On Syntax-Directed Transduction and Tree Transducers

by

David F. Martin* and Steven A. Vere*

SUMMARY

Several topics of theoretical and practical importance to the field of translator writing systems are presented in this paper. These are

(1). implementation of generalized syntax-directed transduction (GSDT) on a finite-state tree transducer;

(2). implementation of GSDT on a tree-walking pushdown store transducer;

(3). transformation of the context-free grammar underlying a GSDT and the resulting transformation of the transduction elements of that GSDT;

(4). tree transduction of parse trees between equivalent context-free grammars.

1. Introduction

Studies of theoretical models of the translation of computer programming languages have led to a better understanding of the nature and design of the compilation process. The study of syntax-directed transduction [1, 2] has been particularly fruitful in this respect. Models of transformational grammar such as tree transductions [3] have related uses.

The following topics are covered in this paper:

(1) relation between generalized syntax-directed transduction (GSDT) and tree transduction;

(2) implementation of GSDT on a tree-walking pushdown store transducer;

(3) transformation of the context-free grammar underlying a GSDT and the resulting transformation of the transduction elements of that GSDT;

(4) tree transductions of parse trees between equivalent context-free grammars.

2. Definitions and Notation

A context-free grammar is a system $G = (V_N, V_T, P, S)$, where V_N and V_T are respectively the nonterminal and terminal alphabets, $S \varepsilon V_N$ is the distinguished symbol, and P, the set of productions, is a finite subset of $V_N \times V^*$. The symbol Φ denotes the empty set, and λ denotes the null string. When x is a string, $lg(x)$ denotes the length of x; $lg(\lambda) = 0$.

A stratified alphabet is a pair (Σ, r), where Σ is a finite, nonempty set of symbols and $r\colon \ \Sigma \to \{0, 1, 2, \ldots\}$ associates a non-negative integer with each element of Σ. The function r partitions Σ into a finite number of subsets $\Sigma_0, \Sigma_1, \ldots, \Sigma_k$, where $\Sigma_i = \{\sigma \varepsilon \Sigma \mid r(\sigma) = i\}$. Often only Σ will denote a stratified alphabet, r being understood. The set of trees T_Σ generated by a stratified alphabet Σ is defined inductively as the smallest set containing Σ_0 and such that whenever $\sigma \varepsilon \Sigma_n$ and $t_0, t_1, \ldots, t_{n-1} \varepsilon T_\Sigma$, then $\sigma(t_0 t_1 \ldots t_{n-1}) \varepsilon T_\Sigma$. It is convenient to introduce variable leaf labels for trees, and this is done following Rounds [3]. Let A be a set. The set of trees $T_\Sigma(A)$, where Σ is a stratified alphabet, is the smallest set containing $\Sigma_0 \cup A$ and such that whenever $\sigma \varepsilon \Sigma_n$ and $t_0, t_1, \ldots, t_{n-1} \varepsilon T_\Sigma(A)$, then $\sigma(t_0\ t_1 \ldots t_{n-1}) \varepsilon T_\Sigma(A)$. Thus $T_\Sigma = T_\Sigma(\Phi) = T_\Sigma(\Sigma_0)$. When t is a tree, $||t||_C$ is the string composed of a left-to-right concatenation of the leaf labels of t.

A tree transducer (TT) is a system $A = (Q, \Sigma, X, \delta, q_0, F)$, where

Q = finite, nonempty set of states

Σ = stratified alphabet

X = set of variables

* Computer Science Department, School of Engineering and Applied Science, University of California, Los Angeles
This research was sponsored by the Office of Naval Research, Information Systems Branch, contract N00014-69-A-0200-4027, and the Atomic Energy Commission, contract AT(11-1) Gen 10 Project 14. Reproduction of whole or part is permitted for any purpose of the United States government.

$q_0 \in Q$ is the initial state

$F \subseteq Q$ is the set of final states

δ, the direct transition function, is in its most general form, a mapping of $T_\Sigma(Q \times T_\Sigma(\{x_0, x_1, \ldots, x_{k-1}\}))$ into the finite subsets of $T_\Sigma(Q \times T_\Sigma(\{x_0, x_1, \ldots, x_{k-1}\}))$, where $x_0, x_1, \ldots, x_{k-1} \in X$. In general tree transducers can operate nondeterministically, although only deterministic ones will be used here. Certain restrictions can be placed on the direct transition function:

(1). If $\delta: Q \times \{\sigma(x_0 x_1 \ldots x_{k-1})\} \to T_\Sigma(Q \times \{x_0, x_1, \ldots, x_{k-1}\})$, where $\sigma \in \Sigma_k$, then $A = (Q, \Sigma, X, q_0, F)$ is a (deterministic) finite-state tree transducer (FSTT).

(2). If $\delta: Q \times \{\sigma(x_0 x_1 \ldots x_{k-1})\} \to T_\Sigma(\{x_0, x_1, \ldots, x_{k-1}\})$, then $A = (Q, \Sigma, X, q_0, F)$ is called a context-free tree transducer.

These tree transducers are almost the same as Rounds' tree grammars [3], except that a transition function rather than tree rewriting rules is used, and the set of trees accepted (and transduced) by the device may be infinite. Viewing the components of δ as tree rewriting rules, the tree transducers opeate in the same manner as the tree grammars of Rounds [3].

An instantaneous description (ID) of a tree transducer is a representation of the partially transformed input tree plus the state symbols labeling the roots of its subtrees, i.e., an element of $T_\Sigma(Q \times T_\Sigma(\Phi))$. Informally, an atomic move $\vdash_A$ of a tree transducer A relates two ID's separated by a single direct transition. The set $Y(A)$ of trees accepted by a finite-state tree transducer $A = (Q, \Sigma, X, q_0, F)$ is

$$Y(A) = \{x \in T_\Sigma(\Phi) \mid (q_0, x) \overset{*}{\vdash_A} y,\ y \in T_\Sigma(F \times \Sigma_0)\} .$$

Final states will not come into play in this paper, but are included here for completeness. The set $Z(A)$ of trees output by a finite-state tree transducer $A = (Q, \Sigma, X, q_0, F)$ is just the set of accepting ID's of A, i.e.,

$$Z(A) = \{y \in T_\Sigma(F \times \Sigma_0) \mid (q_0, x) \overset{*}{\vdash_A} y,\ x \in Y(A)\} .$$

A generalized syntax-directed transduction (GSDT) [2] is a system $G_t = (G, \Gamma, \Delta, R)$, where

$G = (V_N, V_T, P, S)$ is a context-free grammar;

Γ is a finite, nonempty set of translation symbols of the form $t_j(A)$, $A \in \overline{V_N - \{S\}}$, plus the symbol $t_1(S)$;

Δ = output alphabet

R: $P \to$ finite subsets of $(\Gamma \cup \Delta)^*$, such that if $h_0 B_1 h_1 B_2 \ldots h_{m-1} B_m h_m \in R(C \to g_0 A_1 g_1 A_2 \ldots g_{n-1} A_n g_n)$,

then $g_i \in V_T^*$, $h_i \in \Delta^*$, $A_i \in V_N$, $B_i \in \Gamma$, where the B_i are of the form $t_j(A)$, $A \in \{A_1, A_2, \ldots, A_n\}$. Implying an ordering on the elements of $R(A \to x) = \{y_1, \ldots, y_k\}$, the notation used in [2] and here is

$$A \to x \qquad \begin{aligned} t_1(A) &= y_1 \\ t_2(A) &= y_2 \\ &\vdots \\ t_k(A) &= y_k \end{aligned}$$

where $y_1, y_2, \ldots, y_k$ are the transduction elements associated with $A \to x$. The translation induced by $G_t = (G, \Gamma, \Delta, R)$ is a mapping $t : L(G) \to$ subsets of Δ^*.

3. GSDT's and Finite-State Tree Transducers

Theorem 1

Corresponding to each GSDT $G_t = (G, \Gamma, \Delta, R)$ there exists a deterministic FSTT A such that $||Z(A)||_C = t(L(G))$.

Proof:

Let $G_t = (G, \Gamma, \Delta, R)$ be the given GSDT, where $G = (V_N, V_T, P, S)$. Let $P = \{P_1, P_2, \ldots, P_n\}$, and form

$$\overline{P} = \{\overline{P}_{11}, \overline{P}_{12}, \ldots, \overline{P}_{1k_1},\ \overline{P}_{21}, \overline{P}_{22}, \ldots, \overline{P}_{2k_2},\ \ldots,\ \overline{P}_{n1}, \overline{P}_{n2}, \ldots, \overline{P}_{nk_n}\},$$

where k_i is the number of transduction elements associated with P_i.

Let

$$p = \max_{t_j(A) \in \Gamma}(j) \qquad \max_i (k_i)$$

construct the FSTT $A = (\{q_1, \ldots, q_p\} \times V_N,\ P \cup \overline{P} \cup V_T \cup \{\lambda\} \cup \Delta,\ \{x_0, x_1, \ldots\},\ \delta,\ [q_1, S],\ \Phi)$.

The stratified alphabet $P \cup \overline{P} \cup V_T \cup \{\lambda\} \cup \Delta$ is partitioned as follows:

$\Sigma_0 = V_T \cup \{\lambda\} \cup \Delta$

$\Sigma - \Sigma_0 = P \cup \overline{P}$, where

$r(P_i) = r(A \to x) = \max(1, \lg(x))$, where
$P_i = A \to x$ is the i-th production;

$r(\overline{P}_{ij}) = \max(1, \lg(y_j))$, where $P_i = A \to x$ and
$t_j(A) = y_j$.

To construct δ, let, for each production
$P_i = A \to g_0A_1g_1 \cdots g_{n-1}A_ng_n$ in P, the j-th translation of A be

$t_j(A) = h_0B_1h_1 \cdots h_{m-1} B_mh_m$.

Corresponding to the above, include in δ the transition

$$\delta([q_j,A], P_i(x_0x_1 \cdots x_{k-1})) = \overline{P}_{ij}(h_0([q_{j_1},A_{k_1}],x_{\ell_1})h_1([q_{j_2},A_{k_2}], x_{\ell_2}) \cdots h_{m-1}([q_{j_m},A_{k_m}],x_{\ell_m})h_m) ,$$

where if $B_i = t_r(A_s)$, then

$j_i = r$

$k_i = \max(1, \lg(g_0A_1g_1 \cdots g_{s-1}A_s)-1)$

$\ell_i = s$

The translation of the input tree by the FSTT corresponds to the "top-to-bottom" interpretation of the corresponding GSDT along the lines suggested by Lewis and Stearns [1]. The states of the FSTT act as "marks" which pass through the tree, directing the translation in the same manner as the tree grammars of [3]. Once this and the above construction are understood, the proof becomes straightforward without additional details.

Example 1

The GSDT in Example 3.1 of [2] is implicitly defined by

1.	$S \to A$	$t_1(S) = t_2(A)ct_1(A)$
2.	$A \to aA$	$t_1(A) = t_2(A)ct_1(A)$ $t_2(A) = at_2(A)$
3.	$A \to bA$	$t_1(A) = t_2(A)ct_1(A)$ $t_2(A) = bt_2(A)$
4.	$A \to a$	$t_1(A) = \lambda$ $t_2(A) = a$
5.	$A \to b$	$t_1(A) = \lambda$ $t_2(A) = b$

This GSDT produces, for each $w \in \{a,b\}^+$, a string composed of the suffixes of w separated by c's. The corresponding FSTT is

$$A = (\{q_1, q_2\} \times \{S, A\}, \Sigma, \{x_0, x_1\}, \delta, [q_1,S], \Phi),$$

where

$\Sigma = \{P_1,\ldots,P_5, \overline{P}_{11}, \overline{P}_{21}, \overline{P}_{22},\ldots,\overline{P}_{51}, \overline{P}_{52}\}$ and

δ consists of

$\delta([q_1,S], P_1(x_0)) = \overline{P}_{11}(([q_2,A],x_0)c([q_1,A],x_0))$

$\delta([q_1,A], P_2(x_0x_1)) = \overline{P}_{21}(([q_2,A], x_1) c([q_1,A],x_1))$

$\delta([q_2,A], P_2(x_0x_1)) = \overline{P}_{22}(a([q_2,A], x_1))$

$\delta([q_1,A], P_3(x_0x_1)) = \overline{P}_{31}(([q_2,A], x_1) c([q_1,A],x_1))$

$\delta([q_2,A], P_3(x_0x_1)) = \overline{P}_{32}(b([q_2,A], x_1))$

$\delta([q_1,A], P_4(x_0)) = \overline{P}_{41}(\lambda)$

$\delta([q_2,A], P_4(x_0)) = \overline{P}_{42}(a)$

$\delta([q_1,A], P_5(x_0)) = P_{51}(\lambda)$

$\delta([q_2,A], P_5(x_o)) = \overline{P}_{52}(b)$

Theorem 2 (Rounds [3])

Deterministic finite-state tree transductions are effectively closed under composition. Immediately from theorems 1 and 2 we have

Corollary

GSDT's are effectively closed under composition.

4. Tree-Walking Pushdown Store Transducers

A tree-walking pushdown store transducer (TPDT) is a system $A = (Q, G, \Gamma, \Delta, \delta, q_0, Z_0, F)$, where

- Q = finite, nonempty set of states
- $G = (V_N, V_T, P, S)$ is a context-free grammar
- Γ = pushdown store alphabet
- Δ = output alphabet
- $\delta = Q \times (P \cup V_T \cup \{\lambda\}) \times \Gamma \to Q \times \{-1, 0, 1,\ldots,p\} \times \Gamma^* \times \Delta^*$ is the direct transition function
- $q_0 \in Q$ is the initial state
- $Z_0 \in \Gamma$ is the initial pushdown store symbol
- $F \subseteq Q$ is the set of final states

The TPDT operates similarly to the tree automata of [2], except that a pushdown store is involved. Another view of the TPDT is a pushdown store transducer whose input is a parse tree on G and whose input head executes a "two-way" movement over this tree.

The TPDT makes an atomic move as follows. If $(p, d, w, y) \in \delta(q, A, Z)$, the TPDT is in state q, its input head is located at a node labeled A in the input tree, and Z is the topmost symbol of the pushdown store. The TPDT then changes to state p, replaces Z by w, outputs y, and moves its input head in direction d. If $d = 0$, no head movement is made. If $d = -1$, the head moves to the ancestor of the node where it previously resided. If $d = i$, $i > 0$, the head moves to the i-th descendent of the node previously read. The symbols of w, read left-to-right, are the topmost symbols of the pushdown store read top-to-bottom.

Theorem 3

Each GSDT can be implemented on a one-state, deterministic TPDT.

Proof:

Let $G_t = (G, \Gamma, \Delta, R)$, $G = (V_N, V_T, P, S)$ be the given GSDT. A one-state, deterministic TPDT will be constructed that simulates a preorder traversal [4] of the output tree corresponding to the given GSDT, as if it were implemented on a FSTT. As the leaves of this output tree are "visited," their labels are output by the TPDT. Construct

$A = (\{q\}, G, \Gamma \times \{1,2,\ldots,p\}, \Delta, \delta, q,$
$[t_1(S),1], \Phi)$, where

$p = 1 +$ maximum value of m attained in a transduction element $h_0B_1h_1\ldots h_{m-1}B_mh_m$.

Let $P = \{P_1,P_2,\ldots,P_k\}$. The transition function δ is constructed as follows. Let

$$P_i = A \rightarrow g_0A_1g_1\ldots g_{n-1}A_ng_n \quad ,$$

and let the j-th transduction element associated with this production be

$$t_j(A) = h_0B_1h_1\ldots h_{m-1}B_mh_m \quad .$$

Corresponding to the above, include in δ the transitions

$$\delta\ (q,P_i,[t_j(A),1]) = (q,i_1,[B_1,1]\ [t_j(A),2],h_0)$$
$$\delta\ (q,P_i,[t_j(A),2]) = (q,i_2,[B_2,1][t_j(A),3],h_1)$$
$$\vdots$$
$$\delta(q,P_i,[t_j(A),m]) = (q,\ i_m,[B_m,1][t_j(A),m+1],\ h_{m-1})$$

$$\delta(q,P_i,[t_j(A),m+1]) = (q,\ -1,\ \lambda,\ h_m)\ .$$

If $B_r = t_v(A_\ell)$, then $i_r = lg(g_0A_1g_1\ldots g_{\ell-1}A_\ell)$.

The above transitions break up the computation of $t_j(A)$ into m+1 "phases":

1. Output h_0; compute B_1;

$\vdots$

m. Output h_{m-1}; compute B_m;

m+1. Output h_m; return to predecessor node.

The reader can readily convince himself that the recursively applied rules for preorder tree traversal

"visit the root;
"visit the subtrees in left-to-right order"

are indeed being applied to the output tree, and the leaves of this tree are being output in left-to-right order.

Example 2

Let G_t be the GSDT of Example 1. The corresponding TPDT is

$M = (\{q\}, G, \{t_1(S),t_1(A),t_2(A)\} \times \{1,2,3\},$
$\{a,b,c\}, \delta, q, [t_1(S),1], \Phi)$, where δ
consists of

$$\delta(q,P_1,[t_1(S),1]) = (q,\ 1,\ [t_2(A),1][t_1(S),2],\lambda)$$
$$\delta(q,P_1,[t_1(S),2]) = (q,\ 1,\ [t_1(A),1][t_1(S),3],c)$$
$$\delta(q,P_1,[t_1(S),3]) = (q,-1,\lambda,\lambda)$$
$$\delta(q,P_2,[t_1(A),1]) = (q,\ 1,\ [t_2(A),1][t_1(A),2],\lambda)$$
$$\delta(q,P_2,[t_1(A),2]) = (q,\ 1,\ [t_1(A),1][t_1(A),3],c)$$
$$\delta(q,P_2,[t_1(A),3]) = (q,\ -1,\ \lambda,\ \lambda)$$
$$\delta(q,P_2,[t_2(A),1]) = (q,\ 1,\ [t_2(A),1][t_2(A),2],a)$$
$$\delta(q,\ P_2,[t_2(A),2]) = (q,\ -1,\ \lambda,\ \lambda)$$
$$\delta(q,\ P_3,[t_1(A),1]) = (q,\ 1,\ [t_2(A),1][t_1(A),2],\lambda)$$
$$\delta(q,\ P_3,[t_1(A),2]) = (q,\ 1,\ [t_1(A),1][t_1(A),3],c)$$
$$\delta(q,\ P_3,[t_1(A),3]) = (q,\ -1,\ \lambda,\ \lambda)$$
$$\delta(q,\ P_3,[t_2(A),1]) = (q,\ 1,\ [t_2(A),\ 1][t_2(A),2],\ b)$$
$$\delta(q,\ P_3,[t_2(A),2]) = (q,\ -1,\ \lambda,\ \lambda)$$
$$\delta(q,\ P_4,[t_1(A),1]) = (q,\ -1,\ \lambda,\ \lambda)$$
$$\delta(q,\ P_4,[t_2(A),1]) = (q,\ -1,\ \lambda,\ a)$$
$$\delta(q,\ P_5,[t_1(A),1]) = (q,\ -1,\ \lambda,\ \lambda)$$
$$\delta(q,\ P_5,[t_2(A),1]) = (q,\ -1,\ \lambda,\ b)$$

5. Transformations of the Grammars Underlying Syntax-Directed Transductions

Transforming the grammar underlying a syntax-directed transduction generally changes the transduction elements associated with the new productions. Only transformations that result in weakly equivalent grammars will be considered in this section.

(1). Substitution of Productions

Let $G = (V_N, V_T, P, S)$ be a context-free grammar. Let $A \to z_1Bz_2 \in P$, $B \in V_N$, and let $B \to y_i$, $i = 1, 2, \ldots, k$ be all the B-productions in P. Let $G' = (V_N, V_N, P', S)$, where $P' = (P - \{A \to z_1Bz_2\}) \cup \{A \to z_1y_iz_2 \mid i = 1, 2, \ldots, k\}$; clearly $L(G') = L(G)$.

Theorem 4

GSDT's are effectively closed under substitution of productions.

Proof:

Let $G_t = (G, \Gamma, \Delta, R)$, $G = (V_N, V_T, P, S)$ be the given GSDT. Let $A \to z_1Bz_2$ be the production to be deleted, via replacement of B by $B \to y_1, \ldots, B \to y_k$. Construct $G_t' = (G', \Gamma, \Delta, R')$, $G' = (V_N, V_T, P', S)$, where

$P' = (P - \{A \to z_1Bz_2\}) \cup \{A \to z_1y_iz_2 \mid i = 1, 2, \ldots, k\}$.

To construct R', define homomorphisms h_i: $(\Gamma \cup \Delta)^* \to (\Gamma \cup \Delta)^*$ such that if

$$R(B \to y_i) = \{x_{i1}, x_{i2}, \ldots, x_{ik_i}\},$$

then $h_i(t_j(B)) = x_{ij}$, $j = 1,2,\ldots,k_i$
$h_i(X) = X$, $X \neq t_j(B)$ for some $1 \le j \le k_i$.

Obtain R' via

$$R'(X) = X, \; X \in P - \{A \to z_1Bz_2\}$$

$$R'(A \to z_1y_kz_2) = \{h_i(x_1), \ldots, h_i(x_m)\} = h_i(R(A \to z_1Bz_2)),$$

where $R(A \to z_1Bz_2) = \{x_1, \ldots, x_m\}$.

Clearly G_t' and G_t produce the same transduction.

Example 3

$S \to E$	$t_1(S) = t_1(E)\#t_2(E)$
$E \to E + T$	$t_1(E) = t_1(E)t_1(T) +$
	$t_2(E) = + t_2(E)t_2(T)$
$E \to T$	$t_1(E) = t_1(T)$
	$t_2(E) = t_2(T)$
$T \to T*a$	$t_1(T) = t_1(T)a*$
	$t_2(T) = *t_2(T)a$
$T \to a$	$t_1(T) = a$
	$t_2(T) = a$

Eliminating the production $S \to E$ through substitution of $E \to E + T$ and $E \to T$ yields the GSDT

$S \to E + T$	$t_1(S) = t_1(E)t_1(T)+\#+t_2(E)t_2(T)$
$S \to T$	$t_1(S) = t_1(T)\#t_2(T)$
$E \to E+T$	$t_1(E) = t_1(E)t_1(T) +$
	$t_2(E) = + t_2(E)t_2(T)$
$E \to T$	$t_1(E) = t_1(T)$
	$t_2(E) = t_2(T)$
$T \to T*a$	$t_1(T) = t_1(T)a*$
	$t_2(T) = *t_2(T)a$
$T \to a$	$t_1(T) = a$
	$t_2(T) = a$

Corollary

Syntax-directed transductions (SDT's) and simple SDT's are closed under substitution of productions. Simple Polish SDT's are not closed under substitution of productions.

(2). Redefinition

Let $G = (V_N, V_T, P, S)$ be a context-free grammar, and let $A \to y_1y_2y_3$ be a production of P. Construct $G' = (V_N \cup \{Z\}, V_T, P', S)$, where Z is a new nonterminal and

$$P' = (P - \{A \to y_1y_2y_3\}) \cup \{A \to y_1Zy_3, Z \to y_2\}.$$

Clearly $L(G') = L(G)$.

Theorem 5

GSDT's are effectively closed uder redefinition.

Proof: Let $G_t = (G, \Gamma, \Delta, R)$, $G = (V_N, V_T, P, S)$ be the given GSDT, and let $A \to y_1y_2y_3$ be the production that participates in the redefinition.

Construct $G_t' = (G', \Gamma', \Delta, R')$, $G' = (V_N \cup \{Z\}, V_T, P', S)$, where $P' = (P - \{A \to y_1y_2y_3\}) \cup \{A \to y_1Zy_3, Z \to y_2\}$.

Γ' and R' are obtained as follows. Let $A \to y_1y_2y_3 = A \to g_0A_1g_1\cdots g_{n-1}A_ng_n$.

Let $y_2 \in (W \cup V_T)^*$, where $W \subseteq \{A_1, A_2, \ldots, A_n\}$. In particular, suppose $W = \{A_{i_1}, \ldots, A_{i_m}\}$.

Let $R(A \to y_1y_2y_3) = \{x_1, x_2, \ldots, x_k\}$. Then

$$R'(Z \to y_2) = \{t_1(A_{i_1}), t_2(A_{i_1}), \ldots, t_{k_1}(A_{i_1}), \ldots \\ \vdots \\ t_1(A_{i_m}), t_2(A_{i_m}), \ldots, t_{k_m}(A_{i_m})\},$$

and hence $\Gamma' = \Gamma \cup \{t_1(Z), \ldots, t_\ell(Z)\}$, where $\ell = k_1 + k_2 + \ldots + k_m$. Define a homomorphism h: $(\Gamma \cup \Delta)^* \to (\Gamma' \cup \Delta)^*$ such that

$h(t_j(A_{i_s})) = t_r(Z)$, $A_{i_s} \in W$, $r = k_1 + k_2 + \ldots + k_{s-1} + j$

$h(X) = X$, otherwise.

Then $R'(A \to y_1 Z y_2) = h(R(A \to y_1 y_2 y_3))$.

$R'(X) = R(X)$, $X \in P - \{A \to y_1 y_2 y_3\}$.

The above construction is based upon the fact that in the redefinition transformation replacing $A \to y_1 y_2 y_3$ by $A \to y_1 Z y_3$ and $Z \to y_2$, the nonterminal A "loses control" of the translations of the nonterminals in y_2, and must "pass them on" to Z. Careful scrutiny of the example below will convince the reader that G_t' and G_t induce the same transduction.

Example 4

Consider the trivial GSDT defined implicitly by

$A \to aBabcCD$ $\quad t_1(A) = t_2(C)xyt_1(B)t_1(D)t_1(C)xt_2(B)$

$B \to b$ $\quad t_1(B) = \lambda$

$\quad t_2(B) = b$

$C \to c$ $\quad t_1(C) = c$

$\quad t_2(C) = \lambda$

$D = d$ $\quad t_1(D) = d$

Let $y_1 = a$, $y_2 = BabcC$, $y_3 = D$. Then $W = \{B,C\} \subseteq \{B,C,D\}$, and the translation symbols $t_1(Z),\ldots,t_4(Z)$ are added, where the homomorphism of the proof is

$h(t_1(B)) = t_1(Z)$; $h(t_2(B)) = t_2(Z)$;

$h(t_1(C)) = t_3(Z)$; $h(t_2(C)) = t_4(Z)$;

$h(X) = X$, otherwise.

The following GSDT results:

$A \to aZD$ $\quad t_1(A) = t_4(Z)xyt_1(Z)t_1(D)t_3(Z)xt_2(Z)$

$Z \to BabcC$ $\quad t_1(Z) = t_1(B)$; $t_2(Z) = t_2(B)$

$\quad t_3(Z) = t_1(C)$; $t_4(Z) = t_2(C)$

$B \to b$ $\quad t_1(B) = \lambda$; $t_2(B) = b$

$C \to c$ $\quad t_1(C) = c$; $t_2(C) = \lambda$

$D \to d$ $\quad t_1(D) = d$

Corollary

GSDT's are effectively closed under transformation to Chomsky normal form.

Proof: Transformation to Chomsky normal form consists of a finite number of redefinition transformations.

(3). Arden's Transformation

Let $G = (V_N, V_T, P, S)$ be a context-free grammar. Let $A \to A\alpha_i$ $i = 1, 2, \ldots, r$ be the left-recursive A-productions in P, and let $A \to \beta_i$, $i = 1, 2, \ldots, s$ be the remaining A-productions.

Construct $G' = (V_N \cup \{Z\}, V_T, P', S)$, where

$$P' = (P - \{A \to A\alpha_1, \ldots, A \to A\alpha_r\}) \cup \{A \to \beta_i Z \mid i = 1,2,\ldots,s\} \cup \{Z \to \alpha_i \mid i = 1, 2, \ldots, r\} \cup \{Z \to \alpha_i Z \mid i = 1, 2, \ldots, r\} .$$

Then $L(G') = L(G)$.

Arden's Transformation of Syntax-Directed Transductions

Let $G_t = (G, \Gamma, \Delta, R)$, $G = (V_N, V_T, P, S)$ be given, where G_t is not a GSDT, but rather the simpler SDT of [1]. Construct $G_t' = (G', \Gamma, \Delta, R')$, where G' is the grammar constructed above. Let $R(A \to A\alpha_i) = \{x_i A \bar{x}_i\}$, $i = 1, 2, \ldots, r$

$R(A \to \beta_i) = y_i$, $i = 1, 2, \ldots, s$.

Then $R'(A \to \beta_i) = R(A \to \beta_i)$, $i = 1, 2, \ldots, s$

$R'(A \to \beta_i Z) = \{t_1(Z)\, y_i t_2(Z)\}$, $i = 1, 2, \ldots, s$

$R'(Z \to \alpha_i Z) = \{t_1(Z)x_i, \bar{x}_i t_2(Z)\}$, $i = 1, 2, \ldots, r$

$R'(Z \to \alpha_i) = \{x_i, \bar{x}_i\}$, $i = 1, 2, \ldots, r$

$R'(X) = R(X)$, all other $X \in P'$.

G_t' and G_t induce the same transduction. Note that G_t' is a GSDT.

Conjecture

GSDT's are not closed under Arden's transformation or transformation to Greibach normal form.

6. Transformations of Parse Trees

Suppose a GSDT $G_t = (G, \Gamma, \Delta, R)$ is given. Suppose further that it is more convenient to parse strings of L(G) according to grammar G', where $L(G') = L(G)$. It may not always be possible to transform G_t to an equivalent syntax-directed transduction of the same class. Rather than transform G_t, the parse trees on G' could be transformed back to their counterparts on G, and G_t applied. A preliminary result is presented in this section.

Inverse Chomsky Normal Form

Let $G = (V_N, V_T, P, S)$ be a context-free grammar and $G' = (V_N', V_T, P', S)$ be a Chomsky normal form grammar such that $L(G') = L(G)$. A tree transduction can be used to transform parse trees on G' to their counterparts on G. Consider the P-production

$$P_k = A \rightarrow aAbBAb, \quad A, B \in V_N, \; a, b \in V_T$$

and its counterparts in P'

$$\begin{aligned}
P'_i &= A \rightarrow N_a X_1 \\
P'_{i+1} &= X_1 \rightarrow AX_2 \\
P'_{i+2} &= X_2 \rightarrow N_b X_3 \\
P'_{i+3} &= X_3 \rightarrow BX_4 \\
P'_{i+4} &= X_4 \rightarrow AN_b \\
P'_{i+5} &= N_a \rightarrow a \\
P'_{i+6} &= N_b \rightarrow b
\end{aligned}$$

The corresponding direct transitions of an inverse Chomsky normal form general tree transduction are

$$\begin{aligned}
&\delta(q, P'_{i+5}(x_0)) = a \\
&\delta(q, P'_{i+6}(x_0)) = b \\
&\delta(q, P'_i(x_0 P'_{i+1}(x_1 x_2))) = (q, \bar{P}'_{i+1}(x_0 x_1 x_2)) \\
&\delta(q, \bar{P}'_{i+1}(x_0 x_1 \, P'_{i+2}(x_2 x_3))) = (q, \bar{P}'_{i+2}(x_0 x_1 x_2 x_3)) \\
&\delta(q, \bar{P}'_{i+3}(x_0 x_1 x_2 x_3 \, P'_{i+4}(x_4 x_5))) = P_k((q, x_0)\ldots \\
&\qquad (q, x_5))
\end{aligned}$$

Other inverse transformations can be accomplished using tree transductions, including inverse Arden's transformation and inverse Greibach normal form. This and other work is currently being continued.

7. Conclusion

Several topics of theoretical and practical importance to the field of translator writing systems were presented in this paper. It is instructive to note the practical difference between the two implementations of GSDT presented in Sections 3 and 4. The FSTT implementation (Section 3) results in a "program" (transition function) of moderate size, but requires a relatively complex memory management system for manipulating tree structures. On the other hand, the TPDT implementation (Section 4) requires a relatively simple memory management system, since the input tree is fixed and the output can be written into a sequential file. As payment, the TPDT requires a relatively large program.

Much more work than appears in this paper has been done [5]. The properties of a syntax-directed transduction scheme more general than GSDT, and equivalent to the scheme of [6], have been investigated.

BIBLIOGRAPHY

1. Lewis, P. M., and Stearns, R. E., "Syntax-Directed Transduction," J. ACM 15:464-488, July 1968.

2. Aho, A. V., and Ullman, J. D., "Translations on a Context-Free Grammar," Conf. Record of ACM Symposium on Theory of Computing, Marina Del Rey, Calif., May 1969, pp. 93-112.

3. Rounds, W. C., "Context-Free Grammars on Trees," Conf. Record of ACM Symposium on Theory of Computing, Marina Del Rey, Calif., May 1969, pp. 143-148.

4. Knuth, D. E., Fundamental Algorithms, Addison-Wesley, Reading, Mass., 1968.

5. Vere, Steven A., "Syntax Directed Translation of Context-Free Languages," forthcoming Ph.D. Dissertation, Computer Science Department, UCLA.

6. Knuth, D. E., "Semantics of Context-Free Languages," Math. Systems Theory 2:127-145, 1968.