Automates d'arbres

http://list.mpri.master.univ-paris7.fr/wws/info/cours-1-18

cours-1-18@mpri.master.univ-paris7.fr

Cours: Florent Jacquemard
TD: Ștefan Ciobâcă

INRIA Saclay & LSV (CNRS-ENS Cachan)

florent.jacquemard@inria.fr
ciobaca@lsv.ens-cachan.fr

12 janvier 2010

Bibliography

- **TATA book** (Tree Automata Theory and Application)
  Hubert Comon, Max Dauchet, Remi Gilleron, Florent Jacquemard, Denis Lugiez, Christof Löding, Sophie Tison, Marc Tommasi
  `http://tata.gforge.inria.fr`

  chapter 8 :
  - Automata for Unranked Trees

Arbres finis orientés ordonnés étiquettés par des symboles de fonction sans arité

Unranked ordered labeled trees

- ranked terms = first order terms over a signature
  - every symbols has a fixed arity

$\longrightarrow$ automated deduction, program analysis, evaluation strategies in functional languages...

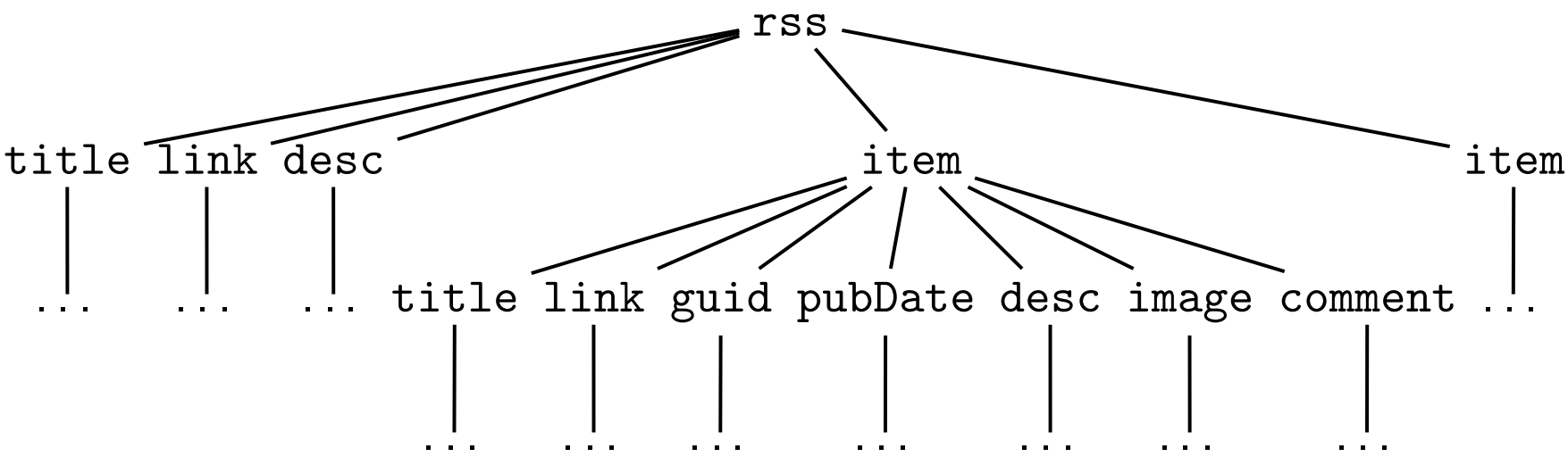
- unranked terms = finite trees (directed, rooted) labelled over a finite alphabet
  - one node can have arbitrarily (though finitely) many childrens
  - the number of children of a node does not depend on its label

$\longrightarrow$ Web data

Web data (XML Document)

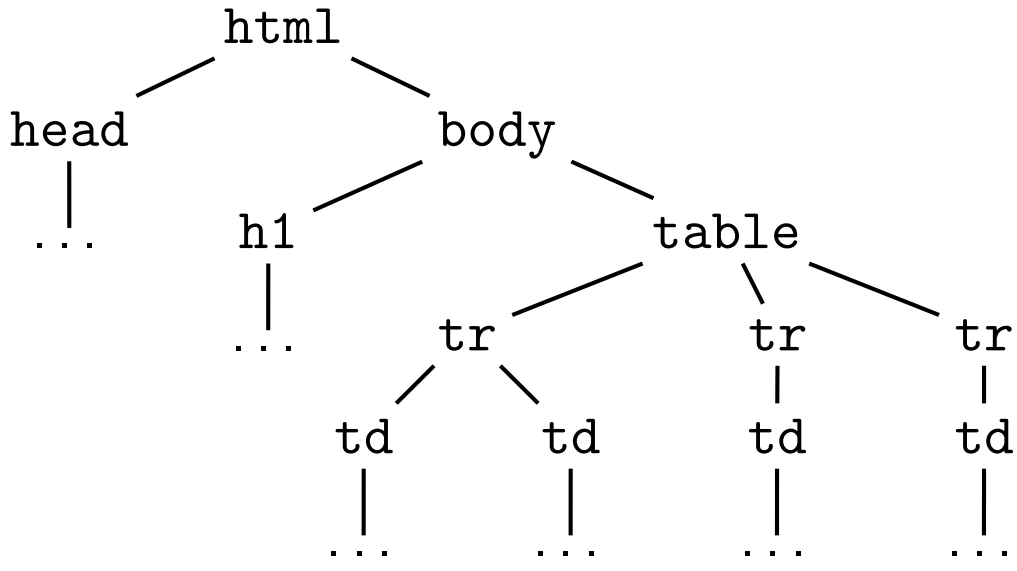
```
<rss version="2.0">
    <title>Mon blog</title>
    <link>http://myblog.blogspot.com</link>
    <description>bla bla bla</description>
    <item>
      <title>Concert</title>
      <link>http://myblog.blogspot.com/me/Mon blog/...</link>
      <guid>5f7da0aa-a593-4a2e</guid>
      <pubDate>Fri, 21 Mar 2009 14:40:02 +0100</pubDate>
      <description>...</description>
      <image href="..."></image>
      <comment link="..." count="0" enabled="0">...</comment>
    </item>
    <item>
      <title>Journée de surf</title>
      ...
    </item>
</rss>
```

Web data


rss
title
link
desc
item
item
...
...
...
title
link
guid
pubDate
desc
image
comment
...
...
...
...
...
...
...
...

HTML Document

```
<html>
 <head>...</head>
 <body>
  <h1>...</h1>
  <table>
   <tr>
    <td>...</td>
    <td>...</td>
   </tr>
   <tr>
    <td>...</td>
   </tr>
   <tr>
    <td>...</td>
   </tr>
  </table>
 </body>
</html>
```



XML Documents

- class of documents with a predefined structure (valid documents)

  ex :

```
<table>
  <tr> <td> c11 </td> <td> c12 </td> </tr>
  <tr> <td> c21 </td> <td> c22 </td> </tr>
</table>
```

- defined by DTD, XML schema... = tree language

XML Documents

ranked	unranked
tree automata	schemas (DTD, XML schema)
membership	validation
emptiness	query satisfiability
inclusion	schema entailment
tree transducers, rewrite systems	transformation languages (XSLT)
rewrite closure	type inference

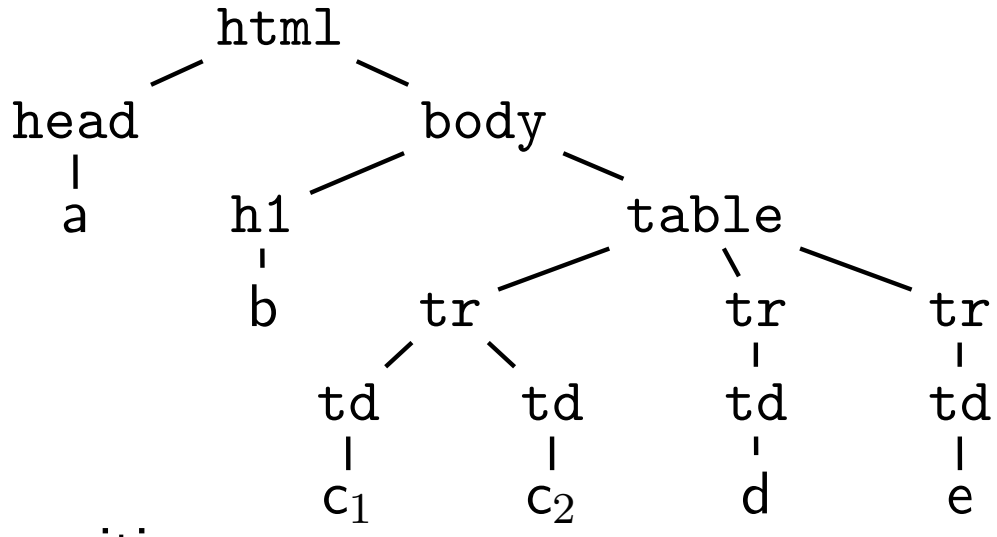
Unranked ordered trees

- Σ is a finite alphabet.

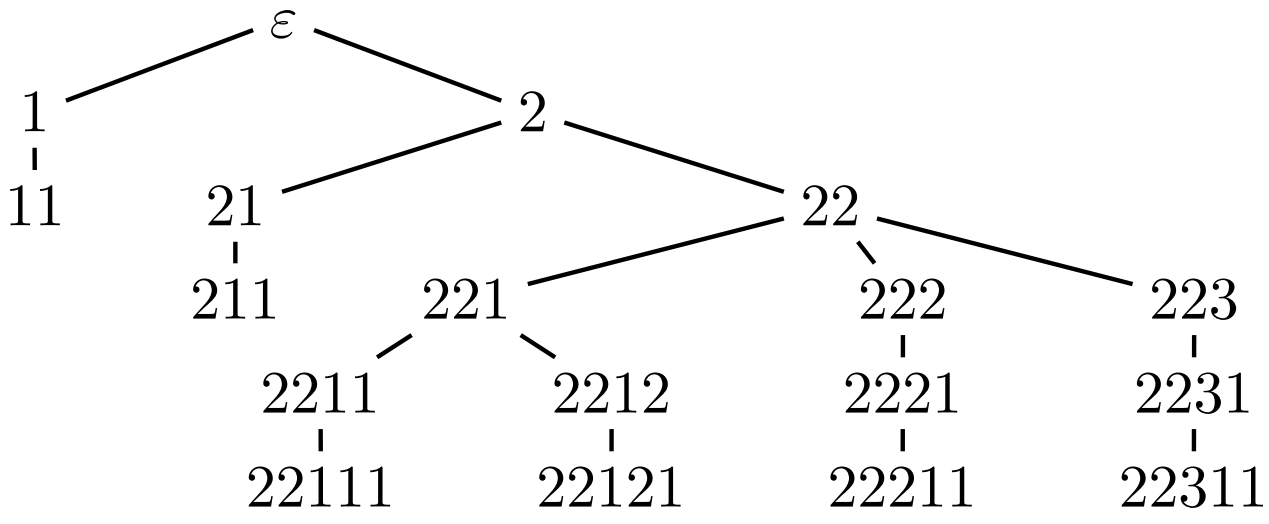
$$\begin{array}{rcll} \text{tree} & := & a(\text{hedge}) & (a \in \Sigma) \\ \text{hedge} & := & \text{tree}^* & \end{array}$$

- a hedge can be empty. $a()$ is denoted by a.
- The set of all unranked ordered trees over Σ is denoted $\mathcal{U}(\Sigma)$.
- The set of hedges over Σ is denoted $\mathcal{H}(\Sigma)$.
- set of positions $\subset \mathbb{N}^*$: as for terms.

Example : tree of $\mathcal{U}(\Sigma)$



positions :



Example : language $\subseteq \mathcal{U}(\Sigma)$

- $\Sigma = \{a, b\}$.
- $L :=$ terms of $\mathcal{U}(\Sigma)$
  - height 1,
  - root is labelled by a,
  - even number of leaves, all leaves labelled by b.
- $L = \{a, a(bb), a(bbbb), ...\}$.
- finite description : $L = a\big((bb)^*\big)$

Hedge Automata (HA)

Definition : Hedge Automata

A Hedge Automaton (HA) over an alphabet Σ is a tuple $\mathcal{A} = (\Sigma, Q, Q^{\mathsf{f}}, \Delta)$ where Q is a finite set of states, $Q^{\mathsf{f}} \subseteq Q$ is the subset of final states and Δ is a set of transition rules of the form : $a(L) \to q$ with $a \in \Sigma$ and $L \subseteq Q^*$ is a regular language.

A run of $\mathcal{A}$ on $t \in \mathcal{U}(\Sigma)$ is a tree $r \in \mathcal{U}(Q)$ such that

- r and t have the same domain,
- for all $p \in \mathcal{P}os(t)$, with $t(p) = a$, $r(p) = q$, there exists $a(L) \to q \in \Delta$ such that $r(p1) \dots r(pn) \in L$, where n is the number of successors of p in $\mathcal{P}os(t)$.

The run r is accepting (succesful) iff $r(\varepsilon) \in Q^{\mathsf{f}}$.

HA Languages

- language of $\mathcal{A}$: $L(\mathcal{A})$ is the set of terms on which there exists an accepting run of $\mathcal{A}$,
- language of $\mathcal{A}$ in state $q \in Q$: $L(\mathcal{A}, q)$ is the set of terms t such that there exists a run r of $\mathcal{A}$ on t with $r(\varepsilon) = q$,
- $L(\mathcal{A}) = \bigcup_{q \in Q_f} L(\mathcal{A}, q)$.

- equivalently, $L(\mathcal{A}, q)$ is the smallest set of terms $a(t_1, \ldots, t_n) \in \mathcal{U}(\Sigma)$ $(n \geq 0)$ such that there exists a transition $a(L) \to q$, and states $q_1, \ldots, q_n \in Q$ with $t_i \in L(\mathcal{A}, q_i)$ s.t. $i \leq n$ and $q_1 \ldots q_n \in L$.

HA language : Example 1

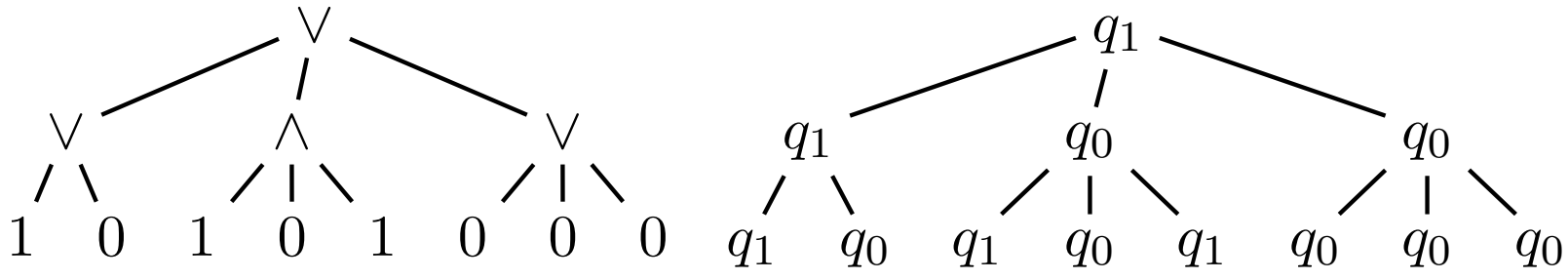
- $\Sigma = \{a, b\}$.
- $L :=$ terms of $\mathcal{U}(\Sigma)$
  - height 1,
  - root is labelled by a,
  - even number of leaves, all leaves labelled by b.
- $L = \{a, a(bb), a(bbbb), ...\}$.
- finite description : $L = L(\mathcal{A})$ with
  $\mathcal{A} := \big(\Sigma, \{q_a, q_b\}, \{q_a\}, \{b \to q_b, a((q_b q_b)^*) \to q_a\}\big)$.

Boolean expressions with variadic $\wedge$ and $\vee$

- $\Sigma = \{\wedge, \vee, 0, 1\}$,
- states $\{q_0, q_1\}$,
- transitions :

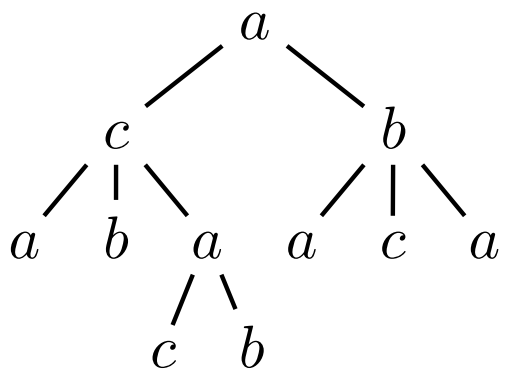
$$
\begin{array}{rcl}
0 & \to & q_0 \\
\wedge(q_1^*\, q_0\, (q_0 \mid q_1)^*) & \to & q_0 \\
\vee(q_0\, q_0^*) & \to & q_0
\end{array}
\qquad
\begin{array}{rcl}
1 & \to & q_1 \\
\wedge(q_1\, q_1^*) & \to & q_1 \\
\vee(q_0^*\, q_1\, (q_0 \mid q_1)^*) & \to & q_1
\end{array}
$$

- example : Boolean expression and associated run



HA language : Example 3

- $\Sigma = \{a, b, c\}$.
- $L :=$ terms of $\mathcal{U}(\Sigma)$
  - with 2 b's at positions p_1 and p_2, and
  - one c on the smallest common ancestor of p_1 and p_2.

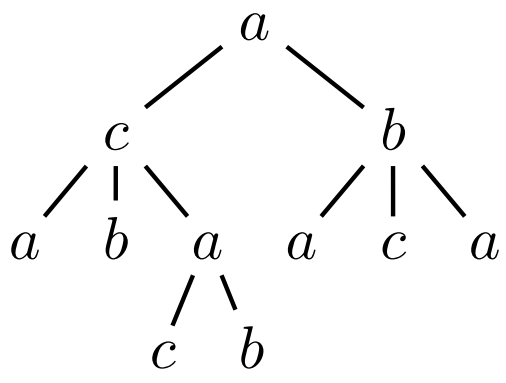


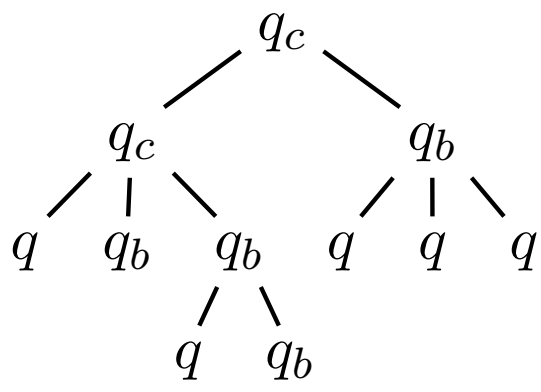
HA language : Example 3

- $\Sigma = \{a, b, c\}$.
- $L :=$ terms of $\mathcal{U}(\Sigma)$
  - with 2 b's at positions p_1 and p_2, and
  - one c on the smallest common ancestor of p_1 and p_2.

$\mathcal{A} = (\Sigma, Q, Q^{\mathsf{f}}, \Delta)$, avec $Q = \{q, q_b, q_c\}$, $Q^{\mathsf{f}} = \{q_c\}$, $\Delta =$

$$
\begin{array}{llll}
a(Q^*) \to q & a(Q^*q_bQ^*) \to q_b & a(Q^*q_cQ^*) \to q_c \\
b(Q^*) \to q_b & c(Q^*q_bQ^*) \to q_b & b(Q^*q_cQ^*) \to q_c \\
c(Q^*) \to q & c(Q^*q_bQ^*q_bQ^*) \to q_c & c(Q^*q_cQ^*) \to q_c
\end{array}
$$





Normalized Hedge Automata

Definition :

A HA $\mathcal{A} = (\Sigma, Q, Q^{\mathsf{f}}, \Delta)$ over Σ is called normalized if for all $a \in \Sigma$ and $q \in Q$, there is at most one transition of the form $a(L) \to q$ in Δ.

When $\mathcal{A}$ is normalized, we denote $a(L_{a,q}) \to q$ the unique transition with a and q.

Proposition :

For all HA $\mathcal{A}$, there exista a normalized HA $\mathcal{A}_{\mathsf{n}}$ recognizing the same language.

The size of $\mathcal{A}_{\mathsf{n}}$ is linear in the size of $\mathcal{A}$.

Complete and deterministic hedge automata

Semantical definitions

Definition :

A HA $\mathcal{A} = (\Sigma, Q, Q^{\mathsf{f}}, \Delta)$ over Σ is complete if for all $t \in \mathcal{U}(\Sigma)$, there exists at least one state $q \in Q$ s.t. $t \in L(\mathcal{A}, q)$.

Definition :

A HA $\mathcal{A} = (\Sigma, Q, Q^{\mathsf{f}}, \Delta)$ over Σ is deterministic if for all $t \in \mathcal{U}(\Sigma)$, there exists at most one state $q \in Q$ s.t. $t \in L(\mathcal{A}, q)$.

Complete and deterministic hedge automata

Syntactical definitions

Definition :

A HA $\mathcal{A} = (\Sigma, Q, Q^{\mathsf{f}}, \Delta)$ over Σ is complete if
for all $a \in \Sigma$ and all finite sequence $q_1, \ldots, q_n \in Q^*$,
there exists a transition $a(L) \to q \in \Delta$ with $q_1 \ldots q_n \in L$.

Definition :

A HA $\mathcal{A} = (\Sigma, Q, Q^{\mathsf{f}}, \Delta)$ over Σ is deterministic if
for all transitions $a(L_1) \to q_1$ and $a(L_2) \to q_2$ in Δ,
either $L_1 \cap L_2 = \emptyset$, or $q_1 = q_2$.

Determinism : examples

HA for Boolean expressions evaluation : deterministic.
$\Sigma = \{\wedge, \vee, 0, 1\}$, $Q = \{q_0, q_1\}$ et Δ :

$$\begin{array}{rcl} 0 & \to & q_0 \\ \wedge(q_1^* \, q_0 \, (q_0 \mid q_1)^*) & \to & q_0 \\ \vee(q_0 q_0^*) & \to & q_0 \end{array} \qquad \begin{array}{rcl} 1 & \to & q_1 \\ \wedge(q_1 q_1^*) & \to & q_1 \\ \vee(q_0^* \, q_1 \, (q_0 \mid q_1)^*) & \to & q_1 \end{array}$$

Language with 2 b's and common ancestor c : not deterministic.
$\Sigma = \{a, b, c\}$, $Q = \{q, q_b, q_c\}$, $Q^{\mathsf{f}} = \{q_c\}$, Δ :

$$\begin{array}{rcl} a(Q^*) & \to & q \\ b(Q^*) & \to & q_b \\ c(Q^*) & \to & q \end{array} \qquad \begin{array}{rcl} a(Q^* q_b Q^*) & \to & q_b \\ c(Q^* q_b Q^*) & \to & q_b \\ c(Q^* q_b Q^* q_b Q^*) & \to & q_c \end{array} \qquad \begin{array}{rcl} a(Q^* q_c Q^*) & \to & q_c \\ b(Q^* q_c Q^*) & \to & q_c \\ c(Q^* q_c Q^*) & \to & q_c \end{array}$$

HA completion

Proposition :

For all HA $\mathcal{A}$, there exists a complete HA $\mathcal{A}_c$ recognizing the same language.

The size of $\mathcal{A}_c$ is linear is the size of $\mathcal{A}$.

pr.: add a *trash* state $q_\perp$ and transitions :

$$a\Big(\bigcap_{q\in Q} Q^* \setminus L_{a,q} \cup Q_\perp^* q_\perp Q_\perp^*\Big) \to q_\perp$$

HA determinization

Proposition :

For all HA $\mathcal{A}$, there exists a deterministic HA $\mathcal{A}_\mathsf{d}$ recognizing the same language.

The size of $\mathcal{A}_\mathsf{d}$ is exponential in the size of $\mathcal{A}$ (lower bound).

pr.: subset construction

HA : membership decision

Proposition : $\in$

The problem of membership is decidable in polynomial time for the HAs whose languages $L_{a,q}$ are given by NFAs.

- linear time for DHA whose languages $L_{a,q}$ are given by DFA,
- NP-complet if the languages $L_{a,q}$ are given by alternating automata.

HA : emptiness decision

Proposition : $\emptyset$

The problem of emptiness is decidable in polynomial time for HAs whose languages $L_{a,q}$ are given by NFAs.

pr.: state marking. $M \subseteq Q$,

- initialy, $M = \emptyset$.
- at each step , if $a(L_{a,q}) \to q \in \Delta$ and $L_{a,q} \cap M^* \neq \emptyset$ then $M := M \cup \{q\}$.

- PSPACE-complete if the languages $L_{a,q}$ are given by alternating automata.

HA : decision of inclusion and equivalence

Proposition : $\subseteq$, $\equiv$

The problem of inclusion (resp. equivalence) is EXPTIME-complete for HAs whose languages $L_{a,q}$ are given by NFAs.

pr.: $L(\mathcal{A}_1) \subseteq L(\mathcal{A}_2)$ iff $L(\mathcal{A}_1) \cap (\mathcal{U}(\Sigma) \setminus L(\mathcal{A}_2)) = \emptyset$.

- in PTIME for DHAs whose languages $L_{a,q}$ are given by DFAs.
- PSPACE-complete for DHAs whose languages $L_{a,q}$ are given by alternating automata.

Currying

Transformation into binary trees with $@$ and constants.

We associate the following signature to an alphabet Σ :

$$\Sigma_@ := \{a : 0 \mid a \in \Sigma\} \cup \{@ : 2\}$$

The operator curry : $\mathcal{U}(\Sigma) \to \mathcal{T}(\Sigma_@)$, is defined recursively :

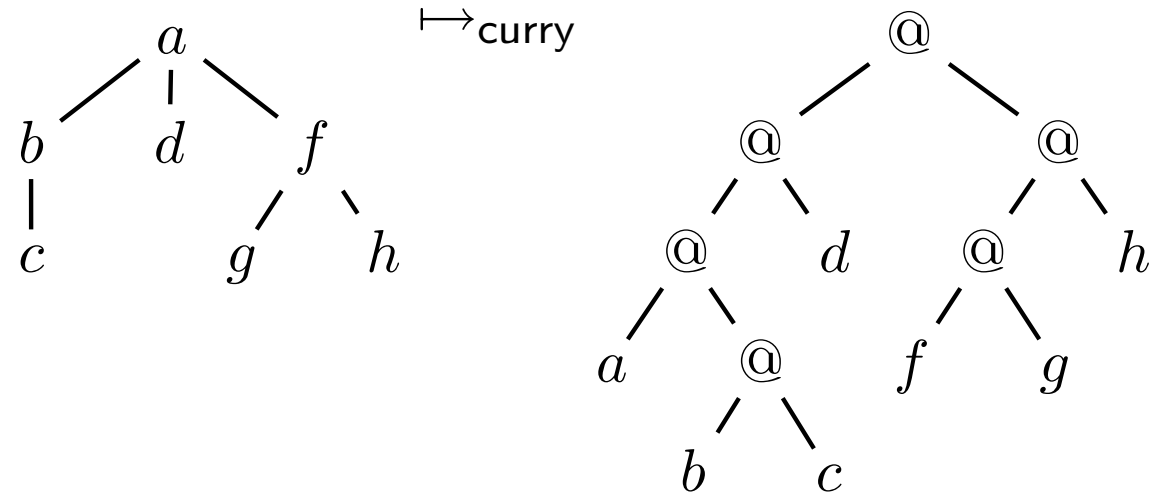
- $\mathsf{curry}(a) := a$,
- $\mathsf{curry}\big(a(t_1, \ldots, t_n)\big) := @\big(\mathsf{curry}\big(a(t_1, \ldots, t_{n-1})\big), \mathsf{curry}(t_n)\big)$.

$$a(t_1 \ldots t_n) \,@\, t_{n+1} = a(t_1 \ldots t_{n+1})$$

$$\mathsf{curry}(t \,@\, t') = @\big(\mathsf{curry}(t), \mathsf{curry}(t')\big)$$

Currying : example 1

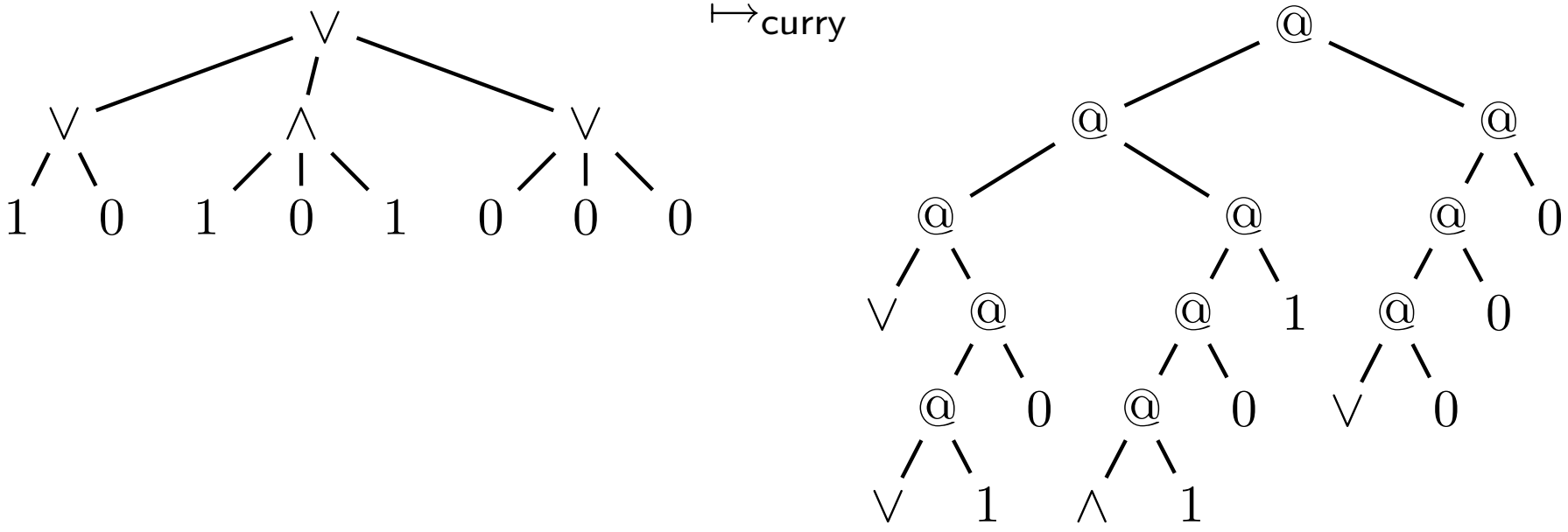
$$\mathsf{curry}\big(a(t_1,\ldots,t_n)\big) := @\big(\mathsf{curry}\big(a(t_1,\ldots,t_{n-1})\big),\mathsf{curry}(t_n)\big)$$



Currying : example 2

$\mathsf{curry}\big(a(t_1,\ldots,t_n)\big) := @\big(\mathsf{curry}\big(a(t_1,\ldots,t_{n-1})\big), \mathsf{curry}(t_n)\big)$

image of unranked Boolean expressions.



Currying : properties

Lemma :

curry is a bijection from $\mathcal{U}(\Sigma)$ into $\mathcal{T}(\Sigma_@)$.

Proposition :

$L \subseteq \mathcal{U}(\Sigma)$ is a HA language iff curry(L) is regular.

HA : Boolean operations

Proposition :

The class of HA languages is closed under union, intersection and complement.

HA : closure under morphisms

projection $h : \mathcal{U}(\Sigma) \to \mathcal{U}(\Sigma')$, defined by extension to trees of an application $h : \Sigma \to \Sigma'$.

$$h(L) = \{h(t) \mid t \in L\} \quad \text{and} \quad h^{-1}(L') = \{t \in \mathcal{U}(\Sigma) \mid h(t) \in L'\}$$

Proposition :

The class of HA languages is closed under projections and inverse projections.

Stepwise Automata

Determinism & Minimization

Stepwise Automata

Definition : stepwise automata

A deterministic stepwise hedge automaton (DSHA) is a tuple $\mathcal{A} = (\Sigma, Q, Q_f, \delta_0, \delta)$, where Σ, Q, and Q_f are as usual, $\delta_0 : \Sigma \to Q$ is a function assigning to each letter of the alphabet an initial state, and $\delta : Q \times Q \to Q$ is the transition function.

For $a \in \Sigma$,
$$\begin{aligned} \delta_a : & \quad Q^* \to Q \\ \delta_a(\varepsilon) &= \delta_0(a) \\ \delta_a(w \cdot q) &= \delta(\delta_a(w), q) \end{aligned}$$

A run of $\mathcal{A}$ on $t \in \mathcal{U}(\Sigma)$ is a tree $r \in \mathcal{U}(Q)$ such that

- r and t have the same domain,
- for all $p \in \mathcal{P}os(t)$, $r(p) = \delta_{t(p)}(r(p1) \dots r(pn))$, where n is the number of successors of p in $\mathcal{P}os(t)$.

The run r is accepting (succesful) iff $r(\varepsilon) \in Q^{\mathsf{f}}$.

Stepwise Automata & Ranked TA

stepwise DSHA $\mathcal{A}$	ranked DTA curry($\mathcal{A}$)
$\delta_0(a) = q$	$a \to q$
$\delta(q_1, q_2) = q$	$@(q_1, q_2) \to q$

Lemma :

For all $t, t' \in \mathcal{U}(\Sigma)$ and $q, q' \in Q$,
if $t \in L(\mathcal{A}, q)$ and $t' \in L(\mathcal{A}, q')$, then $t \,@\, t' \in L(A, \delta(q, q')$.

Lemma :

For all DSHA $\mathcal{A}$, curry$(L(\mathcal{A})) = L($curry$(\mathcal{A}))$.

Minimal Stepwise Automata

Corollary :

DSHA recognize all HA unranked tree languages.

Corollary :

For each HA language $L \subseteq \mathcal{U}(\Sigma)$ there is a unique (up to renaming of states) minimal DSHA accepting L.

Logique monadique faible du second ordre
Weak Second Order Monadic Logic

$\mathrm{MSO}(\rightarrow, \downarrow)$

Logic and Automata

- logic express properties of labeled binary trees
  = language specification,
- compilation of formulae into automata
  = decision algorithms.
- equivalence between both formalisms
  Thatcher & Wright's theorem for ranked trees and ranked TA.
  analogous for unranked trees and HA.

Unranked term as structure

Σ finite alphabet

$$\mathcal{L}_\Sigma := \{=, \downarrow, \rightarrow, L_a \mid a \in \Sigma\}.$$

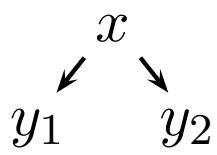
to $t \in \mathcal{U}(\Sigma)$, we associate a structure $\underline{t}$ over $\mathcal{L}_\Sigma$

$$\underline{t} := \langle \mathcal{P}os(t), =^{\underline{t}}, \downarrow^{\underline{t}}, \rightarrow^{\underline{t}}, L_a^{\underline{t}}, L_b^{\underline{t}}, \cdots \rangle$$

$$\begin{array}{rcl}
\text{domain} & := & \text{positions of } t \quad (\mathcal{P}os(t) \subset \mathbb{N}^*), \\
=^{\underline{t}} & := & \text{equality over } \mathcal{P}os(t), \\
\downarrow^{\underline{t}} & := & \left\{ \langle p, p \cdot i \rangle \mid i \in \mathbb{N}, p, p \cdot i \in \mathcal{P}os(t) \right\} \text{ (parent relation)}, \\
\rightarrow^{\underline{t}} & := & \left\{ \langle p \cdot i, p \cdot i + 1 \rangle \mid i \in \mathbb{N}, p, p \cdot i, p \cdot i + 1 \in \mathcal{P}os(t) \right\} \\
 & & \text{(next sibling)}, \\
L_a^{\underline{t}} & := & \{p \in \mathcal{P}os(t) \mid t(p) = a\}.
\end{array}$$

MSO($\rightarrow, \downarrow$) : syntaxe

Rappel : WS2S, $S_1(x, y_1) \wedge S_2(x, y_2)$ for



formulas of MSO($\rightarrow, \downarrow$) :

- first order variables $x \ldots$
- second order variables $X \ldots$
- term ::= x
- form ::= $x = y \mid x \downarrow y \mid x \rightarrow y \mid x \in X \mid L_a(x) \quad a \in \Sigma$
  $\mid$ form $\wedge$ form $\mid$ form $\vee$ form $\mid \neg$form
  $\mid \exists x$ form $\mid \exists X$ form $\mid \forall x$ form $\mid \forall X$ form

MSO($\rightarrow, \downarrow$) : semantics

- $t \in \mathcal{U}(\Sigma)$,
- valuation σ of first order variables into $\mathcal{P}os(t)$,
- valuation δ of second order variables into subsets of $\mathcal{P}os(t)$,
- $\underline{t}, \sigma, \delta \models x = y$ iff $\sigma(x) = \sigma(y)$,
- $\underline{t}, \sigma, \delta \models x \downarrow y$ iff $\exists i \in \mathbb{N}$, $\sigma(y) = \sigma(x) \cdot i$,
- $\underline{t}, \sigma, \delta \models x \rightarrow y$ iff $\exists p_0 \in \mathbb{N}^*$, $i \in \mathbb{N}$, $\sigma(x) = p_0 \cdot i$ and $\sigma(x) = p_0 \cdot i + 1$,
- $\underline{t}, \sigma, \delta \models x \in X$ iff $\sigma(x) \in \delta(X)$,
- $\underline{t}, \sigma, \delta \models L_a(x)$ iff $t(\sigma(x)) = a$,
- $\underline{t}, \sigma, \delta \models \phi_1 \wedge \phi_2$ iff $\underline{t}, \sigma, \delta \models \phi_1$ and $\underline{t}, \sigma, \delta \models \phi_2$,
- $\underline{t}, \sigma, \delta \models \phi_1 \vee \phi_2$ iff $\underline{t}, \sigma, \delta \models \phi_1$ or $\underline{t}, \sigma, \delta \models \phi_2$,
- $\underline{t}, \sigma, \delta \models \neg\phi$ iff $\underline{t}, \sigma, \delta \not\models \phi$,

MSO$(\rightarrow, \downarrow)$: semantics (quantifiers)

- $\underline{t}, \sigma, \delta \models \exists x\ \phi$ iff $x \notin dom(\sigma)$, x free in ϕ and exists $p \in \mathcal{P}os(t)$ s.t. $\underline{t}, \sigma \cup \{x \mapsto p\}, \delta \models \phi$,
- $\underline{t}, \sigma, \delta \models \forall x\ \phi$ iff $x \notin dom(\sigma)$, x free in ϕ and for all $p \in \mathcal{P}os(t)$, $\underline{t}, \sigma \cup \{x \mapsto p\}, \delta \models \phi$,
- $\underline{t}, \sigma, \delta \models \exists X\ \phi$ iff $X \notin dom(\delta)$, X free in ϕ and exists $P \subseteq \mathcal{P}os(t)$ s.t. $\underline{t}, \sigma, \delta \cup \{X \mapsto P\} \models \phi$,
- $\underline{t}, \sigma, \delta \models \forall X\ \phi$ iff $X \notin dom(\delta)$, X free in ϕ and for all $P \subseteq \mathcal{P}os(t)$, $\underline{t}, \sigma, \delta \cup \{X \mapsto P\} \models \phi$.

Second order monadic logic : examples

- root :

  $x = \varepsilon \equiv \forall y \, x \downarrow^* y$, or $\neg \exists y \, y \downarrow x$,

- leaf :

  $\mathsf{leaf}(x) = \neg \exists y \, x \downarrow y$

- first child :

  $\mathsf{first}(x) = \neg \exists y \, y \rightarrow x$

- last child :

  $\mathsf{last}(x) = \neg \exists y \, x \rightarrow y$

- prefixe ordering = transitive closure of $\downarrow$:

  $x \downarrow^* y \equiv$
  $\forall X \left(y \in X \wedge \forall z \, \forall z' \left(z \downarrow z' \wedge z' \in X \Rightarrow z \in X \right) \right) \Rightarrow x \in X$

- transitive closure of $\rightarrow$:

  $x \xrightarrow{*} y \equiv$
  $\forall X \left(y \in X \wedge \forall z \, \forall z' \left(z \rightarrow z' \wedge z' \in X \Rightarrow z \in X \right) \right) \Rightarrow x \in X$

Defining languages of $\mathcal{U}(\Sigma)$ in MSO$(\rightarrow, \downarrow)$

$\Sigma := \{a_1, \ldots, a_n\}$,

Definition : MSO$(\rightarrow, \downarrow)$-definability

For $\phi \in$ MSO$(\rightarrow, \downarrow)$ without free variables over $\mathcal{L}_\Sigma$,
$L(\phi) := \left\{ t \in \mathcal{U}(\Sigma) \;\middle|\; \underline{t} \models \phi \right\}$.

Theorem :

A subset of $\mathcal{U}(\Sigma)$ is a HA language iff it is definable in MSO$(\rightarrow, \downarrow)$.

HA $\longrightarrow$ MSO$(\rightarrow, \downarrow)$

Let $\Sigma = \{a_1, \ldots, a_n\}$.

Theorem :

For all hedge automaton $\mathcal{A}$ over Σ, there exists $\phi_{\mathcal{A}} \in \mathrm{MSO}(\rightarrow, \downarrow)$ such that $L(\phi_A) = L(\mathcal{A})$.

Let $\mathcal{A} = (\Sigma, Q, Q^{\mathsf{f}}, \Delta)$, normalized, with $Q = \{q_1, \ldots, q_m\}$, and $P = \{p_1, \ldots, p_h\}$ disjoint union of state sets of NFAs for the horizontal languages $L_{a,q} \subseteq Q^*$, $a \in \Sigma$, $q \in Q$.
$\phi_{\mathcal{A}}$: existence of an accepting run of $\mathcal{A}$ on $t \in \mathcal{U}(\Sigma)$.

$$\phi_{\mathcal{A}} := \exists Y_1 \ldots \exists Y_m \, \exists Z_0 \ \ldots \exists Z_h \, \phi(\overline{Y}, \overline{Z})$$

HA $\rightarrow$ MSO$(\rightarrow, \downarrow)$

$$
\begin{aligned}
&\phi_{\mathcal{A}} \equiv Y_1, \ldots, Y_m \text{ partition } \mathcal{P}os(t) \\
&\wedge Z_0 = \{\varepsilon\} \wedge Z_0, Z_1, \ldots, Z_h \text{ partition } \mathcal{P}os(t) \\
&\wedge \bigvee_{q_j \in Q^{\mathsf{f}}} \varepsilon \in Y_j \\
&\wedge \forall x \bigvee_{a_i(L) \rightarrow q_j \in \Delta} x \in Y_j \wedge \\
&\forall y \left(\begin{array}{ll}
x \downarrow y \wedge \mathsf{first}(y) & \Rightarrow \bigvee_{p_k = \mathsf{initial}(L)} y \in Z_k \\
\wedge \exists y'\ x \downarrow y \wedge y \rightarrow y' & \Rightarrow \\
\qquad \bigvee_{p_k \xrightarrow{q_{j'}} p_{k'} \in L} (y \in Z_k \wedge y \in Y_{j'} \wedge y' \in Z_{k'}) & \\
\wedge (x \downarrow y \wedge \mathsf{last}(y)) & \Rightarrow \\
\qquad \bigvee_{p_{k'} \in \mathsf{final}(L)} \bigvee_{p_k \xrightarrow{q_{j'}} p_{k'} \in L} (y \in Z_k \wedge y \in Y_{j'}) &
\end{array} \right)
\end{aligned}
$$

MSO$(\rightarrow, \downarrow) \rightarrow$ HA

Theorem :

Every subset of $\mathcal{U}(\Sigma)$ definable in MSO$(\rightarrow, \downarrow)$ is a HA language.

For all formula $\phi \in$ MSO$(\rightarrow, \downarrow)$ over Σ (without free variables) there exists a hedge automaton $\mathcal{A}_\phi$ over Σ, such that $L(\mathcal{A}_\phi) = L(\phi)$.

Corollary :

MSO$(\rightarrow, \downarrow)$ is decidable.

pr.: reduction to emptiness decision for $\mathcal{A}_\phi$.

Operator "first child next sibling"

Correspondence with algebras of ranked terms (symbols of arity 0 or 2) : representation of unranked terms as binary terms with pointers and lists.

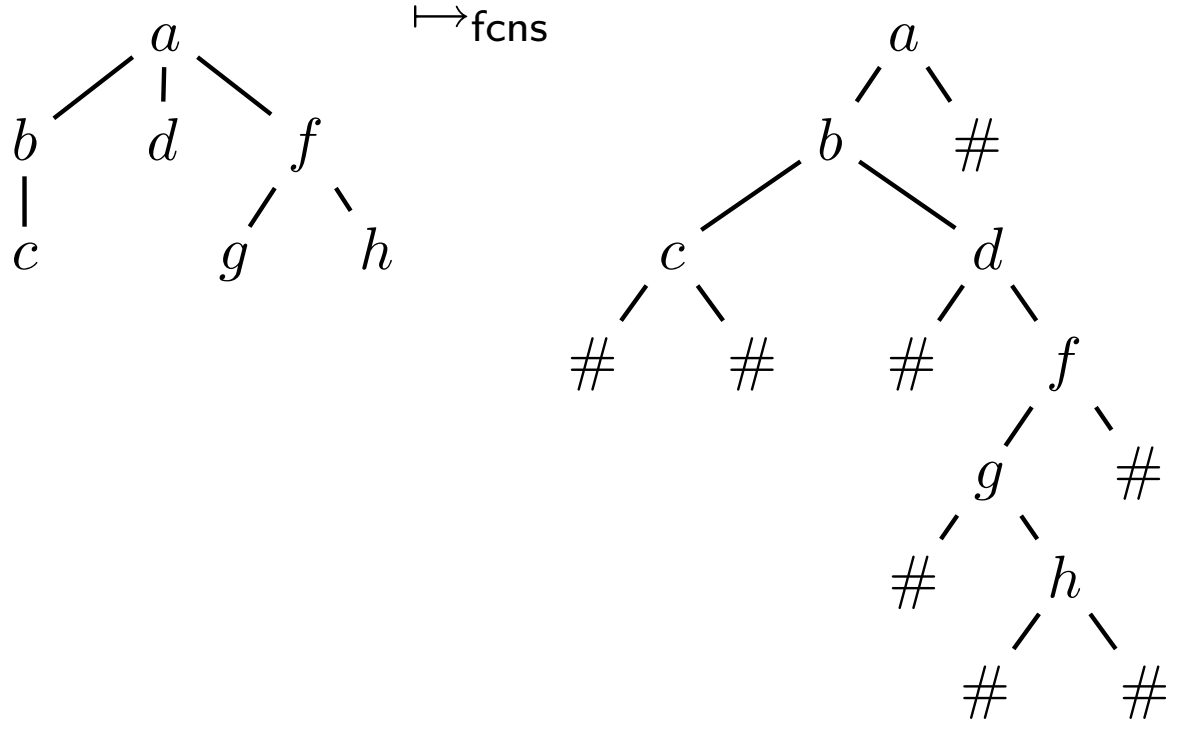
We associate the following signature to an alphabet Σ :

$$\Sigma_{\#} := \{a : 2 \mid a \in \Sigma\} \cup \{\# : 0\}$$

The operator fcns : $\mathcal{H}(\Sigma) \to \mathcal{T}(\Sigma_{\#})$, is defined recursively :

- $\mathsf{fcns}(a) := a(\#, \#)$,
- $\mathsf{fcns}\big(a(t_1, \ldots, t_n)\big) := a\big(\mathsf{fcns}(t_1, \ldots, t_n), \#\big)$,
- $\mathsf{fcns}(t_1, \ldots, t_n) := \mathsf{fcns}(t_1)[\mathsf{fcns}(t_2, \ldots, t_n)]_2$ if $n \geq 2$.

Operator "first child next sibling" : example



Operator "first child next sibling" : properties

Lemma :

fcns is a bijection from $\mathcal{H}(\Sigma)$ into $\mathcal{T}(\Sigma_{\#})$.

For all $t \in \mathcal{T}(\Sigma_{\#})$, $\text{fcns}^{-1}(t) \in \mathcal{U}(\Sigma)$ iff $t|_2 = \#$.

Proposition :

If $L \subseteq \mathcal{U}(\Sigma)$ is a HA language, then $\text{fcns}(L)$ is regular.

Proposition :

If $L \subseteq \mathcal{T}(\Sigma_{\#})$ is regular, then $\text{fcns}^{-1}(L) \cap \mathcal{U}(\Sigma)$ is a HA language.

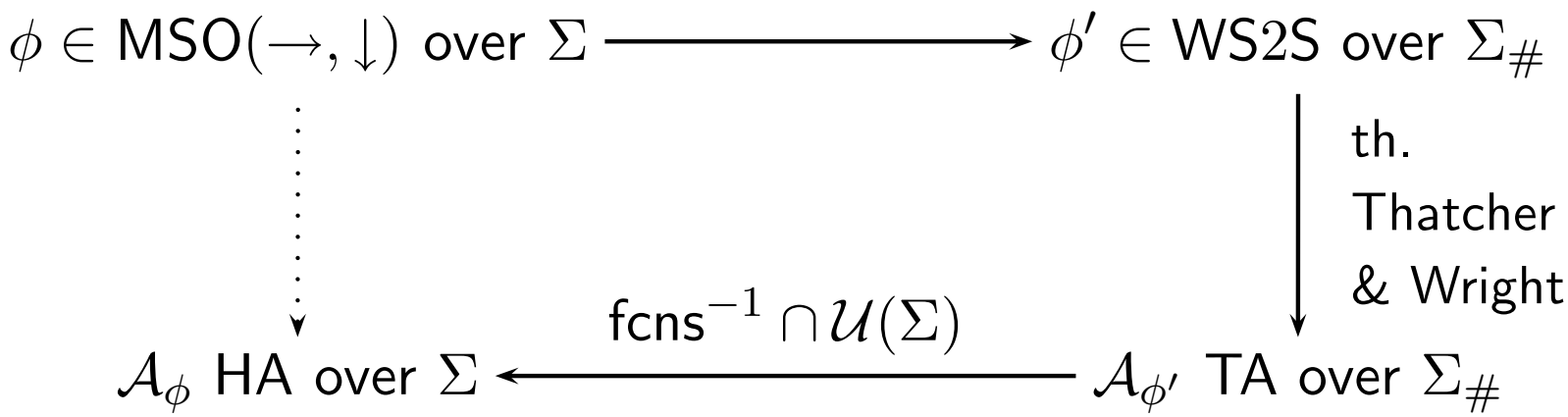
MSO$(\rightarrow, \downarrow) \rightarrow$ HA

$\phi \in$ MSO$(\rightarrow, \downarrow)$ over $\Sigma \mapsto$ $\phi' \in$ WS2S over $\Sigma_\#$ defining fcns$(L(\phi))$.

$$\phi' \equiv L_\#(\varepsilon \cdot 2) \wedge \psi$$

ψ obtained from ϕ by replacements of atoms :

ϕ	ψ
$L_a(x)$	$L_a(x) \wedge \neg L_\#(x)$
$x \downarrow y$	$y \in x \cdot 1 \cdot 2^*$
$x \rightarrow y$	$y = x \cdot 2$



Variantes de Hedge Automates
–
Langages Réguliers modulo Associativité

Extensions des HA

Definition : CF-HA

CF-HA : tuple $(\Sigma, Q, Q_{\mathsf{f}}, \Delta)$ comme un HA dont les transitions sont de la forme $a(L) \to q$ avec $a \in \Sigma$, $q \in Q$, et $L \subseteq Q^*$ est hors contexte (*context-free*).

Definition : CS-HA

CS-HA : tuple $(\Sigma, Q, Q_{\mathsf{f}}, \Delta)$ comme un HA dont les transitions sont de la forme $a(L) \to q$ avec $a \in \Sigma$, $q \in Q$, et $L \subseteq Q^*$ est contextuel (*context-sensitive*).

CF-HA : exemple

$\Sigma = \{a, b, f\}$, langage L des arbres de $\mathcal{U}(\Sigma)$:

- dont les noeuds internes sont étiquettés par f,
- avec le même nombre de feuilles a que de feuilles b sous chaque noeud.

langage du CF-HA $(Q, Q_\mathsf{f}, \Delta)$ avec $Q = \{q, q_a, q_b\}$ et

$$\Delta = \{a \to q_a, \quad b \to q_b, \quad f(L) \to q\}$$

L est le langage engendré par la grammaire context-free

$$\begin{array}{lll} N & := & \varepsilon \mid \text{ toute permutation de } NN_aN_b \mid q \\ N_a & := & q_a \quad N_b := q_b \end{array}$$

Rem. : L n'est pas un langage de HA.

Langages réguliers modulo A (ATA)

Signature $\Sigma = \Sigma_\emptyset \uplus \Sigma_\mathsf{A}$.
Les symboles de Σ_A sont binaires et suivent l'axiome d'associativité :

$$a(x_1, a(x_2, x_3)) = a(a(x_1, x_2), x_3) \qquad \text{(A)}$$

Pour un TA $\mathcal{B}$ sur Σ, on note

$$\mathsf{A}\big(L(\mathcal{B})\big) := \big\{ t \in \mathcal{T}(\Sigma) \;\big|\; t \xleftrightarrow[\mathsf{A}]{*} s \in L(\mathcal{B}) \big\}$$

(langage d'ATA)

Proposition :

- la classe des langages d'arbres réguliers est strictement incluse dans la classe des langages d'ATA.
- La classe des langages d'ATA n'est pas close par intersection.

Correspondences $\mathcal{T}(\Sigma) \leftrightarrow \mathcal{U}(\Sigma)$

On suppose $\Sigma_{\mathsf{A}} = \{a\}$.

$$\begin{array}{rrcl} \mathit{flat}: & \mathcal{T}(\Sigma) & \to & \mathcal{U}(\Sigma) \\ \mathit{hflat}: & \mathcal{T}(\Sigma)^* & \to & \mathcal{H}(\Sigma) \\ \mathit{flat}^{-1}: & \mathcal{U}(\Sigma) & \to & \mathcal{T}(\Sigma) \end{array}$$

Définitions ($g \in \Sigma_n \setminus \Sigma_{\mathsf{A}}$) :

$$\begin{array}{rcl} \mathit{flat}\big(g(t_1,\ldots,t_n)\big) & = & g(\mathit{flat}(t_1)\ldots\mathit{flat}(t_n)) \\ \mathit{flat}\big(a(t_1,t_2)\big) & = & a(\mathit{hflat}(t_1\,t_2)) \\ \\ \mathit{hflat}\big(g(s_1,\ldots,s_n)\,t_2\ldots t_m\big) & = & \mathit{flat}\big(g(s_1,\ldots,s_n)\big)\,\mathit{hflat}(t_2\ldots t_m) \\ \mathit{hflat}\big(a(s_1,s_2)\,t_2\ldots t_m\big) & = & \mathit{hflat}(s_1 s_2\,t_2\ldots t_m) \\ \\ \mathit{flat}^{-1}\big(g(t_1\ldots t_n)\big) & = & g(\mathit{flat}^{-1}(t_1),\ldots,\mathit{flat}^{-1}(t_n)) \\ \mathit{flat}^{-1}\big(a(t_1\ldots t_m)\big) & = & a(t_1, a(t_2,\ldots,a(t_{m-1},t_m))) \\ & & (m \geq 2) \end{array}$$

CF-HA $\leftrightarrow$ ATA

Proposition :

CF-HA $\equiv$ ATA

- $\subseteq$ pour tout CF-HA $\mathcal{A}$ il existe un TA $\mathcal{B}$ tel que $L(\mathcal{B}) = \mathit{flat}^{-1}\big(L(\mathcal{A})\big)$.
- $\supseteq$ pour tout TA $\mathcal{B}$ il existe un CF-HA $\mathcal{A}$ tel que $L(\mathcal{A}) = \mathit{flat}\big(\mathsf{A}(L(\mathcal{B}))\big)$.

Résultats CF-HA

Proposition :

La classe des langages de CF-HA n'est pas close par intersection et complément.

Proposition : $\emptyset$

Le problème du vide est décidable en temps polynomial CF-HA.

Proposition : $\in$

Le problème de l'appartenance est décidable en temps polynomial CF-HA.

pr.: Conséquences de la correspondance avec ATA et des résultats pour ces langages.

Automates d'arbres généralisés (GTA)

Definition : GTA

Un *automate d'arbres généralisé* (GTA) sur une signature Σ est un tuple $\mathcal{B} = (\Sigma, Q, Q^{\mathsf{f}}, \Delta)$ où Q est un ensemble fini d'*états*, $Q^{\mathsf{f}} \subseteq Q$ est le sous-ensemble des états finaux et Δ est un ensemble de règles de transition de TA ou de la forme $f(q_1, \ldots, q_n) \to f(q'_1, \ldots, q'_n)$ avec $f \in \Sigma_n$ $(n \geq 0)$ et $q_1, \ldots, q_n, q'_1, \ldots, q'_n \in Q$.

Pour un TA ou GTA $\mathcal{B}$, on définit les languages :

$$\begin{array}{rcl} L(\mathcal{B}) & := & \{t \in \mathcal{T}(\Sigma) \mid t \xrightarrow[\Delta]{*} q \in Q^{\mathsf{f}}\} \\ L_{\mathsf{A}}(\mathcal{B}) & := & \{t \in \mathcal{T}(\Sigma) \mid t \xrightarrow[\Delta/\mathsf{A}]{*} q \in Q^{\mathsf{f}}\} \end{array}$$

où $\xrightarrow[\Delta/\mathsf{A}]{*}$ est la réécriture modulo A : $\xrightarrow[\Delta/\mathsf{A}]{*} := \xleftrightarrow[\mathsf{A}]{*} \circ \xrightarrow[\Delta]{} \circ \xleftrightarrow[\mathsf{A}]{*}$

Langages de TA et GTA modulo A

Proposition :

Pour tout GTA $\mathcal{B}$, il existe un TA $\mathcal{B}'$ tel que $L(\mathcal{B}') = L(\mathcal{B})$.

Proposition :

Pour tout TA $\mathcal{B}$, $L_{\mathsf{A}}(\mathcal{B}) = \mathsf{A}\big(L(\mathcal{B})\big)$.

Donc le vide de $L_{\mathsf{A}}(\mathcal{B})$ pour TA $\mathcal{B}$ est décidable.

Proposition :

Pour un GTA $\mathcal{B}$, en général $L_{\mathsf{A}}(\mathcal{B}) \neq \mathsf{A}\big(L(\mathcal{B})\big)$.

Résultats GTA

Proposition : $\in$

Le problème de l'appartenance $t \in L_{\mathsf{A}}(\mathcal{B})$ pour GTA $\mathcal{B}$ est PSPACE-complet.

Proposition : $\emptyset$

Le vide de $L_{\mathsf{A}}(\mathcal{B})$ pour GTA $\mathcal{B}$ est indécidable.

CS-HA $\leftrightarrow$ GTA mod. A

Proposition :

CS-HA $\equiv$ AGTA

- $\subseteq$ pour tout CS-HA $\mathcal{A}$ il existe un GTA $\mathcal{B}$ tel que $L_{\mathsf{A}}(\mathcal{B}) = \mathsf{A}\big(\mathit{flat}^{-1}\big(L(\mathcal{A})\big)\big)$.
- $\supseteq$ pour tout GTA $\mathcal{B}$ il existe un CS-HA $\mathcal{A}$ tel que $L(\mathcal{A}) = \mathit{flat}\big(L_{\mathsf{A}}(\mathcal{B})\big)$.

CS-HA : résultats de décision

Proposition :

Le probléme de l'appartenance est PSPACE-complet pour CS-HA.

Proposition :

Le problème du vide est indécidable pour les CS-HA.

Résumé

CS-HA	$\equiv$	AGTA	clotures Booléennes $\emptyset$ indécidable.
CF-HA	$\equiv$	ATA	pas de cloture par $\cap$ et $\neg$ $\emptyset$ décidable.
HA			clotures Booléennes $\emptyset$ décidable.