TREE-TRANSDUCERS AND SYNTAX-CONNECTED TRANSDUCTIONS

Peter Paul Schreiber
Technische Universität Berlin, Informatik PC2
Ernst-Reuter-Platz 8, Berlin 10, West Germany

Abstract

We investigate Finite Tree-Transducers operating top-down, bottom-up or both ways simultaneously. A comparative study of their transductional power is given. Syntax-Connected Transductions extending Syntax-Directed Transductions are investigated. Various types of transductions of local forests defined by Syntax-Connected Transduction Schemes can be performed by Finite Tree-Transducers.

Introduction

Operational automata like tree-transducers are extensions of classical automata. In addition to local processing like symbol-changing and state-switching, they can manipulate (permute, copy or erase) input-structures and output-structures. Finite state and push-down transducers have, so far, been very useful tools for designing and structuring the first phases of a compiler (such as the scanner and the parser). The more complicated phases consisting of semantic analysis, code generation and optimization, however, could not be supplied with such useful tools from automata theory. This is due to the fact that the objects to be dealt with in these phases are trees which have to be manipulated. As long as language translation had been understood as string processing and not as a tree-manipulating process, little effort was made to investigate machines which perform tree transductions. From the point of view of generalized automata theory, trees were used as inputs and (in a further generalization step) as outputs. Comparing these tree-transducers with syntax-directed transduction schemes performing transformations of the derivation trees of an underlying CF-Grammar, one can see that tree-transducers are more powerful than the syntax-directed transduction schemes. Many tree-transforming phases of a compiler cannot be modelled by a syntax-directed transduction scheme, but by a tree-transducer.

1. Trees represented as terms

To represent trees labelled by elements of a set Σ we use terms over Σ. The set T_Σ of terms over Σ is the smallest subset of $(\Sigma \cup \{ (,) \})^*$ satisfying:

(0) $\Sigma \subset T_\Sigma$

(1) If $t_1, \ldots, t_k \in T_\Sigma$ and $a \in \Sigma$ then $a(t_1 \ldots t_k) \in T_\Sigma$

Let M be a set, $M \cap \Sigma = \emptyset$. The set $T_\Sigma[M]$ of terms over Σ indexed by M is the smallest subset of $\Sigma \cup M \cup \{ (,) \})^*$ such that

(0) $\Sigma \cup M \subset T_\Sigma$

(1) If $t_1, \ldots, t_k \in T_\Sigma[M]$ for $k > 0$ and $a \in \Sigma$ then $a(t_1 \ldots t_k) \in T_\Sigma[M]$

A subword t' of $t \in T_\Sigma[M]$ which is a term is called subterm of t.
Notation: $t' \leq t$. Two subterms t' and t" of t are independent iff $t' \not\leq t''$ and $t'' \not\leq t'$.

Let $t' \leq t$ and $r \in T_\Sigma[M]$, then $t(t' \longleftarrow r)$ is the term obtained by replacing t' by r.
Let $s_1, \ldots, s_k$ be pairwise independent subterms of t and $\pi : [k] \longrightarrow [k]$ ($[k] = \{1, \ldots k\}$) any permutation and $r_i \in T_\Sigma[M]$ $(1 \leq i \leq k)$, then

$$t((s_1 \leftarrow r_1) \ldots (s_k \leftarrow r_k)) = t((s_{\pi(1)} \leftarrow r_{\pi(1)}) \ldots (s_{\pi(k)} \leftarrow r_{\pi(k)}))$$

Let $X = \{x_i \mid i \in \mathbb{N}\}$ be a set of parameters and $X_k = \{x_1, \ldots, x_k\}$.

The operation of simultaneous substitutions is defined as:

$$t[t_1, \ldots, t_k] := t((x_1 \leftarrow t_1) \ldots (x_k \leftarrow t_k))$$

The frontier fr(t) of $t \in T_\Sigma[M]$ is the word obtained by concatenating the labels of the leaves from left to right.

The depth $\|t\|$ of $t \in T_\Sigma[M]$ is defined as:

$$\|t\| = \begin{cases} 1 \text{ for } t = a \in \Sigma \\ \max\limits_{i \in [k]} \|t_i\| + 1 \quad \text{for } t = a(t_1 \ldots t_k) \in T_\Sigma[M] \end{cases}$$

2. Finite Tree-Transducers

A Finite Tree-Transducer (FT) $P = (Q, \Sigma, \Delta, R, I)$ consists of a finite set Q of states, an inputalphabet Σ, an outputalphabet Δ, a finite set R of rules and a subset I of Q of distinguished states.

A Top-Down-rule (T-rule) is a rule of the form:

$<q,a>(x_1 \dots x_k) \longrightarrow t$ with $<q,a> \varepsilon Q x \Sigma$ and $t \varepsilon T_\Delta[QxX_k]$

or $<q,a> \longrightarrow t$ with $t \varepsilon T_\Delta$.

T-rules of that type with $\|t\| = n$ are called T(1,n)-rules.

A T-rule $<q,a>(u_1 \dots u_k) \longrightarrow t$ with $a(u_1 \dots u_k) \varepsilon T_\Sigma[X]$, $\|t\| = n$
and $\|a(u_1 \dots u_k)\| = m$ is called T(m,n)-rule.

A Top-Down-Finite-Tree-Transducer (TFT) is a FT with T-rules.
A move of a TFT is defined as a relation $\underset{T}{\vdash}$ on $T_{\Sigma \cup \Delta \cup (Qx\Sigma)}$.

Let r, $s \varepsilon T_{\Sigma \cup \Delta \cup (Qx\Sigma)}$ and R a set of T(1,1)-rules then $r \underset{T}{\vdash} s$ iff

$\exists r' \varepsilon T_{\Sigma \cup (Qx\Sigma)}$ $r' = <q,a>(t_1 \dots t_k) \leq r$

and $\exists (<q,a>(x_1 \dots x_k) \longrightarrow t) \varepsilon R$

such that $s := r(r' \longleftarrow t[t_1, \dots, t_k])$

$\vdash^*$ denotes the reflexive and transitive closure of $\vdash$.

$T(P) = \{<r,s> \varepsilon T_\Sigma x T_\Delta \mid <q_0,r> \vdash^* s \wedge q_0 \varepsilon I\}$ is called Tree-Transduction from T_Σ to T_Δ performed by a TFT P.

A Bottom-Up-rule (B-rule) is a rule of the form

$a \rightarrow <t,p>$ with $a \varepsilon \Sigma$, $t \varepsilon T_\Delta$ and $p \varepsilon Q$

or $a(<x_1,p_1> \dots <x_k,p_k>) \rightarrow <t,p>$ with $a \varepsilon \Sigma$, $p, p_1, \dots, p_k \varepsilon Q$ and $t \varepsilon T_\Delta[X_k]$.

B-rules of that type with $\|t\| = n$ are called B(1,n)-rules.

A B-rule $a(u_1 \dots u_k) \rightarrow <t,p>$ with $u_i \varepsilon T_\Sigma[X_k x Q]$, $\|t\| = n$

and $\|a(u_1 \dots u_k)\| = m$ is called B(m,n)-rule.

A Bottom-Up-Finite-Tree-Transducer is a FT with B-rules.
A move of a BFT with B(1,1)-rules is defined as a relation $\underset{B}{\vdash}$ on $T_{\Sigma \cup \Delta \cup (\Delta x Q)}$.

Let r, $s \varepsilon T_{\Sigma \cup \Delta \cup (\Delta x Q)}$ then $r \underset{B}{\vdash} s$ iff

$\exists r' = a(<t_1,p_1> \dots <t_k,p_k>) \leq r$

and $\exists (a(<x_1,p_1> \dots <x_k,p_k>) \rightarrow <t,p>) \varepsilon R$

such that $s := r(r' \leftarrow <t[t_1, \dots, t_k],p>)$

$T(P) = \{\langle r,s\rangle \varepsilon T_{\Sigma} x T_{\Delta} | r \vdash^{*} \langle s,p_0\rangle \wedge p_0 \varepsilon I\}$ is the <u>Tree-Transduction</u> performed by a BFT P.

A rule is called <u>rank-preserving</u> if each parameter x_i of its left side occurs on its right side.
A rule is called <u>linear</u> if each parameter x_i of its left side occurs not more than once on its right side.
Rank-preserving rules can copy and do not erase subtrees while linear rules can erase and do not copy subtrees.

$$\text{A FT with} \begin{Bmatrix} \text{rank-preserving} \\ \text{linear} \end{Bmatrix} \text{rules is called} \begin{Bmatrix} \text{RFT} \\ \text{LFT} \end{Bmatrix}$$

and LRFT if its rules are linear and rank-preserving. A FT with $|Q| = 1$ is a pure FT

$_{\Sigma}FT_{\Delta} = \{T(P) \epsilon T_{\Sigma} x T_{\Delta} | P \text{ is a FT}\}$ is called the class of F-Tree-Transductions and we write FT for a fixed pair (Σ,Δ).

From now on we only consider Tree-Transducers with T(1,1)-rules or B(1,1)-rules.

Generalized Finite-Tree-Transducers are composed out of a

$$\text{TFT } P_T = (Q_T,\Sigma,\Delta,R_T,I_T) \quad \text{and a} \quad \text{BFT } P_B = (Q_B,\Sigma,\Delta,R_B,I_B).$$

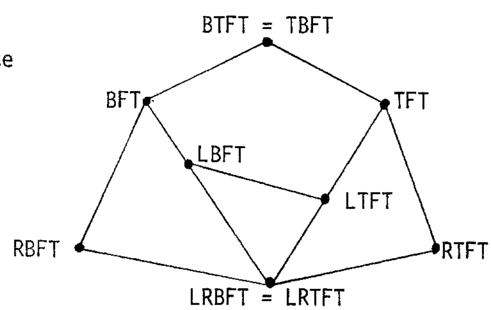
A move of a TBFT $P = (Q_T,Q_B,\Sigma,\Delta,R_T,R_B,I_T,I_B)$ is T-move followed by a B-move and a move of a BTFT $P = (Q_B,Q_T,\Sigma,\Delta,R_B,R_T,I_B,I_T)$ is a B-move followed by a T-move.

Let P be a TBFT, then

$$T(P) = \{\langle r,t\rangle \varepsilon T_{\Sigma} x T_{\Delta} | \langle q_0,r\rangle \vdash_{T} s \vdash^{*} \langle t,p_0\rangle,\ q_0 \varepsilon I_T \wedge p_0 \varepsilon I_B\}$$

and $T(P) = \{\langle r,t\rangle \varepsilon T_{\Sigma} x T_{\Delta} | \langle q_0,r\rangle \vdash_{B} s \vdash^{*} \langle t,p_0\rangle,\ q_0 \varepsilon I_T \wedge p_0 \varepsilon I_B\}$ for a BTFT P.

<u>Theorem 1</u>: For the classes of Finite Tree-Transductions the following lattice exists: (including results by ENGELFRIET, ROUNDS and THATCHER)



3. Syntax-Connected-Transduction-Schemes

Given a CF-Grammar $G = (\Sigma, \Sigma_0, P, S,)$ consisting of a finite alphabet Σ, subset $\Sigma_0 \subset \Sigma$ of terminal symbols, a set $P \subset (\Sigma \setminus \Sigma_0) \times \Sigma^+$ productions and a start symbol $S \varepsilon \Sigma \setminus \Sigma_0$. The elements of $\Sigma \setminus \Sigma_0$ are called syntactic variables.
The set $D_S(G)$ of derivation trees of G with root S is defined as:

(0) $S \varepsilon D_S(G)$

(1) If $r \varepsilon D_S(G)$, $fr(r) = w_1 A w_2$ and $(A \rightarrow w) \varepsilon P$ with $w_1, w_2 \varepsilon \Sigma^*$, $A \varepsilon \Sigma \setminus \Sigma_0$ then $r' = r(A \leftarrow A(w))$ is in $D_S(G)$

The set $D(G) := \{t \varepsilon D_S(G) \mid fr(t) \varepsilon \Sigma_0{}^+\}$ is called local forest of G.

A Syntax-Connected-Transduction-Scheme (SCTS) $G = (G_E, G_A, \kappa)$ consists of a CF-input grammar $G_E = (\Sigma, \Sigma_0, P_E, S)$, a CF-output grammar $G_A = (\Delta, \Delta_0, P_A, S)$ with $\Delta \setminus \Delta_0 \subset \Sigma \setminus \Sigma_0$ and transduction rules $A \rightarrow w, v[i_1, \ldots, i_m]$, where $A \rightarrow w$ is an input production from P_E, $A \rightarrow v$ is an output production from P_A such that $[i_1, \ldots, i_m] \varepsilon \kappa \subset \mathbb{N}^m$ connects positions of the common syntactic variables i.e. $A \rightarrow w, v[i_1, \ldots, i_m]$ has generally the following form:

$$A \rightarrow g_0 A_1 g_1 \ldots g_{k-1} A_k g_k,\ h_0 A_{i_1} h_1 \ldots h_{m-1} A_{i_m} h_m [i_1, \ldots, i_m]$$

where $i_j \varepsilon [k]$ for $1 \leq j \leq m$, $A_i \varepsilon \Sigma \setminus \Sigma_0$, $g_i \varepsilon \Sigma_0{}^*$ $(0 \leq i \leq k)$ and $h_i \varepsilon \Delta_0{}^*$ $(1 \leq i \leq m)$.

The set $T_S(G)$ of pairs of transduction trees is defined as:

(0) $<S,S> \varepsilon T_S(G)$

(1) If $<r,s> \varepsilon T_\Sigma \times T_\Delta$ such that $fr(r) = w_0 B_1 w_1 \ldots w_{m-1} B_m w_m$
$fr(s) = v_0 B_{i_1} v_1 \ldots v_{n-1} B_{i_n} v_n$ and $[i_1, \ldots, i_n]$ with $i_j \varepsilon [m]$,
$w_i \varepsilon \Sigma_0{}^*$ $(0 \leq i \leq m)$, $v_i \varepsilon \Delta_0{}^*$ $(0 \leq i \leq n)$ then $<r,s> \varepsilon T_S(G)$.

A relation $\rightarrow$ is defined on $T_S(G)$ as:

$$<r,s> \rightarrow <r',s'>$$

iff there exists a transduction rule $B_k \rightarrow w, v[i_1, \ldots, i_l]$ such that:

1. $r' = r(B_k \leftarrow B_k(w))$
2. s' derives from s by replacing all B_{i_j} with $i_j = k$ by $B_k(v)$.

or if no i_j with $i_j = k$ exists $r' = r(B_k \leftarrow B_k(w))$ for $(B_k \rightarrow w) \varepsilon P_E$ and $s' = s$.

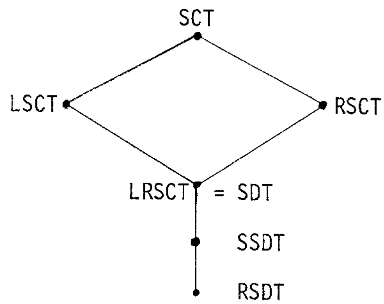
$T(G) := \{<r,s> \varepsilon T_S(G) \mid fr(r) \varepsilon \Sigma_0^+ \wedge fr(s) \varepsilon \Delta_0^+\}$ is called <u>Syntax-Connected-Transduction</u> defined by G.

A transduction rule is called <u>rank-preserving</u> if each $i \varepsilon [k]$ appears at least once in $[i_1,\ldots,i_m]$ and <u>linear</u> if all $i_j (1 \leq i \leq m)$ are pairwise unequal in $[i_1,\ldots, i_m]$. If $[i_1,\ldots,i_m] = [1,\ldots,m]$ the transduction rule is called <u>simple</u> and <u>regular</u> for $[i_1] = [1]$.

$$\text{A SCTS is a} \left\{\begin{array}{c} \text{RSDTS} \\ \text{SSDTS} \\ \text{LRSCTS} \\ \text{LSCTS} \\ \text{RSCTS} \end{array}\right\} \text{if all transduction rules are} \left\{\begin{array}{l} \text{regular} \\ \text{simple} \\ \text{linear and rank-preserving} \\ \text{linear} \\ \text{rank-preserving} \end{array}\right\}$$

${}_{\Sigma}SCT_{\Delta} = \{T(G) \subset T_{\Sigma} x T_{\Delta} \mid G \text{ is a SCTS}\}$ is called the class of SC-Transductions and we write SCT for a fixed pair (Σ,Δ).

<u>Theorem 2</u>: For the classes of Syntax-connected Transductions the following lattice exists:



4. Relations between Schemes and Transducers

A SCT-Scheme defines a pair of local trees, while a tree-transducer operates on an input tree and produces an output tree.

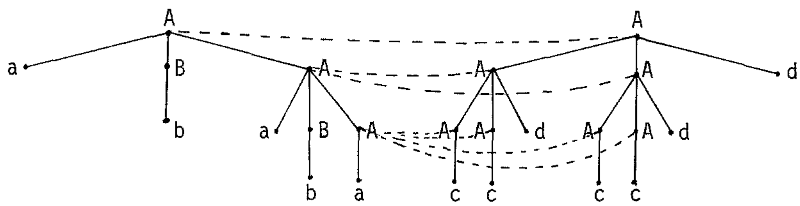
<u>Theorem 3</u>: For each α-SCTS G exists a α-TFT P such that T(G) = T(P).
(α = L or R or LR)

This theorem implies several corollaries delivering a large variety of results dealing with special restricted cases for transducers and transduction schemes as well.

Example: Let $G = (G_E, G_A, \kappa)$ have the transduction rules:

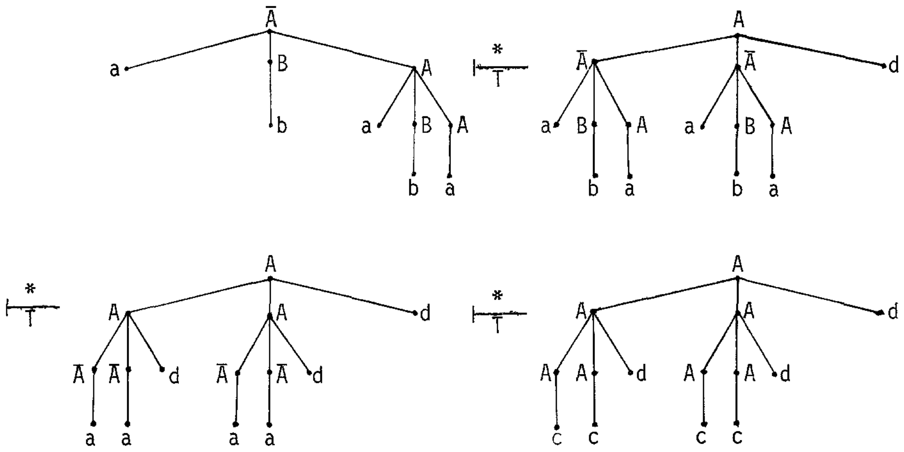
$A \rightarrow aBA, AAd[2,2]$ $\qquad A \rightarrow a,c$ $\qquad (B \rightarrow b) \varepsilon P_E$

The following pair of trees is in T(G):



The dotted lines indicate the connections appearing in the course of generation. This Tree-Transduction can be performed by a TFT with the following rules:

$\bar{A}(x_1x_2x_3) \rightarrow A(\bar{x}_3\bar{x}_3 d)$ $\qquad \bar{A}(x) \rightarrow A(\bar{x})$ $\qquad \bar{a} \rightarrow c$



Acknowledgements:

I thank BLEICKE EGGERS for valuable discussions and GITA SARI for her excellent typing.

References:

ENGELFRIET, J.: Bottom-up and Top-Down Tree Transducers - a comparison. Memorandum No. 19, 1971 Techn.Hogeschool Twente, Netherlands

ROUNDS, W.C.: Mappings and grammars on trees. MST 4, 257 - 287 (1970)

SCHREIBER, P.P.: Baum-Transduktoren (Thesis forthcoming)

SCHREIBER, P.P.: Operational Automata for Compiler Design. Bericht Nr. 75 - 13. Technische Universität Berlin, FB 20 - Kybernetik

THATCHER, J.W.: Generalized^2 Sequential Machine Maps. JCSS 4, 339 - 367 (1970)