# Tree Transductions and Families of Tree Languages

Brenda S. Baker

Center for Research in Computing Technology
Harvard University
Cambridge, Massachusetts 02138

## Introduction

Interest in the study of sets of trees, or tree languages, has led to the definition of finite automata which accept trees [2,11] and of transducers which map trees into other trees [7,9, 10]. These generalized machines may read trees either "top-down" (from the root toward the leaves) or "bottom-up" (from the leaves toward the root). The classes of top-down and bottom-up nondeterministic transductions as defined by Thatcher and by Rounds [7,9,10] have been shown to be incomparable; there are transductions which can be done by a bottom-up transducer which cannot be done by any top-down transducer, and vice versa. Here it is shown that both the class of top-down transductions and the class of bottom-up transductions can be characterized in terms of two restricted classes of tree transductions. From these characterizations, it is shown that the composition of any $n$ bottom-up transductions can be realized by the composition of $n+1$ top-down transductions, and similarly, the composition of any $n$ top-down transductions can be realized by the composition of $n+1$ bottom-up transductions.

Next, we study the families of tree languages which can be obtained from the recognizable sets (sets accepted by finite tree automata) by the composition of $n$ top-down or bottom-up transductions, $n \geq 0$. It is shown that these families form a single hierarchy in which the "bottom-up" families alternate with the "top-down" families. We conjecture that each inclusion in the hierarchy is proper, but we have as yet been unable to prove this conjecture.

The yield operation, which concatenates the leaves of a tree from left to right to form a string, is used to obtain a hierarchy of families of string languages from the hierarchy of families of tree languages. It is shown that each family of string languages in this hierarchy is properly contained in the family of context-sensitive languages; this solves an open problem proposed by Thatcher [9].

Finally, the closure properties of the top-down and bottom-up families in both the tree hierarchy and the string hierarchy are investigated. The conjecture that the tree hierarchy is infinite is shown to be equivalent to conjectures that certain families in the tree hierarchy are not closed under certain tree operations.

†The results stated here will be included in the author's doctoral dissertation (Harvard University, in preparation). This research has been supported in part by the National Science Foundation under Grant NSF-GJ-30409. The results were obtained while the author was an NSF Pre-Doctoral Fellow at Harvard University.

## 1. Definitions and Notation

The definitions given here are based on those of Engelfriet [3].

A finite set of symbols or alphabet $A$ is ranked by specifying a function $r: A \to N$, where $N$ is the set of nonnegative integers. For $n \geq 0$, $A_n = \{b \in A \mid r(b) = n\}$ is the set of symbols of rank $n$.

Given a ranked alphabet $A$, the set of all finite trees labeled over the alphabet $A$ is $A_*$, defined inductively as follows:

(i) If $b \in A_0$, then $b \in A_*$;

(ii) If $n \geq 1$, $b \in A_n$, and $t_1, t_2, \ldots, t_n \in A_*$, then $b[t_1, \ldots, t_n] \in A_*$.

A tree labeled over the alphabet $A$ is thus a string in $(A \cup \pi)^*$, where $\pi$ is the set containing left and right brackets and comma. A subset of $A_*$ is called a tree language.

A relation $R \subseteq \Sigma_* \times \Delta_*$, where $\Sigma$ and $\Delta$ are ranked alphabets, is called a tree transformation. If $\langle s,t \rangle \in R$, then $t$ is a tree obtained from $s$ under the transformation $R$. If $R_1$ and $R_2$ are tree transformations, then the composition of $R_1$ and $R_2$ is $R_2 \circ R_1 = \{\langle s,u \rangle \mid$ for some $t$, $\langle s,t \rangle \in R_1$ and $\langle t,u \rangle \in R_2\}$.

If $R$ is a tree transformation and $T$ is a set of trees, then $R(T) = \{t \mid$ for some $s \in T$, $\langle s,t \rangle \in R\}$. If $C$ is a class of tree transformations and $F$ is a family of tree languages, then $C(F) = \{R(T) \mid R \in C$ and $T \in F\}$.

Of interest in this paper are classes of tree transformations which are generated by tree transducers, which are special automata which read trees and output trees. In order to define tree transducers, it is necessary to consider alphabets indexed by sets of trees. Given a set $S$ of trees over a ranked alphabet $\Delta$, and another ranked alphabet $\Sigma$, the set of $\Sigma$-trees indexed by $S$ is $\Sigma_*(S)$, defined by

(i) $\Sigma_0 \cup S \subseteq \Sigma_*(S)$

(ii) If $n \geq 1$, $b \in \Sigma_n$, and $t_1,\ldots,t_n \in \Sigma_*(S)$, then $b[t_1,\ldots,t_n] \in \Sigma_*(S)$.

We also need to use a special set of variables $X = \{x_1,x_2,x_3,\ldots\}$. Let $X_0 = \phi$, and for $k \geq 1$, let $X_k = \{x_1,x_2,\ldots,x_k\}$.

A nondeterministic tree transducer is a 5-tuple $M = (Q,\Sigma,\Delta,R,P)$ where

(1) $Q$ is a finite ranked alphabet of states, such that each state has rank 1,

(2) $\Sigma$ is a finite input alphabet,

(3) $\Delta$ is a finite output alphabet,

(4) $P \subseteq Q$ and

(5) $R$ is a finite set of rule-schemes restricted according to whether the transducer is top-down or bottom-up. In a top-down transducer, each rule-scheme in $R$ is of one of the forms

$q[b] \to t$, where $q \in Q$, $b \in \Sigma_0$, and $t \in \Delta_*$, or

$q[b[x_1,\ldots,x_n]] \to t$, where $n > 0$, $q \in Q$, $b \in \Sigma_n$, and $t \in \Delta_*^n(Q[X_n])$.

In a bottom-up transducer, each rule in $R$ is of one of the forms

$b \to q[t]$ where $b \in \Sigma_0$, $q \in Q$, and $t \in \Delta_*$, or

$b[q_1[x_1],\ldots,q_n[x_n]] \to q[t]$ where $n > 0$, $b \in \Sigma_n$, $q,q_1,\ldots,q_n \in Q$, and $t \in \Delta_*[X_n]$.

The rule-schemes in $R$ generate a set of rules $S$, defined as follows. If $M$ is a top-down transducer, then define $\text{Range}(X) = \Sigma_*$. If $M$ is a bottom-up transducer, define $\text{Range}(X) = \Delta_*$. Then $S = \{h(u) \to h(v) \mid h: (\Sigma \cup \Delta \cup \pi \cup X)^* \to (\Sigma \cup \Delta \cup \pi)^*$ is a homomorphism with $h(x_i) \in \text{Range}(X)$ for each $i$, $h(b) = b$ for $b \in \Sigma \cup \Delta \cup \pi$, and $u \to v$ is a rule-scheme in $R\}$. Note that the set $S$ is obtained by replacing the variables in the rule by trees in either $\Sigma_*$ or $\Delta_*$, according to whether the transducer is top-down or bottom-up.

The transition relation $\Rightarrow$ of $M$ is defined as follows. For any rule $\alpha \to \beta \in S$ and any trees $\phi\alpha\psi$ and $\phi\beta\psi$, $\phi\alpha\psi \Rightarrow \phi\beta\psi$ (recall that trees have been defined as a special case of strings). Let $\overset{*}{\Rightarrow}$ be the transitive reflexive closure of $\Rightarrow$.

The transduction performed by $M$ is $M = \{\langle t_1,t_2\rangle \in \Sigma_* \times \Delta_* \mid$ for some $q \in P$, $q[t_1] \overset{*}{\Rightarrow} t_2\}$ if $M$ is top-down, while if $M$ is bottom-up it is $M = \{\langle t_1,t_2\rangle \in \Sigma_* \times \Delta_* \mid$ for some $q \in P$, $t_1 \overset{*}{\Rightarrow} q[t_2]\}$.

A tree transducer $M = (Q,\Sigma,\Delta,R,P)$ is

(1) linear if no variable occurs more than once in the right side of any rule-scheme,

(2) one-state if $Q$ is a singleton set,

(3) deterministic if (a) no two rule-schemes have the same left side, and (b) if $M$ is top-down, then $P$ is a singleton set,

(4) a finite tree automaton if $\Sigma = \Delta$ and either

(a) $M$ is top-down and every rule-scheme is of one of the two forms

$q[b] \to b$, where $b \in \Sigma_0$, $q \in Q$ or

$q[b[x_1,\ldots,x_n]] \to b[q_1[x_1],\ldots,q_n[x_n]]$, where $n > 0$, $b \in \Sigma_n$, $q,q_1,\ldots,q_n \in Q$, or

(b) $M$ is bottom-up and every rule is of one of the two forms

$b \to q[b]$, where $b \in \Sigma_0$, $q \in Q$, or

$b[q_1[x_1],\ldots,q_n[x_n]] \to p[b[x_1,\ldots,x_n]]$, where $n > 0$, $b \in \Sigma_n$, and $p,q_1,\ldots,q_n \in Q$.

A set of trees is recognizable if $S = M(\Sigma_*)$ for some finite tree automaton $M = (Q,\Sigma,\Sigma,R,P)$.

## 2. Tree transductions and composition

The classes of top-down and bottom-up transductions are incomparable [3]. Intuitively, the incomparability is derived from the ability of top-down transducers to copy input trees and then generate different output from each copy, and the ability of bottom-up transducers to produce output nondeterministically and then copy it. A natural question is whether these intuitive differences can be used to characterize the classes of top-down and bottom-up transductions. We answer this question in the affirmative; both top-down and bottom-up transductions can be characterized in terms of deterministic one-state transductions (which do the copying) and nondeterministic linear transductions (which apply the nondeterminism).

Let NB, NT, NLT, NLB, and DO denote the classes of nondeterministic bottom-up, nondeterministic top-down, nondeterministic linear top-down, nondeterministic linear bottom-up and deterministic one-state transductions[†], respectively.

[†]Engelfriet has shown that the class of deterministic one-state top-down transductions is the same as the class of deterministic one-state bottom-up transductions [3].

Theorem 1. The following classes of transformations are identical:

(1) NB, the class of nondeterministic bottom-up transductions;

(2) DO∘NLB, the class of transformations obtained by applying first a nondeterministic linear bottom-up transduction and then a deterministic one-state transduction;

(3) DO∘NLT, the class of transformations obtained by applying first a nondeterministic linear top-down transduction and then a deterministic one-state transduction.

Theorem 2. The following classes of transformations are identical:

(1) NT, the class of nondeterministic top-down transductions; and

(2) NLT∘DO, the class of transformations obtained by applying first a deterministic one-state transduction and then a nondeterministic linear top-down transduction.

Note that the class of top-down transductions cannot be characterized in terms of linear bottom-up transductions in Theorem 2, since there are linear bottom-up transductions which are not contained in the class of top-down transductions [3].

Theorem 2 was discovered independently by Engelfriet [3] and the author, and Engelfriet also obtained a decomposition theorem closely related to Theorem 1.

According to Theorems 1 and 2, the classes of top-down and bottom-up transductions can be decomposed into the same two classes of transductions, DO and NLT, applied in different orders. The two theorems show that the difference between top-down and bottom-up transductions lies in the relative order of copying (DO) and nondeterministic behavior (NLT).

Neither top-down nondeterministic transductions nor bottom-up nondeterministic transductions are closed under composition [3,9]. Theorems 1 and 2 show that the composition of n bottom-up or top-down transductions can be factored into alternating linear top-down and one-state transductions. This observation provides the following result.

Theorem 3. For any $n \geq 1$, $NB^n \subseteq NT^{n+1}$ and $NT^n \subseteq NB^{n+1}$. That is, the composition of any n bottom-up transductions can be realized by the composition of n+1 top-down transductions, and the composition of any n top-down transductions can be realized by the composition of n+1 bottom-up transductions.

Corollary 4. $\bigcup_{n=1}^{\infty} NB^n = \bigcup_{n=1}^{\infty} NT^n$. Thus, the closure of the class of nondeterministic bottom-up transductions under composition is the same as the closure of the class of nondeterministic top-down transductions under composition.

According to Theorem 3, any transformation which can be done in one direction by a succession of transductions can also be done in the other direction by doing one extra transduction. From the corollary, it may be seen that it makes no difference whether trees are read from the root toward the leaves or from the leaves toward the root, as long as there is no bound on the number of transductions applied. This corollary is opposite in spirit to the result of Engelfriet that the classes of top-down and bottom-up transductions are incomparable [3].

## 3. Two hierarchies

Neither top-down nor bottom-up transductions in general preserve recognizability [7,9,10]. This suggests that applying successive transductions to the family of recognizable sets generates hierarchies of families of tree languages.

Definition. Let $D_0$ and $U_0$ denote the family of recognizable sets. For any $n \geq 0$, define $D_{n+1} = NT(D_n)$ and $U_{n+1} = NB(U_n)$.

Thus, for $n \geq 0$, $D_n$ is the family of tree languages obtained from the recognizable sets by the composition of n top-down tree transductions. For $n \geq 0$, $U_n$ is the family of tree languages obtained from the recognizable sets by the composition of n bottom-up tree transductions.

Ogden and Rounds [6] studied the top-down families $D_0, D_1, D_2, \ldots$ and conjectured that for every $n \geq 0$, $D_n \subsetneq D_{n+1}$ so that the families form an infinite hierarchy. They proved only that $D_1 \subsetneq D_2$; it was known previously that $D_0 \subsetneq D_1$ [7]. Here, we use the bottom-up hierarchy $U_0, U_1, U_2, \ldots$ to refine both the hierarchy and their conjecture.

Two hierarchies have been defined, one generated by top-down transducers and the other by bottom-up transducers from the recognizable sets. But there is actually only a single hierarchy, as demonstrated by the following theorem. Recall that NLT is the class of nondeterministic linear top-down transductions and that DO is the class of deterministic one-state transductions (either top-down or bottom-up).

Theorem 5. For every $n \geq 0$, $D_n = NLT(U_n)$ and $U_{n+1} = DO(D_n)$. Therefore, for every $n \geq 0$, $D_n \subseteq U_{n+1} \subseteq D_{n+1}$.

From this theorem, it may be seen that the top-down and bottom-up families alternate in a single hierarchy of families of tree languages, $D_0, U_1, D_1, U_2, D_2, \ldots$. The fact that $U_1 = DO(D_0)$ was also observed independently by Engelfriet [3]. Now, Ogden and Rounds conjectured that the families $D_0, D_1, D_2, \ldots$ formed an infinite hierarchy, with $D_n \subsetneq D_{n+1}$ for every $n \geq 0$ [6]. Here we refine this conjecture.

Conjecture 6. For $n \geq 0$, $D_n \subsetneq U_{n+1} \subsetneq D_{n+1}$.

Although we have as yet been unable to prove this conjecture, we present the following related theorem.

Theorem 7. For every $n \geq 0$, if $D_n \subsetneq U_{n+1}$, then $U_{n+1} \not\subseteq D_{n+1}$.

We conjecture that it is also true that for any $n \geq 0$, if $U_n \subsetneq D_n$, then $D_n \subsetneq U_{n+1}$. A proof of this conjecture together with Theorem 7 and the fact that $D_0 \subsetneq U_1$ would prove that the hierarchy is infinite. While the question of whether the hierarchy is infinite remains unsolved, we have refined the result of Ogden and Rounds that $D_0 \subsetneq D_1 \subsetneq D_2$ by showing that $D_0 \subsetneq U_1 \subsetneq D_1 \subsetneq U_2 \subsetneq D_2$.

The above hierarchy is a hierarchy of families of tree languages. There is a related hierarchy of families of string languages, obtained from the tree hierarchy by the yield operation. However, by our definition of trees, the empty word cannot be obtained by the yield operation. It is desirable to let the empty string appear in the families in the hierarchy, since with this addition the families in the string hierarchy become closed under arbitrary homomorphism (Section 5). Therefore, we define an extended yield operation on families of tree languages.

Definition. Let F be a family of tree languages. Define $\text{YIELD}(F) = \{\text{yield}(T), \text{yield}(T) \cup \{e\} \mid T \in F\}$.

Thus, $\text{YIELD}(D_0)$, $\text{YIELD}(U_1)$, $\text{YIELD}(D_1)$, $\text{YIELD}(U_2)$,... form a hierarchy of families of string languages. Again, we conjecture that for $n \geq 0$, $\text{YIELD}(D_n) \subsetneq \text{YIELD}(U_{n+1}) \subsetneq \text{YIELD}(D_{n+1})$, but we do not have a proof of this conjecture.

We are able, however, to produce a bound on the complexity of the languages in the tree hierarchy and the YIELD hierarchy. It is known that every recognizable set is context-free (recall that trees have been defined as a special kind of string ) [1]. Also, the yield of every recognizable set is context-free [8]. Now, both $\text{yield}(D_1)$ and $\text{yield}(U_1)$ contain non-context-free languages [7,9,10]. Thatcher [9] posed the question of whether $\text{yield}(D_1)$ contains only context-sensitive languages. Ogden and Rounds [6] were able to show that for any $n$, $\text{yield}(D_n)$ is recursive. Here we strengthen this result considerably and answer Thatcher's question affirmatively in the following theorem.

Theorem 8. For every $n \geq 0$, $U_n$, $D_n$, $\text{YIELD}(U_n)$, and $\text{YIELD}(D_n)$ are properly contained in the family of context-sensitive languages.

It is natural to ask what the tree families and string families in these two hierarchies are like. In particular, what are the differences between the top-down and the bottom-up families? Sections 4 and 5 investigate the properties of the families in the tree hierarchy and the YIELD hierarchy, respectively.

## 4. Properties of the tree hierarchy

Several properties of the families in the tree hierarchy can be obtained by showing that certain properties of tree families are preserved by tree transduction. The next proposition states that closure of a family of tree languages under linear top-down transductions is preserved by the class of nondeterministic top-down transductions.

Proposition 9. If F is a family of tree languages closed under linear top-down (bottom-up) transductions, then NT(F) is also closed under linear top-down (bottom-up) transductions.

Since the family $D_0$ is closed under linear top-down transductions [9], we obtain the following corollary.

Corollary 10. For $n \geq 0$, $D_n$ is closed under linear top-down transductions and linear bottom-up transductions.

The next group of results is concerned with tree-substitution and string-substitution, which are defined as follows.

Definition. For an alphabet $\Sigma$ and a family of string languages, an *$\mathscr{L}$-string-substitution on $\Sigma$* is a function $\tau: \Sigma \to \mathscr{L}$. The function $\tau$ is extended to $\Sigma^*$ by defining $\tau(e) = e$ and $\tau(a_1 \ldots a_n) = \tau(a_1) \ldots \tau(a_n)$ for $a_1, \ldots, a_n \in \Sigma$, $n \geq 1$. For a language $L \subseteq \Sigma^*$, define $\tau(L) = \bigcup_{w \in L} \tau(w)$. The string-substitution $\tau$ is *nonerasing* if for each $a \in \Sigma$, $e \notin \tau(a)$.

Recall that trees have been defined to be a special class of strings. Suppose that F is a family of tree languages, t is a tree, and $\tau$ is an F-string-substitution which "preserves" symbols of rank greater than one and symbols in $\pi$. In this case, the string language $\tau(t)$ is also a set of trees, and $\tau$ is called a tree substitution. The above conditions are stated formally in the following definition of tree substitution.

Definition. Let F be a family of tree languages, $\Sigma$ a ranked alphabet, and $\tau$ an F-string-substitution on $\Sigma \cup \pi$. If for any symbol $b \in \Sigma_n$, $n \geq 1$, and any symbol $b \in \pi$, $\tau(b) = \{b\}$, then $\tau$ is also an *F-tree-substitution on $\Sigma_0$*.

We say that a family F of string (tree) languages is *closed under string-substitution (tree-substitution)* if for every $L \in F$ and every F-string-substitution (F-tree-substitution) $\tau$, $\tau(L) \in F$. A family F of tree languages is *closed under tree-substitution into the*

recognizable sets if for every recognizable set $L$ and every $F$-tree-substitution $\tau$, $\tau(L) \in F$.

Now, we investigate the extent to which closure under tree substitution is preserved by certain classes of transductions.

Theorem 11. (i) Let $F$ be a family of tree languages which contains the recognizable sets and is closed under tree-substitution. Then $NT(F)$ is also closed under tree-substitution.

(ii) Let $F$ be a family of tree languages which is closed under tree substitution into the recognizable sets and contains the recognizable sets. Then $DT(F)$ is also closed under tree-substitution into the recognizable sets.

(iii) Let $F$ be a family of tree languages which is closed under tree-substitution into the recognizable sets and contains the recognizable sets. Then $DO(F)$ is also closed under tree-substitution into the recognizable sets.

Thatcher and Wright [11] have shown that the family of recognizable sets is closed under an operation of tree "concatenation" which is a restricted form of tree substitution. From this fact it is easy to show that $D_0$ is closed under tree substitution. Thus, we obtain the following corollary.

Corollary 12. For every $n \geq 0$, the family $D_n$ is closed under tree substitution. For every $n \geq 0$, the family $U_n$ is closed under tree substitution into the recognizable sets.

The next theorem was discovered independently by Engelfriet [3] and the author.

Theorem 13. $DB \circ NB \subseteq NB$, that is, the class of nondeterministic bottom-up transductions is closed under composition with deterministic bottom-up transductions.

From the definition of the families $U_n$ as $U_n = NB(U_{n-1})$ and the above theorem, we obtain the following corollary.

Corollary 14. For $n > 0$, $U_n$ is closed under deterministic bottom-up transductions.

Ogden and Rounds have shown that for $n \geq 0$, the family $D_n$ is closed under intersection with the recognizable sets [6]. An argument similar to theirs shows that for $n \geq 0$, the family $U_n$ is closed under intersection with the recognizable sets.

One of the differences between the known closure properties of the top-down families and the known closure properties of the bottom-up families is that for each $n$, $D_n$ is known to be closed under tree substitution, while $U_n$, $n > 0$, is only known to be closed under tree sutstitution into the recognizable sets. We conjecture that in fact, for $n > 0$, $U_n$ is not closed under tree substitution. In particular, this conjecture is equivalent to the conjecture that for $n > 0$, $D_n \subsetneq U_{n+1} \subsetneq D_{n+1}$.

Theorem 15. For every $n \geq 0$, $D_n \subsetneq U_{n+1} \subsetneq D_{n+1}$ if and only if $U_{n+1}$ is not closed under tree substitution.

Further, from our characterization of the families in the hierarchy as $D_n = NLT(U_n)$ and $U_{n+1} = DO(D_n)$, it is obvious that the conjecture that the hierarchy is infinite is equivalent to the conjecture that each $D_n$ is not closed under deterministic one-state transductions and each $U_n$ is not closed under linear top-down transductions.

Proposition 16. For every $n \geq 0$, $D_n \subsetneq U_{n+1}$ if and only if $D_n$ is not closed under deterministic one-state transductions. For every $n > 0$, $U_n \subsetneq D_n$ if and only if $U_n$ is not closed under nondeterministic linear top-down transductions.

## 5. Properties of the YIELD hierarchy

In order to use the closure properties of the families $D_n$ and $U_n$ to obtain information about the closure properties of the families $YIELD(D_n)$ and $YIELD(U_n)$ for $n \geq 0$, we investigate the relationships between certain tree operations and certain string operations. First, we reveal the connection between tree substitution and string substitution under the yield operation.

Definition. If $\mathscr{L}_1$ and $\mathscr{L}_2$ are families of string languages, define $Sub_N(\mathscr{L}_1, \mathscr{L}_2) = \{\tau(L) \mid L \in \mathscr{L}_2$ and $\tau$ a nonerasing $\mathscr{L}_1$-string-substitution$\}$. If $F_1$ and $F_2$ are families of tree languages, define $Sub_T(F_1, F_2) = \{\tau(L) \mid L \in F_2$ and $\tau$ an $F_1$-tree-substitution$\}$.

Theorem 17. Let $F_1$ and $F_2$ be families of tree languages. Then

$$YIELD(sub_T(F_1, F_2)) = Sub_N(YIELD(F_1), YIELD(F_2)).$$

Corollary 18. If a family $F$ of tree languages is closed under tree substitution, then $YIELD(F)$ is closed under nonerasing string substitution. If a family $F$ of tree languages is closed under tree substitution into the recognizable sets, then $YIELD(F)$ is closed under union, concatenation, and Kleene *.

Next, we consider how a bottom-up transducer can perform a homomorphism on the yields of its input trees.

It is easy to see that a nonerasing homomorphism can be performed by a deterministic linear bottom-up transducer which does a local transformation at each leaf. In fact, the ability of bottom-up transducers to delete subtrees enables them to carry out homomorphisms

which erase.

Theorem 19. Let F be a family of tree languages. For any tree language T in F and any homomorphism h, there is a deterministic linear bottom-up transducer M such that h(yield(T)) = yield(M(T)) if e is not in h(yield(T)), and h(yield(T)) = yield(M(T))$\cup${e}, otherwise. Consequently, if F is closed under deterministic linear bottom-up transductions, then YIELD(F) is closed under arbitrary homomorphism.

The last general result relating closure of families of tree languages under tree operations and closure of families of string languages under string operations is the following. Recall that a full abstract family of languages (full AFL) is a family of string languages which is closed under union, concatenation, Kleene *, arbitrary homomorphism, inverse homomorphism, and intersection with regular sets.

Theorem 20. Let F be a family of tree languages which contains the recognizable sets and is closed under deterministic linear bottom-up tree transductions and under tree substitution. Then YIELD(F) is a substitution-closed full AFL. Thus, YIELD(F) is closed under union, concatenation, Kleene *, arbitrary homomorphism, inverse homomorphism, intersection with regular sets, and string substitution.

Finally, the above results can be applied to the families in the tree hierarchy to obtain closure properties of the families in the string hierarchy. In particular, since each $D_n$ contains the recognizable sets and is closed under linear bottom-up transductions and under tree substitution, Theorem 20 is applicable to the the top-down families in the hierarchy.

Theorem 21. For every $n \geq 0$, the family $YIELD(D_n)$ is a substitution-closed full AFL. Thus, $YIELD(D_n)$ is closed under union, concatenation, Kleene *, arbitrary homomorphism, inverse homomorphism, intersection with regular sets, and string substitution.

The fact that for $n \geq 0$, $yield(D_n)$ is closed under intersection with regular sets was known previously [6].

Since each $U_n$, $n > 0$, is closed under deterministic bottom-up transductions and under tree substitution into the recognizable sets, we obtain the following result.

Theorem 22. For every $n \geq 0$, $YIELD(U_n)$ is closed under substitution into the context-free sets, union, concatenation, Kleene *, arbitrary homomorphism, and intersection with regular sets.

It is not known whether for $n > 0$, $YIELD(U_n)$ is closed under inverse homomorphism or under string substitution. If $w \in \Delta^*$ and $h: \Sigma^* \to \Delta^*$ is a homomorphism, then there may be several ways of factoring $w$ as $w = h(a_1)h(a_2)\ldots h(a_n)$ for some $a_1,\ldots,a_n \in \Sigma$. Given a tree language $T_1 \in U_n$, there does not seem to be any intuitive way of obtaining another tree language $T_2 \in U_n$ with $yield(T_2) = h^{-1}(yield(T_1))$, since the copying needed to obtain the overall structure of $T_1$ seems to interfere with obtaining all possible factorings of $yield(T_1)$. Therefore, we conjecture that for $n > 0$, $YIELD(U_n)$ is not closed under inverse homomorphism and that it is therefore not an AFL. A proof of this conjecture would prove that both the tree hierarchies and the string hierarchies are infinite, since for $n \geq 0$, $YIELD(D_n)$ is closed under inverse homomorphism (by Theorem 21).

## 6. Conclusions

Although the classes of top-down and bottom-up transductions have been shown by Engelfriet [3] to be incomparable, we have shown that the two classes are closely related; the composition of n transductions in one direction can always be realized by the composition of n+1 transductions in the other direction. From this fact it was shown that there is a single hierarchy of families of tree languages obtained from the recognizable sets by the composition of top-down or bottom-up transductions. Although we have as yet been unable to prove that the hierarchy is infinite, our conjecture that the inclusion of each family of the hierarchy in the next is proper has been found to be equivalent to conjectures that certain families in the hierarchy are not closed under certain tree operations.

A hierarchy of families of string languages was obtained from the tree hierarchy by an extended yield operation. An open question suggested by Thatcher [9] was answered by showing that each family in the YIELD hierarchy is properly contained in the family of context-sensitive languages. Finally, the top-down and bottom-up families in both the tree hierarchy and the YIELD hierarchy were shown to be closed under a number of operations. Since many of the results in this paper do not depend on any specification of the families of languages involved, but rather on properties of the families, they are also applicable to families other than those in the hierarchies studied here.

## References


1. W. Brainerd, Tree generating regular systems, Info. and Control 14 (1969), 217-231.
2. J. Doner, Tree acceptors and some of their applications, J. of Computer and System Sciences 4 (1979), 406-451.
3. J. Engelfriet, Bottomup and topdown tree transformations--a comparison, tech. report, Technische Hogeschool Twente, Netherlands, July, 1971.
4. S. Ginsburg and E. Spanier, Substitution in families of languages, Information Sciences 2 (1970), 83-110.
5. L. Levy and A. Joshi, Some results in tree automata, Proceedings of Third Annual ACM

Symposium on Theory of Computing (1971), 78-85.

6. W. Ogden and W. Rounds, Compositions of n tree transducers, Proceedings of Fourth Annual ACM Symposium on Theory of Computing (1972), 198-206.

7. W. Rounds, Mappings and grammars on trees, Math. Systems Theory 4 (1970), 257-287.

8. J. Thatcher, Characterizing derivation trees of context-free grammars through a generalization of finite automata theory, JCSS 1 (1967), 317-322.

9. J. Thatcher, Generalized$^2$ sequential machine maps, J. of Computer and System Sciences 4 (1970), 339-367.

10. J. Thatcher, A survey of tree automata, in Currents in the Theory of Computing, (A. Aho, Ed.), Prentice-Hall, 1973.

11. J. Thatcher and J. Wright, Generalized finite automata theory with an application to a decisin problem of second-order logic, Math. Systems Theory 2 (1968), 57-81.