : \text{vars}(\ell) \rightarrow 2^{(2^Q)^*} \)

- invariant: determinism.
- fixpoint: rewrite closure of \( L(A_0) \).
Parametrized Rewrite Systems

Given a fixed HA $\mathcal{B}$ with state set $Q$
parametrized rewrite rule : symbols of $Q$ allowed in leaves of rhs

\[
\begin{align*}
\text{book}(x) & \rightarrow \text{book}(q_e x) \quad \text{(insert first)} \\
\text{book}(x_1 x_2) & \rightarrow \text{book}(x_1 q_e x_2) \quad \text{(insert into)} \\
\text{name}(x) & \rightarrow \text{name}(x) q_h \quad \text{(insert after)} \\
\text{name}(x) & \rightarrow q_1 \ldots q_n \quad \text{(replace/delete), } n \geq 0
\end{align*}
\]

semantics of parametrized rewrite system $\mathcal{R}$:

- (possibly infinite) rewrite system $\mathcal{R}/\mathcal{B}$ obtained by replacement of $q$ in rhs by a (ground) tree in $L(\mathcal{B}, q)$.
- Different occurrences of $q$ can be replaced by different trees.

see also [Gilleron 91 STACS], [Löding 02 STACS]
# Forward and Backward Closure of Update Primitives

> [JR 2010 PPDP]

<table>
<thead>
<tr>
<th>Original</th>
<th>Update</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a(x) \rightarrow b(x)$</td>
<td></td>
<td>ren</td>
</tr>
<tr>
<td>$a(x) \rightarrow a(px)$</td>
<td></td>
<td>ins$_{\text{first}}$</td>
</tr>
<tr>
<td>$a(x) \rightarrow a(xp)$</td>
<td></td>
<td>ins$_{\text{last}}$</td>
</tr>
<tr>
<td>$a(x_1x_2) \rightarrow a(x_1px_2)$</td>
<td></td>
<td>ins$_{\text{into}}$</td>
</tr>
<tr>
<td>$a(x) \rightarrow q_1$</td>
<td></td>
<td>rpl$_1$</td>
</tr>
<tr>
<td>$a(x) \rightarrow \varepsilon$</td>
<td></td>
<td>del</td>
</tr>
<tr>
<td>$a(x_1) \rightarrow p_a(x)$</td>
<td></td>
<td>ins$_{\text{before}}$</td>
</tr>
<tr>
<td>$a(x) \rightarrow a(x)p$</td>
<td></td>
<td>ins$_{\text{after}}$</td>
</tr>
<tr>
<td>$a(x) \rightarrow q_1 \ldots q_n$</td>
<td></td>
<td>rpl</td>
</tr>
<tr>
<td>$a(x) \rightarrow x$</td>
<td></td>
<td>dels</td>
</tr>
</tbody>
</table>

- preserve HA
- polynomial construction
- do not preserve HA
- inverse-preserve HA
- exponential construction
HA preservation: by transformation of the horizontal languages (NFA) \( L \) in rules \( a(L) \rightarrow q \).

ex: for \( a(x) \rightarrow b(x) \in \mathcal{R} \) and \( a(L) \rightarrow q \in \mathcal{A} \), add \( b(L) \rightarrow q \) to \( \mathcal{A} \).

ex: for \( a(x) \rightarrow a(p \ x) \in \mathcal{R} \), and \( a(L) \rightarrow q \in \mathcal{A} \), add a loop labeled with \( p \) on the initial state of \( L \).

HA inverse-preservation: automata completion as before.
Unranked Tree Rewriting

CF Hedge Automata
Non preservation of HA

linear & flat rewrite rules do not preserve HA.

RE rewrite closure

\[ g(x \ q \ a \ y) \rightarrow g(x \ b \ q' \ y) \]

CF rewrite closure (simultaneously insert first and last)

\[ c(x) \rightarrow c(a \ x \ b) \]

CF rewrite closure (simultaneously insert and rename)

\[ c_0(x) \rightarrow c_1(a \ x), \ c_1(x) \rightarrow c_0(x \ b) \]

flat and right-ground rewrite rules do not preserve HA.

\[ a(x) \rightarrow b \ a \ c \]

collapsing rewrite rules do not preserve HA.
Non preservation of HA (collapse)

collapsing rewrite rules do not preserve HA.

\[ R = \{ c(x) \rightarrow x \}, \quad L_{\text{in}} = \{ \} \text{ (regular)} \]

\[ R^* (L_{\text{in}}) \cap a^* b^* = \{ a^n b^n \mid n \geq 0 \} \]

The rewrite closure is a CF-HA language.

- all these examples are in contrast with the case of terms.
- an extension of HA is needed.
CF Hedge Automata  

\[ \mathcal{A} = (\Sigma, Q, F, \Delta) \] with

- \( \Sigma \) ranked alphabet, every symbol has a fixed arity
- \( Q \) finite state set,
- \( F \subseteq Q \) final states,
- \( \Delta \) set of transitions \( a(L) \rightarrow q \)
  - \( a \in \Sigma, q \in Q, \)
  - \( L \) is a CF language over \( Q^* \).

\( \Delta \) represents the (possible infinite) rewrite system

\[ \Delta_\infty = \{ a(q_1 \ldots q_n) \rightarrow q \mid a(L) \rightarrow q \in \Delta, q_1 \ldots q_n \in L \} \]

\( \text{HA} \equiv \) ranked tree automata

\( \text{CF-HA} \equiv \) ranked tree automata modulo A
CF Hedge Automata: Main Properties

- closure of recognized languages under union
- no closure under intersection and complementation
- closure under intersection with HA

- membership $t \in L(A)$ is decidable in PTIME
- emptiness: $L(A) = \emptyset$ is decidable in PTIME
  (when horizontal languages are presented by CFG)
## Forward Closure of Update Primitives

**[JR 2010 PPDP]**

<table>
<thead>
<tr>
<th>Transformation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( a(x) \rightarrow b(x) )</td>
<td>ren</td>
</tr>
<tr>
<td>( a(x) \rightarrow a(q \ x) )</td>
<td>( \text{ins}_{\text{first}} )</td>
</tr>
<tr>
<td>( a(x) \rightarrow a(x \ q) )</td>
<td>( \text{ins}_{\text{last}} )</td>
</tr>
<tr>
<td>( a(x_1 \ x_2) \rightarrow a(x_1 \ q \ x_2) )</td>
<td>( \text{ins}_{\text{into}} )</td>
</tr>
<tr>
<td>( a(x) \rightarrow q_1 )</td>
<td>( \text{rpl}_1 )</td>
</tr>
<tr>
<td>( a(x) \rightarrow \epsilon )</td>
<td>del</td>
</tr>
<tr>
<td>( a(x) \rightarrow p \ a(x) )</td>
<td>( \text{ins}_{\text{before}} )</td>
</tr>
<tr>
<td>( a(x) \rightarrow a(x) \ q )</td>
<td>( \text{ins}_{\text{after}} )</td>
</tr>
<tr>
<td>( a(x) \rightarrow q_1 \ldots q_n )</td>
<td>rpl</td>
</tr>
<tr>
<td>( a(x) \rightarrow x )</td>
<td>dels</td>
</tr>
</tbody>
</table>

**Preservation:**

- **HA**
- **CF-HA**
- Polynomial construction
- Inverse-preserve **HA**
- Exponential construction
Verification of Inconsistency for Rule Based ACP

An ACP $\langle \mathcal{R}_+, \mathcal{R}_- \rangle$ is locally inconsistent for $t$ if there exists $u$ such that $t \xrightarrow{\mathcal{R}_-} u$ and $t \xrightarrow{\mathcal{R}_+} u$.

equivalently $\mathcal{R}_-(t) \cap \mathcal{R}_+(t) \neq \emptyset$.

[JR 2010 PPDP]
Local inconsistency is decidable in PTIME for update rules.

proof:
- compute a HA recognizing $\tau_- = \{u \mid t \xrightarrow{\mathcal{R}_-} u\}$
- compute a CF-HA recognizing $\tau_+ = \mathcal{R}_+^*(\{t\})$
- check that $\tau_- \cap \tau_+ \neq \emptyset$. 
[JR 2008 RTA]

restricted CF rewrite rules $a(x) \rightarrow r$ with

1. $r$ linear
2. $x$ at depth $\leq 1$ in $r$

preserve CF-HA.

\[ a(x) \rightarrow b(x), \quad a(x) \rightarrow a(entry(name(Homer)),x) \]

polynomial construction
finite transformation of CF grammars for horizontal language.
Non-preservation of CF-HA

**flat and non-linear CF rewrite rules do not preserve CF-HA.**

\[ g(x) \rightarrow g(xx) \quad g(a) \rightarrow g(a^{2^k}) \]

**non-shallow CF rewrite rules do not preserve CF-HA.**

\[ a(x) \rightarrow b(a(x\ e)) \quad a(c) \rightarrow b(\ldots b(a(c\ e^n))) \]
Unranked Tree Rewriting

CF$^2$ Hedge Automata
transitions are rewrite rules!

\[ \mathcal{A} = (\Sigma, Q, F, \Delta) \] with

- \( \Sigma \) ranked alphabet, every symbol has a fixed arity
- \( Q \) finite state set,
- \( F \subseteq Q \) final states,
- \( \Delta \) set of transitions of the form,

\[ a \rightarrow q \]
\[ p(x) \rightarrow q(x) \]
\[ p_1(x) p_2 \rightarrow q(x) \text{ horizontal transitions} \]
\[ p_1 p_2(x) \rightarrow q(x) \]
\[ p_1(p_2(x)) \rightarrow q(x) \text{ vertical transitions} \]

where \( p, p_1, p_2 \in Q \cup \Sigma, q \in Q \).

equivalently, \( x \) can be set to \( \varepsilon \).
\( \langle \{a, b, c\}, \{q_0, q_1, q_2\}, \{q_0\}, \Delta \rangle \) with

\[
\Delta = \left\{ \begin{array}{ccc}
  b(x) & \rightarrow & q_2(x), \\
  a q_1(x) & \rightarrow & q_0(x), \\
  q_2(x) c & \rightarrow & q_1(x), \\
  q_0(b(x)) & \rightarrow & q_2(x)
\end{array} \right\}
\]

...not a CF-HA language.
CF^2HA ⊃ HA and CF-HA

**CF-HA** with variable-free transitions

\[
\begin{align*}
q_1 & \rightarrow q \\
a(q_1) & \rightarrow q_2 \\
a & \rightarrow q
\end{align*}
\]

where \( a \in \Sigma \) and \( q_1, q_2, q \) are states.

**HA** with \( Q = Q_h \uplus Q_v \), transitions:

\[
\begin{align*}
q_h & \rightarrow q' \\
a(q_h) & \rightarrow q_h \upharpoonright q_v \\
a & \rightarrow q_h \upharpoonright q_v
\end{align*}
\]

where \( a \in \Sigma, q_h, q'_h \in Q_h, q_v \in Q_v \).
CF$^2$HA: properties

**Membership** is decidable in PTIME for CF$^2$HA.

Dynamic programming procedure (CKY like)

**Emptiness** is decidable in PTIME for CF$^2$HA.

State marking with 2 marks.
CF^2HA Preservation

[JR 2013 LATA]

CF rewrite rule $a(x) \rightarrow r$ where $x$
- is the only variable in $r$,
- has at most 1 occurrence in $r$,
- has no siblings in $r$.

preserve CF^2HA.

Example: T-patterns

$$\mathcal{R} = \left\{ q_0(x) \rightarrow a \; q_1(x), \quad q_1(x) \rightarrow q_2(x) \; c, \quad q_2(x) \rightarrow q_0(b(x)) \right\}$$

$$q_0 \xrightarrow{\mathcal{R}} a \; q_1 \xrightarrow{\mathcal{R}} a \; q_2 \; c \xrightarrow{\mathcal{R}} a \; q_0(b) \; .c \xrightarrow{\mathcal{R}} a \; a \; q_1(b) \; c \xrightarrow{\mathcal{R}} a \; a \; q_2(b) \; c \; c \xrightarrow{\mathcal{R}} a \; a \; q_0(b(b)) \; c \; c \xrightarrow{\mathcal{R}} \ldots$$
CF$^2$HA Preservation

Given a CF$^2$HA $\mathcal{A}_{in} = \langle \Sigma, Q_{in}, F_{in}, \Delta_{in} \rangle$, we construct a CF$^2$HA $\mathcal{A}$ with state set

$$Q = Q_{in} \uplus \{h \mid h \text{ non-var subhedge of a rhs of } R\} \uplus \{a \mid a \in \Sigma\} \uplus \{q\}$$

and transitions

- $q_1(x_1) \ldots q_n(x_n) \rightarrow p(x_1 \ldots x_n)$
- $q_1(q_2(x)) \rightarrow p(x)$
- $q_1(x_1) \ldots q_n(x_n) \rightarrow p(\bar{x}) \in \Delta_{in}$
- $q_1(q_2(x)) \rightarrow p(x) \in \Delta_{in}$

- $t(x) h \rightarrow t h(x)$ if $x \in t, t h \in Q$
- $t(x) h \rightarrow q(x)$ if $x \in t, t h \notin Q$
- $t h(x) \rightarrow t h(x)$ if $x \notin t, t h \in Q$
- $t h(x) \rightarrow q(x)$ if $x \notin t, t h \notin Q$
- $h(x) \rightarrow a(x)$ if $a(x) \rightarrow h \in R$
1-childvar Condition

\[ R = \{ a(x) \rightarrow c \ a(e \ x \ g) \ d \} \]

\[ R^* (\{a\}) = \{ c^n \ a(e^n g^n) \ d^n | n \geq 1 \} \]

seemingly not \( \text{CF}^2 \text{HA} \).
Forward Closure of $\text{Ext}^d$ Update Primitives [JR 2013 LATA]

Given a fixed HA $\mathcal{B}$ with state set $Q$

- $a(x) \rightarrow b(x)$ node renaming (ren)
- $a(x) \rightarrow a(u_1 x u_2)$ $u_1, u_2 \in Q^*$ insertion of child nodes (ins.c)
- $a(x) \rightarrow v_1 a(x) v_2$ $v_1, v_2 \in Q^*$ insertion of sibling nodes (ins.s)
- $a(x) \rightarrow b(a(x))$ insertion of a parent node (ins.p)
- $a(x) \rightarrow u$ $u \in Q^*$ node replacement (rpl)
- $a(x) \rightarrow x$ recursive deletion (del)

preserve CF-HA.

The proof uses the CF$^{2}$HA presentation of CF-HA.
Loop-Free Update Rewrite Systems

\( R \) loop-free if there exists no sequence \( a_1, \ldots, a_n \) (\( n > 1 \)) such that for all \( 1 \leq i < n \), \( a_i(x) \rightarrow a_{i+1}(x) \in R \) and \( a_1 = a_n \).

Transformation of an update rewrite system \( R \) into a loop-free update rewrite system \( \hat{R} \) by selecting a representative \( \hat{a} \) of \( a \) in every \( a \)'s loop and suppressing loops.

In the construction of an automaton for \( R^* \), it is sufficient to consider the rewrite closure by \( \hat{R} \).
**Closure under Update Rewrite Systems**

Rewrite closure of CF-HA is CF-HA for every loop-free update rewrite system.

**Initialize**, given CF-HA $\mathcal{A}_{in} = \langle \Sigma, Q_{in}, F_{in}, \Delta_{in} \rangle$

\[
\Delta_0 = \Delta_{in} \cup \{q_{a_1} \rightarrow q\} \cup \{a_n(q^{a_1...a_n}) \rightarrow q_{a_1...a_n} \mid a_1, \ldots, a_n \text{ is a renaming chain}\}
\]

**Completion**

<table>
<thead>
<tr>
<th>$\mathcal{R}$ contains</th>
<th>$\Delta_{i+1} = \Delta_i \cup$</th>
</tr>
</thead>
<tbody>
<tr>
<td>(ren) $a_n(x) \rightarrow b(x)$</td>
<td>${q^{a_1...a_n} \rightarrow q^{a_1...a_nb} \mid q^{a_1...a_nb} \in Q}$</td>
</tr>
<tr>
<td>(ins.c) $a_n(x) \rightarrow a_n(u \ x \ v)$</td>
<td>$\cup {q_{a_1...a_nb} \rightarrow q_{a_1...a_n} \mid q_{a_1...a_nb} \in Q}$</td>
</tr>
<tr>
<td>(ins.s) $a_n(x) \rightarrow u \ a_n(x) \ v$</td>
<td>${u \ q^{a_1...a_n} \ v \rightarrow q^{a_1...a_n} \mid q^{a_1...a_n} \in Q}$</td>
</tr>
<tr>
<td>(ins.p) $a_n(x) \rightarrow b(a_n(x))$</td>
<td>${u \ q_{a_1...a_n} \ v \rightarrow q_{a_1...a_n} \mid q_{a_1...a_n} \in Q}$</td>
</tr>
<tr>
<td>(rpl) $a_n(x) \rightarrow u$</td>
<td>${b(q_{a_1...a_n}) \rightarrow q_{a_1...a_n} \mid q_{a_1...a_n} \in Q}$</td>
</tr>
<tr>
<td>(del) $a_n(x) \rightarrow x$</td>
<td>${u \rightarrow q_{a_1...a_n} \mid q_{a_1...a_n} \in Q}$</td>
</tr>
<tr>
<td></td>
<td>${q^{a_1...a_n} \rightarrow q_{a_1...a_n} \mid q^{a_1...a_n} \in Q}$</td>
</tr>
</tbody>
</table>
Synchronized rename and insert

\[ R = \{ a(x) \rightarrow c\ a(e\ x\ g)\ d \} \]

\[ R^* (\{a\}) = \{ c^n\ a(e^n g^n)\ d^n \mid n \geq 1 \} \]

\[ R' = \begin{cases} 
  a(x) & \rightarrow \ c\ a'(x)\ d, \\
  a'(x) & \rightarrow \ a(e\ x\ g) 
\end{cases} \quad \text{inv-monadic, 1-childvar}
\]

\[ \in (\text{ins.s}) + (\text{ren}) \]

\[ \notin 1\text{-childvar} \]

\[ \in (\text{ins.c}) + (\text{ren}) \]
Conclusion

- decidable models CF-HA, CF^2HA of unranked ordered tree recognizers extending hedge automata
- captures forward/backward rewrite closure under
  - various families of unranked tree rewrite systems with (inverse-) CF rules
  - parametric rewrite systems modeling update primitives

Perspectives

- case of unranked unordered trees
- counting constraints on horizontal and vertical paths

...Thank You!