Tree Automata and Applications

M1 course, 2023/2024

◆□ ▶ < □ ▶ < Ξ ▶ < Ξ ▶ < Ξ < つ < ℃ 1/89</p>

Organization

Timetable

- Exercises: Thursday 8:30 10:30 (Luc Lapointe)
- Course: Thursday 10:45 12:45 (Stefan Schwoon)

Exams

- DM or CC (to be specified by Luc)
- Final Exam: 2h, 11 January
- First session: DM/CC + Exam (50/50)
- Second session: DM/CC + Repeat Exam (50/50)

Course materials

- Website: lecturer's homepage + Wiki MPRI, course 1-18 (exercise sheets, slides, former exams)
- Hubert Comon et al.

Tree Automata Techniques and Applications. http://tata.gforge.inria.fr/

Motivations

- Natural extension of formal-language notions (automata, logic, ...)
- Treatment of tree-like data structures: parse tree, XML documents (XPath, CSS selectors)
- O Applications e.g. in compiler construction, formal verification

Trees

We consider *finite ordered ranked* trees.

- ordered : internal nodes have children 1...n
- *ranked* : number of children fixed by node's label

Let N denote the set of positive integers.

Nodes (*positions*) of a tree are associated with elements of N^* :



Definition: Tree

A (finite, ordered) *tree* is a non-empty, finite, prefix-closed set $Pos \subseteq N^*$ such that $w(i + 1) \in Pos$ implies $wi \in Pos$ for all $w \in N^*$, $i \in N$.

Ranked Trees

Ranked symbols

Let $\mathcal{F}_0, \mathcal{F}_1, \ldots$ be disjoint sets of symbols of arity $0, 1, \ldots$ We note $\mathcal{F} := \bigcup_i \mathcal{F}_i$.

• Notation (example): $\mathcal{F} = \{f(2), g(1), a, b\}$

Let \mathcal{X} denote a set of variables (disjoint from the other symbols).

Definition: Ranked tree

A ranked tree is a mapping $t : Pos \rightarrow (\mathcal{F} \cup \mathcal{X})$ satisfying:

- Pos is a tree;
- for all $p \in Pos$, if $t(p) \in \mathcal{F}_n$, $n \ge 1$ then $Pos \cap pN = \{p1, \dots, pn\}$;
- for all $p \in Pos$, if $t(p) \in \mathcal{X} \cup \mathcal{F}_0$ then $Pos \cap pN = \emptyset$.

Trees and Terms

Definition: Terms

The set of *terms* $T(\mathcal{F}, \mathcal{X})$ is the smallest set satisfying:

•
$$\mathcal{X} \cup \mathcal{F}_0 \subseteq T(\mathcal{F}, \mathcal{X});$$

• if $t_1, \ldots, t_n \in T(\mathcal{F}, \mathcal{X})$ and $f \in \mathcal{F}_n$, then $f(t_1, \ldots, t_n) \in T(\mathcal{F}, \mathcal{X})$. We note $T(\mathcal{F}) := T(\mathcal{F}, \emptyset)$. A term in $T(\mathcal{F})$ is called *ground term*. A term of $T(\mathcal{F}, \mathcal{X})$ is *linear* if every variable occurs at most once.

Example:
$$\mathcal{F} = \{f(2), g(1), a, b\}$$
, $\mathcal{X} = \{x, y\}$

- $f(g(a), b) \in T(\mathcal{F});$
- $f(x, f(b, y)) \in T(\mathcal{F}, \mathcal{X})$ is linear;
- $f(x,x) \in T(\mathcal{F},\mathcal{X})$ is non-linear.

We confuse terms and trees in the obvious manner.

Height and size

Definition

Let $t \in T(\mathcal{F}, \mathcal{X})$. We note $\mathcal{H}(t)$ the *height* of t and |t| the *size* of t.

- if $t \in \mathcal{X}$, then $\mathcal{H}(t) := 0$ and |t| := 0; (for notational convenience)
- if $t \in \mathcal{F}_0$, then $\mathcal{H}(t) := 1$ and |t| := 1;
- if $t = f(t_1, ..., t_n)$, then $\mathcal{H}(t) := 1 + \max\{\mathcal{H}(t_1), ..., \mathcal{H}(t_n)\}$ and $|t| := 1 + |t_1| + \cdots + |t_n|$.

< □ > < @ > < E > < E > ○ Q @ 7/89

Subterms / subtrees

Definition: Subtree

Let $t, u \in T(\mathcal{F}, \mathcal{X})$ and p a position. Then $t|_p : Pos_p \to T(\mathcal{F}, \mathcal{X})$ is the ranked tree defined by

- $Pos_p := \{ q \mid pq \in Pos \};$
- $t|_{p}(q) := t(pq).$

Moreover, $t[u]_p$ is the tree obtained by replacing $t|_p$ by u in t.

 $t \ge t'$ (resp. t > t') denotes that t' is a (proper) subtree of t.

Substitutions and Context

Definition: Substitution

- (Ground) substitution σ : mapping from \mathcal{X} to $\mathcal{T}(\mathcal{F}, \mathcal{X})$ resp. $\mathcal{T}(\mathcal{F})$
- Notation: $\sigma := \{x_1 \leftarrow t_1, \dots, x_n \leftarrow t_n\}$, with $\sigma(x) := x$ for all $x \in \mathcal{X} \setminus \{x_1, \dots, x_n\}$
- Extension to terms: for all $f \in \mathcal{F}_m$ and $t'_1, \ldots, t'_m \in \mathcal{T}(\mathcal{F}, \mathcal{X})$ $\sigma(f(t'_1, \ldots, t'_m)) = f(\sigma(t'_1), \ldots, \sigma(t'_m))$
- Notation: $t\sigma$ for $\sigma(t)$

Definition: Context

A context is a linear term $C \in T(\mathcal{F}, \mathcal{X})$ with variables x_1, \ldots, x_n . We note $C[t_1, \ldots, t_n] := C\{x_1 \leftarrow t_1, \ldots, x_n \leftarrow t_n\}.$

 $\mathcal{C}^{n}(\mathcal{F})$ denotes the contexts with *n* variables and $\mathcal{C}(\mathcal{F}) := \mathcal{C}^{1}(\mathcal{F})$. Let $\mathcal{C} \in \mathcal{C}(\mathcal{F})$. We note $\mathcal{C}^{0} := x_{1}$ and $\mathcal{C}^{n+1} = \mathcal{C}^{n}[\mathcal{C}]$ for $n \geq 0$.

Tree automata

Basic idea: Extension of finite automata from words to trees Direct extension of automata theory when words seen as unary terms:

 $abc \cong a(b(c(\$)))$

Finite automaton: labels every prefix of a word with a state. Tree automaton: labels every position/subtree of a tree with a state. Two variants: bottom-up vs top-down labelling

Basic results (preview)

- Non-deterministic bottom-up and top-down are equally powerful
- Deterministic bottom-up equally powerful
- Deterministic top-down less powerful

Bottom-up automata

Definition: (Bottom-up tree automata)

A (finite bottom-up) tree automaton (NFTA) is a tuple $\mathcal{A} = \langle Q, \mathcal{F}, G, \Delta \rangle$, where:

- Q is a finite set of *states*;
- \mathcal{F} a finite ranked alphabet;
- G ⊆ Q are the final states;
- Δ is a finite set of rules of the form

$$f(q_1,\ldots,q_n) \to q$$

for $f \in \mathcal{F}_n$ and $q, q_1, \ldots, q_n \in Q$.

Example: $Q := \{q_0, q_1, q_f\}, \ \mathcal{F} = \{f(2), g(1), a\}, \ G := \{q_f\}, \ \text{and rules}$ $a \to q_0 \quad g(q_0) \to q_1 \quad g(q_1) \to q_1 \quad f(q_1, q_1) \to q_f$

Move relation and computation tree

Move relation

Let $t, t' \in T(\mathcal{F}, Q)$. We write $t \to_{\mathcal{A}} t'$ if the following are satisfied:

- $t = C[f(q_1, \ldots, q_n)]$ for some context C;
- t' = C[q] for some rule $f(q_1, \ldots, q_n) \rightarrow q$ of \mathcal{A} .

Idea: successively reduce t to a single state, starting from the leaves. As usual, we write \rightarrow_{A}^{*} for the transitive and reflexive closure of \rightarrow_{A} .

Computation

Let $t : Pos \to \mathcal{F}$ a ground tree. A *run* or *computation* of \mathcal{A} on t is a labelling $t' : Pos \to Q$ compatible with Δ , i.e.:

• for all $p \in Pos$, if $t(p) = f \in \mathcal{F}_n$, t'(p) = q, and $t'(pj) = q_j$ for all $pj \in Pos \cap pN$, then $f(q_1, \ldots, q_n) \to q \in \Delta$

Regular tree languages

◆□ ▶ ◆ □ ▶ ◆ □ ▶ ◆ □ ▶ ● □ ● ○ ○ ○ 13/89

A tree *t* is *accepted* by \mathcal{A} iff $t \to_{\mathcal{A}}^{*} q$ for some $q \in G$.

 $\mathcal{L}(\mathcal{A})$ denotes the set of trees accepted by \mathcal{A} .

L is *regular/recognizable* iff $L := \mathcal{L}(\mathcal{A})$ for some NFTA \mathcal{A} .

Two NFTAs A_1 and A_2 are *equivalent* iff $\mathcal{L}(A_1) = \mathcal{L}(A_2)$.

NFTA with ε -moves

Definition:

An ε -NFTA is an NFTA $\mathcal{A} = \langle Q, \mathcal{F}, G, \Delta \rangle$, where Δ can additionally contain rules of the form $q \rightarrow q'$, with $q, q' \in Q$.

Semantics: Allow to re-label a position from q to q'.

Equivalence of ε -NFTA

For every ε -NFTA \mathcal{A} there exists an equivalent NFTA \mathcal{A}' .

Proof (sketch): Construct the rules of \mathcal{A}' by a saturation procedure.

Deterministic, complete, and reduced NFTA

An NFTA is *deterministic* if no two rules have the same left-hand side. An NFTA is *complete* if for every $f \in \mathcal{F}_n$ and $q_1, \ldots, q_n \in Q$, there exists at least one rule $f(q_1, \ldots, q_n) \rightarrow q \in \Delta$.

As usual, a DFTA has *at most* one run per tree. A DCFTA as *exactly* one run per tree.

A state q of \mathcal{A} is accessible if there exists a tree t s.t. $t \to_{\mathcal{A}}^{*} q$. \mathcal{A} is said to be reduced if all its states are accessible.

A pumping lemma for tree languages

Lemma

Let *L* be recognizable. Then there exists a constant *k* such that for all $t \in L$ with $\mathcal{H}(t) > k$ there exist contexts $C, D \in \mathcal{C}(\mathcal{F})$ and $u \in \mathcal{T}(\mathcal{F})$ satisfying:

- D is non-trivial (i.e. not just a variable);
- t = C[D[u]];
- for all $n \ge 0$, we have $C[D^n[u]] \in L$.

Proof: Let *k* be the number of states of an NFTA \mathcal{A} recognizing *L*. Then an accepting run for *t* has positions $p, pp' (p' \neq \varepsilon)$ labelled with the same state *q*. Let $C := t[x]_p$, $D := t|_p[x]_{p'}$, and $u := t|_{pp'}$. We have $t = C[D[u]] \in L$, $D[u] \rightarrow_{\mathcal{A}}^* q$, and $u \rightarrow_{\mathcal{A}}^* q$, hence the accepting run of *t* implies $D[q] \rightarrow_{\mathcal{A}}^* q$ and $C[q] \rightarrow_{\mathcal{A}}^* q_f$, for some final q_f . Therefore, $C[u] \rightarrow_{\mathcal{A}}^* q_f$ and for any $n \ge 0$, (by induction) $C[D^{n+1}[u]] \rightarrow_{\mathcal{A}}^* C[D^n[D[q]]] \rightarrow_{\mathcal{A}}^* C[D^n[q]] \rightarrow_{\mathcal{A}}^* C[q] \rightarrow_{\mathcal{A}}^* q_f$

Illustration of pumping lemma

Let $L = \{ f(g^i(a), g^i(a)) \mid i \ge 0 \}$ for $\mathcal{F} = \{ f(2), g(1), a \}$. Suppose (by contradiction) that *L* is recognizable by NFTA \mathcal{A} with *k* states. Let $t = f(g^k(a), g^k(a))$.



Pumping D creates trees outside $L \Rightarrow L$ not recognizable.

Top-down tree automata

Definition

A top-down tree automaton (T-NFTA) is a tuple $\mathcal{A} = \langle Q, \mathcal{F}, I, \Delta \rangle$, where Q, \mathcal{F} are as in NFTA, $I \subseteq Q$ is a set of *initial states*, and Δ contains rules of the form

$$q(f) \rightarrow (q_1, \ldots, q_n)$$

for $f \in \mathcal{F}_n$ and $q, q_1, \ldots, q_n \in Q$.

Move relation: $t \rightarrow_{\mathcal{A}} t'$ iff

t = C[q(f(t₁,...,t_n))] for some context C, f ∈ F_n, and t₁,..., t_n ∈ T(F);
t' = C[f(q₁(t₁),...,q_n(t_n))] for some rule q(f) → (q₁,...,q_n).
t is accepted by A if q(t) →^{*}_A t for some q ∈ I.

From top-down to bottom-up

Theorem (T-NFTA = NFTA)

L is recognizable by an NFTA iff it is recognizable by a T-NFTA.

Claim: *L* is accepted by NFTA $\mathcal{A} = \langle Q, \mathcal{F}, G, \Delta \rangle$ iff it is accepted by T-NFTA $\mathcal{A}' = \langle Q, \mathcal{F}, G, \Delta' \rangle$, with

$$\Delta' := \{ q(f) \rightarrow (q_1, \ldots, q_n) \mid f(q_1, \ldots, q_n) \rightarrow q \in \Delta \}$$

Proof: Let $t \in T(\mathcal{F})$. We show $t \rightarrow^*_{\mathcal{A}} q$ iff $q(t) \rightarrow^*_{\mathcal{A}'} t$.

• Base:
$$t = a$$
 (for some $a \in \mathcal{F}_0$)
 $t = a \rightarrow^*_{\mathcal{A}} q \iff a \rightarrow_{\Delta} q \iff q(a) \rightarrow_{\Delta'} \varepsilon \iff q(a) \rightarrow^*_{\mathcal{A}'} a$

• Induction:
$$t = f(t_1, \ldots, t_n)$$
, hypothesis holds for t_1, \ldots, t_n
 $f(t_1, \ldots, t_n) \rightarrow^*_{\mathcal{A}} q \iff \exists q_1, \ldots q_n : f(q_1, \ldots, q_n) \rightarrow_{\Delta} q \land \forall i : t_i \rightarrow^*_{\mathcal{A}} q_i$
 $\iff \exists q_1, \ldots, q_n : q(f) \rightarrow_{\Delta'} (q_1, \ldots, q_n) \land \forall i : q_i(t_i) \rightarrow^*_{\mathcal{A}'} t_i$
 $\iff q(f(t_1, \ldots, t_n)) \rightarrow_{\mathcal{A}'} f(q_1(t_1), \ldots, q_n(t_n)) \rightarrow^*_{\mathcal{A}'} f(t_1, \ldots, t_n)$

From NFTA to DFTA

Theorem (NFTA=DFTA)

If L is recognizable by an NFTA, then it is recognizable by a DFTA.

Claim (subset construct.): Let $\mathcal{A} = \langle Q, \mathcal{F}, G, \Delta \rangle$ an NFTA recognizing L. The following DCFTA $\mathcal{A}' = \langle 2^Q, \mathcal{F}, G', \Delta' \rangle$ also recognizes L:

•
$$G' = \{ S \subseteq Q \mid S \cap G \neq \emptyset \}$$

• for every $f \in \mathcal{F}_n$ and $S_1, \ldots, S_n \subseteq Q$, let $f(S_1, \ldots, S_n) \to S \in \Delta'$, where $S = \{ q \in Q \mid \exists q_1 \in S_1, \ldots, q_n \in S_n : f(q_1, \ldots, q_n) \to q \in \Delta \}$

Proof: For $t \in T(\mathcal{F})$, show $t \rightarrow^*_{\mathcal{A}'} \{ q \mid t \rightarrow^*_{\mathcal{A}} q \}$, by structural induction.

DFTA with accessible states

In practice, the construction of \mathcal{A}' can be restricted to accessible states: Start with transitions $a \to S$, then saturate.

Deterministic top-down are less powerful E.g., $L = \{f(a, b), f(b, a)\}$ can be recognized by DFTA but not by T-DFTA.

Closure properties

Theorem (Boolean closure)

Recognizable tree languages are closed under Boolean operations.

Negation (invert accepting states) Let $\langle Q, \mathcal{F}, G, \Delta \rangle$ be a DCFTA recognizing *L*. Then $\langle Q, \mathcal{F}, Q \setminus G, \Delta \rangle$ recognizes $\mathcal{T}(\mathcal{F}) \setminus L$.

Union (juxtapose)

Let $\langle Q_i, \mathcal{F}, G_i, \Delta_i \rangle$ be NFTA recognizing L_i , for i = 1, 2. Then $\langle Q_1 \uplus Q_2, \mathcal{F}, G_1 \cup G_2, \Delta_1 \cup \Delta_2 \rangle$ recognizes $L_1 \cup L_2$.

Cross-product construction

Direct intersection

Let $A_i = \langle Q_i, \mathcal{F}, G_i, \Delta_i \rangle$ be NFTA recognizing L_i , for i = 1, 2. Then $A = \langle Q_1 \times Q_2, \mathcal{F}, G_1 \times G_2, \Delta \rangle$ recognizes $L_1 \cap L_2$, where

$$\frac{f(q_1,\ldots,q_n) \to q \in \Delta_1 \quad f(q_1',\ldots,q_n') \to q' \in \Delta_2}{f(\langle q_1,q_1'\rangle,\ldots,\langle q_n,q_n'\rangle) \to \langle q,q'\rangle \in \Delta}$$

Remarks:

- If A_1, A_2 are D(C)FTA, then so is A.
- If A₁, A₂ are complete, replace G₁ × G₂ with (G₁ × Q₂) ∪ (Q₁ × G₂) to recognize L₁ ∪ L₂.

▲□▶▲□▶▲三▶▲三▶ 三 のへで 22/89

Tree languages and context-free languages

Front

Let t be a ground tree. Then $fr(t) \in \mathcal{F}_0^*$ denotes the word obtained from reading the leaves from left to right (in increasing lexicographical order of their positions).

Example:
$$t = f(a, g(b, a), c)$$
, $fr(t) = abac$

Leaf languages

- Let L be a recognizable tree language. Then fr(L) is context-free.
- Let L be a context-free language that does not contain the empty word. Then there exists an NFTA A with $L = fr(\mathcal{L}(A))$.

Proof (idea):

- Given a T-NFTA recognizing L, construct a CFG from it.
- L is generated by a CFG using productions of the form A → BC | a only. Replace A → BC by A → A₂ and A₂ → BC, construct a T-NFTA from the result.

Visibly pushdown automata

Visibly pushdown automaton

Let $\mathcal{A} = \langle Q, \Sigma, \Gamma, T, q_0 z_0, F \rangle$ be a pushdown automaton.

 \mathcal{A} is called visibly pushdown (VPA) if there exist $\Sigma_0, \Sigma_1, \Sigma_2$ such that

• $\Sigma = \Sigma_0 \uplus \Sigma_1 \uplus \Sigma_2$

•
$$T \subseteq \bigcup_{i=0}^{2} (Q \times \Gamma) \times \Sigma_{i} \times (Q \times \Gamma^{i})$$

Closure properties

Languages accepted by VPA are closed under boolean operations.

VPA and tree languages

Let $L \subseteq T(\mathcal{F})$ be a recognizable tree language.

Then L, seen as a word language of terms, is accepted by a VPA.

From TA to VPA

Let $\mathcal{A} = \langle Q, \mathcal{F}, I, \Delta \rangle$ be a T-NFTA accepting *L*.

For convenience, assume $I = \{q_0\}$ is a singleton (closure under union). We construct a single-state VPA $\mathcal{B} = \langle \Sigma, \Gamma, T, q_0 \rangle$ accepting by empty stack and recognizing the terms of L (can be converted into a normal VPA).

•
$$\Sigma_0 = \mathcal{F}_0 \cup \{ \}$$
, $\Sigma_1 = \mathcal{F} \setminus \mathcal{F}_0$, $\Sigma_2 = \{ , , (\}$
• $\Gamma = Q \cup \{ r_i \mid r \in \Delta, r = q(f) \rightarrow (q_1, \dots, q_n), n \ge 1, 0 \le i \le n \}$
• $T = \bigcup_{r \in \Delta} T_r$
• for $r = q(a) \rightarrow \varepsilon$, we have $T_r = \{ \langle q, a, \varepsilon \rangle \}$;
• for $r = q(f) \rightarrow (q_1, \dots, q_n), n \ge 1$, we have
 $T_r = \{ \langle q, f, r_0 \rangle, \langle r_0, (, q_1 r_1), \langle r_n,) , \varepsilon \rangle \}$
 $\cup \{ \langle r_i, , , q_{i+1} r_{i+1} \rangle \mid 1 \le i < n \}$

▲□▶▲@▶▲≣▶▲≣▶ ≣ のQ@ 25/89

Idea: $q \stackrel{t}{\rightarrow} {}^{*}_{\mathcal{B}} \varepsilon$ iff $q(t) \rightarrow {}^{*}_{\mathcal{A}} t$

From TA to VPA: Example

Consider a T-NFTA $\langle Q, \mathcal{F}, I, \Delta \rangle$ accepting $L = \{ f(g^i(a)) \mid i \geq 0 \}$:

•
$$Q = \{q_0, q_1, q_f\}, \ \mathcal{F} = \{f(2), g(1), a\}, \ I = \{q_f\};$$

• $\Delta := \{ \alpha : q_0(a) \to \varepsilon, \quad \beta : q_1(g) \to q_0, \quad \gamma : q_1(g) \to q_1, \quad \delta : q_f(f) \to (q_1, q_1) \}.$

We construct the single-state VPA $\langle \Sigma, \Gamma, T, q_f \rangle$, where:

•
$$\Sigma_0 = \{a, \}, \Sigma_1 = \{f, g\}, \Sigma_2 = \{,, (\}\}$$

•
$$\Gamma = Q \cup \{\beta_0, \beta_1, \gamma_0, \gamma_1, \delta_0, \delta_1, \delta_2\};$$

•
$$T_{\alpha} = \{ \langle q_0, a, \varepsilon \rangle \};$$

•
$$T_{\beta} = \{ \langle q_1, g, \beta_0 \rangle, \langle \beta_0, (, q_0 \beta_1 \rangle, \langle \beta_1,) \varepsilon \rangle \};$$

•
$$T_{\gamma} = \{ \langle q_1, g, \gamma_0 \rangle, \langle \gamma_0, (, q_1 \gamma_1 \rangle, \langle \gamma_1,) \varepsilon \rangle \};$$

•
$$T_{\delta} = \{ \langle q_f, f, \delta_0 \rangle, \langle \delta_0, (, q_1 \delta_1 \rangle, \langle \delta_1, , , q_1 \delta_2 \rangle, \langle \delta_2,) \varepsilon \rangle \}.$$

Run on
$$f(g(a), g(g(a)))$$
:

$$q_{f} \xrightarrow{f} \delta_{0} \xrightarrow{(} q_{1}\delta_{1} \xrightarrow{g} \beta_{0}\delta_{1} \xrightarrow{(} q_{0}\beta_{1}\delta_{1} \xrightarrow{a} \beta_{1}\delta_{1} \xrightarrow{)} \delta_{1} \xrightarrow{,} q_{1}\delta_{2} \xrightarrow{g} \gamma_{0}\delta_{2} \xrightarrow{(} q_{1}\gamma_{1}\delta_{2} \xrightarrow{g} \beta_{0}\gamma_{1}\delta_{2} \xrightarrow{(} q_{0}\beta_{1}\gamma_{1}\delta_{2} \xrightarrow{a} \beta_{1}\gamma_{1}\delta_{2} \xrightarrow{)} \gamma_{1}\delta_{2} \xrightarrow{)} \delta_{2} \xrightarrow{)} \varepsilon$$

Tree homomorphism

▲□▶▲@▶▲≣▶▲≣▶ ≣ のQ@ 27/89

Definition

Let $\mathcal{X}_n := \{x_1, \ldots, x_n\}$ and $\mathcal{F}, \mathcal{F}'$ ranked alphabets. A tree homomorphism is a mapping $h : \mathcal{F} \to T(\mathcal{F}', \mathcal{X})$, with $h(f) \in T(\mathcal{F}, \mathcal{X}_n)$ if $f \in \mathcal{F}_n$.

Extension of h to trees $(T(\mathcal{F}) \rightarrow T(\mathcal{F}'))$:

•
$$h(f(t_1,\ldots,t_n)) = h(f)\{x_1 \leftarrow h(t_1),\ldots,x_n \leftarrow h(t_n)\}$$

Intuition:

- h(f) "explodes" f-positions into trees
- reorders/copies/deletes subtrees.

Examples

Example



Example (ternary to binary tree)

•
$$\mathcal{F} = \{f(3), a, b\}, \ \mathcal{F}' = \{g(2), a, b\}$$

•
$$h_{32}(f) = g(x_1, g(x_2, x_3)), h_{32}(a) = a, h_{32}(b) = b$$

로▶▲ 로▶ 로 *) 역 (* 28/89

Properties of homomorphisms

A homomorphism h is

- *linear* if h(f) linear for all f;
- non-erasing if $\mathcal{H}(h(f)) > 0$ for all f;
- flat if $\mathcal{H}(h(f)) = 1$ for all f;
- complete if $f \in \mathcal{F}_n$ implies that h(f) contains all of \mathcal{X}_n ;
- permuting if h is complete, linear, and flat;
- alphabetic if h(f) has the form $g(x_1, \ldots, x_n)$ for all f.

Example: h_{32} is linear, non-erasing, and complete.

Non-linear homomorphisms do not preserve recognizability

- Example: $h(f) = f'(x_1, x_1)$, $h(g) = g(x_1)$, h(a) = a
- $L = \{ f(g^i(a)) \mid i \ge 0 \}$ (recognizable)
- $h(L) = \{ f'(g^i(a), g^i(a)) \mid i \ge 0 \}$ (not recognizable)

Linear homomorphisms

Theorem: Linear homomorphisms preserve recognizability

Let $L \subseteq T(\mathcal{F})$ be recognizable and $h : \mathcal{F} \to \mathcal{F}'$ a linear tree homomorphism. Then h(L) is recognizable.

Illustrating example:

•
$$\mathcal{F} = \{f(2), g(1), a\}, \ \mathcal{F}' = \{f'(1), g'(2), c, d\}$$

• $h(f) = f'(g'(x_2, d)), \ h(g) = g'(x_1, c), \ h(a) = g'(c, d)$
• $L = \{f(g^i(a), g^k(a)) \mid i, k \ge 0\}$
• $\mathcal{A} = \langle \{q_0, q_1, q_f\}, \mathcal{F}, \{q_f\}, \Delta \rangle$ recognizes L with
 $\Delta := \{\alpha : a \to q_0, \beta : g(q_0) \to q_1, \gamma : g(q_1) \to q_1, \delta : f(q_1, q_1) \to q_f\}$



Automaton construction for h(L)

Given a reduced NFTA $\mathcal{A} = \langle Q, \mathcal{F}, G, \Delta \rangle$ for *L*, construct NFTA $\mathcal{A}' = \langle Q', \mathcal{F}', G, \Delta' \rangle$ for *h*(*L*).

- $Q' := Q \cup \{ \langle r, p \rangle \mid r \in \Delta, \exists f \in \mathcal{F} : r = f(\ldots) \rightarrow \ldots, p \in Pos_{h(f)} \};$
- Δ' contains, for each transition $r : f(s_1, \ldots, s_n) \to s$ in Δ and $p \in Pos_{h(f)}$:
 - $f'(\langle r, p1 \rangle, \dots, \langle r, pk \rangle) \rightarrow \langle r, p \rangle$ if $h(f)(p) = f' \in \mathcal{F}'_k$ • $s_i \rightarrow \langle r, p \rangle$ if $h(f)(p) = x_i$
 - $s_i \rightarrow \langle r, p \rangle$ if $h(f)(p) = x_i$
 - $\langle r, \varepsilon \rangle \rightarrow s$



Correctness

To prove: \mathcal{A}' accepts h(L).

- $h(L) \subseteq \mathcal{L}(\mathcal{A}')$: For $t \in \mathcal{T}(\mathcal{F})$, prove that $t \to_{\mathcal{A}}^{*} q$ implies $h(t) \to_{\mathcal{A}'}^{*} q$, by structural induction over t.
- h(L) ⊇ L(A'): For t' ∈ T(F'), prove that if t' →^{*}_{A'} q ∈ Q, then there exists t ∈ T(F) ∩ h⁻¹(t') with t →^{*}_A q, by induction on number of states (of Q) in the computation t' →^{*}_{A'} q.

Inverse tree homomorphisms

Theorem: Inverse homomorphisms preserve recognizability

Let $L \subseteq T(\mathcal{F}')$ be recognizable and $h : \mathcal{F} \to \mathcal{F}'$ a tree homomorphism (not necessarily linear). Then $h^{-1}(L)$ is recognizable.

Given an NFTA
$$\mathcal{A}' = \langle Q, \mathcal{F}', G, \Delta' \rangle$$
 for L ,
construct NFTA $\mathcal{A} = \langle Q \uplus \{!\}, \mathcal{F}, G, \Delta \rangle$ for $h^{-1}(L)$.
For all $n \ge 0$ and $f \in \mathcal{F}_n$, and $p_1, \ldots, p_n \in Q$,
• add $f(!, \ldots, !) \to !$ to Δ ;
• if $h(f)\{x_1 \leftarrow p_1, \ldots, x_n \leftarrow p_n\} \to_{\mathcal{A}'}^* q$, add $f(q_1, \ldots, q_n) \to q$ to Δ ,
with:
 $q_i = \begin{cases} p_i & \text{if } x_i \text{ appears in } h(f) \\ ! & \text{otherwise} \end{cases}$

◆□▶◆□▶◆豆▶◆豆▶ 豆 のへで 33/89

Proof: Show $t \to_{\mathcal{A}}^{*} q$ iff $h(t) \to_{\mathcal{A}'}^{*} q$, for all $t \in T(\mathcal{F})$.

Intersection problem

Theorem

The following problem is EXPTIME-complete: Given tree automata $\mathcal{A}_1, \ldots, \mathcal{A}_n$, is $\mathcal{L}(\mathcal{A}_1) \cap \cdots \cap \mathcal{L}(\mathcal{A}_n) \neq \emptyset$?

Proof (sketch):

- Membership: Compute the accessible tuples of states in $\mathcal{A}_1 \times \cdots \times \mathcal{A}_n$.
- Hardness: Simulate an polynomial-space ATM M with input of length n and space p(n) (using EXPTIME=APSPACE).

If $\ensuremath{\mathcal{M}}$ accepts the input, there is an accepting run.

Encode runs of \mathcal{M} as configuration trees.

Construct a collection of T-NFTA A_i , for i = 1, ..., p(n), such that the intersection of their languages is non-empty iff M has an accepting run. A_i checks the following:

- **(**) if \mathcal{M} starts with the correct configuration;
- 2) if all configurations in the run are of length p(n);
- if all final configurations are accepting;
- If the part of the configurations around the i-th symbol are coherent.

Detailed proof: Veanes, 1997

Congruences on trees

Definition: Congruence

Let \equiv be an equivalence relation on $T(\mathcal{F})$.

• \equiv is called a *congruence* if for any $n \ge 0$ and $f \in \mathcal{F}_n$,

 $u_1 \equiv v_1, \ldots, u_n \equiv v_n$ we have

$$f(u_1,\ldots,u_n)\equiv f(v_1,\ldots,v_n)$$

• \equiv saturates *L* if $u \equiv v$ implies $u \in L \iff v \in L$.

For
$$L \subseteq T(\mathcal{F})$$
, write $u \equiv_L v$ if
 $\forall C \in \mathcal{C}(\mathcal{F}) : C[u] \in L \Leftrightarrow C[v] \in L$

Myhill-Nerode Theorem for trees

The following are equivalent:

- $L \subseteq T(\mathcal{F})$ is recognizable.
- 2 L is saturated by some congruence of finite index.
- $\Im \equiv_L$ is of finite index.

Myhill-Nerode Theorem

Application:

Consider $L = \{ f(g^i(a), g^i(a)) \mid i \ge 0 \}$. For any pair $i \ne k$, consider $C = f(x, g^i(a))$. Then $C[g^i(a)] \in L$ but $C[g^k(a)] \notin L \Rightarrow g^i(a) \not\equiv_L g^k(a)$ Therefore \equiv_L is not of finite index, and L is not recognizable.

Proof of the theorem (sketch):

- 1 → 2: Let A be DCFTA and let u ≡ v iff u →^{*}_A q ^{*}_A ← v. Then ≡ is a congruence of finite index and saturates L.
- 2 → 3: Let ≡ be a saturating congruence, u ≡ v implies u ≡_L v (prove u ≡ v implies C[u] ≡ C[v] for all C, by recurrence over height of position of x in C).

•
$$3 \rightarrow 1$$
: Let $\mathcal{A} = \langle T(\mathcal{F}) / \equiv_L, \mathcal{F}, L / \equiv_L, \Delta \rangle$, with
 $f([u_1], \dots, [u_n]) \rightarrow [f(u_1, \dots, u_n)]$
for all $n \ge 0, f \in \mathcal{F}_n, u_1, \dots, u_n \in T(\mathcal{F})$,
where $[u]$ is the equivalence class of $u \in T(\mathcal{F})$;

Remark: This can be shown to be the canonical minimal DCFTA - one 36/89
Path languages

Path languages

Let $t \in T(\mathcal{F})$. The path language $\pi(t)$ is defined as follows:

• if
$$t = a \in \mathcal{F}_0$$
, then $\pi(t) = \{a\}$

• if
$$t = f(t_1, \ldots, t_n)$$
, for $f \in \mathcal{F}_n$, then $\pi(t) = \{ fiw \mid w \in \pi(t_i) \}$.

We write $\pi(L) = \bigcup \{ \pi(t) \mid t \in L \}$ for $L \subseteq T(\mathcal{F})$.

Example: $L = \{f(a, b), f(b, a)\}, \pi(L) = \{f1a, f2b, f1b, f2a\}.$

Path closure

Let $L \subseteq T(\mathcal{F})$ be a tree language.

- The path closure of L is $pc(L) = \{ t \mid \pi(t) \subseteq \pi(L) \} \supseteq L$.
- L is called *path-closed* if L = pc(L).

Example: $pc(L) = \{f(a, a), f(a, b), f(b, a), f(b, b)\}$, so L is not path-closed.

Path closure and T-NFTA

Lemma

Let $L \subseteq T(\mathcal{F})$ be a recognizable tree language. Then:

- $\pi(L)$ is a recognizable word language.
- pc(L) is a recognizable tree language.

Proof: Let $\mathcal{A} = \langle Q, \mathcal{F}, G, \Delta \rangle$ be a reduced T-NFTA for L.

- Construct a finite (word) automaton out of A. (Easy, but does require A to be reduced!)
- Construct A' = ⟨Q, F, G, Δ'⟩ for pc(L) as follows: for all a ∈ F₀:

 $q(a) \rightarrow_{\Delta} \varepsilon \quad \rightarrow \quad q(a) \rightarrow_{\Delta'} \varepsilon$

for all $n \ge 1$, $f \in \mathcal{F}_n$: $\forall i : q(f) \rightarrow_{\Delta} (q_{i,1}, \dots, q_{i,n}) \rightarrow q(f) \rightarrow_{\Delta'} (q_{1,1}, \dots, q_{n,n})$ Let $L_q = \mathcal{L}(\langle Q, \mathcal{F}, \{q\}, \Delta \rangle)$ and $L'_q = \mathcal{L}(\langle Q, \mathcal{F}, \{q\}, \Delta' \rangle)$. Prove $t \in L'_q \Leftrightarrow \pi(t) \subseteq \pi(L_q)$ for all $q \in Q$, $t \in \mathcal{T}(\mathcal{F})$ by induction.

Path closure and T-NFTA

Corollary

It is decidable whether a recognizable tree language is path-closed.

Theorem

Let $L \subseteq T(\mathcal{F})$ be a recognizable tree language. *L* is path-closed iff it is recognized by a T-DFTA.

Proof (sketch):

"→": Let A = ⟨Q, F, G, Δ⟩ be a reduced T-NFTA for L. Construct a T-DFTA A' = ⟨2^Q, F, G, Δ'⟩ as follows: for all a ∈ F₀, S(a) →_{Δ'} ε if ∃q ∈ S, q(a) →_Δ ε; for all n ≥ 1, f ∈ F_n, S(f) →_{Δ'} (S₁,..., S_n) where S_i = { q_i | ∃q ∈ S, q(f) →_Δ (q₁,..., q_n) }.
"←":

Let \mathcal{A} be a complete T-DFTA for L, define L_q as before. Prove that $\pi(t) \subseteq \pi(L_q)$ implies $t \in L_q$, for all $q \in Q, t \in T(\mathcal{F})$.

Logic over trees

Alternative specification for sets of trees

E.g., to describe valid HTML documents:

- A p tag may only appear inside a body tag.
- A dl tag must contain pairs of dt and dd tags.

Roadmap

- We shall define a logic that defines such properties of trees.
- The sets of trees definable in that language will be recognizable.

Recall: First-/second-order logic

First-order logic (FO)

Let $\sigma = ((R_i)_{1 \le i \le n})$ be a relation signature and $\mathcal{X}_1 = \{x_1, x_2, \ldots\}$ a set of variables. The first-order formulas $FO(\sigma)$ are:

$$R_i(x_{j_1},\ldots,x_{j_i}) \mid x = x' \mid \neg \phi \mid \phi \land \phi' \mid \exists x.\phi$$

Second-order logic: allow quantifying over relations *Monadic:* only quantify over sets

Monadic second-order logic (MSO) Let σ as before and $\mathcal{X}_1 = \{x_1, x_2, \ldots\}$, $\mathcal{X}_2 = \{X_1, X_2, \ldots\}$ sets of first-/second-order variables. The set of $MSO(\sigma)$ formulae are:

$$R_i(x_{j_1},\ldots,x_{j_i}) \mid x = x' \mid x \in X \mid \neg \phi \mid \phi \land \phi' \mid \exists x.\phi \mid \exists X.\phi$$

Weak second-order: only quantify over finite sets

WSkS (weak MSO over with k successors) WSkS = $MSO(<_1, ..., <_k)$

Semantics of MSO

Definition

Let ${\mathfrak M}$ a domain, σ a signature, ν a valuation with

•
$$\nu(x) \in \mathfrak{M}$$
 for $x \in \mathcal{X}_1$

•
$$u(X) \subseteq \mathfrak{M} \text{ for } X \in \mathcal{X}_2$$

$$\begin{array}{lll}\mathfrak{M}, \sigma, \nu \models R_{i}(x_{j_{1}}, \dots, x_{j_{i}}) & \text{if} & (\nu(x_{j_{1}}), \dots, \nu(x_{j_{i}})) \in R_{i} \\ \mathfrak{M}, \sigma, \nu \models x = x' & \text{if} & \nu(x) = \nu(x') \\ \mathfrak{M}, \sigma, \nu \models x \in X & \text{if} & \nu(x) \in \nu(X) \\ \mathfrak{M}, \sigma, \nu \models \neg \phi & \text{if} & \mathfrak{M}, \sigma, \nu \not\models \phi \\ \mathfrak{M}, \sigma, \nu \models \phi \land \phi' & \text{if} & \mathfrak{M}, \sigma, \nu \models \phi \land \mathfrak{M}, \sigma, \nu \models \phi' \\ \mathfrak{M}, \sigma, \nu \models \exists x. \phi & \text{if} & \exists m \in \mathfrak{M}. & \mathfrak{M}, \sigma, \nu[x \mapsto m] \models \phi \\ \mathfrak{M}, \sigma, \nu \models \exists X. \phi & \text{if} & \exists M \subseteq \mathfrak{M}. & \mathfrak{M}, \sigma, \nu[X \mapsto M] \models \phi \end{array}$$

We omit \mathfrak{M}, σ when clear from context.

Recall: Common abbreviations

∀x, ∀X, ∨, etc can be expressed in the usual way.
X ⊆ Y:

$$\forall x.(x \in X \to x \in Y)$$

• $Z = X \cup Y$:

$$\forall x.(x \in Z \leftrightarrow x \in X \lor x \in Y)$$

(ロト・日本)・(日本)・日本(日本) 日、(1)への(43/89)

• Partition (X, X_1, \dots, X_m) : $\left(\forall x. \left(x \in X \leftrightarrow \bigvee_{i=1}^m x \in X_i \right) \right) \land \left(\bigwedge_{i=1}^m \bigwedge_{j \neq i} \forall x. (x \notin X_i \lor x \notin X_j) \right)$

• Similarly, $X = \emptyset$, $X = \{x\}$, X = Y,...

WSkS and trees

Let $\mathfrak{M} = N^*$, we fix $\langle i$ to be the relation $\langle i = \{ \langle p, pip' \rangle \mid p, p' \in N^* \}$. We define $\langle = \bigcup_{i=1}^k \langle i \rangle$ and \leq as usual, and ε for the minimal element. We write *xi* to denote the least *q* s.t. $\nu(x) \langle i \rangle q$.

Coding of a tree

Let $t \in T(\mathcal{F})$ and k the maximal arity in \mathcal{F} . As a shorthand, define $S_{\mathcal{F}} := (S_f)_{f \in \mathcal{F}}$. We note $C(t) := (S, S_{\mathcal{F}})$, where:

•
$$S = \bigcup_{f \in \mathcal{F}} S_f;$$

• for all
$$f \in \mathcal{F}$$
, $S_f = \{ p \in Pos_t \mid t(p) = f \}$.

 $\begin{array}{ll} (S,S_{\mathcal{F}}) \text{ encodes a tree if } Tree(S,S_{\mathcal{F}}) \text{ holds:} \\ Tree(S,S_{\mathcal{F}}) & := & S \neq \emptyset \land Partition(S,S_{\mathcal{F}}) \\ & \land \forall x. \forall y. (x \in S \land y < x) \rightarrow y \in S \\ & \land & \bigwedge_{n=1}^{k} \bigwedge_{f \in \mathcal{F}_n} \bigwedge_{i=1}^{n} (x \in S_f \rightarrow xi \in S) \\ & \land & \bigwedge_{n=1}^{k} \bigwedge_{f \in \mathcal{F}_n} \bigwedge_{i=n+1}^{k} (x \in S_f \rightarrow xi \notin S) \end{array}$

Semantics of WSkS on trees

Coded valuation

Let
$$\mathcal{F}' := \mathcal{F} \times 2^{\mathcal{X}_1 \cup \mathcal{X}_2}$$
. The arity of (f, τ) is *n* if $f \in \mathcal{F}_n$.
Let $t \in T(\mathcal{F})$ and ν a valuation. The tuple $\langle t, \nu \rangle$ is *coded* by a tree $t' \in T(\mathcal{F}')$, as follows, for all $p \in Pos$ and $t'(p) = \langle f, \tau \rangle$:

• if
$$x \in \mathcal{X}_1$$
 then $\tau(x) = 1$ iff $p = \nu(x)$;

• if
$$X \in \mathcal{X}_2$$
 then $\tau(X) = 1$ iff $p \in \nu(X)$.

A tree
$$t' \in \mathcal{T}(\mathcal{F}')$$
 is valid $(t' \in \mathcal{T}_v(\mathcal{F}'))$ if it codes some $\langle t, \nu
angle.$

Semantics of WSkS

Let ϕ be a formula of WSkS and $V \subseteq (\mathcal{X}_1 \cup \mathcal{X}_2) \uplus (\{S\} \cup S_{\mathcal{F}})$ its free variables.

$$\mathcal{L}(\phi) := \{ \langle t, \nu \rangle \in T_{\nu}(\mathcal{F}') \mid \nu[(S, S_{\mathcal{F}}) \mapsto C(t)] \models \phi \}$$

Examples



• We have $C(t) = (S, S_f, S_g, S_a)$ with $S = \{\varepsilon, 1, 11, 2\}$, $S_f = \{\varepsilon\}, S_g = \{1\}, S_a = \{11, 2\}.$ • $\nu'[(S, S_F) \mapsto C(t)] \models x \in S_g$, thus $\langle t, \nu' \rangle \in \mathcal{L}(x \in S_g)$ • $t \in \mathcal{L}(\exists x. x \in S_g)$

(ロト・日本)・(目)・(目)・(目)・(10,00) 46/89

WSkS and recognizability

Theorem

A tree language $L \subseteq T(\mathcal{F})$ is recognizable iff $L = \mathcal{L}(\phi)$ for some formula $\phi(S, S_{\mathcal{F}})$ of WSkS.

Proof: (sketch)

- DCFTA A → WSkS: Construct formula φ that
 (i) verifies that the structure is a tree;
 (ii) guesses a computation of A, i.e. partitioning of S onto states;
 (iii) verifies that the computation is locally correct;
 (iv) verifies that the root is labelled by an accepting state.
- WSkS φ → NFTA A: Proceed by recurrence on φ, show that all subformulae of φ are recognizable.

Example: DCFTA \rightarrow WSkS

• Let $Q := \{q_0, q_1, q_f\}$, $\mathcal{F} = \{f(2), g(1), a\}$, $G := \{q_f\}$, and rules $a \to q_0 \quad g(q_0) \to q_1 \quad g(q_1) \to q_1 \quad f(q_1, q_1) \to q_f$ (automate à compléter !)

• Corresponding formula:

$$\phi = Tree(S, S_{\mathcal{F}}) \land \exists Q_0, Q_1, Q_f. Partition(S, Q_0, Q_1, Q_f) \land \forall x. (x \in S_a \to x \in Q_0) \land \forall x. ((x \in S_g \land x1 \in Q_0) \to x \in Q_1) \land \forall x. ((x \in S_g \land x1 \in Q_1) \to x \in Q_1) \land \forall x. ((x \in S_f \land x1 \in Q_1 \land x2 \in Q_1) \to x \in Q_f) \land \cdots \land \varepsilon \in Q_f$$

◆□▶◆□▶◆∃▶◆∃▶ ∃ のへで 48/89

Example: $WSkS \rightarrow NFTA$

Consider $\mathcal{F} = \{f(2), g(1), a\}.$

•
$$\phi = x \in S_g$$

 $\mathcal{A}_{\phi} = \langle \{q, q'\}, \mathcal{F} \times 2^{\{x\}}, \{q'\}, \Delta \rangle$ with transitions
 $\langle a, 0 \rangle \rightarrow q$
 $\langle g, 1 \rangle (q) \rightarrow q'$ $\langle g, 0 \rangle (q) \rightarrow q$ $\langle g, 0 \rangle (q') \rightarrow q'$
 $\langle f, 0 \rangle (q, q) \rightarrow q$ $\langle f, 0 \rangle (q, q') \rightarrow q'$ $\langle f, 0 \rangle (q', q) \rightarrow q'$
accepts $\mathcal{L}(x \in S_g)$ (scans for a single g-position with $\tau(x) = 1$).
• $\phi' = \exists x.\phi$
Obtain $\mathcal{A}_{\phi'}$ from \mathcal{A}_{ϕ} by stripping $\tau(x)$:
 $\mathcal{A}_{\phi'} = \langle \{q, q'\}, \mathcal{F}, \{q'\}, \Delta \rangle$
 $a \rightarrow q$
 $g(q) \rightarrow q'$ $g(q) \rightarrow q$ $g(q') \rightarrow q'$
 $f(q, q) \rightarrow q$ $f(q, q') \rightarrow q'$ $f(q', q) \rightarrow q'$

Unranked trees

We now consider *finite ordered* unranked trees.

- ordered : internal nodes have children 1...n
- unranked : nodes may have an arbitrary number of children

Motivation: e.g., XML documents

- "A html tag contains an optional head and an obligatory body."
- "A div tag contains an unlimited number of p, ol, ul, ... tags."

Definition: Tree (recall)

A (finite, ordered) *tree* is a non-empty, finite, prefix-closed set $Pos \subseteq N^*$.

Hedge automata

Definition: (Bottom-up) hedge automaton

A hedge automaton (NHA) is a tuple $\mathcal{A} = \langle Q, \Sigma, G, \Delta \rangle$, where:

- Q is a finite set of *states*;
- Σ a finite alphabet;
- $G \subseteq Q$ are the *final states*;
- Δ is a finite set of rules of the form

a(R)
ightarrow q

for $a \in \Sigma$, $q \in Q$, and R a regular (word) language over Q.

Example: $Q := \{q_x, q_h, q_b, q_p\}, \Sigma = \{x, h, b, p\}, G := \{q_x\}, \text{ and rules}$ $x(q_h^2 q_b) \rightarrow q_x \quad h(\varepsilon) \rightarrow q_h \quad b(q_p^*) \rightarrow q_b \quad p(\varepsilon) \rightarrow q_p$

This accepts trees of the form x(h, b(p, ..., p)) and x(b(p, ..., p)).

Semantics of hedge automata

Remark:

- The R in $a(R) \rightarrow q$ are called *horizontal languages*.
- They are (finitely) represented by regular expressions or finite automata.

Computation of NHA

Let $t \in T(\Sigma)$ be a tree. A *run* or *computation* of \mathcal{A} on t is a tree $t' \in T(Q)$, i.e. for all $p \in Pos$:

if t(p) = a ∈ Σ, t'(p) = q ∈ Q, and Pos ∩ pN = {p1,..., pn}, there exists a(R) → q ∈ Δ such that t'(p1) ··· t'(pn) ∈ R.
Acceptance condition: t'(ε) ∈ G

 $L \subseteq T(\Sigma)$ is called *hedge-recognizable* if $L = \mathcal{L}(\mathcal{A})$ for some NHA \mathcal{A} .

Complete / normalized / deterministic HA

An NHA is ...

- *complete* if for all $t \in T(\Sigma)$, $t \rightarrow^*_{\mathcal{A}} q$ for some q;
- full if for all $a \in \Sigma$, $q \in Q$, there is some $a(R) \rightarrow q$;
- reduced if $a(R_1) \rightarrow q, a(R_2) \rightarrow q \in \Delta$ implies $R_1 = R_2$;
- deterministic (DHA) if $a(R_1) \rightarrow q_1, a(R_2) \rightarrow q_2 \in \Delta$ implies $R_1 \cap R_2 = \emptyset$ or $q_1 = q_2$.

Any NHA has an equivalent complete / full / reduced / deterministic NHA.

- complete: add garbage state, as usual
- full: add rules $a(\emptyset) \to q$ where necessary
- reduced: replace $a(R_1) \to q$ and $a(R_2) \to q$ with $a(R_1 \cup R_2) \to q$ where necessary

Determinization

Determinization of NHA

Let $\mathcal{A} = \langle Q, \Sigma, G, \Delta \rangle$ be a complete, full, reduced NHA. The complete, full, reduced DHA $\mathcal{A}' = \langle 2^Q, \Sigma, G', \Delta' \rangle$ is equivalent to \mathcal{A} where:

•
$$G' = \{ S \subseteq Q \mid S \cap G \neq \emptyset \};$$

• let $R_{a,q}$ denote the (unique) language s.t. $a(R_{a,q})
ightarrow q \in \Delta;$

- $R'_{a,q} := R_{a,q}[q' \to (S \cup \{q'\}) \mid q' \in Q, S \subseteq Q]$
- for all $a \in \Sigma$, $S \subseteq Q$, we have $a(R_{a,S}) \rightarrow S \in \Delta'$;

$$R_{a,S} := \left(\bigcap_{q \in S} R'_{a,q}\right) \setminus \left(\bigcup_{q \notin S} R'_{a,q}\right)$$

▲□▶▲@▶▲≧▶▲≧▶ ≧ のへで 54/89

Encoding unranked trees

Bijective encoding of unranked into ranked trees

- Let Σ an alphabet; $\mathcal{F}_{\Sigma} := \{ \mathbb{Q}(2) \} \cup \{ a(0) \mid a \in \Sigma \}.$
- Define the coding $C_{\mathbb{Q}}(t) \in T(\mathcal{F}_{\Sigma})$ of $t \in T(\Sigma)$ as $C_{\mathbb{Q}}(a(t_1, \ldots, t_n)) = \underbrace{\mathbb{Q}(\mathbb{Q}(\ldots, \mathbb{Q}(t_1), C_{\mathbb{Q}}(t_2)), \ldots), C_{\mathbb{Q}}(t_n))$

Example:



Recognizing encoded trees

Theorem

A language $L \subseteq T(\Sigma)$ is hedge-recognizable iff $C_{\mathbb{Q}}(L)$ is recognizable.

• NHA \rightarrow NFTA: Let $\mathcal{A} = \langle Q, \Sigma, G, \Delta \rangle$ an NHA; $\Delta = \{a_1(R_1) \rightarrow q_1, \dots, a_n(R_n) \rightarrow q_n\}$; R_i represented by det.compl. FA $\mathcal{A}_i = \langle S_i, Q, s_0^{(i)}, F_i, \delta_i \rangle$.

◆□▶◆舂▶◆≧▶◆≧▶ ≧ のへで 56/89

Construct NFTA
$$\mathcal{A}' = \langle Q', \mathcal{F}_{\Sigma}, G, \Delta' \rangle$$
, where:
• $Q' = Q \cup \biguplus_{i=1}^{n} S_i$
• $\Delta' = \bigcup_{i=1}^{n} (\Delta_1^i \cup \Delta_2^i \cup \Delta_3^i)$
 $\Delta_1^i = \{a_i \rightarrow s_0^{(i)}\}$
 $\Delta_2^i = \{\mathfrak{O}(s, q) \rightarrow \delta_i(s, q) \mid s \in S_i, q \in Q\}$
 $\Delta_3^i = \{s_f \rightarrow q_i \mid s_f \in F_i\}$

Example: $NHA \rightarrow NFTA$

• $Q := \{q_x, q_h, q_b, q_p\}, \Sigma = \{x, h, b, p\}, G := \{q_x\}, \text{ and rules}$ $x(q_h^? q_b) \rightarrow q_x \quad h(\varepsilon) \rightarrow q_h \quad b(q_p^*) \rightarrow q_b \quad p(\varepsilon) \rightarrow q_p$



- Automaton for first rule:
- Single-state automata with s_h, s_b, s_p for the other rules



Recognizing encoded trees

Theorem

A language $L \subseteq T(\Sigma)$ is hedge-recognizable iff $C_{\mathbb{Q}}(L)$ is recognizable.

• NFTA
$$\rightarrow$$
 NHA:
Let $\mathcal{A} = \langle Q, \mathcal{F}_{\Sigma}, G, \Delta \rangle$ an NFTA (without ε -moves).

Define
$$\Delta_R := \{ \langle q_0, q_1, q_2 \rangle \mid @(q_0, q_1) \rightarrow_{\Delta} q_2 \}$$

and $Out := G \cup \{ q \mid \exists q', q'' : @(q', q) \rightarrow_{\Delta} q'' \}$.
For $q \in Q, q' \in Out$, let $A_{q,q'} := \langle Q, Q, q, \{q'\}, \Delta_R \rangle$ a word automaton.

Construct NHA
$$\mathcal{A}' := \langle Q, \Sigma, G, \Delta' \rangle$$
, where
 $\Delta' = \{ a(\mathcal{L}(\mathcal{A}_{q,q'})) \rightarrow q' \mid a \rightarrow_{\Delta} q, q' \in Out \}$

Corollary

Hedge-recognizable languages are closed under boolean operations.

Unranked trees and logic

▲□▶▲□▶▲≡▶▲≡▶ ≡ のへで 59/89

UTL = weak MSO(*child*,*next*) interpreted over $\mathfrak{M} = N^*$, where

- child(x, y) iff y = xi for some $i \in N$
- next(x, y) iff $\exists z, i : x = zi \land y = z(i+1)$

Further predicates can be defined from this:

• desc(x, y) = "y is a descendant of $x" = "x \le y"$

Notions like $\mathcal{L}(\phi)$ are defined in analogy with WSkS.

Theorem: UTL = NHA

A language $L \subseteq T(\Sigma)$ is hedge-recognizable iff $L = \mathcal{L}(\phi)$ for some formula $\phi(S, S_{\Sigma})$ of UTL.

UTL = NHA: Proof sketch

 UTL → NHA: Let φ be an UTL formula. Define φ' of WS2S s.t. L(φ') = C_Q(L(φ)).

Define leftmost(x, y) as $\forall X : (x \in X \land \forall z, z' : (z \in X \land z' = z1 \rightarrow z' \in X))$ $\land \forall z : (z \in X \rightarrow z = x \lor (\exists z' : z' \in X \land z = z'1)))$ $\rightarrow (y \in X \land \forall z : z \in X \rightarrow z \le y)$ ("y is the maximal position in x1*")

Then *child* and *next* can be translated as follows: $child(x,y) := \exists z : leftmost(z,x) \land leftmost(z2,y)$ $next(x,y) := \exists z : leftmost(z12,x) \land leftmost(z2,y)$

UTL = NHA: Proof sketch

• NHA \rightarrow UTL:

Let ${\mathcal A}$ be a complete, full, normalized, deterministic NHA.

Construct formula $\phi(S, S_{\Sigma})$ of UTL that

(i) verifies that the structure is a tree;

(ii) guesses a computation of A, i.e. partitioning of S onto states;

(iii) verifies that the computation is locally correct;

(iv) verifies that the root is labelled by an accepting state.

The major difference with the NFTA \rightarrow WSkS construction is (iii): (iii): whenever the computation puts q on an a-labelled position p, guess a run of the automaton for $R_{a,q}$ over p and its children

Tuples of trees

Let $t_1, t_2 \in T(\mathcal{F})$ ranked trees. Add a fresh symbol – to \mathcal{F}_0 and let $\mathcal{F}' := \{ \langle f, g \rangle(k) \mid f \in \mathcal{F}_m, g \in \mathcal{F}_n, k = \max\{m, n\} \}.$

 $\langle t_1, t_2
angle$ denotes the ranked tree $t \in \mathcal{T}(\mathcal{F}')$ as follows:

- $Pos_t = Pos_{t_1} \cup Pos_{t_2}$
- for all $p \in Pos_t$,

$$t(p) = \begin{cases} \langle f, g \rangle & \text{if } t \in \textit{Pos}_{t_1} \cap \textit{Pos}_{t_2}, t_1(p) = f, t_2(p) = g \\ \langle f, - \rangle & \text{if } t \in \textit{Pos}_{t_1} \setminus \textit{Pos}_{t_2}, t_1(p) = f \\ \langle -, g \rangle & \text{if } t \in \textit{Pos}_{t_2} \setminus \textit{Pos}_{t_1}, t_2(p) = g \end{cases}$$

Example:



Tree relations We consider (binary) relations $R \subseteq T(\mathcal{F})^2$.

- Let \$\mathcal{R}_2\$ be the class of recognizable relations
 (= recognizable languages over \$\mathcal{F}'\$).
- Let \mathfrak{X}_2 be the class of *finite unions of cross products* $R \in \mathfrak{X}_2$ iff $R = \bigcup_{i=1}^n \left(L_1^{(i)} \times L_2^{(i)} \right)$, for some $n \ge 0$ and $L_1^{(i)}, L_2^{(i)}$ recognizable for all i
- Let \mathfrak{T}_2 be the class of relations recognizable by GTT.

Definition: Ground Tree Transducer

A ground tree transducer (GTT) is pair $\mathcal{G} = \langle \mathcal{A}_1, \mathcal{A}_2 \rangle$ of bottom-up NFTA over \mathcal{F} . (The states of \mathcal{A}_1 and \mathcal{A}_2 may overlap.) The relation accepted by \mathcal{G} is

$$\{ \langle t, u \rangle \mid \exists n \ge 0, \ C \in \mathcal{C}^n(\mathcal{F}), \\ t_1, \dots, t_n \in T(\mathcal{F}), \ u_1, \dots, u_n \in T(\mathcal{F}), \ q_1, \dots, q_n : \\ t = C[t_1, \dots, t_n] \land u = C[u_1, \dots, u_n] \\ \land \forall i : t_i \rightarrow^*_{\mathcal{A}_1} q_i \ \mathcal{A}_2^* \leftarrow u_i \}$$

Relations between $\mathfrak{R}_2, \mathfrak{X}_2, \mathfrak{T}_2$

▲□▶▲□▶▲三▶▲三▶ 三 のへで 64/89

Propositions

- $\textcircled{2} \ \mathfrak{R}_2 \not\subseteq \mathfrak{T}_2 \ \text{and} \ \mathfrak{X}_2 \not\subseteq \mathfrak{T}_2$
- ${\color{black} \bullet} \ {\mathfrak X}_2 \subseteq {\mathfrak R}_2$
- $\ \, \mathfrak{X}_2 \cup \mathfrak{T}_2 \subsetneq \mathfrak{R}_2$

Proofs:

- 2 \emptyset is in $\mathfrak{X}_2 \cap \mathfrak{R}_2$ but not \mathfrak{T}_2
- see next slides
- see next slides
- see next slides

Proof of $\mathfrak{X}_2 \subseteq \mathfrak{R}_2$

③ Let
$$A_i = \langle Q_i, \mathcal{F}, G_i, \Delta_i \rangle$$
 (for $i = 1, 2$) be NFTA
and let $R = \mathcal{L}(\mathcal{A}_1) \times \mathcal{L}(\mathcal{A}_2) \in \mathfrak{X}_2$.

Construct NFTA $\mathcal{A} = \langle Q, \mathcal{F}', G_1 \times G_2, \Delta \rangle$ with $\mathcal{L}(\mathcal{A}) = R$:

•
$$Q = (Q_1 \cup \{-\}) \times (Q_2 \cup \{-\})$$

• for every $f \in \mathcal{F}_m$, $g \in \mathcal{F}_n$, $m \ge n$, $\neg(f = g = -)$
 Δ contains

•
$$\langle f, g \rangle (\langle q_1, q'_1 \rangle, \dots, \langle q_n, q'_n \rangle, \langle q_{n+1}, - \rangle, \dots, \langle q_m, - \rangle) \rightarrow \langle q, q' \rangle$$
 if
 $f(q_1, \dots, q_m) \rightarrow q \in \Delta_1$ and $g(q'_1, \dots, q'_n) \rightarrow q' \in \Delta_2$
• $\langle g, f \rangle (\langle q_1, q'_1 \rangle, \dots, \langle q_n, q'_n \rangle, \langle -, q'_{n+1} \rangle, \dots, \langle -, q_m \rangle) \rightarrow \langle q, q' \rangle$ if
 $f(q'_1, \dots, q'_m) \rightarrow q \in \Delta_2$ and $g(q_1, \dots, q_n) \rightarrow q' \in \Delta_1$

(reminder: we assume that - is a fresh symbol in \mathcal{F}_0)

Intuition: Modified cross-product construction.

Proof of $\mathfrak{T}_2 \subseteq \mathfrak{R}_2$

• Let $\mathcal{G} = \langle \mathcal{A}_1, \mathcal{A}_2 \rangle$, $\mathcal{A}_i = \langle Q_i, \mathcal{F}, G_i, \Delta_i \rangle$ (for i = 1, 2). We construct NFTA $\mathcal{A}' = \langle Q', \mathcal{F}', \{q_f\}, \Delta' \rangle$ with $\mathcal{L}(\mathcal{A}') = \mathcal{L}(\mathcal{G})$.

Construct NFTA $\mathcal{A} = \langle Q, \mathcal{F}', G, \Delta \rangle$ from $\mathcal{A}_1, \mathcal{A}_2$ as in previous proof. Then:

•
$$Q' = Q \uplus \{q_f\}$$

• $\Delta' = \Delta \cup \Delta_1 \cup \Delta_2$
 $\Delta_1 = \{ \langle q, q \rangle \rightarrow q_f \mid q \in Q_1 \cap Q_2 \}$
 $\Delta_2 = \{ \langle f, f \rangle (q_f, \dots, q_f) \rightarrow q_f \mid f \in \mathcal{F}_n, f \neq - \}$

Intuition:

 Δ reads pairs of trees from $\mathcal{A}_1, \mathcal{A}_2$;

 Δ_1 allows to plug pairs of subtrees into some context C;

 Δ_2 reads the remaining context *C*.

Proof of $\mathfrak{X}_2 \cup \mathfrak{T}_2 \subsetneq \mathfrak{R}_2$

- **③** Let $\mathcal{F} = \{f(1), g(1), a\}$. Let $R = \{ \langle t_1, t_2 \rangle \mid \exists C \in C(\mathcal{F}), t \in T(\mathcal{F}) : t_1 = C[t] \land t_2 = C[f(t)] \}.$
 - *R* ∉ 𝔅₂: By pigeonhole principle using ⟨*fⁱ*(*a*), *fⁱ⁺¹*(*a*)⟩, *i* ≥ 0.
 - $R \notin \mathfrak{T}_2$:

Suppose that *R* is accepted by GTT $\langle A_1, A_2 \rangle$ with *n* states in common. For all $i \ge 0$, let q_i such that $g^i(a) \rightarrow^*_{A_1} q_i$ and $f(g^i(a)) \rightarrow^*_{A_2} q_i$. Contradiction follows from pigeon-hole principle.

▲□▶▲圖▶▲≣▶▲≣▶ ■ のへで 67/89

•
$$R \in \mathfrak{R}_2$$
:
Let $\mathcal{A} = \langle \{q_a, q_f, q_g, q\}, \mathcal{F}', \{q\}, \Delta \rangle$ with:
 $\langle -, a \rangle \rightarrow q_a \quad \langle x, y \rangle(q_x) \rightarrow q_y \quad q_f \rightarrow q \quad \langle x, x \rangle(q) \rightarrow q$
for $x, y \in \{f, g, a\}$

Closure properties

Boolean closure

 \mathfrak{X}_2 and \mathfrak{R}_2 are closed under boolean operations.

Transitive closure If $R \in \mathfrak{T}_2$, then $R^* \in \mathfrak{T}_2$.

Proof: Let $\langle A_1, A_2 \rangle$ with states Q_1, Q_2 a GTT accepting R. We construct $\langle B_1, B_2 \rangle$ accepting R^* by adding transitions to A_1 and A_2 using the following saturation rule:

• For $i \neq j$ and all $q \in Q_1 \cap Q_2$, $q' \in Q_j$, if there exists a tree t s.t. $t \rightarrow^*_{\mathcal{B}_i} q$ and $t \rightarrow^*_{\mathcal{B}_j} q'$ then add $q \rightarrow q'$ to \mathcal{B}_j .

Transitive closure: Intuition

Suppose that $\langle t, v \rangle, \langle v, u \rangle \in R$. The interesting case is illustrated below:



Suppose that $\langle t, v \rangle$ differ in a position p and $\langle v, u \rangle$ in positions pp_1, \ldots, pp_n .

Then in A_2 we want the subtrees of u at pp_1, \ldots, pp_n to be substitutable for the corresponding subtrees in v.

Transitive closure: Intuition

Consider the runs of t, v, u in $\langle A_1, A_2 \rangle$:



Adding $q_i \rightarrow q'_i$ to the right-hand side automaton achieves the objective.

Transitive closure: $R^* \subseteq \mathcal{L}(\langle \mathcal{B}_1, \mathcal{B}_2 \rangle)$

Proof by induction: Let $\langle t, u \rangle \in R^i$, for $i \ge 0$.

• i = 0: trivial

i → i + 1: Let v s.t. ⟨t, v⟩ ∈ Rⁱ and ⟨v, u⟩ ∈ R. Then (by induction) ⟨t, v⟩ is accepted by ⟨B₁, B₂⟩. Let P be the positions in which ⟨t, v⟩ differ and P' be the positions in which ⟨v, u⟩ differ. All incomparable pairs in P × P' are handled by the definition of GTT. For p ∈ P and pp1,..., ppn ∈ P' consider the previous drawings. The case pp1,..., ppn ∈ P and p ∈ P' is symmetric.

◆□▶ ◆□▶ ◆ 臣▶ ◆ 臣▶ 臣 • • ○ Q (? 71/89)

Transitive closure: $R^* \supseteq \mathcal{L}(\langle \mathcal{B}_1, \mathcal{B}_2 \rangle)$

Let $\langle \mathcal{B}_1^i, \mathcal{B}_2^i \rangle$ denote the GTT after adding *i* transitions and show that its language is included in R^* .

- i = 0: trivial
- $i \rightarrow i + 1$: Let $q \rightarrow q'$ be the transition added in the (i + 1)-th step (to \mathcal{B}_1 , say) and let $q \rightarrow q'$ be used j times in accepting some $\langle t, u \rangle$.

If j = 0, then $\langle t, u \rangle \in R^*$ by induction hypothesis. Otherwise:

- there exist $n \ge 0$, $C \in \mathcal{C}^n(\mathcal{F})$ etc such that $t = C[t_1, \ldots, t_n]$, $u = C[u_1, \ldots, u_n]$ and $\forall k : t_k \rightarrow^*_{\mathcal{B}^{l+1}} q_k \overset{*}{\underset{\mathcal{B}^{k+1}}{\xrightarrow{}}} \leftarrow u_k$.
- **3** Suppose $t_k = C'[t'] \rightarrow^*_{\mathcal{B}_1^{i+1}} C'[q] \xrightarrow{} C'[q'] \rightarrow^*_{\mathcal{B}_1^{i+1}} q_k$ for some k, C', t'.
- 3 There must be some $v \in T(\mathcal{F})$ with $v \to_{\mathcal{B}_1^i}^* q$ and $v \to_{\mathcal{B}_1^i}^* q'$.
- From (2) et (3) we have $C'[v] \rightarrow^*_{\mathcal{B}_1^{i+1}} q_k$.
- So Replacing t_k by C'[v] in (1) we get $\langle t[t'/v], u \rangle \in \mathcal{L}(\langle \mathcal{B}_1^{i+1}, \mathcal{B}_2^{i+1} \rangle)$ with fewer than j times $q \to q'$, thus by ind.hyp. $\langle t[t'/v], u \rangle \in R^*$.
- From (2) and (3), $t' \rightarrow^*_{\mathcal{B}_1^{i+1}} q_{\mathcal{B}_2^i} \leftarrow v$, with fewer than j times $q \rightarrow q'$.
- From (6) by ind.hyp. $\langle t, t[t'/v] \rangle \in R^*$.
Application: XML

XML = Extensible Markup Language

- Conceived for platform-independent exchange of structured data
- An XML document consists of *tags* with *attributes* and text (parsed character data, *pcdata*)

Example:

<html><head><meta charset="UTF-8"/> <title>My web page</title></head> <body>Bonne année !</body></html>

- A *well-formed* XML document forms a tree (balanced tags, one single root tag)
- Testing for well-formedness / generating tree from document: visibly pushdown automaton, LL/LR parser

Valid XML documents

- Languages of XML documents defined by schemas (DTD, XML Schema, Relax NG)
- Schemas define permissible tags (+attributes) and their nesting
- Examples of XML languages: HTML, SVG, KML, ...

• Valid XML document: well-formed document satisfying a schema

◆□▶▲□▶★■▶★■▶ ■ のへで 74/89

• Example: XML-Schema for KML

DTD for XML

DTD = Document Type Definition DTD define a (restricted) subclass of XML languages. Essentially, defines a regular language of child tags for each tag type.

The language of XML documents defined by DTD is accepted by NHA.

Restrictions on DTD

Expressivity of DTD

There are hedge-recognizable languages that cannot be defined by DTD.

Example: $\{f(g(a)), f'(g(b))\}$

DTD contain another restriction:

It is an error if the content model allows an element to match more than one occurrence of an element type in the content model.

E.g., (ab|ac) is not allowed (but a(b|c) is).

Deterministic regular expressions

Definition: Marked RE

Let *e* be a RE over Σ . The *marked* RE \overline{e} is a RE over $\Sigma \times \mathbb{N}$ obtained by adding a unique subscript to each letter in *e*.

Example: e = (ab|ac), then $\bar{e} = (a_1b_2|a_3c_4)$

Definition: Deterministic RE

Let *e* a RE over Σ . We call *e* deterministic if \overline{e} satisfies the following: for all $u, v, w \in (\Sigma \times \mathbb{N})^*$ and $a \in \Sigma$, if $ua_i v, ua_j w \in L(\overline{e})$ then i = j.

▲□▶▲圖▶▲≣▶▲≣▶ ≣ のへで 77/89

Example: e = (ab|ac), $\bar{e} = (a_1b_2|a_3c_4)$, not deterministic because $a_1b_2, a_3c_4 \in L(\bar{e})$

Parsing deterministic RE

Parsing det. RE

Let *e* be a deterministic RE. A DFA for *e* can be constructed in polynomial (linear) time. [Brüggemann-Klein 1993, Groz et al 2012]

◆□▶◆□▶◆豆▶◆豆▶ 豆 のQ (* 78/89)

Proof (sketch): Construction of Glushkov automaton from *e*.

Expressivity of det. RE

Not every regular language can be defined by a deterministic RE.

XML Schema

XML Schema can define more expressive XML languages. Example:

```
<rpre><xsd:complexType name="track">
<xsd:sequence minOccurs="1" maxOccurs="unbounded">
 <re><xsd:choice>
  <xsd:element name="invSession" type="invSession"</pre>
   minOccurs="1" maxOccurs="1"/>
  <xsd:element name="conSession" type="conSession"</pre>
   minOccurs="1" maxOccurs="1"/>
 </xsd:choice>
 <rpre><xsd:element name="break" type="xsd:string"</pre>
   minOccurs="0" maxOccurs="1"/>
</xsd:sequence>
</r>sd:complexType>
```

XML Schema and Hedge Automata

XML Schema = NHA

XML Schema (restricted to occurrence and nesting conditions) correspond to the class of hedge-recognizable languages.

◆□▶◆□▶◆三▶◆三▶ 三 のへで 80/89

Moreover, XML Schema also permit non-hedge-recognizable features:

- constraints on data types in attributes and pcdata
- consistency constraints (e.g., unique keys)

XSL Transformation

- XSLT allows to transform XML documents into other documents (incl. non XML)
- XQuery used to specify nodes on which to apply a transformation

Example (from Wikipedia):

```
<rsl:template match="//title">
    <em>
        <rsl:apply-templates/>
        </em>
</rsl:template>
<rsl:for-each select="book">
        <rsl:sort select="price" order="ascending" />
</rsl:for-each>
```

Tree transducers

Definition: Bottom-up tree transducer

A (finite bottom-up) tree transducer (NUTT) is a tuple $\mathcal{U} = \langle Q, \mathcal{F}, \mathcal{F}', G, \Delta \rangle$, where:

- Q is a finite set of *states* and $G \subseteq Q$ are *final* states;
- $\mathcal{F}, \mathcal{F}'$ are finite ranked alphabets;
- Δ is a finite set of rules of the form

$$f(q_1(x_1),\ldots,q_n(x_n)) \rightarrow q(u)$$

for
$$f\in \mathcal{F}_n$$
 and $q,q_1,\ldots,q_n\in Q, u\in \mathcal{T}(\mathcal{F}',\mathcal{X}_n)$, or $q(x_1) o q'(u)$

for $q,q' \in Q, u \in T(\mathcal{F}', \mathcal{X}_1)$ (ε -rule).

Example: $\mathcal{F} = \{f(1), a\}, \ \mathcal{F}' = \mathcal{F} \cup \{h(2), g(1)\};$ $\mathcal{U}_1 = \langle \{q, q_f\}, \mathcal{F}, \mathcal{F}', \{q_f\}, \Delta \rangle$, with rules $a \rightarrow q(a) \qquad f(q(x_1)) \rightarrow q(f(x_1)) \mid q(g(x_1)) \mid q_f(h(x_1, x_1))$

NUTT move relation

Move relation

Let $t, t' \in T(\mathcal{F}, \mathcal{F}', Q)$. We write $t \rightarrow_{\mathcal{U}} t'$ if the following are satisfied:

• $t = C[f(q_1(u_1), \ldots, q_n(u_n))]$ for some context C and $u_1, \ldots, u_n \in T(\mathcal{F}');$

•
$$t' = C[q(u\{x_1 \leftarrow u_1, \ldots, x_n \leftarrow u_n\}]$$
 for some rule $f(q_1(x_1), \ldots, q_n(x_1)) \rightarrow q(u)$ of \mathcal{U} .

Idea: Like an NFTA, but can additionally reorder/copy/delete subtrees and "explode" symbols into subtrees like a homomorphism.

▲□▶▲圖▶▲≣▶▲≣▶ ≣ のへで 83/89

A NUTT \mathcal{U} defines the relation $\mathcal{R}(\mathcal{U}) = \{ \langle t, t' \rangle \mid t \rightarrow^*_{\mathcal{U}} q(t'), \ q \in G \}.$

Relations of NUTT

We write $\mathcal{U}(t)$ for $\{ t' \mid \langle t, t' \rangle \in \mathcal{R}(\mathcal{U}) \}$. Examples:

- Example 1: $U_1(fffa) = \{h(ffa, ffa), h(fga, fga), h(gfa, gfa), h(gga, gga)\}$
- Example 2: $\mathcal{F} = \{f(2), g(1), a\}, \mathcal{F}' = \mathcal{F};$ $\mathcal{U}_2 = \langle \{q, q', q''\}, \mathcal{F}, \mathcal{F}', \{q''\}, \Delta \rangle$, with rules $a \to q(a) \qquad g(q(x_1)) \to q(g(x_1)) \qquad f(q(x_1), q(x_2)) \to q(f(x_1, x_2))$ $a \to q'(a) \qquad g(q'(x_1)) \to q'(g(x_1))$ $f(q(x_1), q'(x_2)) \to q''(g(x_1))$ $\mathcal{R}(\mathcal{U}_2) = \{ \langle f(t, g^m(a)), g(t) \rangle \mid t \in T(\mathcal{F}), m > 0 \}$

▲□▶▲□▶▲□▶▲□▶ □ のへで 84/89

Properties of NUTT

A NUTT ${\mathcal U}$ is

- ε-free if it contains no ε-rule;
- *linear* if in rules of Δ , u is linear;
- non-erasing if in every rule, $\mathcal{H}(u) > 0$ (not just a variable);
- *complete* if for every rule with $f \in \mathcal{F}_n$ on the left-hand side, u on the right-hand side contains all of \mathcal{X}_n ;
- deterministic (DUTT) if it is ε-free and no two rules have the same left-hand side.

Examples:

- \mathcal{U}_1 is non-deterministic, non-linear, complete.
- \mathcal{U}_2 is non-deterministic, linear, non-complete.

NUTT and other relation classes

(Linear) NUTT and \mathfrak{R}_2 are incomparable

• $\mathcal{R}(\mathcal{U}_2)^{-1}$ is in \mathfrak{R}_2 , accepted by the following (with q'' final): $\langle a, a \rangle \to q \quad \langle g, g \rangle(q) \to q \quad \langle f, f \rangle(q, q) \to q$ $\langle a, - \rangle \to q' \quad \langle g, - \rangle(q') \to q' \quad \langle f, g \rangle(q, q') \to q''$

But $\mathcal{R}(\mathcal{U}_2)^{-1}$ is in \mathfrak{R}_2 is not definable by NUTT: Suppose that such a NUTT existed with k rules.

▲□▶▲@▶▲≣▶▲≣▶ ≣ のへで 86/89

• \mathfrak{R}_2 is incapable of copying or reordering subtrees.

Top-down transducers

Definition: Top-down tree transducer

A top-down tree transducer (NDTT) is a tuple $\mathcal{D} = \langle Q, \mathcal{F}, \mathcal{F}', I, \Delta \rangle$, where:

- Q is a finite set of *states* and $I \subseteq Q$ are *initial* states;
- $\mathcal{F}, \mathcal{F}'$ are finite ranked alphabets;
- Δ is a finite set of rules of the form

$$q(f)
ightarrow u[q_1(x_{i_1}), \dots, q_k(x_{i_k})]$$

for $f \in \mathcal{F}_n$, $q, q_1, \dots, q_k \in Q$, $u \in \mathcal{C}^k(\mathcal{F}')$, $x_{i_1}, \dots, x_{i_k} \in \mathcal{X}_n$, or
 $q(x_1)
ightarrow u[q_1(x_1), \dots, q_k(x_1)]$
for $q, q' \in Q$ and $u \in \mathcal{C}^k(\mathcal{F}')$ (ε -rule).

NDTT move relation

Move relation

Let $t, t' \in T(\mathcal{F}, \mathcal{F}', Q)$. We write $t \rightarrow_{\mathcal{D}} t'$ if the following are satisfied:

- $t = C[q(f(u_1, \ldots, u_n))]$ for some context C and $u_1, \ldots, u_n \in T(\mathcal{F})$;
- $t' = C[u[q_1(u_{i_1}), ..., q_k(u_{i_k})]]$ for some rule $q(f) \to u[q_1(x_{i_1}), ..., q_k(x_{i_k})]$ of \mathcal{D} .

The relation defined by \mathcal{D} is $\mathcal{R}(\mathcal{D}) = \{ \langle t, t' \rangle \mid q(t) \rightarrow t', q \in I \}.$

Example:
$$\mathcal{F} = \{f(1), a\}, \ \mathcal{F}' = \mathcal{F} \cup \{f(1), g(1), h(2), a\};$$

 $\mathcal{D}_1 = \langle \{q, q'\}, \mathcal{F}, \mathcal{F}', \{q\}, \Delta \rangle$, with rules
 $q(f(x)) \rightarrow h(q'(x), q'(x)) \qquad q'(f(x)) \rightarrow f(q'(x)) \mid g(q'(x)) \qquad q'(a) \rightarrow a$
Then $\mathcal{D}_1(ffa) = \{h(fa, fa), h(fa, ga), h(ga, fa), h(ga, ga)\}.$

Closure properties

Properties of NUTT and DUTT

- There exist relations expressible by NUTT but not NDTT.
- There exist relations expressible by NDTT but not NUTT.
- NUTT are closed under union, but not intersection.
- NUTT are not closed under composition, but linear NUTT are.

◆□▶◆□▶◆三▶◆三▶ 三 のへで 89/89

• Linear complete NUTT and NDTT are equivalent.