Triangle: Language, Literature, Computation, n. 8, 2012
Publicacions Universitat Rovira i Virgili · ISSN: 2013-939X
https://revistes.urv.cat/index.php/triangle

Transformations Induced by Transducers

Guangwu Liu

Research Group on Mathematical Linguistics
Universitat Rovira i Vigili
Tarragona, Spain

School of Transportation
Wuhan University of Technology
Wuhan, China
E-mail: guangwu.liu@estudiants.urv.cat

Summary. This paper is a summary of my seminars given in the Research Group on Mathematical Linguistics in the year 2005. It is a short survey on automata theory, including finite state automata and tree automata. The transformations (transductions) induced by finite state automata and tree automata are given.

1 Preliminaries and Notations

In this section, some basic notations about formal language theory and automata theory are given. Other concepts can be found in the references [1, 2, 5, 7].

A *set* is a collection of objects, we use $|S|$ to denote the cardinality of S, i.e., the number of elements of S. If $|S| < \infty$, then S is called a *finite* set; otherwise, S is called an *infinite set*. An alphabet is a finite set of symbols. A *(formal) language* is a set of strings of symbols from some alphabet. An element x in a language L is called a word, written $x \in L$.

A *semigroup* consists of a set M and a binary associative operation on M, denoted by $(M, *)$. For $\forall m_1, m_2, m_3 \in M$, we have $(m_1 * m_2) * m_3 = m_1 * (m_2 * m_3)$. A semigroup which has a neutral element, denoted by 1_M, is a *monoid*. Given two subsets A, B of a monoid M, the *product* AB is defined by

$$AB = \{c \in M | \exists a \in a, \exists b \in B : c = ab\}. \tag{1}$$

Recognizable sets and rational sets are two important language families. They are of distinct nature and Kleene's Theorem precisely asserts that they coincide in finitely generated free monoids. Some properties of regular languages like closure properties can be proved for recognizable subsets, others for the rational subsets of a monoid. Here we give the formal definition of recognizable sets and rational sets.

Definition 1. *Let M be a monoid. A subset A of M is* recognizable *if there exist a finite monoid N, a morphism α from M into N and a subset P of N such that $A = \alpha^{-1}(P)$.*

The set of all recognizable subsets of M is denoted by $Rec(M)$.

Example 1. Let M be any monoid, $N = \{1\}$. Let α be the unique morphism from M onto N. Then $M, \emptyset \in Rec(M)$.

Example 2. If M is a finite monoid, then any subset of M is recognizable.

Example 3. If $M = X^*$ and X is an alphabet, then $A \in Rec(X^*)$ iff A is recognized by a finite automaton.

Definition 2. *Let M be a monoid. The family Rat(M) of* rational subsets *of M is the least family $\mathcal{R}$ of M satisfying the following conditions:*

(i) $\emptyset \in \mathcal{R}; \{m\} \in \mathcal{R}$ for all $m \in M$,
(ii) if $A, B \in \mathcal{R}$,then $A \cup B, AB \in \mathcal{R}$,
(iii) if $A \in \mathcal{R}$,then

$$A^* = \bigcup_{n \geqslant 0} A^n \in \mathcal{R}$$

Remark 1. A rational subset of M is either empty or can be expressed, starting with singletons, by a finite number of unions, products, and plus or stars (*Rational expression*).

Definition 3. *Let X and Y be alphabets. A rational (resp. recognizable) relation over X and Y is a rational (resp. recognizable) subset of the monoid $X^* \times Y^*$.*

Definition 4. *Let M and M' be monoids. A rational relation over M and M' is a rational subsets of $M \times M'$*

The "dynamic" notion of rational relation is a rational transduction. A transduction τ from X^* into Y^* is a function from X^* into the power set $\mathcal{P}(Y^*)$. For convenience, we write $\tau : X^* \to Y^*$.

Theorem 1 (Nivat's, 1968). *Let X and Y be alphabets. The following conditions are equivalent:*

(i) $A \in Rat(X^* \times Y^*)$,

(ii) There exist an alphabet Z, two morphisms $\varphi : Z^* \to X^*$, $\psi : Z^* \to Y^*$ and a regular language $K \subset Z^*$ such that

$$A = \{(\varphi h, \psi h) : h \in K\}$$

(iii) There exist an alphabet Z, two alphabetic morphisms $\alpha : Z^* \to X^*$, $\beta : Z^* \to Y^*$ and a regular language $K \subset Z^*$ such that

$$A = \{(\alpha h, \beta h) : h \in K\}$$

(iv) There exist an alphabet Z, two alphabetic morphisms $\alpha : Z^* \to X^*$,$\beta : Z^* \to Y^*$ and a local regular language $K \subset Z^*$ such that

$$A = \{(\alpha h, \beta h) : h \in K\}$$

2 Transformations Induced by FSTs

The machines realizing rational transductions are called transducers. The automaton reads input words over alphabet X, and emits output words over alphabet Y. Thus, the automaton realizes a rational transduction. The following is the mathematical definition of a transducer.

2.1 Finite transducer

Definition 5. *A finite transducer is a 6-tuple $T = (Q, \Sigma, \Delta, \delta, s, F)$, where*

1. *Q is the finite set of states,*
2. *Σ is the input alphabet,*
3. *Δ is the output alphabet,*

4. *δ is the transition-and-output function from a finite subset of $Q \times \Sigma^*$ to finite subset of $Q \times \Delta^*$,*
5. *$s \in Q$ is the starting state,*
6. *$F \subseteq Q$ is the set of final states.*

Example 4. Figure 1 is the graph representation of the transducer $T = (\{0,1,2\}, \{a,b\}, \{0,1\}, \delta, 0, \{2\})$. Where the transition-and-output function consists of the following transition rules: $\delta(0, aa) = (0, 0)$, $\delta(0, b) = (1, 101)$, $\delta(1, \lambda) = (2, 11)$, $\delta(1, b) = \{(1, \lambda), (1, 101)\}$, $\delta(2, a) = (2, 0)$.

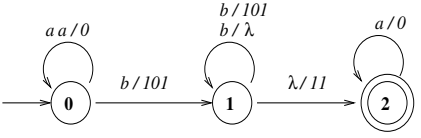


Fig. 1. Finite-state transducer

For a given word $u \in \Sigma^*$, we say that $v \in \Delta^*$ is an output of T for u if there exists a state transition sequence of T, $(q_1, v_1) \in \delta(s, u_1), (q_2, v_2) \in \delta(q_1, u_2), \ldots, (q_n, v_n) \in \delta(q_{n-1}, u_n)$, and $q_n \in F$, i.e.,

$$s \xrightarrow{u_1/v_1} q_1 \xrightarrow{u_2/v_2} \cdots\cdots \xrightarrow{u_n/v_n} q_n \in F$$

such that $u = u_1 \cdots u_n, u_1, \ldots, u_n \in \Sigma^*$, and $v = v_1 \cdots v_n, v_1, \ldots, v_n \in \Delta^*$. We write $v \in T(u)$, where $T(u)$ denotes the set of all outputs of T for the input word u.

Definition 6. *A finite transducer $T = (Q, \Sigma, \Delta, \delta, s, F)$ is called a generalized sequential machine(GSM) if δ is a function from $Q \times \Sigma$ to finite subsets of $Q \times \Delta^*$, i.e., T reads exactly one symbol at each transition.*

2.2 Finite transduction

For each finite transducer $T = (Q, \Sigma, \Delta, \delta, s, F)$, the transduction induced by T is $T : \Sigma^* \longrightarrow 2^{\Delta^*}$. For a language $L \subseteq \Sigma^*$,

$$T(L) = \bigcup_{w \in L} T(w)$$

Example 5. Consider the transducer in Example 4, the following relations are in the transductions induced by transducer $T = (\{0,1,2\}, \{a,b\}, \{0,1\}, \delta, 0, \{2\})$:

1. $T(aabb) = \{010111, 010110111\}$,
2. $T(bbba) = \{101110, 101101110, 101101101110\}$,
3. $T(\lambda) = \emptyset$, $T(aaab) = \emptyset$,
4. $T(\{b, ba\}) = \{10111, 101110\}$.

2.3 Rational relation (transduction)

Transductions induced by finite transducers can be viewed as a relation $R_T \subseteq \Sigma^* \times \Delta^*$ defined by

$$R_T = \{(u, v) | v \in T(u)\}$$

Definition 7. *A transduction $\tau : X^* \to Y^*$ is rational iff τ is realized by a transducer.*

Corollary 1. *Any rational transduction $\tau : X^* \to Y^*$ can be realized by a transducer $T = (Q, \Sigma, \Delta, \delta, s, F)$ such that*

$$\delta \subset Q \times (X \cup \{\lambda\}) \times (Y \cup \{\lambda\}) \times Q \tag{2}$$

and further F consists of a single state $q_+ \neq s$, and $(p, u, v, q) \in \delta$ implies $p \neq q_+$ and $q \neq s$.

Theorem 2 (Nivat's, 1968). *Let Σ and Δ be finite alphabets. $R \subseteq \Sigma^* \times \Delta^*$ is a rational relation iff there are a finite alphabet Γ, a regular language $L \subseteq \Gamma^*$ and a morphisms $g : \Gamma^* \to \Sigma^*$ and $h : \Gamma^* \to \Delta^*$ such that*

$$R = \{(g(w), h(w)) | w \in L\}$$

Definition 8. *Two finite transducers are said to be equivalent if they define exactly the same finite transduction.*

Remark 2. Equivalence problem of finite transducers is undecidable, equivalence problem for *single-valued* finite transducers is decidable.

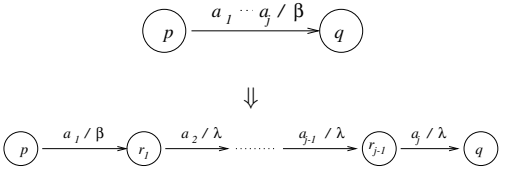
Inverse transduction: Let $T : \Sigma^* \to 2^{\Delta^*}$ be a finite transduction. Then the inverse of T, i.e. $T^{-1} : \Delta^* \to 2^{\Sigma^*}$, is also a finite transduction.

Definition 9. *A finite transducer $T = (Q, \Sigma, \Delta, \delta, s, F)$ is in the standard form if $\delta : Q \times (\Sigma \cup \{\lambda\}) \to 2^{Q \times (\Delta \cup \{\lambda\})}$. Intuitively, the standard restricts the input and output of each transition to be only a single letter or λ.*

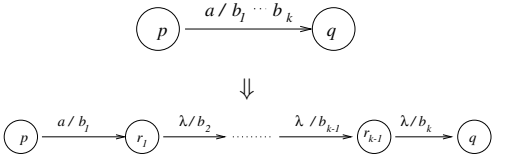
Theorem 3. *Each finite transducer can be transformed into an equivalent finite transducer in the standard form.*

The transformation of an arbitrary finite transducer to an equivalent one in the standard form consists of two steps: First, each transition that reads more than one letter is transformed into several transitions reading exactly one letter. Secondly, each transition that has a string of more than one letter as output is transformed into several transitions such that each of them has exactly one letter as output.

The two transformation steps can be represented as follows:



$p, q \in Q, a_1, \ldots, a_j \in \Sigma, j \geqslant 2, \beta \in \Delta^*$, $r_1, \ldots, r_{j-1}$ new states.



$p, q \in Q, a \in \Sigma \cup \{\lambda\}, b_1, \ldots, b_k \in \Delta, k \geqslant 2$, $r_1, \ldots, r_{k-1}$ new states.

3 Tree Automata and Tree Transducers

Tree automata were first introduced by J.W. Thatcher and J.B. Wright, and independently by J. Doner around 1965.

The trees can be regarded as ΣX-terms over ranked alphabet Σ and frontier alphabet X. Forest is the set of ΣX-terms. Here just some basic definitions and theorems will be given. Some details can be checked in the book *Tree Automa* and some related references.

The ranked alphabet will be denoted by Σ, Δ, Γ, etc., in which every element has a rank (arity) $m \geq 0$. And the frontier alphabet (or variables) will be denoted by X, Y, Z, etc.

3.1 Tree automata

Definition 10. *A $\Sigma-$algebra $\mathcal{A}$ is a pair consisting of a nonempty set A (of elements of $\mathcal{A}$) and a mapping that assigns to every operator $\sigma \in \Sigma$ an $m-$ary operation*

$$\sigma^{\mathcal{A}} : A^m \to A.$$

where m is the arity of σ. The operation $\sigma^{\mathcal{A}}$ is called the realization of σ in $\mathcal{A}$.

Definition 11. *A homomorphism from a Σ-algebra $\mathcal{A}$ to a Σ-algebra $\mathcal{B}$ is a mapping $\varphi : A \to B$ such that for all $m \geq 0, \sigma \in \Sigma_m$ and $a_1, \ldots, a_m \in A$,*

$$\sigma^{\mathcal{A}}(a_1, \ldots, a_m)\varphi = \sigma^{\mathcal{B}}(a_1\varphi, \ldots, a_m\varphi).$$

and then write $\varphi : \mathcal{A} \to \mathcal{B}$.

Definition 12. *The set $T_\Sigma(X)$ of ΣX-terms in X, or ΣX-terms for short, is defined as follows:*

1. *$X \subseteq T_\Sigma(X)$;*
2. *$\sigma(t_1, \ldots, t_m) \in T_\Sigma(X)$ whenever $m \geq 0, \sigma \in \Sigma_m$ and $t_1, \ldots, t_m \in T_\Sigma(X)$;*
3. *every ΣX-term can be obtained by applying the rules (1) and (2) a finite number of times.*

Example 6. Let $\Sigma = \Sigma_0 \cup \Sigma_1 \cup \Sigma_2$ be a ranked alphabet, where $\Sigma_0 = \{\gamma\}$, $\Sigma_1 = \{\omega\}$ and $\Sigma_2 = \{\sigma\}$. The frontier alphabet $X = \{x, y\}$. Then $t = \omega(\sigma(y, \sigma(\gamma, x)))$ is a ΣX-tree.

Tree automata (or tree recognizers) define forests. There are four basic types of these recognizers. A *frontier-to-root tree recognizer* (or F-recognizer) reads its input trees from the frontier towards the root. A *root-to-frontier tree recognizer* (or R-recognizer) reads the trees starting at the root towards the frontier.

Definition 13. *A frontier-to-root ΣX-recognizer* **A** *consists of*

1. *a finite Σ-algebra $\mathcal{A} = (A, \Sigma)$;*

2. *an initial assignment* $\alpha : X \to A$;
3. *a set* $A' \subseteq A$ *of final states.*

We can write $\mathbf{A} = (\mathcal{A}, \alpha, A')$ *or* $\mathbf{A} = (A, \Sigma, X, \alpha, A')$. *The forest recognized by* $\mathbf{A}$ *is the* ΣX*-forest*

$$T(\mathbf{A}) = \{t \in T_\Sigma(X) | t^{\mathcal{A}}(\alpha) \in A'\}.$$

Definition 14. *A nondeterministic frontier-to-root* ΣX*-recognizer* $\mathbf{A}$ *consists of*

1. *a finite ND* Σ*-algebra* $\mathcal{A} = (A, \Sigma)$;
2. *an initial assignment* $\alpha : X \to \mathcal{P}A$;
3. *a set* $A' \subseteq A$ *of final states.*

we write $\mathbf{A} = (\mathcal{A}, \alpha, A')$ *or* $\mathbf{A} = (A, \Sigma, X, \alpha, A')$. *The forest recognized by* $\mathbf{A}$ *is the* ΣX*-forest*

$$T(\mathbf{A}) = \{t \in T_\Sigma(X) | t\widehat{\alpha} \cap A' \neq \emptyset\}$$

Definition 15. *A nondeterministic root-to-frontier (NDR)* ΣX*-recognizer* $\mathbf{A}$ *consists of*

1. *a finite NDR* Σ*-algebra* $\mathcal{A} = (A, \Sigma)$;
2. *a set* $A' \subseteq A$ *of initial states;*
3. *a final assignment* $\alpha : X \to \mathcal{P}A$.

we write $\mathbf{A} = (\mathcal{A}, A', \alpha)$ *or* $\mathbf{A} = (A, \Sigma, X, A', \alpha)$. *The forest recognized by* $\mathbf{A}$ *is the* ΣX*-forest*

$$T(A) = \{t \in T_\Sigma(X) | t\widehat{\alpha} \cap A' \neq \emptyset\}.$$

A *deterministic root-to-frontier* ΣX-recognizer (DR) is an NDR ΣX-recognizer $\mathbf{A} = (\mathcal{A}, A', \alpha)$ such that A' and all of the sets $\sigma^{\mathcal{A}}(a)$ $(\sigma \in \Sigma_m, m \geq 1, a \in A)$ and $\sigma^{\mathcal{A}}$ with $\sigma \in \Sigma_0$ contain exactly one element.

Deterministic and nondeterministic frontier-to-root ΣX-recognizers and nondeterministic root-to-frontier ΣX-recognizers are of the same power, the forests by these three types of recognizers are exactly the recognizable forests. But the deterministic root-to-frontier recognizers are weaker and they define a proper subfamily of *Rec*.

For a recognizable forest, some operations can be defined on it. The most common operations are:

1. *forest product*: $T(x \leftarrow T_x | x \in X) = \cup(t(x \leftarrow T_x | x \in X) | t \in T)$. Where T is a ΣX-forest and $(T_x | x \in X)$ is an X-indexed family of ΣX-forests;

2. *z-product*: $S \cdot_z T = T(x \leftarrow T_x | x \in X)$, where S and T are ΣX-forests, $T_z = S,\ T_x = x$ for all $x \in X,\ x \neq z$;
3. *x-quotient*: $S^{-x}T = \{p \in T_\Sigma(X) | S \cdot_x \{p\} \cap T \neq \emptyset\}$, where S and T are ΣX-forests;
4. *σ-product*: $\sigma(T_1, \ldots, T_m) = \{\sigma(t_1, \ldots, t_m) | t_1 \in T_1, \ldots, t_m \in T_m\}$, where $\sigma \in \Sigma$ is an m-ary operator and $T_1, \ldots, T_m$ are m ΣX-forests;

3.2 Tree transducers

Tree transformation is defined as a binary relation $\tau \subseteq T_\Sigma(X) \times T_\Omega(Y)$, in which $T_\Sigma(X)$ and $T_\Omega(Y)$ are the set of trees as defined in the previous section. An inclusion $(p, q) \in \tau$ means that τ transforms p into q. Denote a countably infinite set of auxiliary variables by $\Xi = \{\xi_1, \xi_2, \ldots\}$. Its role is to indicate an occurrence of a subtree in a tree.

There are two main kinds of tree transducers, named *frontier-to-root tree transducers* which process a tree from the leaves to the root and *root-to-frontier tree transducers* which work in the opposite direction.

Definition 16. *A frontier-to-root tree transducer (F-transducer) is a system $\mathcal{A} = (\Sigma, X, A, \Omega, Y, P, A')$, where Σ, Ω are ranked alphabet, X, Y are the frontier alphabet, A is a ranked alphabet consisting of unary operators (the state set of $\mathcal{A}$), $A' \subseteq A$ is the set of final states and P is a finite set of productions of the following two types:*

1. $x \to a(q)$ $(x \in X, a \in A, q \in T_\Omega(Y))$;
2. $\sigma(a_1(\xi_1), \ldots, a_m(\xi_m)) \to a(q(\xi_1, \ldots, \xi_m))$ $(\sigma \in \Sigma_m, m \geq 0, a_1, \ldots, a_m, a \in A, q(\xi_1, \ldots, \xi_m) \in T_\Omega(Y \cup \Xi_m))$.

Given an F-transducer $\mathcal{A} = (\Sigma, X, A, \Omega, Y, P, A')$, the transformation induced by $\mathcal{A}$ is the relation

$$\tau_\mathcal{A} = \{(p, q) | p \in T_\Sigma(X), q \in T_\Omega(Y), aq \in p\tau_\mathcal{A}^* \text{ for some } a \in A'\} \qquad (3)$$

For every $p \in T_\Sigma[X \cup A\Xi], p\tau_\mathcal{A}^*$ is the subset of $AT_\Omega(Y \cup \Xi)$ given as follows:

1. if $p = a\xi \quad (a \in A, \xi \in \Xi)$, then $a\xi \in p\tau_\mathcal{A}^*$;
2. if $p \in X \cup \Sigma_0$, then $aq \in p\tau_\mathcal{A}^*$ for all $(p, aq) \in P$,
3. if $p = \sigma(p_1, \ldots, p_m) \quad (\sigma \in \Sigma_m, m > 0)$ then $aq(q_1, \ldots, q_m) \in p\tau_\mathcal{A}^*$ for all $(\sigma(a_1, \ldots, a_m), aq) \in P$ and $a_i q_i \in p_i \tau_\mathcal{A}^* (a, a_i \in A, i = i, \ldots, m)$;
4. nothing is in any $p\tau_\mathcal{A}^*$ unless this follows from (1)-(3).

Example 7. Let $\mathcal{A} = (\Sigma, \{x\}, \{a_0, a_1\}, \Omega, \{y\}, P, \{a_0\})$, where $\Sigma = \Sigma_2 = \{\sigma\}$,$\Omega = \Omega_1 = \{\omega\}$ and P consists of the productions $x \to a_1 y$ and $\sigma(a_1, a_1) \to a_0\omega(\xi_1)$.

Consider the tree $\sigma(x, x)$. One of the possible derivations

$$\sigma(x, x) \Rightarrow \sigma(a_1 y, x) \Rightarrow \sigma(a_1 y, a_1 y) \Rightarrow a_0\omega(y)$$

is illustrated as below.

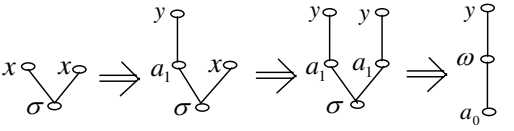


Fig. 2. A transformation by an F-transducer

Definition 17. *A root-to-frontier tree transducer (R-transducer) is a system $\mathcal{A} = (\Sigma, X, A, \Omega, Y, P, A')$, where Σ, X, A, Ω, Y and A' are specified the same way as F-transducer, A' is the set of initial states, P is a finite set of productions of the following two types:*

1. $ax \to q \quad (a \in A, x \in X, q \in T_\Omega(y))$;
2. $a\sigma(\xi_1, \ldots, \xi_m) \to q \quad (a \in A, \sigma \in \Sigma_m, m \geq 0, q \in T_\Omega[Y \cup A\Xi_m])$.

Given an R-transducer $\mathcal{A} = (\Sigma, X, A, \Omega, Y, P, A')$. Then transformation induced by $\mathcal{A}$ is the relation

$$\tau_\mathcal{A} = \{(p, q) | p \in T_\Sigma(X), q \in T_\Omega(Y), q \in p\tau_{\mathcal{A},a} \quad \textit{for some } a \in A'\} \qquad (4)$$

For any $a \in A$ and $p \in T_\Sigma(X)$, $p\tau_{\mathcal{A},a}$ is defined as follows:

1. if $p \in \Sigma_0 \cup X$ and $(ap, q) \in P$ then $q \in p\tau_{\mathcal{A},a}$;
2. if $p = \sigma(p_1, \ldots, p_m)$ $(\sigma \in \Sigma_m, m > 0)$, then for any $(a\sigma, q(\mathbf{a_1}\xi_1^{n_1}, \ldots, \mathbf{a_m}\xi_m^{n_m}))$ $\in P$ and $q_{i_j} \in p_i\tau_{\mathcal{A},a_{i_j}}$ $(1 \leq i \leq m, 1 \leq j \leq n_i), q(\mathbf{q_1}, \ldots, \mathbf{q_m}) \in p\tau_{\mathcal{A},a}$ *where* $\mathbf{q_i} = (q_{i_1}, \ldots, q_{i_{n_i}})$ $(i = 1, \ldots, m)$;
3. nothing is in any $p\tau_{\mathcal{A},a}$ unless this follows from (1)-(2).

Example 8. Let $\mathcal{A} = (\Sigma, \{x\}, \{a_0, a_1, a_2\}, \Omega, \{y_1, y_2\}, P, \{a_0\})$ be an R-transducer, where $\Sigma = \Sigma_1 = \{\sigma\}$, $\Omega = \Omega_1 \cup \Omega_2$, $\Omega_1 = \{\omega_1\}$, $\Omega_2 = \{\omega_2\}$ and P consists of the productions $a_0\sigma \to \omega_2(a_1\xi_1, a_2\xi_1)$, $a_1\sigma \to \omega_1(a_1\xi_1)$, $a_2\sigma \to$

$\omega_1(a_2\xi_1)$, $a_1x \to y_1$, $a_2x \to y_2$. Consider the trees $p = \sigma(\sigma(\sigma(x)))$ and $q = \omega_2(\omega_1(\omega_1(y_1)), \omega_1(\omega_1(y_2)))$. Then a derivation of q from a_0p is shown in Figure 3.

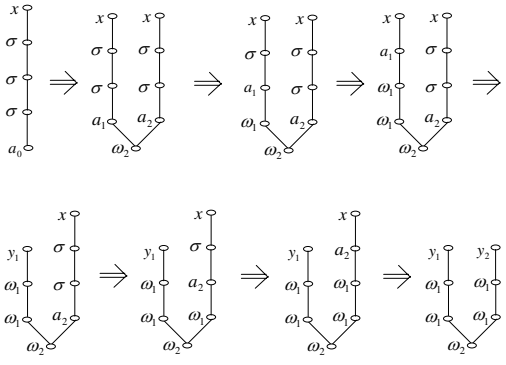


Fig. 3. A transformation by an R-transducer

3.3 Some classes of tree transformations

Let $\mathcal{A} = (\Sigma, X, A, \Omega, Y, P, A')$ be an F-transducer. Then:

1. A production of $\mathcal{A}$ is *linear* if each auxiliary variable occurs at most once in it. Moreover, A is a *linear* F-transducer (LF-transducer) if all of its productions are linear.
2. $\mathcal{A}$ is a *totally defined* F-transducer (TF-transducer) if
   (i) for each $x \in X$ there is a production in P with left-hand side x and
   (ii) for all $m > 0, \sigma \in \Sigma_m$ and $a_1, \ldots, a_m \in A$ there is a production in P with left-hand side $\sigma(a_1, \ldots, a_m)$.
3. $\mathcal{A}$ is a *nondeleting* F-transducer (NF-transducer) if for every production $\sigma\ (a_1, \ldots, a_m) \to aq,\ (\sigma \in \Sigma_m, m \geq 0)$ from P each $\xi_i \in \Xi_m$ occurs at least once in q.
4. $\mathcal{A}$ is a *deterministic* F-transducer (DF-transducer) if there are no two distinct productions in P with the same left-hand side.
5. $\mathcal{A}$ is an *F-relabeling* if each of its productions is of the form
   (i) $x \to ay\ (x \in X, a \in A, y \in Y)$ or

(ii) $\sigma(a_1, \ldots, a_m) \rightarrow a\omega(\xi_1, \ldots, \xi_m)$, where $\sigma \in \Sigma_m, a_1, \ldots, a_m, a \in A, \omega \in \Omega_m$.

Example 9. Let $\mathcal{A} = (\Sigma, \{x\}, \{a_0, a_1\}, \Omega, \{y\}, P, \{a_1\})$ be the F-transducer with $\Sigma = \Sigma_2 = \{\sigma\}$ and $\Omega = \Omega_2 = \{\omega\}$, where P consists of the productions $x \rightarrow a_0 y$, $\sigma(a_0, a_0) \rightarrow a_1\omega(\xi_1, \xi_2)$, $\sigma(a_0, a_1) \rightarrow a_0\omega(\xi_1, \xi_2)$, $\sigma(a_1, a_0) \rightarrow a_1\omega(\xi_1, \xi_2)$, $\sigma(a_1, a_1) \rightarrow a_1\omega(\xi_1, \xi_2)$.

Then $\mathcal{A}$ is a linear, totally defined, nondeleting, and deterministic F-transducer. $\mathcal{A}$ is also an F-relabeling.

References

1. Aho, A., J. Hopcroft and J. Ullman (1974). *The design and analysis of computer algorithms*. Reading: Addison-Wesley.
2. Berstel, J. (1979). *Transductions and context-free languages*. Stuttgart: Teubner.
3. Gécseg, F. and M. Steinby (1984). *it Tree automata*. Budapest: Akadémiai Kiadó.
4. Harrison, M. (1978). *Introduction to formal language theory*. Philippines: Addison-Wesley.
5. Hopcroft, J. and J. Ullman (1979). *Introduction to automata theory, languages, and computation*. Reading: Addison-Wesley.
6. Linz, P. (2001). *An introduction to formal languages and automata*. Massachusetts: Jones and Bartlett Publishers.
7. Rozenberg, G. and A. Salomaa (eds.) (1997). *Handbook of formal languages*. Berlin: Springer.
8. Yu, S. (2001). *Finite Automata*. Technical Report GRLMC. Tarragona.