Introduction to Tree Language Theory

Hitoshi Ohsaki

National Institute of
Advanced Industrial Science and Technology (AIST)

seminar talk (3/10)

2009

III. Trees

Signature and trees

Given a finite set $F = \{ f_0, \ldots, f_n \}$ of symbols

F is a signature if F is equipped with mapping $\mathsf{ar}_F : F \to \mathbb{N}$

f has arity i if $\mathsf{ar}(f) = i$

$F_{(i)} = \{ f \in F \mid \mathsf{ar}_F(f) = i \}$

The set of (finite) trees over F, denoted by $\mathcal{T}_F$:

$f(t_1, \ldots, t_n) \in \mathcal{T}_F$ if $f \in F_{(n)}$ & $t_1, \ldots, t_n \in \mathcal{T}_F$ $(n \geqslant 0)$

$f()$ is simply denoted by f, called constant

Note

By definition, there is no empty tree, though the empty word exists

Positions

Let p be a word over $\mathbb{N} - \{0\}$

p is a position of t $(t \in \mathcal{T}_F)$

if $p = \varepsilon$ or

$t = f(t_1, \ldots, t_n)$ & $p = i\,q$ $(1 \leqslant i \leqslant n)$ & q is a position of t_i

pos(t) : set of positions of t (ε is root position)

size(t) : the number of elements in pos(t)

height(t) : max$\{\ |w|\ \mid\ w \in$ pos(t) $\}$

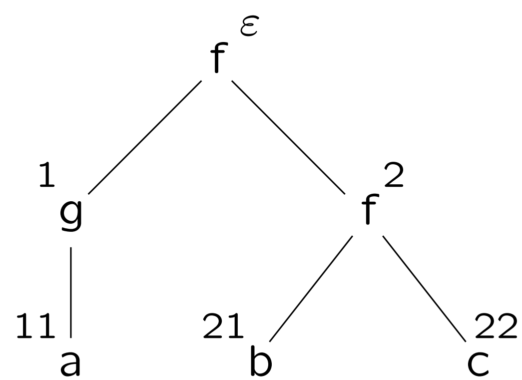
Example

Let t be the tree f(g(a), f(b, c))

pos(t) = { ε, 1, 11, 2, 21, 22 }

size(t) = 6

height(t) = 2



f $\in F_{(2)}$

g $\in F_{(1)}$

a, b, c $\in F_{(0)}$

Contexts and substitutions

Let $\Box$ be a fresh constant symbol, called hole

$\mathcal{C}_F$	:	set of trees over $F \cup \{\Box\}$, called contexts
$C[\,]$	:	context with <u>one</u> $\Box$ in C
$C[\,t\,]$	:	tree whose $\Box$ in $C[\,]$ is replaced by t

Let V be set of symbols with $F \cap V = \varnothing$ & $\mathrm{ar}_{F \cup V}(\,x\,) = 0$ $(x \in V)$

$\mathcal{T}_{F,V}$	:	set $\mathcal{T}_{F \cup V}$ of trees with variables
θ	:	mapping $V \to \mathcal{T}_{F,V}$, called assignment
$t\,\theta$	:	$\theta(x)$ if $x \in V$; $f(t_1\theta, \ldots, t_n\theta)$ if $t = f(t_1, \ldots, t_n)$ (homomorphic extension† of θ, called substitution)

Let R be binary relation over $\mathcal{T}_{F,V}$

R is closed under contexts	:	$s\,R\,t \;\Rightarrow\; C[s]\;R\;C[t]$
R is closed under substitutions	:	$s\,R\,t \;\Rightarrow\; s\theta\;R\;t\theta$

Tree automata (TA)

tree automaton $(F, Q, Q_{\text{fin}}, \Delta)$

F : signature

Q : finite set of state symbols

Q_{fin} : finite set Q_{fin} $(\subseteq Q)$ of final states

Δ : finite set of transition rules with the following forms

$$f(p_1, \ldots, p_n) \rightarrow q \qquad \text{[regular rule]}$$

$$p \rightarrow q \qquad \text{[epsilon rule]}$$

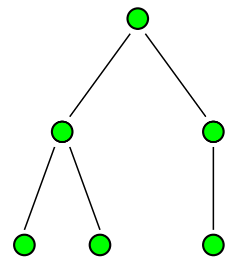
for $f \in F_{(n)}$, $p_1, \ldots, p_n, p, q \in Q$

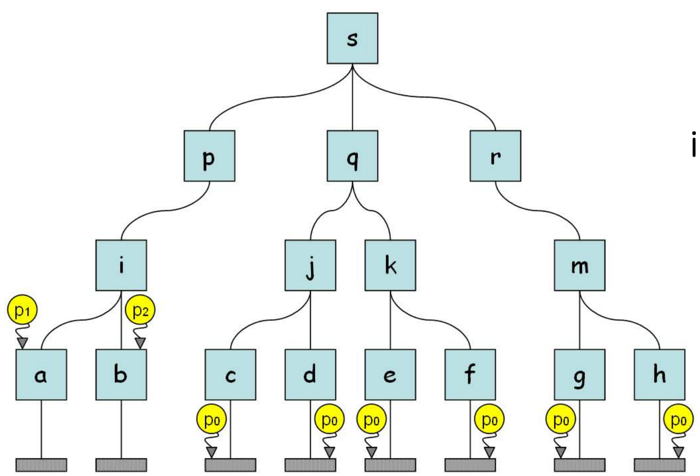
regular tree automaton (RTA) if Δ does not contain an epsilon rule

Note

RFA (= epsilon-rule free FA) & FA $\propto$ RTA & TA

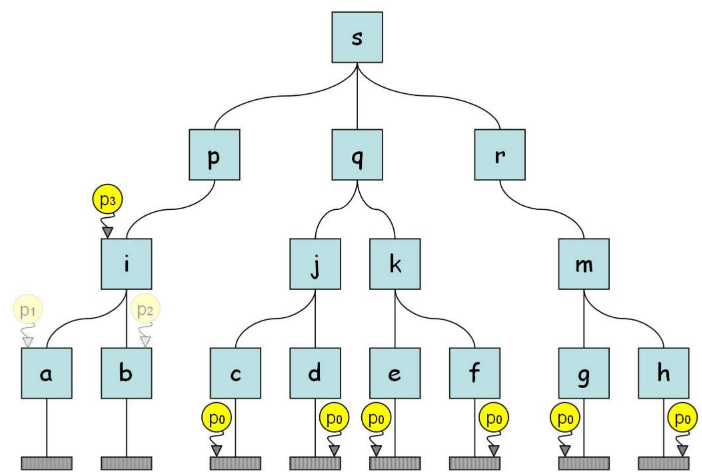
Tree automata vs. finite automata

	tree automata	automata
input		
transition rules	$f(\alpha_1, \ldots, \alpha_n) \longrightarrow \beta$ $\alpha \longrightarrow \beta$	$\alpha \xrightarrow{f} \beta$ $\alpha \longrightarrow \beta$



$i(p_1, p_2) \rightarrow p_3$

$\Rightarrow$



Accepted trees

Given a tree automaton $\mathcal{A} = (F, Q, Q_{\text{fin}}, \Delta)$

$\to_{\mathcal{A}}$ (move relation)	:	$s \to_{\mathcal{A}} t$ if $\exists\, l \to r \in \Delta$, $C[\,] \in \mathcal{C}_{F \cup Q}$: $s = C[l]$ & $t = C[r]$
t is accepted by $\mathcal{A}$ ($\mathcal{A}$ accepts t)	:	$\exists\, t \in \mathcal{T}_F,\ q \in Q_{\text{fin}} :\ t \to_{\mathcal{A}} \cdots \to_{\mathcal{A}} q$
tree language L	:	$L \subseteq \mathcal{T}_F$
tree language $\mathcal{L}(\mathcal{A})$	:	set of trees accepted by $\mathcal{A}$
$\mathcal{A}$ accepts L	:	$\mathcal{L}(\mathcal{A}) = L$
regular tree language	:	tree language accepted by a regular TA

Example

Consider $\mathcal{A} = (\{0, 1, \vee, \wedge, \neg\}, Q, Q_{\text{fin}}, \Delta)$ where

$$
\begin{array}{lll}
Q & : & q_0 \;\; q_1 \\
Q_{\text{fin}} & : & q_1 \\
\Delta & : & 0 \to q_0 \quad 1 \to q_1 \quad \neg(q_0) \to q_1 \quad \neg(q_1) \to q_0 \\
 & & \vee(q_0, q_0) \to q_0 \quad \vee(q_0, q_1) \to q_1 \quad \vee(q_1, q_0) \to q_1 \quad \vee(q_1, q_1) \to q_1 \\
 & & \wedge(q_0, q_0) \to q_0 \quad \wedge(q_0, q_1) \to q_0 \quad \wedge(q_1, q_0) \to q_0 \quad \wedge(q_1, q_1) \to q_1
\end{array}
$$

We take $\wedge(\vee(0, 1), \wedge(1, 0))$:

$$
\begin{array}{lll}
\wedge(\vee(0, 1), \wedge(1, 0)) & \to_{\mathcal{A}} & \wedge(\vee(q_0, 1), \wedge(1, 0)) \\
 & \to_{\mathcal{A}} & \wedge(\vee(q_0, q_1), \wedge(1, 0)) \\
 & \to_{\mathcal{A}} & \wedge(\vee(q_0, q_1), \wedge(q_1, 0)) \\
 & \to_{\mathcal{A}} & \wedge(\vee(q_0, q_1), \wedge(q_1, q_0)) \\
 & \to_{\mathcal{A}} & \wedge(q_1, \wedge(q_1, q_0)) \\
 & \to_{\mathcal{A}} & \wedge(q_1, q_0) \\
 & \to_{\mathcal{A}} & q_0
\end{array}
$$

$\wedge(\vee(0, 1), \wedge(1, 1)) \to^{*}_{\mathcal{A}} q_1$ accepted by $\mathcal{A}$

Deterministic tree automata (DTA)

TA $(F, Q, Q_{\text{fin}}, \Delta)$ is deterministic if

1. Δ contains regular transition rules only (so it is a regular TA)
2. there are no rules $f(p_1, \ldots, p_n) \to q_1$ and $f(p_1, \ldots, p_n) \to q_2$ with $q_1 \not\equiv q_2$

Proposition (TA = DTA)

Given a TA $\mathcal{A}$, one can construct a DTA $\mathcal{B}$ such that $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{B})$

Proof

Similar to the case of finite automata (see page 8–9 in the first talk), the procedure for constructing DTA consists of the two steps: Let $\mathcal{A} = (F, Q, Q_{\text{fin}}, \Delta)$,

(1) eliminate epsilon rules from Δ

(2) eliminate non-determinisity by subset construction for the rules $f(p_1, \ldots, p_n) \to q_1$ and $f(p_1, \ldots, p_n) \to q_2$ with $q_1 \not\equiv q_2$

Complete the proof in Exercise. □

Example

Consider $\mathcal{A} = (\{\,f,\,a\,\},\, Q,\, Q_{fin},\, \Delta\,)$ where

Q : q_a q

Q_{fin} : q

Δ : $q_a \to q$ $\quad a \to q_a$ $\quad f(\,q_a,\, q_a\,) \to q_a$

Step (1) Eliminate epsilon rules by adding transition rules

$a \to q$ as $a \to q_a \quad q_a \to q$

$f(\,q_a,\, q_a\,) \to q$ as $f(\,q_a,\, q_a\,) \to q_a \quad q_a \to q$

Step (2) Eliminate non-determinisity by subset construction (p.10 in seminar 1)

Q_q : $\{q_a\}$ $\{q\}$ $\{q_a, q\}$ $\quad * \varnothing$ is eliminated for optimization

$Q_{q\,fin}$: $\{q\}$ $\{q_a, q\}$

Δ_d : $a \to \{q_a, q\}$ as $a \to q_a$ & $a \to q$

$f(\,\{q_a\},\, \{q_a\}\,) \to \{q_a, q\}$ $\quad f(\,\{q_a, q\},\, \{q_a\}\,) \to \{q_a, q\}$

$f(\,\{q_a\},\, \{q_a, q\}\,) \to \{q_a, q\}$ $\quad f(\,\{q_a, q\},\, \{q_a, q\}\,) \to \{q_a, q\}$

as $f(\,q_a,\, q_a\,) \to q_a$ & $f(\,q_a,\, q_a\,) \to q$

Let $\mathcal{A}_d = (\,\{f, a\},\, Q_d,\, Q_{dfin},\, \Delta_d\,)$, then one can show that $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{A}_d)$

Closure properties

Let $C(TA_F)$: the class of regular tree languages over F

Proposition

The class $C(TA_F)$ is closed under union, intersection, complement

Proof for $\cup$

Suppose $\mathcal{A}_1 = (F, P, P_{\text{fin}}, \Delta_1)$ and $\mathcal{A}_2 = (F, Q, Q_{\text{fin}}, \Delta_2)$ are tree automata whose sets P, Q of state symbols are disjoint. Define $\mathcal{B} = (F, P \cup Q, P_{\text{fin}} \cup Q_{\text{fin}}, \Delta_1 \cup \Delta_2)$, then $\mathcal{L}(\mathcal{B}) = \mathcal{L}(\mathcal{A}_1) \cup \mathcal{L}(\mathcal{A}_2)$. One should note that $t \to_{\mathcal{B}} \cdots \to_{\mathcal{B}} q$ $(q \in P_{\text{fin}} \cup Q_{\text{fin}})$ if and only if $t \to_{\mathcal{A}_1} \cdots \to_{\mathcal{A}_1} q$ $(q \in P_{\text{fin}})$ or $t \to_{\mathcal{A}_2} \cdots \to_{\mathcal{A}_2} q$ $(q \in Q_{\text{fin}})$, whose derivation consists of only $\to_{\mathcal{A}_1}$ or $\to_{\mathcal{A}_2}$. $\square$

Proof for $\cap$

Suppose $\mathcal{A}_1 = (F, P, P_{\text{fin}}, \Delta_1)$ and $\mathcal{A}_2 = (F, Q, Q_{\text{fin}}, \Delta_2)$ are tree automata. First, construct the tree automata $\mathcal{A}'_1 = (F, P, P_{\text{fin}}, \Delta'_1)$ and $\mathcal{A}'_2 = (F, Q, Q_{\text{fin}}, \Delta'_2)$ such that $\mathcal{L}(\mathcal{A}_1) = \mathcal{L}(\mathcal{A}'_1)$ and $\mathcal{L}(\mathcal{A}_2) = \mathcal{L}(\mathcal{A}'_2)$ and there is no epsilon rules in Δ'_1 and Δ'_2. This transformation can be done by the procedure (1) on page 9. (Proof cont'd)

Proof for $\cap$ (cont'd)

Define $\mathcal{C} = (F, P \times Q, P_{\text{fin}} \times Q_{\text{fin}}, \Delta)$ as follows.

$$P \times Q \quad : \quad \{ \langle p, q \rangle \mid p \in P,\ q \in Q \}$$

$$P_{\text{fin}} \times Q_{\text{fin}} \quad : \quad \{ \langle p, q \rangle \mid p \in P_{\text{fin}},\ q \in Q_{\text{fin}} \}$$

$$\Delta \quad : \quad \left\{ f(\langle p_1, q_1 \rangle, \ldots, \langle p_n, q_n \rangle) \to \langle p, q \rangle \;\middle|\; \begin{array}{l} f(p_1, \ldots, p_n) \to p \in \Delta'_1 \\ f(q_1, \ldots, q_n) \to q \in \Delta'_2 \end{array} \right\}$$

$\mathcal{C}$ simulates $\mathcal{A}_1$ and $\mathcal{A}_2$ move by move, simultaneously. This means that $s \to_{\mathcal{C}} t$ using the rule $f(\langle p_1, q_1 \rangle, \ldots, \langle p_n, q_n \rangle) \to \langle p, q \rangle$ if and only if there exists the corresponding moves $s_1 \to_{\mathcal{A}_1} t_1$ by $f(p_1, \ldots, p_n) \to p$ and $s_2 \to_{\mathcal{A}_2} t_2$ by $f(q_1, \ldots, q_n) \to q$, where s_i $(1 \leqslant i \leqslant 2)$ is obtained from s by taking i-th projection of the product states. This observation implies that $s \to_{\mathcal{C}} \cdots \to_{\mathcal{C}} \langle p, q \rangle$ $(\langle p, q \rangle \in P_{\text{fin}} \times Q_{\text{fin}})$ if and only if $s \to_{\mathcal{A}_1} \cdots \to_{\mathcal{A}_1} p$ $(p \in P_{\text{fin}})$ and $s \to_{\mathcal{A}_2} \cdots \to_{\mathcal{A}_2} q$ $(q \in Q_{\text{fin}})$. □

Proof for $(\ )^{\mathsf{c}}$

Using the subset construction, define $\mathcal{D} = (F, 2^Q, \{p \in 2^Q \mid p \cap Q_{\text{fin}} \neq \varnothing\}, \Delta')$, where for each $p_1, \ldots, p_n \in 2^Q$, $f(p_1, \ldots, p_n) \to p \in \Delta'$ if $p = \{q \in Q \mid \exists q_i \in p_i\ (1 \leqslant i \leqslant n) \colon f(q_1, \ldots, q_n) \to q \in \Delta\}$. Note that $\mathcal{D}$ is deterministic and completely defined (which means for each $p_1, \ldots, p_n \in 2^Q$, there exists $f(p_1, \ldots, p_n) \to p$ in Δ'). Moreover, $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{D})$ (Exercise). Let $\mathcal{D}' = (F, 2^Q, \{p \in 2^Q \mid p \cap Q_{\text{fin}} = \varnothing\}, \Delta')$,then $\mathcal{L}(\mathcal{D}') = (\mathcal{L}(\mathcal{D}))^{\mathsf{c}}$. □

Example

Consider $\mathcal{A}_1 = (\{\, \mathsf{f},\, \mathsf{a}, \mathsf{b}\,\}, \{\, \mathsf{p}\,\}, \{\, \mathsf{p}\,\}, \Delta_1)$ and $\mathcal{A}_2 = (\{\, \mathsf{f},\, \mathsf{a}, \mathsf{b}\,\}, \{\, \mathsf{q},\, \mathsf{q_a},\, \mathsf{q_b}\,\}, \{\, \mathsf{q}\,\}, \Delta_2)$ where

$$\begin{array}{lllll} \Delta_1 & : & \mathsf{a} \to \mathsf{p} \quad & \mathsf{b} \to \mathsf{p} \quad & \mathsf{f}(\, \mathsf{p},\, \mathsf{p}\,) \to \mathsf{p} \\ \Delta_2 & : & \mathsf{a} \to \mathsf{q_a} \quad & \mathsf{b} \to \mathsf{q_b} \quad & \mathsf{f}(\, \mathsf{q_a},\, \mathsf{q_b}\,) \to \mathsf{q} \quad \mathsf{f}(\, \mathsf{q_b},\, \mathsf{q_a}\,) \to \mathsf{q} \end{array}$$

Note that

$$\begin{array}{lll} \mathcal{L}(\mathcal{A}_1) & = & \mathcal{T}_{\{\mathsf{f},\mathsf{a},\mathsf{b}\}} \\ \mathcal{L}(\mathcal{A}_2) & = & \{\, \mathsf{f}(\mathsf{a}, \mathsf{b}),\ \mathsf{f}(\mathsf{b}, \mathsf{a})\,\} \end{array}$$

Define by product construction (p.14 in seminar 1) that

$$\begin{array}{lll} Q_\times & : & (\mathsf{p}, \mathsf{q_a})\ \ (\mathsf{p}, \mathsf{q_b})\ \ (\mathsf{p}, \mathsf{q}) \\ Q_{\times\,\mathsf{fin}} & : & (\mathsf{p}, \mathsf{q}) \\ \Delta_\times & : & \mathsf{a} \to (\mathsf{p}, \mathsf{q_a}) \quad \mathsf{b} \to (\mathsf{p}, \mathsf{q_b}) \\ & & \mathsf{f}(\, (\mathsf{p}, \mathsf{q_a}),\, (\mathsf{p}, \mathsf{q_b})\,) \to (\mathsf{p}, \mathsf{q}) \quad \mathsf{f}(\, (\mathsf{p}, \mathsf{q_b}),\, (\mathsf{p}, \mathsf{q_a})\,) \to (\mathsf{p}, \mathsf{q}) \\ & & \text{superimpose transition rules of } \Delta_1 \text{ onto ones in } \Delta_2 \end{array}$$

Let $\mathcal{A}_\times = (\{\mathsf{f}, \mathsf{a}, \mathsf{b}\}, Q_\times, Q_{\times\mathsf{fin}}, \Delta_\times)$, then one can show that $\mathcal{L}(\mathcal{A}_\times) = \mathcal{L}(\mathcal{A}_1) \cap \mathcal{L}(\mathcal{A}_2)$

Example of $(\mathcal{L}(\mathcal{A}))^c$

Consider $\mathcal{A} = (\{\mathsf{f}, \mathsf{a}\}, \{\mathsf{q}, \mathsf{q_a}\}, \{\mathsf{q}\}, \Delta)$ where

$$\Delta \quad : \quad \mathsf{a} \to \mathsf{q_a} \qquad \mathsf{f}(\mathsf{q_a}, \mathsf{q_a}) \to \mathsf{q}$$

Note that

$$\mathcal{L}(\mathcal{A}) \;=\; \{\mathsf{f}(\mathsf{a}, \mathsf{a})\}$$

Define $\mathcal{A}_\mathsf{c} = (\{\mathsf{f}, \mathsf{a}, \mathsf{b}\}, \mathcal{Q}_\mathsf{c}, \mathcal{Q}_{\mathsf{c\,fin}}, \Delta_\mathsf{c})$ where

$$\mathcal{Q}_\mathsf{c} \quad : \quad \varnothing \quad \{\mathsf{q}\} \quad \{\mathsf{q_a}\} \quad \{\mathsf{q}, \mathsf{q_a}\}$$

$$\mathcal{Q}_{\mathsf{c\,fin}} \quad : \quad \varnothing \quad \{\mathsf{q_a}\}$$

$$\Delta_\mathsf{c} \quad : \quad \mathsf{a} \to \{\mathsf{q_a}\}$$

$$\mathsf{f}(\varnothing, \varnothing) \to \varnothing \qquad \mathsf{f}(\varnothing, \{\mathsf{q}\}) \to \varnothing$$

$$\mathsf{f}(\varnothing, \{\mathsf{q_a}\}) \to \varnothing \qquad \mathsf{f}(\varnothing, \{\mathsf{q}, \mathsf{q_a}\}) \to \varnothing$$

$$\mathsf{f}(\{\mathsf{q}\}, \varnothing) \to \varnothing \qquad \mathsf{f}(\{\mathsf{q}\}, \{\mathsf{q}\}) \to \varnothing$$

$$\mathsf{f}(\{\mathsf{q}\}, \{\mathsf{q_a}\}) \to \varnothing \qquad \mathsf{f}(\{\mathsf{q}\}, \{\mathsf{q}, \mathsf{q_a}\}) \to \varnothing$$

$$\mathsf{f}(\{\mathsf{q_a}\}, \varnothing) \to \varnothing \qquad \mathsf{f}(\{\mathsf{q_a}\}, \{\mathsf{q}\}) \to \varnothing$$

$$\mathsf{f}(\{\mathsf{q_a}\}, \{\mathsf{q_a}\}) \to \{\mathsf{q}\} \qquad \mathsf{f}(\{\mathsf{q_a}\}, \{\mathsf{q}, \mathsf{q_a}\}) \to \{\mathsf{q}\}$$

$$\mathsf{f}(\{\mathsf{q}, \mathsf{q_a}\}, \varnothing) \to \varnothing \qquad \mathsf{f}(\{\mathsf{q}, \mathsf{q_a}\}, \{\mathsf{q}\}) \to \varnothing$$

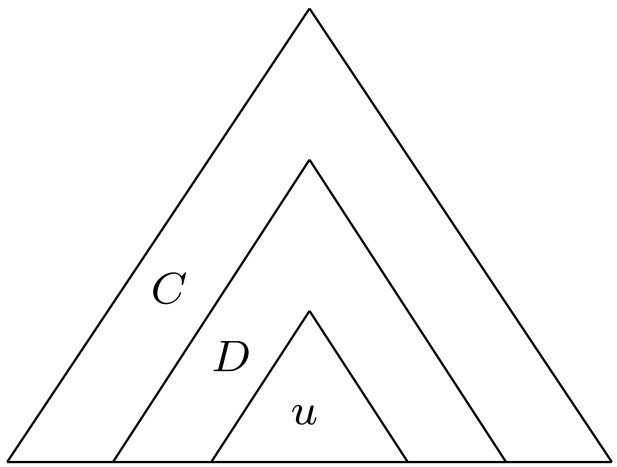
$$\mathsf{f}(\{\mathsf{q}, \mathsf{q_a}\}, \{\mathsf{q_a}\}) \to \{\mathsf{q}\} \qquad \mathsf{f}(\{\mathsf{q}, \mathsf{q_a}\}, \{\mathsf{q}, \mathsf{q_a}\}) \to \{\mathsf{q}\}$$

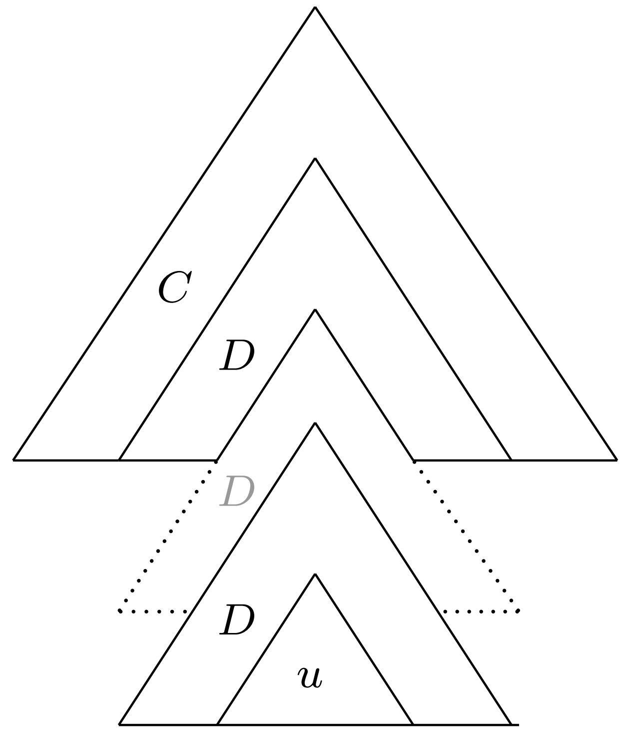
Pumping lemma for TA

Given regular tree language L,

$\exists\ k \geqslant 0\ :\ $ if $t \in L$ & height$(t) \geqslant k$

then $t = C[D[u]]$





such that size$(D) > 1$ & $C[D^n[u]] \in L\ \ (n \geqslant 0)$

Proof

Recall the proof of Pumping lemma for CFG (pages 9-10 in seminar talk 2). Suppose $L = \mathcal{L}(\mathcal{A})$ and $\mathcal{A} = (F, Q, Q_{\mathsf{fin}}, \Delta)$ is regular. If $t \in \mathcal{L}(\mathcal{A})$ and height$(t) \geqslant$ min$(|Q|, |\Delta|)$, then $t \to^*_{\mathcal{A}} q$ $(q \in Q)$ implies that $t = C[D[u]]$ such that $u \to^*_{\mathcal{A}} p$ and $D[p] \to^*_{\mathcal{A}} p$ and $C[p] \to^*_{\mathcal{A}} q$ for some $p \in Q$. Since $D^n[p] \to^*_{\mathcal{A}} p$ $(n \geqslant 0)$, we have $C[D^n[u]] \to^*_{\mathcal{A}} q$. $\square$

Example

Consider $\mathcal{A}_1 = (\{\, \mathsf{f}, \mathsf{a} \,\}, \{\, \mathsf{q}, \mathsf{q_a} \,\}, \{\, \mathsf{q} \,\}, \Delta_1)$ and $\mathcal{A}_2 = (\{\, \mathsf{f}, \mathsf{a} \,\}, \{\, \mathsf{q}, \mathsf{q_a} \,\}, \{\, \mathsf{q} \,\}, \Delta_2)$ where

$$\Delta_1 \quad : \quad \mathsf{a} \to \mathsf{q_a} \qquad \mathsf{f}(\, \mathsf{q_a}, \mathsf{q_a} \,) \to \mathsf{q}$$

$$\Delta_2 \quad : \quad \mathsf{a} \to \mathsf{q_a} \qquad \mathsf{f}(\, \mathsf{q_a}, \mathsf{q_a} \,) \to \mathsf{q_a} \qquad \mathsf{f}(\, \mathsf{q_a}, \mathsf{q_a} \,) \to \mathsf{q}$$

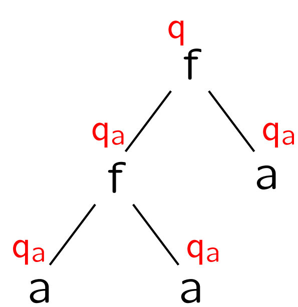
Let $k_1 = \min(\, \{\mathsf{q}, \mathsf{q_a}\}, \, \Delta_1 \,) = 2$

There is no tree $t \in \mathcal{L}(\mathcal{A}_1)$ such that $\mathsf{height}(t) \geqslant k_1$

No tree needed to be decomposed as the height of trees in $\mathcal{L}(\mathcal{A}_1)$ is $< k_1$

Let $k_2 = \min(\, \{\mathsf{q}, \mathsf{q_a}\}, \, \Delta_2 \,) = 2$

For every tree $t \in \mathcal{L}(\mathcal{A}_2)$ such that $\mathsf{height}(t) \geqslant k_2$, e.g., $\mathsf{f}(\mathsf{f}(\mathsf{a}, \mathsf{a}), \mathsf{a})$:



$$C = \mathsf{f}(\square, \mathsf{a})$$
$$D = \mathsf{f}(\square, \mathsf{a})$$
$$u = \mathsf{a}$$

we have $t = C[D[u]]$ such that $\mathsf{size}(D) > 1$ & $C[D^n[u]]$ $(n \geqslant 0)$

Decidability

The following problems are decidable for the class of tree automata :

$t \in \mathcal{L}(\mathcal{A})$?	(membership problem)
$\mathcal{L}(\mathcal{A}) = \varnothing$?	(emptiness problem)
$\mathcal{L}(\mathcal{A}) = \mathcal{T}_F$?	(universality problem)
$\mathcal{L}(\mathcal{A}) \subseteq \mathcal{L}(\mathcal{B})$?	(inclusion problem)
$\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{B})$?	(equivalence problem)

Proof

Observe that the universality, inclusion, equivalence problems are special cases of the emptiness problem under the closure property of TA stating that for any TA $\mathcal{A}, \mathcal{B}$, one can construct tree automata, each of which accepts the union, intersection and complement. Moreover, according to the previous proof, $\mathcal{A} = (F, Q, Q_{\mathsf{fin}}, \Delta)$ accepts *some* tree (so $\mathcal{L}(\mathcal{A}) \neq \varnothing$) if and only if $\mathcal{A}$ accepts a tree whose height is at most $\min(|Q|, |\Delta|)$. The set T of trees with a bounded height is finite, because every symbol in F has a fixed arity. So, if the membership problem is decidable, one can determine whether there exists a tree in T that is accepted by A. The decidability of the membership problem is explained in the next proof. □

Proof for the membership problem

Use König's lemma (1936) stating that :

every finitely branching tree is finite if and only if it has only finite paths.

A tree with a finite number of branches at each fork and with a finite number of leaves at the end of each branch is called a finitely branching tree. Suppose $\mathcal{A} = (F, Q, Q_{\mathsf{fin}}, \Delta)$ does not contain an epsilon rule, because it is possible to obtain an equivalent RTA from a given TA. Observe that for each move $s \to_{\mathcal{A}} t$, the number of symbols in F occurring in s decreases. This implies that every sequence $t_1\, t_2 \ldots$ of trees of $\mathcal{T}_{F \cup Q}$ is finite if $t_i \to_{\mathcal{A}} t_{i+1}$ $(i \geqslant 1)$. Now we consider computation trees of $\mathcal{A}$ which are (possibly infinite) trees, and each of whose node is labeled by a tree in $\mathcal{T}_{F \cup Q}$ such that nodes s, t are connected by an edge from s to t if and only if $s \to_{\mathcal{A}} t$. Then, by definition, every computation tree represents all possible trails of the computation by $\mathcal{A}$ (starting from some tree in $\mathcal{T}_F$). Because $\mathcal{A}$ contains finitely many transition rules, all these computation trees are finitely branching. Moreover, because of the above observation, every path from the root toward a leaf (called run) is finite. Hence, every computation tree of $\mathcal{A}$ is finite according to König's lemma. The membership problem for TA can then be rephrased to the question if a computation tree whose root node is labeled by a given tree in $\mathcal{T}_F$ has a leaf of some state in Q_{fin}. Since the comprehensive search of a finite tree can be done by finitely many recursive calls, the membership problem for TA is decidable. $\square$

Myhill-Nerode's theorem (for words)

Given a language L over Σ

Let $x \approx_L y$: binary relation over Σ^* such that for all $z \in \Sigma^*$,
$xz \in L$ if and only if $yz \in L$ (called L-equivalent)

Proposition

The statements 1 & 2 are equivalent : for a language L over Σ,

1. L is regular
2. $\{\, \{\, x \in \Sigma^* \mid x \approx_L w \,\} \mid \exists w \in \Sigma^* \,\}$ is finite

Proof

To show "1 $\Rightarrow$ 2", suppose that DTA $\mathcal{A} = (F, Q, \mathsf{q}_0, Q_{\mathsf{fin}}, \Delta)$ accepts L. Define $u \sim_{\mathcal{A}} w$ if and only if $\mathcal{A}$ halts with the same state on u and w. $\mathcal{A}$ is not necessarily completely defined. Then, $u \sim_{\mathcal{A}} w$ implies $u \approx_L w$, because for any $z \in \Sigma^*$, if $u \sim_{\mathcal{A}} w$, $\mathcal{A}$ halts with the same state on uz and wz. Since the size of $\{\, \{\, x \in \Sigma^* \mid x \sim_{\mathcal{A}} w \,\} \mid \exists w \in \Sigma^* \,\}$ is $|Q|$, the size of $\{\, \{\, x \in \Sigma^* \mid x \approx_L w \,\} \mid \exists w \in \Sigma^* \,\}$ is at most $|Q|$. (Proof cont'd)

Proof (cont'd)

To show "2 ⇒ 1", define $Q = \{\{ x \in \Sigma^* \mid x \approx_L w \} \mid \exists w \in \Sigma^* \}$, $\mathsf{q}_0 = \{ x \in \Sigma^* \mid x \approx_L \varepsilon \}$, $Q_{\mathsf{fin}} = \{\{ x \in \Sigma^* \mid x \approx_L w \} \mid \exists w \in L \}$, $\Delta = \{ p \xrightarrow{a} q \mid p, q \in Q,\ a \in \Sigma \colon \exists x \in p \ \&\ xa \in q \}$. By assumption, Q is finite, and q_0 must be an element in Q. Besides, Q_{fin} is finite, because Q is finite. Let $\mathcal{B} = (\Sigma, Q, \mathsf{q}_0, Q_{\mathsf{fin}}, \Delta)$, then $\mathcal{B}$ is completely defined DFA, because each $q \in Q$ is non-empty, and $p \not\equiv q$ $(p, q \in Q)$ implies $p \cap q = \varnothing$. By case analysis of input, one can show that $\mathcal{L}(\mathcal{B}) = L$: Obviously, $\varepsilon \in L$ if and only if $\mathsf{q}_0 \in Q_{\mathsf{fin}}$. If $\mathcal{B}$ reads w and halts with the state p, then wa $(a \in \Sigma)$ is accepted by $\mathcal{B}$ if and only if there exists a rule $p \xrightarrow{a} q$ in Δ with some q in Q_{fin}. Because $x \approx_L y$ and $y \in L$ imply $x \in L$, $wa \approx_L u$ for some $u \in L$ (so $wa \in L$) if and only if $q \in Q_{\mathsf{fin}}$. □

Corollary

For every regular language L over Σ, the following statements are equivalent :

1. the number of elements in $\{ \{ x \in \Sigma^* \mid x \approx_L w \} \mid w \in \Sigma^* \}$ is k
2. the number of states of DFA that accepts L is at least k

Proof

"1 ⇒ 2" is an easy consequence of the previous proof. For the reverse, it suffices to show that $u \sim_{\mathcal{A}} w$ if and only if $u \approx_L w$, provided that $\mathcal{A}$ is minimal DFA. Here DFA $\mathcal{A}$ is minimal if there is no DFA that accepts $\mathcal{L}(\mathcal{A})$ and whose number of states is less than $\mathcal{A}$'s. To complete the proof (showing that $u \nsim_{\mathcal{A}} w$ implies $u \not\approx_L w$) is Exercise. □

Myhill-Nerode's theorem for trees

Given a tree language L over F

Let $s \approx_L t$: binary relation over $\mathcal{T}_F$ such that for all $C \in \mathcal{C}_F$,

$C[s] \in L$ if and only if $C[t] \in L$

Proposition

The statements 1 & 2 are equivalent : for a tree language L over F,

1. L is regular
2. $\{ \{ s \in \mathcal{T}_F \mid s \approx_L t \} \mid \exists t \in \mathcal{T}_F \}$ is finite

Proof

To show "1 $\Rightarrow$ 2", similar to the previous proof, one can show that the size of the set $\{\{ s \in \mathcal{T}_F \mid s \approx_L t \} \mid \exists t \in \mathcal{T}_F \}$ is at most the number of state symbols of DTA that accepts L. To show "2 $\Rightarrow$ 1", define the TA $\mathcal{B} = (F, Q, Q_{\mathsf{fin}}, \Delta)$, where $Q = \{\{ s \in \mathcal{T}_F \mid s \approx_L t \} \mid \exists t \in \mathcal{T}_F \}$, $Q_{\mathsf{fin}} = \{\{ s \in \mathcal{T}_F \mid s \approx_L t \} \mid \exists t \in L \}$, $\Delta = \{ f(p_1, \ldots, p_n) \to q \mid p_1, \ldots, p_n, q \in Q, f \in F : \exists t_i \in p_i (1 \leqslant i \leqslant n) \ \&\ f(t_1, \ldots, t_n) \in q \}$. One can show that for all $t \in \mathcal{T}_F$, t is accepted by $\mathcal{B}$ if and only if $t \in L$. $\square$

Exercise

1. Complete the proof on page 9.
2. Show the claim in the proof on page 11 that $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{D})$.
3. Verify Corollary on page 20. (Complete the proof of "2 $\Rightarrow$ 1".)
4. Show that given a TA $\mathcal{A}$, one can construct minimal DTA that accepts $\mathcal{L}(\mathcal{A})$.
5. Define $\mathsf{leaf}(t) = \mathsf{leaf}(t_1) \cdots \mathsf{leaf}(t_n)$ if $t = f(t_1, \ldots, t_n)$ and $n \geqslant 1$; $\mathsf{leaf}(t) = t$ if $t \in F_{(0)}$. Let $\mathsf{C}(\mathsf{RLT}_F)$ be the class of tree languages over F such that $L \in \mathsf{C}(\mathsf{RLT}_F)$ if there exists a regular grammar $\mathcal{G}$ with $L = \{ t \in \mathcal{T}_F \mid \mathsf{leaf}(t) \in \mathcal{L}(\mathcal{G}) \}$. Show that (1) $\mathsf{C}(\mathsf{RLT}_F) \subsetneq \mathsf{C}(\mathsf{TA}_F)$, and (2) $\mathsf{C}(\mathsf{RLT}_F)$ is closed under Boolean operations.
6. Let L be the tree language over $\{\mathsf{f}, \mathsf{a}, \mathsf{b}\}$ such that $\mathsf{f}(\mathsf{a}, \mathsf{b}) \in L$; $\mathsf{f}(t_1, t_2) \in L$ if $t_1, t_2 \in L$. Show that L is a regular tree language.
7. Let $F = \{\mathsf{f}, \mathsf{a}, \mathsf{b}\}$ with $\mathsf{ar}_F(\mathsf{f}) = 2$ and $\mathsf{ar}_F(\mathsf{a}) = \mathsf{ar}_F(\mathsf{b}) = 0$, and $L = \{t \in \mathcal{T}_F \mid$ the numbers of occurrences of a and b in t are the same$\}$. Show that L is not a regular tree language.

Appendix : "On the Myhill-Nerode Theorem for Trees"

In late 1950's, Nerode stated in [1] the well-known property originated from Myhill's early work [2]. After a while, Brainerd generalized this result for automata on finite trees [3]. His generalized result, however, seemed less accessible for those who are not familiar with universal algebra or category theory. In 1977, Kozen showed an accessible presentation of the tree version of Myhill-Nerode's theorem in his thesis (without proof). Fülüp & Vágvölgyi also but independently showed a similar result (with proof) in 1989. Kozen then provided a simple proof and a simple notion of "tree automata" in [4] which is quite acceptable for ones familiar with the original Myhill-Nerode's theorem.

---

[1] A. Nerode: *Linear automaton transformations*, Proc. of the American Mathematical Society (AMS), vol.9, pp 541-544, 1958.

[2] J. Myhill: *Finite Automata and the Representation of Events*, technical report 57-264, Wright Air Development Center (WADC), 1957.

[3] W.S. Brainerd: *The Minimization of Tree Automata*, Information and Control 13, pp.484–491, 1968.

[4] D. Kozen: *On the Myhill-Nerode Theorem for Trees*, Bulletin of the European Association for Theoretical Computer Science 47, pp.170–173, 1992.

National Institute of Advanced Industrial Science and Technology (AIST) – Senri-site, AIST Kansai.

Office: Shin-Senri Nishi 1–2–14 (MSK bldg. 5th floor), Toyonaka, Osaka 560–0083, Japan

URL: **http://staff.aist.go.jp/hitoshi.ohsaki/**