

AUTOMATA FOR UNRANKED TREES

CHRISTOF LÖDING
RWTH AACHEN

WORKSHOP ON TREE AUTOMATA
BONN, JUNE 7–9, 2006

UNRANKED TREES

- Finite trees, each node can have an arbitrary number of successors
 - Distinguish **ordered** and unordered trees
 - Serve as model for, e.g.,
 - terms with associative operators
 - derivation trees of extended context-free grammars
 - tree-structured data (XML)
- ↪ Automata theory for unranked trees similar to the one for ranked trees

UNRANKED TREES

- Finite trees, each node can have an arbitrary number of successors
- Distinguish **ordered** and unordered trees
- Serve as model for, e.g.,
 - terms with associative operators
 - derivation trees of extended context-free grammars
 - tree-structured data (XML)

↪ Automata theory for unranked trees similar to the one for ranked trees

History

- Originally in the 60's/70's: Pair and Quere, Takahashi, Thatcher
- Resurrected by Brüggemann-Klein, Murata, Wood [95–01] in the context of XML

Basic models

- Definitions

- Deterministic bottom-up automata

Encodings by ranked trees

- Child sibling

- Currying

Top-down determinism

- Decision problem

Extensions

- Numerical constraints

- Equality constraints

BASIC MODELS

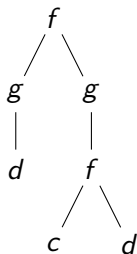
TREES AND HEDGES

Ranked trees:

ranked alphabet $\Sigma = \Sigma_0 \cup \dots \cup \Sigma_n$

$f \in \Sigma_i$ and t_1, \dots, t_i ranked trees

$\Rightarrow f(t_1, \dots, t_i)$ is a ranked tree



$$\Sigma_2 = \{f\}$$

$$\Sigma_1 = \{g\}$$

$$\Sigma_0 = \{c, d\}$$

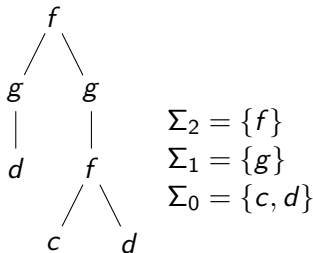
TREES AND HEDGES

Ranked trees:

ranked alphabet $\Sigma = \Sigma_0 \cup \dots \cup \Sigma_n$

$f \in \Sigma_i$ and t_1, \dots, t_i ranked trees

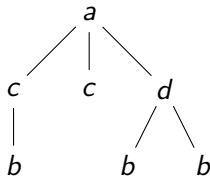
$\Rightarrow f(t_1, \dots, t_i)$ is a ranked tree



Hedges and unranked trees:

alphabet Σ

- Hedges are sequences of unranked trees
- $a \in \Sigma$ and $t_1 \cdots t_n$ hedge
 $\Rightarrow a(t_1 \cdots t_n)$ unranked tree

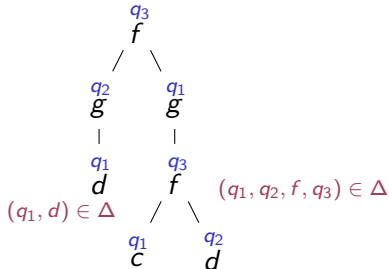


NONDETERMINISTIC TREE AUTOMATA

Ranked

Unranked

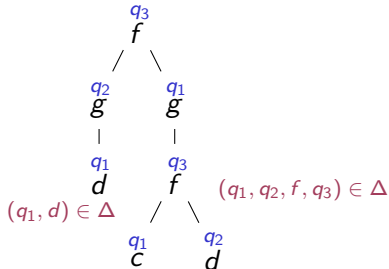
- Q states
- $\Delta \subseteq \bigcup Q^i \times \Sigma_i \times Q$ transitions



NONDETERMINISTIC TREE AUTOMATA

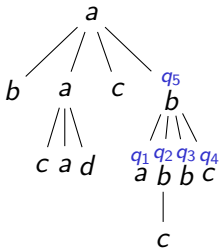
Ranked

- Q states
- $\Delta \subseteq \bigcup Q^i \times \Sigma_i \times Q$ transitions



Unranked

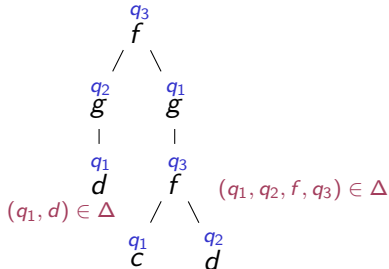
- Q states



NONDETERMINISTIC TREE AUTOMATA

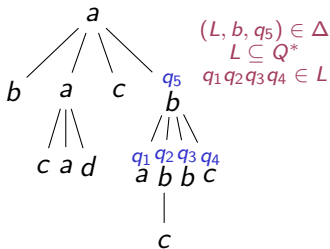
Ranked

- Q states
- $\Delta \subseteq \bigcup Q^i \times \Sigma_i \times Q$ transitions



Unranked

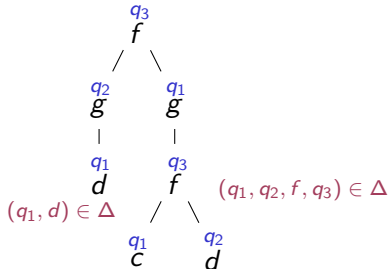
- Q states



NONDETERMINISTIC TREE AUTOMATA

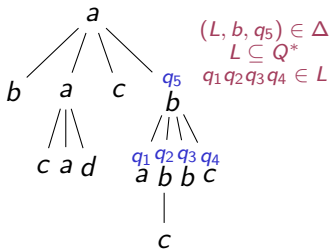
Ranked

- Q states
- $\Delta \subseteq \bigcup Q^i \times \Sigma_i \times Q$ transitions



Unranked

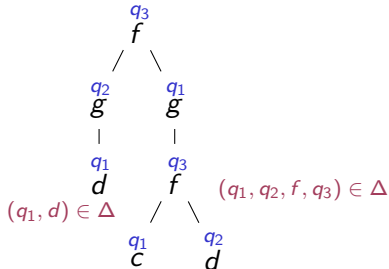
- Q states
- $\Delta \subseteq \text{REG}(Q) \times \Sigma \times Q$



NONDETERMINISTIC TREE AUTOMATA

Ranked

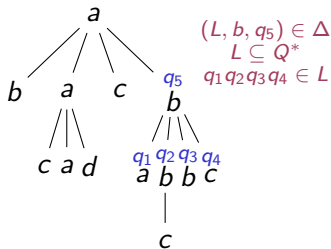
- Q states
- $\Delta \subseteq \bigcup Q^i \times \Sigma_i \times Q$ transitions



- Bottom-up: F final states
- Top-down: q_0 initial state

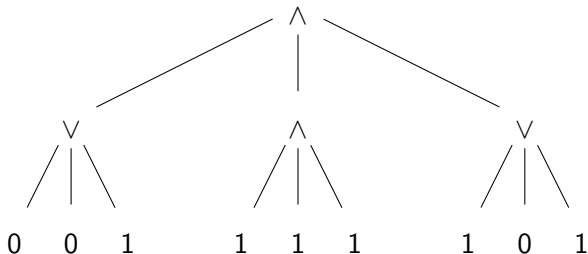
Unranked

- Q states
- $\Delta \subseteq \text{REG}(Q) \times \Sigma \times Q$



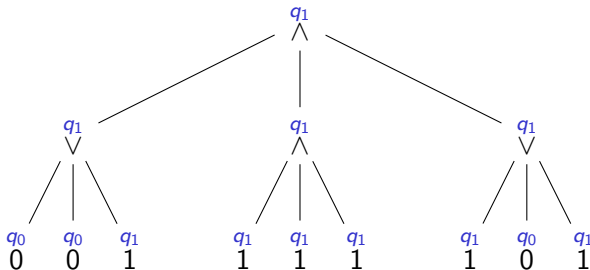
- Bottom-up: F final states
- Top-down: q_0 initial state

EXAMPLE



- States: q_0, q_1
- Transitions: $(\varepsilon, 0, q_0)$, $(\varepsilon, 1, q_1)$
 $(q_0 q_0^*, \vee, q_0)$, $(q_1 q_1^*, \wedge, q_1)$
 $((q_0 + q_1)^* q_0 (q_0 + q_1)^*, \wedge, q_0)$
 $((q_0 + q_1)^* q_1 (q_0 + q_1)^*, \vee, q_1)$

EXAMPLE



- States: q_0, q_1
- Transitions: $(\varepsilon, 0, q_0)$, $(\varepsilon, 1, q_1)$
 $(q_0 q_0^*, \vee, q_0)$, $(q_1 q_1^*, \wedge, q_1)$
 $((q_0 + q_1)^* q_0 (q_0 + q_1)^*, \wedge, q_0)$
 $((q_0 + q_1)^* q_1 (q_0 + q_1)^*, \vee, q_1)$

REPRESENTATIONS

How to represent transitions (L, a, q) ?

How to represent transitions (L, a, q) ?

- Any formalism for regular languages can be used:
regular expressions, NFA, DFA, 2AFA, ...

How to represent transitions (L, a, q) ?

- Any formalism for regular languages can be used:
regular expressions, NFA, DFA, 2AFA, ...
- Merging transitions: $(L_1, a, q), (L_2, a, q) \rightsquigarrow (L_1 \cup L_2, a, q)$

How to represent transitions (L, a, q) ?

- Any formalism for regular languages can be used:
regular expressions, NFA, DFA, 2AFA, ...
- Merging transitions: $(L_1, a, q), (L_2, a, q) \rightsquigarrow (L_1 \cup L_2, a, q)$

Influence on size of representation and complexities

DETERMINISTIC BOTTOM-UP AUTOMATA

Properties for ranked trees (and words):

- efficient minimization
- unique minimal automata (via congruences)

DETERMINISTIC BOTTOM-UP AUTOMATA

Properties for ranked trees (and words):

- efficient minimization
- unique minimal automata (via congruences)

Definition on unranked trees:

- Semantic condition:

$$(L_1, a, q_1), (L_2, a, q_2) \in \Delta \Rightarrow L_1 \cap L_2 = \emptyset \text{ or } q_1 = q_2$$

- Use DFAs to represent transitions

DETERMINISTIC BOTTOM-UP AUTOMATA

Properties for ranked trees (and words):

- efficient minimization
- unique minimal automata (via congruences)

Definition on unranked trees:

- Semantic condition:

$$(L_1, a, q_1), (L_2, a, q_2) \in \Delta \Rightarrow L_1 \cap L_2 = \emptyset \text{ or } q_1 = q_2$$

- Use DFAs to represent transitions

Theorem [Martens, Niehren '05]:

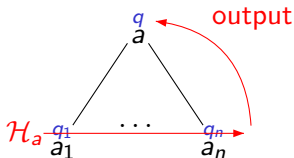
With the above definition the good properties do not transfer.

NORMALIZED DETERMINISTIC BOTTOM-UP AUTOMATA

Model used in [Raeymaekers,Bruynooghe'04], [Cristau,L.,Thomas'05]:

- Transitions represented by det. automata with output (one for each letter from Σ)
- After reading a sequence from Q^* the automaton outputs an element from Q .

$$\mathcal{A} = (Q, (\mathcal{H}_a)_{a \in \Sigma}, F)$$



EXAMPLE

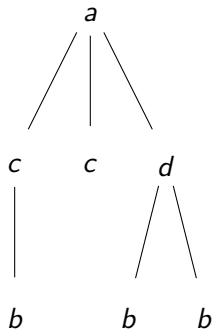
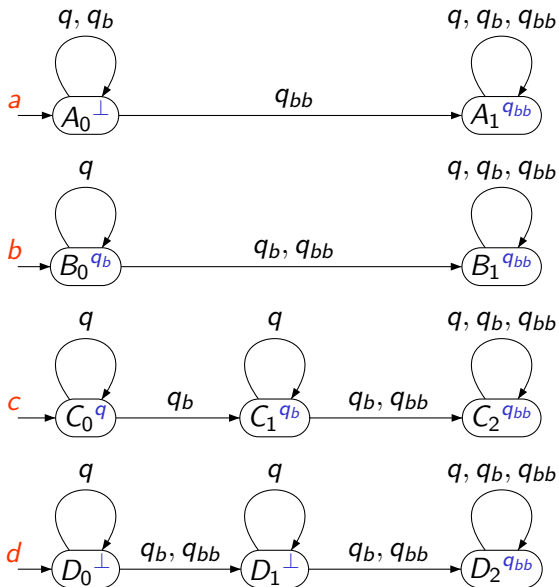
Trees over $\Sigma = \{a, b, c, d\}$ with:

- below each a exists subtree containing at least two b
- below each d exists two subtrees containing at least one b

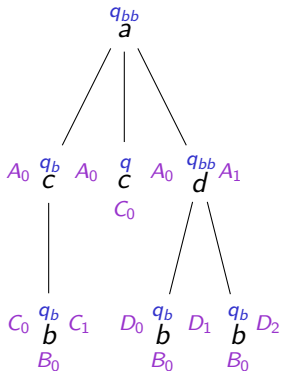
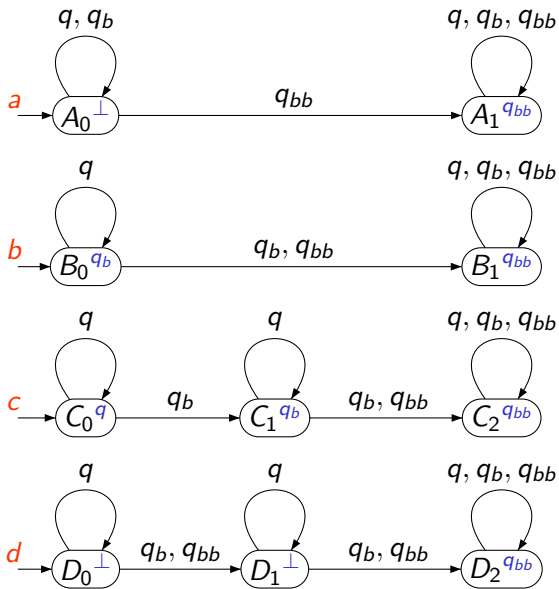
Use states

- q, q_b, q_{bb} signaling the number of b in the subtree (all final states)
- and rejecting sink state \perp

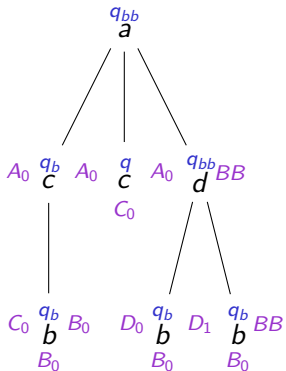
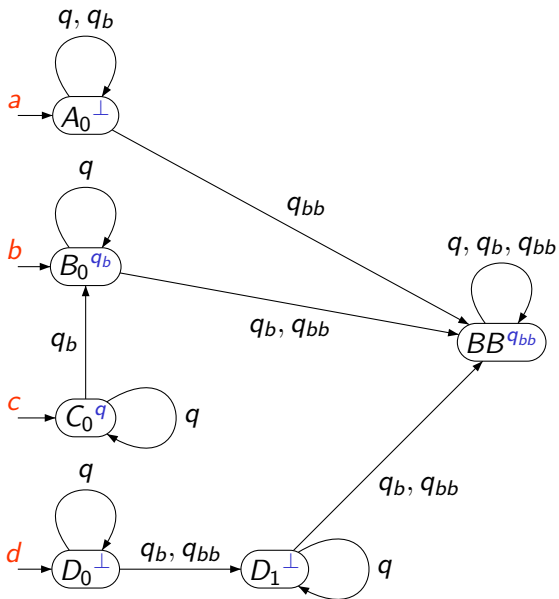
EXAMPLE



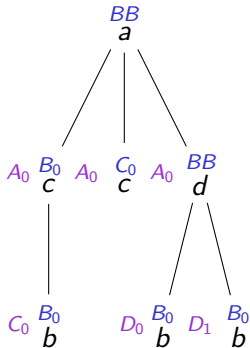
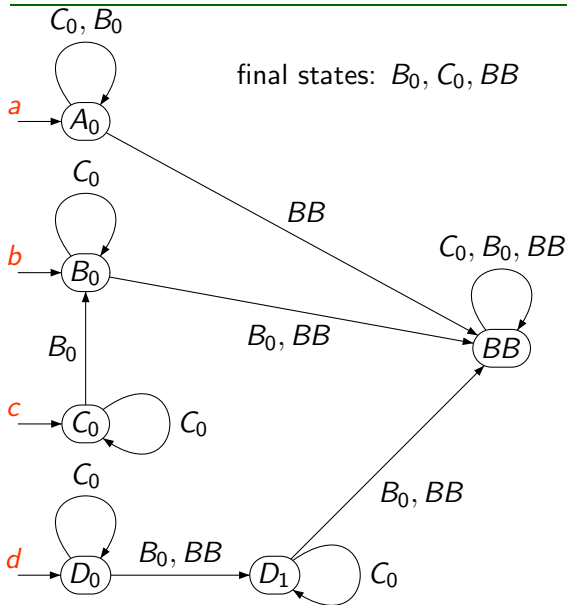
EXAMPLE



EXAMPLE



EXAMPLE



RESULT

Start with model $(Q, (\mathcal{H}_a)_{a \in \Sigma}, F)$



Apply transformations as in example



Obtain $(S, (s_a^{\text{in}})_{a \in \Sigma}, \delta, F')$

- Second model used in [Martens, Niehren'05]
- Both models have good properties w.r.t. minimization

ENCODINGS BY RANKED TREES

ENCODINGS

- Idea: code unranked trees by ranked (binary) trees
“trading width for depth”
- Encoding should preserve regularity of sets of trees
- Allows to use automata theory on ranked trees

ENCODINGS

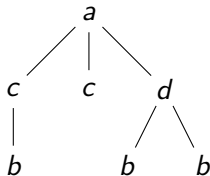
- Idea: code unranked trees by ranked (binary) trees
“trading width for depth”
- Encoding should preserve regularity of sets of trees
- Allows to use automata theory on ranked trees

In this talk:

- First child next sibling / last child previous sibling
- Currying

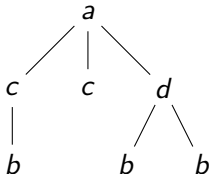
CHILD SIBLING

unranked

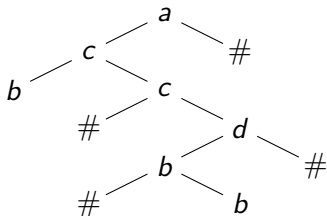


CHILD SIBLING

unranked

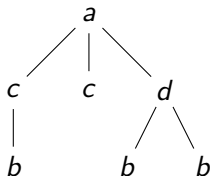


first child next sibling

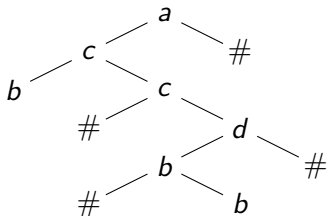


CHILD SIBLING

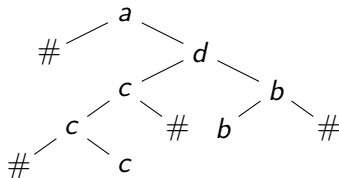
unranked



first child next sibling



last child previous sibling



Remark

A set T of unranked trees is regular iff its encoding $\text{FCNS}(T)$ is.

Consequences

Closure properties and decidability results carry over from ranked trees:

- union, intersection, complement
- emptiness, inclusion, equivalence

PROPERTIES OF CHILD SIBLING CODINGS

Remark

A set T of unranked trees is regular iff its encoding $\text{FCNS}(T)$ is.

Consequences

Closure properties and decidability results carry over from ranked trees:

- union, intersection, complement
- emptiness, inclusion, equivalence

Problems

- Not all complexity results carry over directly

Remark

A set T of unranked trees is regular iff its encoding $\text{FCNS}(T)$ is.

Consequences

Closure properties and decidability results carry over from ranked trees:

- union, intersection, complement
- emptiness, inclusion, equivalence

Problems

- Not all complexity results carry over directly, e.g.
 - Transitions represented by 2AFA on unranked trees: emptiness PSPACE-complete [Martens, Neven '03]
 - 2-way alternating automata on ranked trees: emptiness EXPTIME-complete [Vardi '98, KPV'02]

PROPERTIES OF CHILD SIBLING CODINGS

Remark

A set T of unranked trees is regular iff its encoding $\text{FCNS}(T)$ is.

Consequences

Closure properties and decidability results carry over from ranked trees:

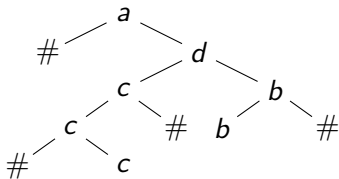
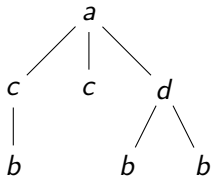
- union, intersection, complement
- emptiness, inclusion, equivalence

Problems

- Not all complexity results carry over directly, e.g.
 - Transitions represented by 2AFA on unranked trees:
emptiness PSPACE-complete [Martens, Neven '03]
 - 2-way alternating automata on ranked trees:
emptiness EXPTIME-complete [Vardi '98, KPV'02]
- Deterministic automata can be exponentially larger on the encodings

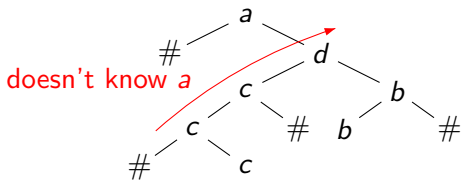
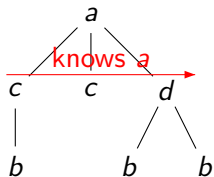
TRANSFERRING DETERMINISTIC AUTOMATA

Illustration of the Problem



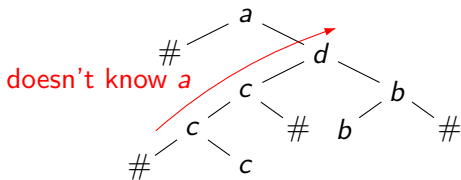
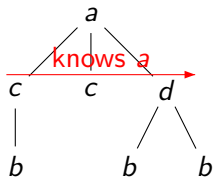
TRANSFERRING DETERMINISTIC AUTOMATA

Illustration of the Problem



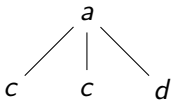
TRANSFERRING DETERMINISTIC AUTOMATA

Illustration of the Problem

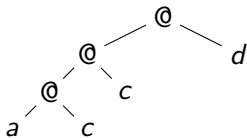
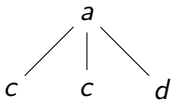


Need an encoding where the label of a node is read "before" the labels of its children.

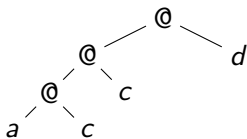
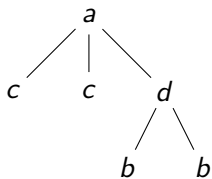
CURRYING



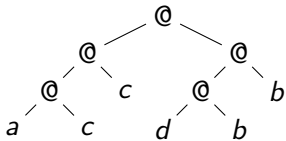
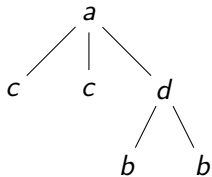
CURRYING



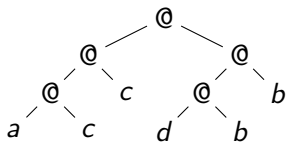
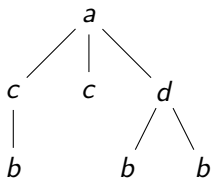
CURRYING



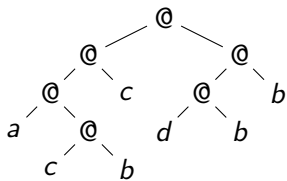
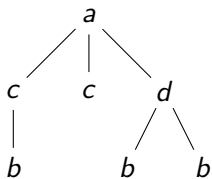
CURRYING

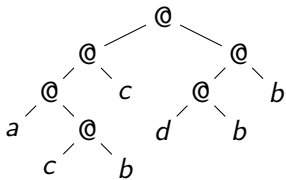
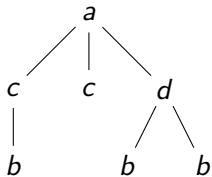


CURRYING



CURRYING



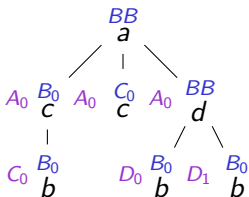
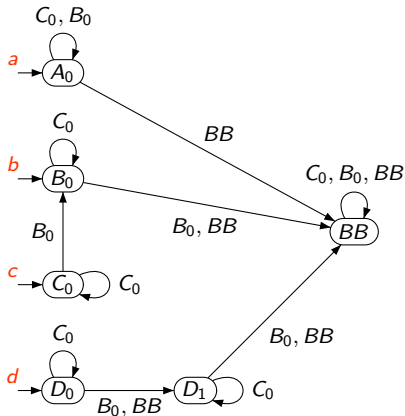


Stepwise automata [Carme,Niehren,Tommasi'04]

- Unranked alphabet \rightsquigarrow symbols at leafs
- Only one binary symbol @
- Automata working on these encodings are called “stepwise automata”

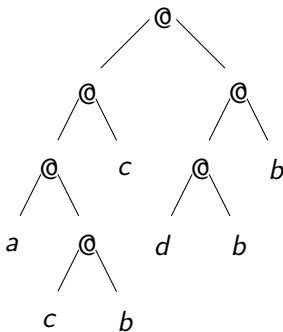
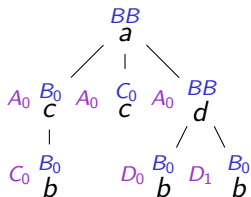
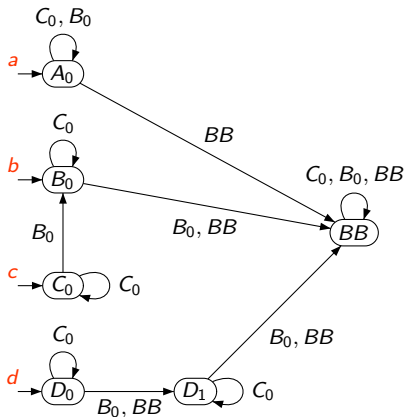
COMPARISON TO UNRANKED TREE AUTOMATA

Previous example:



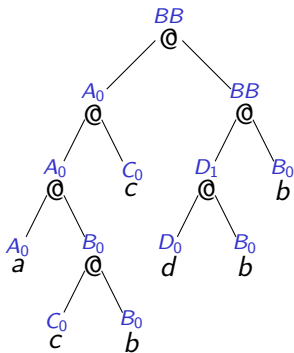
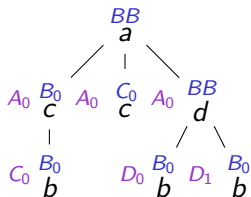
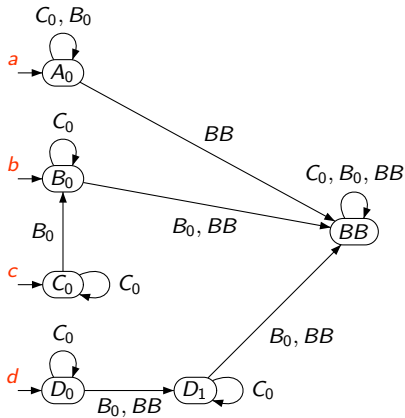
COMPARISON TO UNRANKED TREE AUTOMATA

Previous example:



COMPARISON TO UNRANKED TREE AUTOMATA

Previous example:



CONSEQUENCES

Stepwise automata are unranked automata:

$$\begin{array}{ll} \text{unranked} & \text{stepwise} \\ \xrightarrow{a} q & \leftrightarrow (a, q) \\ q_1 \xrightarrow{q_2} q_3 & \leftrightarrow (q_1, q_2, \textcircled{0}, q_3) \end{array}$$

CONSEQUENCES

Stepwise automata are unranked automata:

$$\begin{array}{lcl} \text{unranked} & & \text{stepwise} \\ \xrightarrow{a} q & \leftrightarrow & (a, q) \\ q_1 \xrightarrow{q_2} q_3 & \leftrightarrow & (q_1, q_2, @, q_3) \end{array}$$

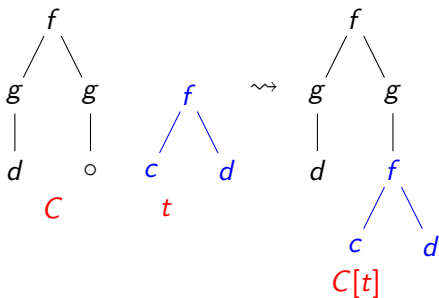
All properties carry over from ranked automata

\rightsquigarrow robust model for deterministic bottom-up automata on unranked trees

Example: Myhill-Nerode Theorem

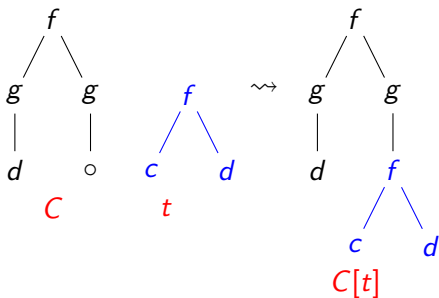
MYHILL-NERODE – RANKED CASE

- Context = tree with a “hole”
- For context C , tree t : $C[t]$ the tree obtained by plugging the root of t into the hole of C



MYHILL-NERODE – RANKED CASE

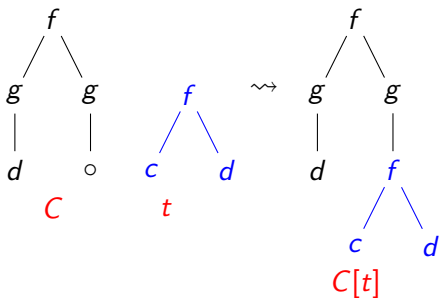
- Context = tree with a “hole”
- For context C , tree t : $C[t]$ the tree obtained by plugging the root of t into the hole of C



$$t_1 \sim_T t_2 \\ \text{iff} \\ \forall C : C[t_1] \in T \Leftrightarrow C[t_2] \in T$$

MYHILL-NERODE – RANKED CASE

- Context = tree with a “hole”
- For context C , tree t : $C[t]$ the tree obtained by plugging the root of t into the hole of C



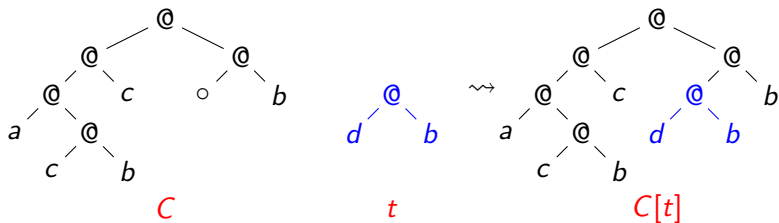
$$t_1 \sim_{\mathcal{T}} t_2 \\ \text{iff} \\ \forall C : C[t_1] \in \mathcal{T} \Leftrightarrow C[t_2] \in \mathcal{T}$$

Theorem

A set \mathcal{T} of ranked trees is regular iff $\sim_{\mathcal{T}}$ has finite index.

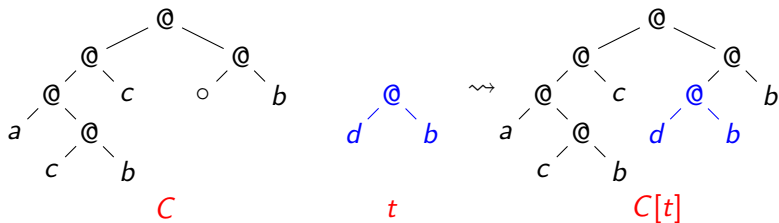
MYHILL-NERODE – UNRANKED CASE

On encodings:

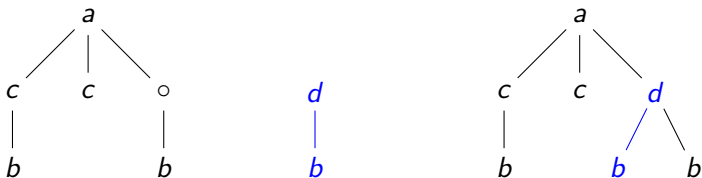


MYHILL-NERODE – UNRANKED CASE

On encodings:

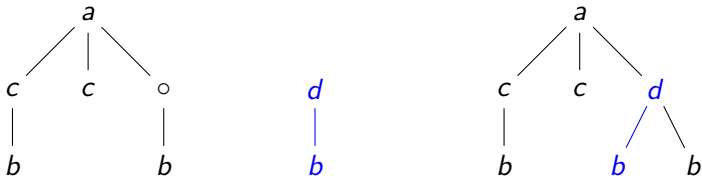


Transferred to unranked trees:



MYHILL-NERODE – UNRANKED CASE

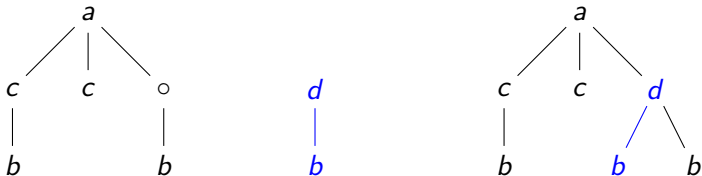
Transferred to unranked trees:



- Context: the hole can appear anywhere in the tree
- Define $C[t]$ as illustrated in example
- Transfer definition of $\sim_{\mathcal{T}}$

MYHILL-NERODE – UNRANKED CASE

Transferred to unranked trees:



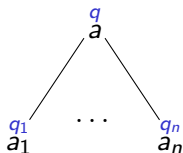
- Context: the hole can appear anywhere in the tree
- Define $C[t]$ as illustrated in example
- Transfer definition of $\sim_{\mathcal{T}}$

Theorem [Martens,Niehren'05]

A set T of unranked trees is regular iff $\sim_{\mathcal{T}}$ has finite index.

TOP-DOWN DETERMINISM

DETERMINISTIC TOP-DOWN AUTOMATA



$q_1 \cdots q_n$ determined by q , a , and n

Semantic definition

$(q, a, L) \in \Delta$: L contains exactly one word for each length n

Syntactic definition

$(q, a, \mathcal{B}) \in \Delta$: \mathcal{B} defines transduction transforming $\underbrace{\bullet \cdots \bullet}_n$ into $q_1 \cdots q_n$

DECIDABILITY QUESTION

Remark

Deterministic top-down automata are strictly weaker than nondeterministic ones.

Question

Given a nondeterministic automaton for unranked trees. Can we decide if there is an equivalent deterministic top-down automaton?

DECIDABILITY QUESTION

Remark

Deterministic top-down automata are strictly weaker than nondeterministic ones.

Question

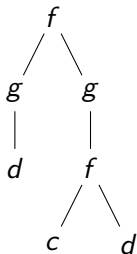
Given a nondeterministic automaton for unranked trees. Can we decide if there is an equivalent deterministic top-down automaton?

Remark

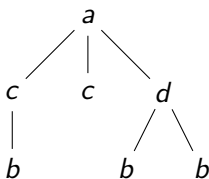
We cannot use encodings as top-down determinism does not transfer.

PATH LANGUAGE

Ranked

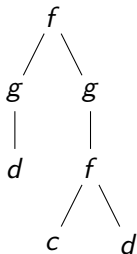


Unranked



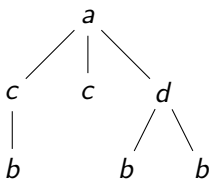
PATH LANGUAGE

Ranked



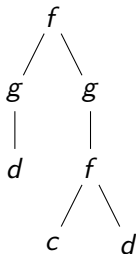
f1g1d
f2g1f1c
f2g1f2d

Unranked



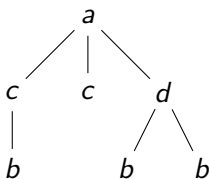
PATH LANGUAGE

Ranked



$f1g1d$
 $f2g1f1c$
 $f2g1f2d$

Unranked

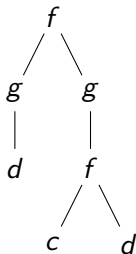


Proposition[Virágh'80]

A regular set T of ranked trees is deterministic top-down recognizable iff $T = \{t \mid \text{Path}(t) \subseteq \text{Path}(T)\}$.

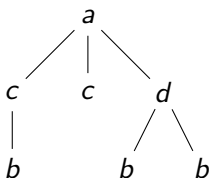
PATH LANGUAGE

Ranked



$f1g1d$
 $f2g1f1c$
 $f2g1f2d$

Unranked



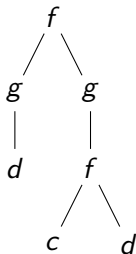
$a \bullet \circ \circ c \bullet b$
 $a \circ \bullet \circ c$
 $a \circ \circ \bullet d \bullet \circ b$
 $a \circ \circ \bullet d \circ \bullet b$

Proposition[Virágh'80]

A regular set T of ranked trees is deterministic top-down recognizable iff $T = \{t \mid \text{Path}(t) \subseteq \text{Path}(T)\}$.

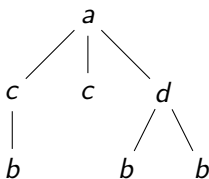
PATH LANGUAGE

Ranked



$f1g1d$
 $f2g1f1c$
 $f2g1f2d$

Unranked



$a \bullet \circ \circ c \bullet b$
 $a \circ \bullet \circ c$
 $a \circ \circ \bullet d \bullet \circ b$
 $a \circ \circ \bullet d \circ \bullet b$

Proposition[Virágh'80]

A regular set T of ranked trees is deterministic top-down recognizable iff $T = \{t \mid \text{Path}(t) \subseteq \text{Path}(T)\}$.

Proposition[Cristau,L.,Thomas'05]

A regular set T of unranked trees is deterministic top-down recognizable iff $T = \{t \mid \text{Path}(t) \subseteq \text{Path}(T)\}$.

Theorem

One can decide, given a nondeterministic automaton for unranked trees, whether there exists an equivalent deterministic top-down automaton.

Theorem

One can decide, given a nondeterministic automaton for unranked trees, whether there exists an equivalent deterministic top-down automaton.

- Construct deterministic word automaton for $\text{Path}(T)$
- Construct deterministic top-down automaton for $\{t \mid \text{Path}(t) \subseteq \text{Path}(T)\}$
- Check equivalence with given automaton

Theorem

One can decide, given a nondeterministic automaton for unranked trees, whether there exists an equivalent deterministic top-down automaton.

- Construct deterministic word automaton for $\text{Path}(T)$
- Construct deterministic top-down automaton for $\{t \mid \text{Path}(t) \subseteq \text{Path}(T)\}$
- Check equivalence with given automaton

Remark

- The result extends to the setting where the transition does not only depend on the number of children but also on their labels.
- Closely related to special types of DTDs as studied in [Martens, Neven, Schwentick'05] and [Martens, Niehren'05]

EXTENSIONS

NUMERICAL CONSTRAINTS

Idea: Enrich transitions with comparisons on the number of occurrences of states.

NUMERICAL CONSTRAINTS

Idea: Enrich transitions with comparisons on the number of occurrences of states.

Transitions (L, a, q) with L given by Presburger regular expression ϕ , a Boolean combination of

- regular expressions over Q
- Presburger formulas with free variables y_q for $q \in Q$.
- $w \in Q^*$ is a model of ϕ if
 - w is in the language defined by the regular expressions and
 - if the Presburger formulas are satisfied with y_q interpreted by the number of q appearing in w .
 - $L(\phi) = \{w \in Q^* \mid w \models \phi\}$

NUMERICAL CONSTRAINTS

Idea: Enrich transitions with comparisons on the number of occurrences of states.

Transitions (L, a, q) with L given by Presburger regular expression ϕ , a Boolean combination of

- regular expressions over Q
- Presburger formulas with free variables y_q for $q \in Q$.
- $w \in Q^*$ is a model of ϕ if
 - w is in the language defined by the regular expressions and
 - if the Presburger formulas are satisfied with y_q interpreted by the number of q appearing in w .
 - $L(\phi) = \{w \in Q^* \mid w \models \phi\}$

Example

$$\phi = \underbrace{(q + p)^* q (q + p)^*}_{\text{at least one } q} \wedge \underbrace{\exists x : y_p = y_q + x}_{\text{more } p \text{ than } q}$$

$$qppqp \models \phi \quad qppq \not\models \phi$$

PROPERTIES OF PRESBURGER AUTOMATA

Results from [Lugiez,Dal Zilio'02],[Seidl,Schwentick,Muscholl'03]:

Theorem

Emptiness is decidable for Presburger automata.

Theorem

Universality is undecidable for Presburger automata.

Corollary

Presburger automata are not closed under complement.

Automata with equality constraints between siblings:

- Check (in)equality of subtrees
- Transitions can be guarded by Boolean combinations of such constraints
- Example: A transition guarded by $1 = 2 \wedge 1 \neq 3$ can only be executed if
 - the first and second subtree of the current node are equal, and
 - the first and third subtree are different.

Theorem [Bogaert, Tison'92]

Automata with equality constraints between siblings on ranked trees are closed under Boolean operations and their emptiness problem is decidable.

Current work (with W. Karianto)

Extension to unranked trees.

Unranked tree automata

- Transitions (L, a, q) with $L \subseteq Q^*$ regular
- A lot of results carry over from ranked trees by using the FCNS-encoding
- For complexity results the detour via encodings can be problematic
- For deterministic automata the currying-encoding gives very good results and compact representations of automata
- Top-down determinism can be handled similarly as for ranked trees
- Emptiness remains decidable for automata extended by numerical comparisons