Universality on Tree Automata of Moore Type

Eija Jurvanen

Turku Centre for Computer Science, and
Department of Mathematics, University of Turku
jurvanen@utu.fi

Marjo Lipponen

Turku Centre for Computer Science, and
Department of Mathematics, University of Turku
marlip@utu.fi

Turku Centre for Computer Science
TUCS Technical Report No 189
June 1998

ISBN 952-12-0241-6
ISSN 1239-1891

Abstract

We define a new type of tree automaton with output as a generalization of Moore automata. The new structure allows us to study how Moore's gedanken-experiments, where the experimenter knows only the inputs and outputs of the automaton, are applicable to trees. In particular, we study simulation, minimality and universality of such automata.

TUCS Research Group
Mathematical Structures of Computer Science

1 Introduction

To study Moore experiments on tree automata we define a new type of tree automaton; we equip a deterministic frontier-to-root tree automaton with output but leave out the initial assignment and final states. The outputs will depend only on states as in Moore automata.

In a Moore experiment an unknown automaton, a "black box", is tested. The experimenter can decide the input and observe the output but has no knowledge of the transition relation or the state in which the machine is. The main conclusion is that it is in general impossible to recover the initial state of the automaton. In a way, the experiment disturbs the machine so much that the knowledge of the initial state is lost. This even suggests a computational counterpart of physical complementarity, such as Heisenberg uncertainty principle, see Moore [8] and Calude, Calude, Svozil, Yu [2].

The tree automaton with output can be viewed as a tree transducer of a special type, a deterministic frontier-to-root relabeling, see Gécseg and Steinby [5, 6]. However, an ordinary deterministic frontier-to-root tree recognizer can easily be restored from a tree automaton with output. First, we have to distinguish the set of accepting symbols among the output symbols and let the tree recognizer accept any tree whose root is entered in a state outputting an accepting symbol. Secondly, the tree automaton with output reads trees having states as leaves. Adding an initial assignment makes the tree recognizer read trees with any set of leaves.

In Calude, Calude, Khoussainov [1] simulation, minimality and universality were studied for finite deterministic automata with output but without initial or final states. Similar studies were done for incomplete automata in Calude and Lipponen [4, 3]. Continuing these works, we compare various simulations of tree automata, define universal tree automata, and study minimal and equivalent tree automata. The main result shows that if a class has a universal tree automaton then it has a minimal universal tree automaton unique up to isomorphism. Furthermore, since ordinary words or strings can be viewed as unary trees, the model we have presented is a generalization of the finite automaton model and all the results in [1] can be drawn from our results.

2 Notations

The definitions concerning trees follow Gécseg and Steinby [5, 6].

A *ranked alphabet* Σ is a finite nonempty set of symbols such that each symbol $\sigma \in \Sigma$ has a unique positive *rank*. For each $m \geq 1$, the set of m-ary symbols in the ranked alphabet Σ is denoted by Σ_m.

If Σ is a ranked alphabet and X a leaf alphabet (a finite set of symbols), then the set $T_\Sigma(X)$ of *Σ-terms with variables in X* is defined inductively as the smallest set T such that

- $X \subseteq T$, and
- $\sigma(t_1, \ldots, t_m) \in T$ whenever $m \geq 1$, $\sigma \in \Sigma_m$, and $t_1, \ldots, t_m \in T$.

A Σ-term with variables in X is called a *ΣX-tree.*

The *height* $\mathrm{hg}(t)$ of a ΣX-tree t is defined so that $\mathrm{hg}(t) = 0$ if $t \in X$ and $\mathrm{hg}(t) = 1 + \max\{hg(t_i) \mid 1 \leq i \leq m\}$ if $t = \sigma(t_1, \ldots, t_m)$.

A Σ-*algebra* $\mathcal{A}$ is a pair $(Q_A, (\sigma^{\mathcal{A}} \mid \sigma \in \Sigma))$ which consists of a nonempty set Q_A, the set of *elements* of $\mathcal{A}$, and a Σ-indexed family of operations on Q_A. If $\sigma \in \Sigma_m$, then $\sigma^{\mathcal{A}} : Q_A^m \to Q_A$ is an m-ary operation. For each $u \in T_\Sigma(Q_A)$, the mapping $\cdot^{\mathcal{A}} : T_\Sigma(Q_A) \to Q_A$ is defined so that $u^{\mathcal{A}} = a$ if $u = a \in Q_A$, and $u^{\mathcal{A}} = \sigma^{\mathcal{A}}(u_1^{\mathcal{A}}, \ldots, u_m^{\mathcal{A}})$ if $u = \sigma(u_1, \ldots, u_m)$.

A *(homo)morphism* of a Σ-algebra $\mathcal{A} = (Q_A, \Sigma)$ into a Σ-algebra $\mathcal{B} = (Q_B, \Sigma)$ is a mapping $\varphi : Q_A \to Q_B$ such that for all $\sigma \in \Sigma_m$, $m \geq 1$, and $a_1, \ldots, a_m \in Q_A$, $\sigma^{\mathcal{A}}(a_1, \ldots, a_m)\varphi = \sigma^{\mathcal{B}}(a_1\varphi, \ldots, a_m\varphi)$. Hence, a morphism is an operation-preserving mapping between Σ-algebras and will be denoted by $\varphi : \mathcal{A} \to \mathcal{B}$.

Let Σ be a ranked alphabet and O a finite alphabet. A *(deterministic frontier-to-root) tree automaton with output* is a system $\mathbf{A} = (\mathcal{A}, F_A)$. Here $\mathcal{A} = (Q_A, \Sigma)$ is a finite Σ-algebra whose elements are called *states*, and $F_A : Q_A \to O$ is a mapping from the set of states Q_A to the output alphabet O.

For the definition of the output tree we define a new ranked alphabet $\Omega = \bigcup_{m \geq 1} \Omega_m$ such that the set Ω_m is empty if Σ_m is empty; otherwise, $\Omega_m = \{\omega_m \mid \omega \in O\}$. An *output tree* is an ΩO-tree and can be obtained from the extension $F_A^* : T_\Sigma(Q_A) \to T_\Omega(O)$ of the output mapping F_A by

- $F_A^*(a) = F_A(a)$ for $a \in Q_A$, and
- $F_A^*(t) = F_A(t^{\mathcal{A}})_m(F_A^*(t_1), \ldots, F_A^*(t_m))$ for $t = \sigma(t_1, \ldots, t_m)$.

Two tree automata with output $\mathbf{A}$ and $\mathbf{B}$ are *isomorphic* if there is a bijective morphism $h : \mathcal{A} \to \mathcal{B}$ such that $F_A(a) = F_B(ah)$ for all $a \in Q_A$.

An input signal of a tree automaton is defined as follows. Let $X_n = \{x_1, \ldots, x_n\}$, with the convention that $X_0 = \emptyset$, and let $X = \bigcup_{n \geq 0} X_n$. Take a new symbol $x \notin X$. For any $n \geq 0$, the tree $p(x_1, \ldots, x_n; x) \in T_\Sigma(X_n \cup \{x\})$ is a *polynomial* if

1. $p(x_1, \ldots, x_n; x) = x$, or
2. $p(x_1, \ldots, x_n; x) = \sigma(y_1, \ldots, y_{j-1}, p'(x_1, \ldots, x_n; x), y_{j+1}, \ldots, y_m)$ where $\sigma \in \Sigma_m$, $y_i \in \{x_1, \ldots, x_n\}$, and $p'(x_1, \ldots, x_n; x)$ is a polynomial.

The class of polynomials with n variables is denoted by $\mathcal{P}_\Sigma(X_n)$ and the set of all polynomials by $\mathcal{P}_\Sigma(X) = \bigcup_{n \geq 0} \mathcal{P}_\Sigma(X_n)$. An *input signal* of an automaton $\mathbf{A}$ is a polynomial $p(a_1, \ldots, a_n; x)$ where each x_i is replaced by some $a_i \in Q_A$. The set of all input signals of $\mathbf{A}$ is denoted by $\mathcal{P}_\Sigma(Q_A)$. In the beginning of each computation x is replaced by a *starting state* $a \in Q_A$ and we write $p(a_1, \ldots, a_n; a) \in T_\Sigma(Q_A)$ or sometimes simply $p(a_1, \ldots, a_n)$ if $a \in \{a_1, \ldots, a_n\}$. Note that polynomials and input signals are special trees, see e.g. Jurvanen [7], since exactly one of the leaves is labelled by x; however, the height of a polynomial is always the distance of x from the root.

There are two ways to define how a tree automaton $\mathbf{A} = (\mathcal{A}, F_A)$ *responds* to an input signal $p(a_1, \ldots, a_n; x) \in \mathcal{P}_\Sigma(Q_A)$.

1. The *total response* of $\mathbf{A}$ is the function $R_A : Q_A \times \mathcal{P}_\Sigma(Q_A) \to T_\Omega(O)$ defined for all $a, a_i \in Q_A$, by

$$\begin{aligned} &R_A(a, x) = F_A(a), \text{ and} \\ &R_A(a, p(a_1, \ldots, a_n; x)) = F_A^*(p(a_1, \ldots, a_n; a)). \end{aligned}$$

2. The *final response* of $\mathbf{A}$ is the function $f_A : Q_A \times \mathcal{P}_\Sigma(Q_A) \to O$ defined by $f_A(a, p(a_1, \ldots, a_n; x)) = F_A(p^{\mathcal{A}}(a_1, \ldots, a_n; a))$ for all $a, a_i \in Q_A$.

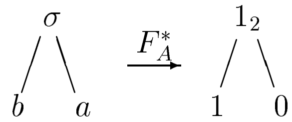
Hence the total response of $\mathbf{A}$ to a polynomial is an output tree and the final response is the output of the state reached at the root of the polynomial.

Example 2.1 Let $\mathbf{A} = (\mathcal{A}, F_A)$ be a tree automaton such that $\Sigma = \{\sigma\}$, $O = \{0, 1\}$, $\mathcal{A} = (Q_A, \Sigma)$, $Q_A = \{a, b, c\}$, and $F_A(a) = 0$, $F_A(b) = 1$, $F_A(c) = 0$. Define the binary operation $\sigma \in \Sigma_2$ by the table below.

$\sigma^{\mathcal{A}}$	a	b	c
a	b	c	c
b	b	c	c
c	c	c	c

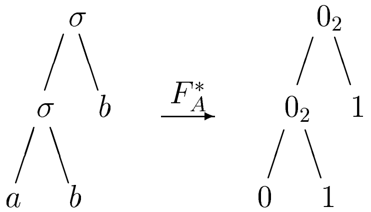
Let $p_1(x_1, x_2, x_3; x) = \sigma(x_2, x)$ and $p_2(x_1, x_2, x_3; x) = \sigma(\sigma(x_1, x), x_2)$ be two polynomials. The responses of $\mathbf{A}$ to the input signal $p_1(a, b, c; x) = \sigma(b, x)$ are

$$\begin{aligned} R_A(a, p_1(a, b, c; x)) &= F_A^*(\sigma(b, a)) \\ &= F_A(\sigma^{\mathcal{A}}(b, a))_2(F_A(b), F_A(a)) \\ &= F_A(b)_2(F_A(b), F_A(a))) \\ &= 1_2(1, 0), \text{ and} \\ f_A(b, p_1(a, b, c; x)) &= F_A(\sigma^{\mathcal{A}}(b, b)) = F_A(c) = 0. \end{aligned}$$



The responses to an input $p_2(a, b, c; x) = \sigma(\sigma(a, x), b)$ are

$$\begin{aligned} R_A(b, p_2(a, b, c; x)) &= F_A^*(\sigma(\sigma(a, b), b)) \\ &= 0_2(0_2(0, 1), 1), \text{ and} \\ f_A(a, p_2(a, b, c; x)) &= F_A(\sigma^{\mathcal{A}}(\sigma^{\mathcal{A}}(a, a), b)) \\ &= F_A(c) \\ &= 0. \end{aligned}$$



3 Simulation

Simulation usually means that a computer is able to perform the same computations as another one. Hence we can require that after receiving the input the operations are done in the same way in both devices producing the same outputs or that the inner operations do not matter as long as the outputs are the same. Following Calude, Calude and Khoussainov [1], we call the first type strong simulation and the second type behavioral simulation (or simply simulation) and define them in terms of responses for automata.

In what follows the ranked alphabet Σ and the output alphabet O are fixed. Let $\mathbf{A} = (\mathcal{A}, F_A)$ and $\mathbf{B} = (\mathcal{B}, F_B)$ be two tree automata. We say that

- $\mathbf{A}$ is *strongly simulated* by $\mathbf{B}$, $\mathbf{A} \ll \mathbf{B}$, if there is a morphism $h : \mathcal{A} \to \mathcal{B}$ such that for all $a, a_i \in Q_A$ and $p(x_1, \ldots, x_n; x) \in \mathcal{P}_\Sigma(X)$,

$$R_A(a, p(a_1, \ldots, a_n; x)) = R_B(ah, p(a_1h, \ldots, a_nh; x)).$$

- **A** is *strongly f-simulated* by **B**, $\mathbf{A} \ll_f \mathbf{B}$, if there is a morphism $h : \mathcal{A} \to \mathcal{B}$ such that for all $a, a_i \in Q_A$ and $p(x_1, \ldots, x_n; x) \in \mathcal{P}_\Sigma(X)$,

$$f_A(a, p(a_1, \ldots, a_n; x)) = f_B(ah, p(a_1 h, \ldots, a_n h; x)).$$

- **A** is *simulated* by **B**, $\mathbf{A} \leq \mathbf{B}$, if there is a mapping $h : Q_A \to Q_B$ such that for all $a, a_i \in Q_A$ and $p(x_1, \ldots, x_n; x) \in \mathcal{P}_\Sigma(X)$,

$$R_A(a, p(a_1, \ldots, a_n; x)) = R_B(ah, p(a_1 h, \ldots, a_n h; x)).$$

- **A** is *f-simulated* by **B**, $\mathbf{A} \leq_f \mathbf{B}$, if there is a mapping $h : Q_A \to Q_B$ such that for all $a, a_i \in Q_A$ and $p(x_1, \ldots, x_n; x) \in \mathcal{P}_\Sigma(X)$,

$$f_A(a, p(a_1, \ldots, a_n; x)) = f_B(ah, p(a_1 h, \ldots, a_n h; x)).$$

Further on, we say that **A** and **B** are *equivalent* (resp. *strongly equivalent*) if $\mathbf{A} \leq \mathbf{B}$ and $\mathbf{B} \leq \mathbf{A}$ (resp. $\mathbf{A} \ll \mathbf{B}$ and $\mathbf{B} \ll \mathbf{A}$). Both of these relations are clearly equivalence relations.

If **A** is strongly simulated by **B**, then **A** is also simulated by **B**. The following simple example due to Steinby shows that the converse is not true. If **A** has one state, **B** has two states and the output alphabet consists of only one element, then there is a Σ-algebra $\mathcal{B}$ such that no mapping $h : Q_A \to Q_B$ is a morphism.

Example 3.1 Let $\mathbf{A} = (\mathcal{A}, F_A)$ and $\mathbf{B} = (\mathcal{B}, F_B)$ be two tree automata such that $\Sigma = \{\sigma\}$, $O = \{0, 1\}$, $\mathcal{A} = (Q_A, \Sigma)$, $Q_A = \{a, b, c\}$, $F_A(a) = F_A(c) = 0$, $F_A(b) = 1$, and $\mathcal{B} = (Q_B, \Sigma)$, $Q_B = \{q, r, s, t\}$, $F_B(q) = F_B(t) = 0$, $F_B(r) = F_B(s) = 1$. Moreover, we define the Σ-algebras $\mathcal{A}$ and $\mathcal{B}$ by the tables below.

$\sigma^{\mathcal{A}}$	a	b	c
a	b	c	c
b	b	c	c
c	c	c	c

$\sigma^{\mathcal{B}}$	q	r	s	t
q	r	t	t	t
r	s	t	t	t
s	s	t	t	t
t	t	t	t	t

Let $h : Q_A \to Q_B$ be a mapping such that $ah = q$, $bh = s$ and $ch = t$. To prove that **A** is simulated by **B** via h, we first show that

$$p^{\mathcal{B}}(ah, bh, ch) = \begin{cases} q, & \text{if } p^{\mathcal{A}}(a, b, c) = a, \\ r \text{ or } s, & \text{if } p^{\mathcal{A}}(a, b, c) = b, \\ t, & \text{if } p^{\mathcal{A}}(a, b, c) = c \end{cases} \tag{1}$$

for all $p \in T_\Sigma(X_3)$. The claim is clear if $p(x_1, x_2, x_3) = x_i$, $1 \leq i \leq 3$. Suppose then that $p = \sigma(p_1, p_2)$ for some p_1, $p_2 \in T_\Sigma(X_3)$. By inductive assumption, (1) holds for p_1 and p_2, and by Table 1, where the sequence a, b, c is abbreviated by $\bar{a}$ and ah, bh, ch by $\bar{a}h$, (1) holds for p, too.

Now (1) implies that

$$F_A(p^{\mathcal{A}}(a, b, c)) = F_B(p^{\mathcal{B}}(ah, bh, ch)) \tag{2}$$

$p_1^{\mathcal{A}}(\bar{a})$	$p_2^{\mathcal{A}}(\bar{a})$	$p^{\mathcal{A}}(\bar{a})$	$p_1^{\mathcal{B}}(\bar{a}h)$	$p_2^{\mathcal{B}}(\bar{a}h)$	$p^{\mathcal{B}}(\bar{a}h)$
a	a	b	q	q	r
a	b	c	q	$r,\ s$	$t,\ t$
a	c	c	q	t	t
b	a	b	$r,\ s$	q	$s,\ s$
b	b	c	$r,\ s$	$r,\ s$	$t,\ t,\ t,\ t$
b	c	c	$r,\ s$	t	$t,\ t$
c	a	c	t	q	t
c	b	c	t	$r,\ s$	$t,\ t$
c	c	c	t	t	t

Table 1: The inductive step of Example 2.1.

for all $p \in T_\Sigma(X_3)$.

Next we show that

$$F_A^*(p(a,b,c)) = F_B^*(p(ah,bh,ch))$$

for all $p \in T_\Sigma(X_3)$. Indeed, if $p(x_1,x_2,x_3) = x_i$, $1 \le i \le 3$, then $F_A^*(d) = F_A(d) = F_B(dh) = F_B^*(dh)$ for all $d = a,b,c$. On the other hand, if $p = \sigma(p_1,p_2)$ for some $p_1,p_2 \in T_\Sigma(X_3)$, then, by the inductive assumption and (2), we obtain

$$\begin{aligned} F_A^*(p(a,b,c)) &= F_A^*(\sigma(p_1(a,b,c),p_2(a,b,c))) \\ &= F_A(p^{\mathcal{A}}(a,b,c))_2(F_A^*(p_1(a,b,c)),F_A^*(p_2(a,b,c))) \\ &= F_B(p^{\mathcal{B}}(ah,bh,ch))_2(F_B^*(p_1(ah,bh,ch)),F_B^*(p_2(ah,bh,ch))) \\ &= F_B^*(p(ah,bh,ch)). \end{aligned}$$

Further on, this implies that

$$\begin{aligned} R_A(d,p(a,b,c;x)) &= F_A^*(p(a,b,c;d)) \\ &= F_B^*(p(ah,bh,ch;dh)) \\ &= R_B(dh,p(ah,bh,ch;x)) \end{aligned}$$

for any $p \in \mathcal{P}_\Sigma(Q_A)$ and $d \in Q_A$, which completes the proof.

However, **A** is not strongly simulated by **B** via h since $\sigma^{\mathcal{A}}(a,a)h = bh = s$ but $\sigma^{\mathcal{B}}(ah,ah) = \sigma^{\mathcal{B}}(q,q) = r$. In the same way we can prove that no other morphism can make **B** strongly simulate **A**.

Example 3.2 Let $\mathbf{A} = (\mathcal{A},F_A)$ and $\mathbf{B} = (\mathcal{B},F_B)$ be two tree automata such that $\Sigma = \{\sigma\}$, $O = \{0,1\}$, $\mathcal{A} = (Q_A,\Sigma)$, $Q_A = \{a,b,c,d\}$, $F_A(a) = F_A(b) = F_A(d) = 0$, $F_A(c) = 1$ and $\mathcal{B} = (Q_B,\Sigma)$, $Q_B = \{p,q,r,s\}$, $F_B(p) = F_B(s) = 0$, $F_B(q) = F_B(r) = 1$. Moreover,

$\sigma^{\mathcal{A}}$	a	b	c	d
a	c	c	d	d
b	c	c	d	d
c	c	c	d	d
d	d	d	d	d

$\sigma^{\mathcal{B}}$	p	q	r	s
p	q	s	s	s
q	r	s	s	s
r	q	s	s	s
s	s	s	s	s

If we choose two mappings $h_1 : Q_A \to Q_B$ and $h_2 : Q_B \to Q_A$ such that

$$ah_1 = p,\ bh_1 = p,\ ch_1 = q,\ dh_1 = s$$
$$ph_2 = a,\ qh_2 = c,\ rh_2 = c,\ sh_2 = d$$

then **A** and **B** are equivalent via h_1 and h_2 but they are not strongly equivalent. Since $\sigma^{\mathcal{A}}(c, a)h_1 = ch_1 = q$ and $\sigma^{\mathcal{B}}(ch_1, ah_1) = \sigma^{\mathcal{B}}(q, p) = r$, **A** cannot be strongly simulated by **B** via h_1.

Notice that by slightly changing the definition of $\sigma^{\mathcal{B}}$ so that $\sigma^{\mathcal{B}}(q, p) = q$, **A** and **B** become strongly equivalent but still not isomorphic.

Our next result shows that f-simulation and simulation hold simultaneously.

Theorem 3.1 *Let* **A** *and* **B** *be two tree automata. Then* **A** *is simulated (resp. strongly simulated) by* **B** *if and only if* **A** *is f-simulated (resp. strongly f-simulated) by* **B**.

Proof: If **A** is simulated by **B**, then **A** is also f-simulated by **B**. So assume that **A** is f-simulated by **B** via $h : Q_A \to Q_B$. Then for any a, $a_i \in Q_A$ and $p(x_1, \ldots, x_n; x) \in \mathcal{P}_\Sigma(X)$,

$$F_A(p^{\mathcal{A}}(a_1, \ldots, a_n; a)) = F_B(p^{\mathcal{B}}(a_1h, \ldots, a_nh; ah)),$$

or simply

$$F_A(p^{\mathcal{A}}(a_1, \ldots, a_n)) = F_B(p^{\mathcal{B}}(a_1h, \ldots, a_nh))$$

for all $p \in T_\Sigma(X)$ and $a_1, \ldots, a_n \in Q_A$. Hence we only need to prove that

$$F_A^*(p(a_1, \ldots, a_n)) = F_B^*(p(a_1h, \ldots, a_nh))$$

for all $p \in T_\Sigma(X)$ and $a_1, \ldots, a_n \in Q_A$ since this implies that

$$R_A(a, p(a_1, \ldots, a_n; x)) = R_B(ah, p(a_1h, \ldots, a_nh; x)).$$

If $p = x_i$, then

$$F_A^*(a_i) = F_A(a_i^{\mathcal{A}}) = F_B(a_ih^{\mathcal{B}}) = F_B^*(a_ih).$$

Otherwise, if $p = \sigma(p_1, \ldots, p_m)$ for some $m \geq 1$, $p_1, \ldots, p_m \in T_\Sigma(X_n)$, $\sigma \in \Sigma_m$, then

$$\begin{aligned}
&F_A^*(p(a_1, \ldots, a_n)) \\
&= F_A(p^{\mathcal{A}}(a_1, \ldots, a_n))_m(F_A^*(p_1(a_1, \ldots, a_n)), \ldots, F_A^*(p_m(a_1, \ldots, a_n))) \\
&= F_B(p^{\mathcal{B}}(a_1h, \ldots, a_nh))_m(F_B^*(p_1(a_1h, \ldots, a_nh)), \ldots, F_B^*(p_m(a_1h, \ldots, a_nh))) \\
&= F_B^*(p(a_1h, \ldots, a_nh)),
\end{aligned}$$

which completes the proof. □

Theorem 3.2 *Two isomorphic tree automata are strongly equivalent.*

Proof: Let $\mathbf{A}$ and $\mathbf{B}$ be two isomorphic tree automata. By definition, there is a bijective morphism $h : \mathcal{A} \to \mathcal{B}$ such that for all $a \in Q_A$, $F_A(a) = F_B(ah)$. Because of Theorem 3.1, it suffices to verify that $\mathbf{A}$ is strongly f-simulated by $\mathbf{B}$. Indeed, we have

$$\begin{aligned} f_A(a, p(a_1, \ldots, a_n; x)) &= F_A(p^{\mathcal{A}}(a_1, \ldots, a_n; a)) \\ &= F_B(p^{\mathcal{A}}(a_1, \ldots, a_n; a)h) \\ &= F_B(p^{\mathcal{B}}(a_1h, \ldots, a_nh; ah)) \\ &= f_B(ah, p(a_1h, \ldots, a_nh; x)). \end{aligned}$$

□

4 Universal Tree Automata

Assume that we have a class of tree automata with fixed starting states. We want to define a tree automaton which both simulates and is simulated by each member of the class. The only restriction compared with simulation defined in the previous section is that the starting states of the members in the class are fixed whereas the starting state of the universal tree automaton varies.

Formally, let $\mathcal{C}$ be a finite class of pairs $(\mathbf{C}, c)$ where $\mathbf{C}$ is a tree automaton and $c \in Q_C$ is a starting state. A tree automaton $\mathbf{U} = (\mathcal{U}, F_U)$ is *universal* for the class $\mathcal{C}$ if the following two conditions hold.

Universality condition 1: For any pair $(\mathbf{C}, c) \in \mathcal{C}$, there is a starting state $u \in Q_U$ and a mapping $h : Q_C \to Q_U$ such that $ch = u$ and

$$R_C(c, p(c_1, \ldots, c_n; x)) = R_U(u, p(c_1h, \ldots, c_nh; x))$$

for all $c_i \in Q_C$ and $p(x_1, \ldots, x_n; x) \in \mathcal{P}_\Sigma(X)$.

Universality condition 2: For any state $u \in Q_U$, there is a pair $(\mathbf{C}, c) \in \mathcal{C}$ and a mapping $h : Q_U \to Q_C$ such that $uh = c$ and

$$R_U(u, p(u_1, \ldots, u_n; x)) = R_C(c, p(u_1h, \ldots, u_nh; x))$$

for all $u_i \in Q_U$ and $p(x_1, \ldots, x_n; x) \in \mathcal{P}_\Sigma(X)$.

Every tree automaton $\mathbf{A} = (\mathcal{A}, F_A)$ is clearly universal for a class $\mathcal{C}(\mathbf{A}) = \{(\mathbf{A}, a) \mid a \in Q_A\}$. The same idea can be used to prove that even though there are finite classes that do not have a universal automaton because of the second condition, each of them can be embedded to a larger class having this property.

Theorem 4.1 *Two tree automata* $\mathbf{A}$ *and* $\mathbf{B}$ *are equivalent if and only if they are universal for the same class.*

Proof: Assume first that **A** and **B** are equivalent (via mappings $h_1 : Q_A \to Q_B$ and $h_2 : Q_B \to Q_A$) and let $\mathcal{C}$ be some finite class for which **A** is universal. We will show that **B** is also universal for $\mathcal{C}$.

Indeed, take any pair $(\mathbf{C}, c) \in \mathcal{C}$ and $p(x_1, \ldots, x_n; x) \in \mathcal{P}_\Sigma(X)$. By Universality condition 1 there is a state $a \in Q_A$ and a mapping $g : Q_C \to Q_A$ such that $cg = a$. We obtain

$$\begin{aligned} R_C(c, p(c_1, \ldots, c_n; x)) &= R_A(a, p(c_1 g, \ldots, c_n g; x)) \\ &= R_B(ah_1, p(c_1 g h_1, \ldots, c_n g h_1; x)). \end{aligned}$$

Hence **B** satisfies Universality condition 1 via the mapping $gh_1 : Q_C \to Q_B$.

To prove that also Universality condition 2 is satisfied, take any $b \in Q_B$. By Universality condition 2 applied to the state $bh_2 \in Q_A$, we can find a pair $(\mathbf{C}, c)$ and a mapping $g' : Q_A \to Q_C$ such that $bh_2 g' = c$. Hence for any $p(x_1, \ldots, x_n; x) \in \mathcal{P}_\Sigma(X)$,

$$\begin{aligned} R_B(b, p(b_1, \ldots, b_n; x)) &= R_A(bh_2, p(b_1 h_2, \ldots, b_n h_2; x)) \\ &= R_C(bh_2 g', p(b_1 h_2 g', \ldots, b_n h_2 g'; x)). \end{aligned}$$

Assume now that **A** and **B** are universal for the same class $\mathcal{C}$. By symmetry, we only need to construct a mapping $h : Q_A \to Q_B$ such that **A** is simulated by **B** via h. Enumerate the class $\mathcal{C} = \{(\mathbf{C}_1, c_1), \ldots, (\mathbf{C}_k, c_k)\}$ and take any $a \in Q_A$. By Universality condition 2 we can choose a pair $(\mathbf{C}, c) \in \mathcal{C}$ in such a way that it has has the smallest index in the class. Let $g_1 : Q_A \to Q_C$ be the mapping such that $ag_1 = c$ and

$$R_A(a, p(a_1, \ldots, a_n; x)) = R_C(c, p(a_1 g_1, \ldots, a_n g_1; x))$$

for all $a_i \in Q_A$ and $p(x_1, \ldots, x_n; x) \in \mathcal{P}_\Sigma(X)$.

Now by Universality condition 1 applied to the pair $(\mathbf{C}, c)$ we can choose a state $b \in Q_B$ to be the minimal one among the states in **B** according to some enumeration. Then there is a mapping $g_2 : Q_C \to Q_B$ such that $cg_2 = b$ and

$$R_C(c, p(a_1 g_1, \ldots, a_n g_n; x)) = R_B(b, p(a_1 g_1 g_2, \ldots, a_n g_1 g_2; x)).$$

Hence if we choose $ah = ag_1 g_2$ for each $a \in Q_A$, **A** is simulated by **B** via h. $\square$

A tree automaton **A** which is universal for the class $\mathcal{C}$ is said to be *minimal* if it has the least number of states compared with all other tree automata universal for the same class. By Theorem 4.1, we may also define minimality by equivalence relation: **A** is minimal if and only if $|Q_A| \leq |Q_B|$ for every automaton **B** which is equivalent with **A**. Furthermore, this means that if **A** is minimal for some class $\mathcal{C}$, it is minimal for all classes for which it is universal.

To construct a minimal universal tree automaton for the natural class $\mathcal{C}(\mathbf{A})$ we will generalize Nerode congruence (see Nerode [9]) for trees.

Two states a and b from Q_A are *R_A-equivalent*, $a \equiv_{R_A} b$, if

$$R_A(a, p(a_1, \ldots, a_n; x)) = R_A(b, p(a_1, \ldots, a_n; x))$$

for all $a_i \in Q_A$, $p(x_1,\ldots,x_n;x) \in \mathcal{P}_\Sigma(X)$. Informally, a and b are R_A-equivalent if the responses are the same no matter which one we choose as the starting state. Similarly, we define f_A-equivalence by using final responses instead of total responses. It is clear that both $\equiv_{R_A}$ and $\equiv_{f_A}$ are equivalence relations.

Lemma 4.2 *Let $\mathbf{A} = (\mathcal{A}, F_A)$ be a tree automaton. For any two states $a, b \in Q_A$,*

1. *$a \equiv_{R_A} b$ if and only if $a \equiv_{f_A} b$.*
2. *if $a \equiv_{f_A} b$ then $p^{\mathcal{A}}(a_1,\ldots,a_n;a) \equiv_{f_A} p^{\mathcal{A}}(a_1,\ldots,a_n;b)$ for all $a_i \in Q_A$ and $p(x_1,\ldots,x_n;x) \in \mathcal{P}_\Sigma(X)$.*
3. *if $a \equiv_{f_A} b$ then $F_A(a) = F_A(b)$.*

Proof: 1. If $a \equiv_{R_A} b$ then also $a \equiv_{f_A} b$. On the other hand, if $a \equiv_{f_A} b$ then the proof similar to that in Theorem 3.1 shows that $a \equiv_{R_A} b$.

2. Assume the contrary; there is a polynomial $p(x_1,\ldots,x_n;x) \in \mathcal{P}_\Sigma(X)$ such that $p^{\mathcal{A}}(a_1,\ldots,a_n;a) \not\equiv_{f_A} p^{\mathcal{A}}(a_1,\ldots,a_n;b)$ and let $q(x_1,\ldots,x_n;x) \in \mathcal{P}_\Sigma(X)$ be the polynomial for which

$$f_A(p^{\mathcal{A}}(a_1,\ldots,a_n;a), q(a_1,\ldots,a_n;x)) \neq f_A(p^{\mathcal{A}}(a_1,\ldots,a_n;b), q(a_1,\ldots,a_n;x)).$$

Then for the polynomial $r(x_1,\ldots,x_n;x)$ which is obtained from q and p so that the leaf labelled by x in q is replaced by a tree p we have

$$f_A(a, r(a_1,\ldots,a_n;x)) \neq f_A(b, r(a_1,\ldots,a_n;x)),$$

a contradiction.

3. The claim follows by choosing the polynomial $p(x_1,\ldots,x_n;x) = x$. □

From now on we will simply use $\equiv$ to denote both equivalence relations $\equiv_{R_A}$ and $\equiv_{f_A}$. For any state $a \in Q_A$, let $[a]$ denote the equivalence class of a, $[a] = \{b \in Q_A \mid a \equiv b\}$. Define the quotient automaton $\mathbf{M}(\mathbf{A}) = (\mathcal{M}, F_M)$ in such a way that $Q_M = \{[a] \mid a \in Q_A\}$, and for all $[a] \in Q_M$ and $p(x_1,\ldots,x_n;x) \in \mathcal{P}_\Sigma(X)$,

$$\begin{aligned} &F_M([a]) = F_A(a), \text{ and} \\ &\sigma^{\mathcal{M}}([a_1],\ldots,[a_m]) = [\sigma^{\mathcal{A}}(a_1,\ldots,a_m)]. \end{aligned}$$

Because of Lemma 4.2, $\mathbf{M}(\mathbf{A})$ is well-defined.

We will show that $\mathbf{M}(\mathbf{A})$ is a minimal universal tree automaton for the class $\mathcal{C}(\mathbf{A})$, and moreover, it is unique up to isomorphism.

Lemma 4.3 *For any tree automaton $\mathbf{A}$,*

1. *$\mathbf{M}(\mathbf{A})$ and $\mathbf{A}$ are equivalent,*
2. *$\mathbf{M}(\mathbf{A})$ is minimal, and*
3. *if $\mathbf{B}$ and $\mathbf{A}$ are equivalent and $\mathbf{B}$ is minimal, then $\mathbf{M}(\mathbf{A})$ and $\mathbf{B}$ are isomorphic.*

Proof: 1. The automaton $\mathbf{A}$ is clearly simulated by $\mathbf{M}(\mathbf{A})$ via the canonical morphism $a \mapsto [a]$ for all $a \in Q_A$. To show the other inclusion, enumerate the set of states $Q_A = \{a_1, \ldots, a_n\}$ and denote by a^{min} the minimal representative of each $[a] \in Q_M$ with respect to this enumeration. Then $\mathbf{M}(\mathbf{A})$ is simulated by A via the mapping $h : Q_M \to Q_A$, $[a]h = a^{min}$.

2. To prove that $\mathbf{M}(\mathbf{A})$ is minimal (for the class $\mathcal{C}(\mathbf{A})$), let $\mathbf{B}$ be another tree automaton which is equivalent to $\mathbf{M}(\mathbf{A})$ via the mappings $h : Q_M \to Q_B$ and $g : Q_B \to Q_M$. We will prove that $|Q_M| \leq |Q_B|$ by showing that the mapping h is injective. Indeed, let $a, b \in Q_M$ such that $ah = bh$. Now

$$\begin{aligned} R_M(a, p(a_1, \ldots, a_n; x)) &= R_B(ah, p(a_1h, \ldots, a_nh; x)) \\ &= R_B(bh, p(a_1h, \ldots, a_nh; x)) \\ &= R_M(b, p(a_1, \ldots, a_n; x)) \end{aligned}$$

implies that $a \equiv b$, and hence, $a = b$ by the definition of $\mathbf{M}(\mathbf{A})$.

3. Let $\mathbf{B}$ be a minimal tree automaton which is equivalent to $\mathbf{A}$. Then $\mathbf{B}$ is also equivalent to $\mathbf{M}(\mathbf{A})$. Furthermore, the previous proof shows that $|Q_M| \leq |Q_B|$ and the mapping $h : Q_M \to Q_B$ is injective. But since B is also minimal, it follows from the definition that $|Q_B| \leq |Q_M|$. Hence $|Q_M| = |Q_B|$ and h has to be bijective. To complete the proof, we only need to show that h is a morphism. So take any $a_1, \ldots, a_k \in Q_M$ and $\sigma \in \Sigma_k$. Since h is a bijection and $\mathbf{M}(\mathbf{A})$ and $\mathbf{B}$ are equivalent, we have

$$R_M(a_1, p(c_1, \ldots, c_n; x)) = R_B(a_1h, p(c_1h, \ldots, c_nh; x)) \tag{3}$$

for all $c_i \in Q_M$ and $p(x_1, \ldots, x_n; x) \in \mathcal{P}_\Sigma(X)$. In particular, (3) holds for all possible trees $p'(c_1, \ldots, c_n, a_1, \ldots, a_k; x)$ which are obtained from $p(c_1, \ldots, c_n; x)$ by replacing x with $\sigma(x, a_2, \ldots, a_k)$. Hence

$$R_M(\sigma^{\mathcal{M}}(a_1, \ldots, a_k), p(c_1, \ldots, c_n; x)) = R_B(\sigma^{\mathcal{B}}(a_1h, \ldots, a_kh), p(c_1h, \ldots, c_nh; x))$$

for all $c_i \in Q_M$ and $p(x_1, \ldots, x_n; x) \in \mathcal{P}_\Sigma(X)$. But since $\mathbf{M}(\mathbf{A})$ and $\mathbf{B}$ are minimal this implies that

$$\sigma^{\mathcal{M}}(a_1, \ldots, a_k)h = \sigma^{\mathcal{B}}(a_1h, \ldots, a_kh).$$

Hence $h : Q_M \to Q_B$ is an isomorphism. $\square$

We have actually proved the following two results.

Theorem 4.4 *If a class $\mathcal{C}$ has a universal tree automaton, then it has a unique minimal universal tree automaton up to isomorphism.*

Theorem 4.5 *Two minimal tree automata $\mathbf{A}$ and $\mathbf{B}$ are equivalent if and only if they are isomorphic.*

5 Further work

Simulation, universality and minimality have been extended to tree automata with output. It seems natural to ask how these new notions relate to tree languages. Tree automata with output read trees having states as leaves and can be regarded as partly evaluated trees by ordinary frontier-to-root tree automata. Hence if we define that a tree automaton with output accepts a tree when the output symbol at the root is accepting, it seems that simulation, minimality and universality have natural interpretations in terms of tree languages. For instance, two equivalent tree automata accept the same tree languages.

Another interesting question is whether the results can be extended to root-to-frontier or incomplete tree automata, see e.g. Jurvanen [7].

Acknowledgement

We thank professor Magnus Steinby for his comments on the paper.

References

[1] C. Calude, E. Calude, and B. Khoussainov. Deterministic automata: simulation, universality and minimality. *Annals of Pure and Applied Logic*, 1105, 1997.

[2] C. Calude, E. Calude, K. Svozil, and S. Yu. Physical versus computational complementarity I. *International Journal of Theoretical Physics*, 36:1495–1523, 1997.

[3] E. Calude and M. Lipponen. Minimal deterministic incomplete automata. *Journal of Universal Computer Science*, 3(11):131–149, 1997.

[4] E. Calude and M. Lipponen. Deterministic incomplete automata: simulation, universality and complementarity. In C. S. Calude, J. Casti, and M. J. Dinneen, editors, *Unconventional Models of Computation*, pages 131–149. Springer-Verlag, Singapore, 1998.

[5] F. Gécseg and M. Steinby. *Tree automata.* Akadémiai Kiadó, Budapest, 1984.

[6] F. Gécseg and M. Steinby. Tree languages. In G. Rozenberg and A. Salomaa, editors, *Handbook of Formal Languages*, volume 3, pages 1–68. Springer, Berlin, 1997.

[7] E. Jurvanen. *On tree languages defined by deterministic root-to-frontier recognizers.* PhD thesis, University of Turku, 1995.

[8] E. F. Moore. Gedanken-experiments on sequential machines. In C. E. Shannon and J. McCarthy, editors, *Automata Studies*, pages 128–153. Princeton University Press, Princeton, 1956.

[9] A. Nerode. Linear automaton transformations. *Proceedings of the American Mathematical Society*, 9:541–544, 1958.

Turku Centre for Computer Science
Lemminkäisenkatu 14
FIN-20520 Turku
Finland

http://www.tucs.abo.fi

University of Turku
- Department of Mathematical Sciences

Åbo Akademi University
- Department of Computer Science
- Institute for Advanced Management Systems Research

Turku School of Economics and Business Administration
- Institute of Information Systems Science