JOURNAL OF COMPUTER AND SYSTEM SCIENCES 25, 418–435 (1982)

The Copying Power of One-State Tree Transducers

JOOST ENGELFRIET

Department of Computer Science, Twente University of Technology, Enschede, The Netherlands

AND

SVEN SKYUM

Department of Computer Science, Aarhus University, Aarhus, Denmark

Received October 1978; revised June 23, 1982

One-state deterministic top-down tree transducers (or, tree homomorphisms) cannot handle "prime copying," i.e., their class of output (string) languages is not closed under the operation $L \to \{\$(w\$)^{f(n)} \mid w \in L,\ f(n) \geqslant 1\}$, where f is any integer function whose range contains numbers with arbitrarily large prime factors (such as a polynomial). The exact amount of nonclosure under these copying operations is established for several classes of input (tree) languages. These results are relevant to the extended definable (or, restricted parallel level) languages, to the syntax-directed translation of context-free languages, and to the tree transducer hierarchy.

1. INTRODUCTION

To measure the copying power of tree transducers, we consider copying operations c_f on languages L defined by $c_f(L) = \{\$(w\$)^{f(n)} \mid w \in L,\ f(n) \geqslant 1\}$, where f is an integer function such as $f(n) = 2$, $f(n) = n$, or $f(n) = 2^n$. Such operations were studied before in relation to parallel rewriting systems and tree transducers in [5, 8, 16], among others. For instance, in [8, 16] it was proved that (nondeterministic) top-down tree transducer languages are not closed under c_2, and in [5] it was shown that bounded copying top-down tree transducers cannot do any infinite copying (i.e., any operation c_f where f has an infinite positive range). In this paper we investigate the copying power of one-state top-down tree transducers and (multi-state) bottom-up tree transducers with respect to these copying operations. Our main result is that deterministic one-state top-down tree transducers (or, tree homomorphisms) cannot do "prime copying," i.e., any operation c_f such that the range of f contains numbers with arbitrarily large prime factors. Although these transducers are clearly able to handle finite copying (c_f, where f has a finite positive range) and "pure exponential" copying (c_f, where $f(n) = ab^n$ for positive integers a, b), it is a consequence of the main result that they can do neither infinite polynomial copying (i.e., any operation c_f such that f is a polynomial with an infinite positive range) nor exponential copying in general (e.g., not c_f with $f(n) = 2^n - 1$). These results are also true for arbitrary (multi-state, nondeterministic) bottom-up tree transducers. To determine the copying power of nondeterministic one-state top-down tree transducers,

0022-0000/82/060418–18$02.00/0

we show that they cannot do finite "coloured" copying, i.e., the operation $\bar{c}_2(L) = \{w\$\bar{w} \mid w \in L\}$.

These results are made concrete in the following two cases: In the first part of the paper (Section 3), we consider the class RPLL of "restricted parallel level" languages, generated by the level grammars of [15], which are a special type of parallel rewriting system. In [15] it is shown that RPLL equals the class of "extended definable" languages of [13]. More important, RPLL equals the class of bottom-up tree transformation languages, i.e., all (string) languages produced by bottom-up tree transducers (or equivalently, tree homomorphisms), when applied to derivation trees of context-free grammars. By means of an intercalation theorem for RPLL, we show that no prime copying can be done in RPLL, in particular no language of the form $c_p(L)$ belongs to RPLL, where L is any infinite language and p is any polynomial with infinite positive range. We note that, as far as tree transducers are concerned, in this part of the paper we only discuss tree homomorphisms.

In the second part of the paper (Sections 4 and 5), we consider the (top-down) tree transducer hierarchy of [5]. We prove in Section 4 that, at each level of this hierarchy, the class of deterministic one-state top-down tree transformation languages is not closed under prime copying (and in particular not under infinite polynomial copying). This result implies that, for every n, the class of (string) languages obtained as output from a composition of n bottom-up tree transducers (applied to derivation trees of context-free grammars) is not closed under infinite polynomial copying. We also show in Section 4 that every class of deterministic (multi-state) top-down tree transformation languages is closed under polynomial copying (under a few conditions on the class of input tree languages). In Section 5, we refine the top-down tree transducer hierarchy by showing that, at each of its levels, nondeterministic one-state top-down tree transducers cannot do coloured copying (clearly multi-state transducers can do this). In this section we also show the result from [20] that one-state top-down tree transducers already give rise to a proper hierarchy (even with respect to the classes of tree transformation languages).

From all these results together we may conclude that infinite polynomial copying and coloured copying distinguish sharply the multi-state from the one-state top-down tree transducers, and the top-down from the bottom-up transducers. For the relevance of (one-state) top-down tree transducers to the syntax-directed translation of context-free languages, we refer to [12, 19].

Sections 2 and 3 of this paper are self-contained, but Sections 4 and 5 are a continuation of the first three sections of [5], and there we assume familiarity of the reader with the notation and terminology of these sections of [5].

2. PRELIMINARIES

The reader is assumed to be familiar with the basic concepts of formal language theory [10, 14]. In this section we discuss some elementary notation, the concept of tree homomorphism, and various copying operations.

2.1. *Elementary Notation*

For a string w, $|w|$ denotes its length; $|\lambda| = 0$. If σ is a symbol, then $\#_\sigma(w)$ denotes the number of occurrences of σ in w. We consider languages which differ only by λ to be identical.

Let $x_1, x_2, x_3, \ldots$ be special symbols. Let $X_k = \{x_1, \ldots, x_k\}$. For strings $w_1, \ldots, w_k$ and a string $w \in (\Sigma \cup X_k)^*$, where Σ is some alphabet, we denote by $w[w_1, \ldots, w_k]$ the result of substituting w_i for x_i in w.

Let $\mathbb{Z}$ denote the set of integers and let $\mathbb{N} = \{1, 2, 3, \ldots\}$. For $A \subseteq \mathbb{Z}$, we denote by ΠA the subset of $\mathbb{Z}$ consisting of all products of elements of A, i.e., $\Pi A = \{k_1 k_2 \cdots k_n \mid n \geqslant 1, k_i \in A\} \subseteq \mathbb{Z}$. For a partial function $f: \mathbb{Z} \to \mathbb{Z}$, we denote as usual by range(f) the set $\{y \in \mathbb{Z} \mid f(x) = y \text{ for some } x \in \mathbb{Z}\}$.

An alphabet Σ is ranked if $\Sigma = \bigcup \{\Sigma_n \mid n \geqslant 0\}$, where the Σ_n are (not necessarily disjoint) subsets of Σ such that only finitely many of them are nonempty. If $\sigma \in \Sigma_n$, then we say that σ has rank n. A tree over Σ is either a symbol of rank 0 or a string of the form $\sigma(t_1 \cdots t_n)$, where σ has rank n and t_i is a tree over Σ, $1 \leqslant i \leqslant n$. The set of all trees over Σ is denoted T_Σ. We shall use $T_\Sigma[X_k]$ to denote $T_{\Sigma \cup X_k}$, where the elements of X_k are given rank 0. Since trees are special strings, $t[t_1, \ldots, t_k]$ denotes the result of substituting the trees t_i for x_i in the tree t; this may be called "tree concatenation" (as it generalizes string concatenation). The yield of a tree t, denoted by yield(t) or $y(t)$ or even yt, is defined as usual by $y(\sigma(t_1 \cdots t_n)) = y(t_1) \cdots y(t_n)$ and $y(\sigma') = \sigma'$ for $\sigma \in \Sigma_n$ and $\sigma' \in \Sigma_0$. It is easy to show that for arbitrary trees $t, t_1, \ldots, t_k$: $y(t[t_1, \ldots, t_k]) = y(t)[y(t_1), \ldots, y(t_k)]$. If L is a tree language, then $yL = \{yt \mid t \in L\}$, and if K is a class of tree languages, then $yK = \{yL \mid L \in K\}$. We also employ the usual intuitive terminology concerning these labeled ordered trees. We assume the reader to be familiar with the concepts of node, label, root, leaf, path, and subtree (a node together with all its descendants).

2.2. *Tree Homomorphisms*

In this subsection we recall the definition of a tree homomorphism and show that it has a property which is basic to the rest of the paper, viz., that the number of copies of the translation made of a subtree by the tree homomorphism has only small prime factors.

Let Σ and Δ be ranked alphabets. A *tree homomorphism* H from T_Σ to T_Δ is a mapping determined by a family $\{H_k\}_{k \geqslant 0}$ of mappings $H_k: \Sigma_k \to T_\Delta[X_k]$ as follows: For $\sigma \in \Sigma_0$, $H(\sigma) = H_0(\sigma)$. For $\sigma \in \Sigma_k$ and $t_1, \ldots, t_k \in T_\Sigma$, $H(\sigma(t_1 \cdots t_k)) = H_k(\sigma)[H(t_1), \ldots, H(t_k)]$.

The class of tree homomorphisms is denoted by HOM. It is easy to see that HOM is closed under composition. If K is a class of tree languages, then $\mathrm{HOM}(K) = \{H(L) \mid H \in \mathrm{HOM} \text{ and } L \in K\}$. Clearly, the tree homomorphism is the generalization to trees of the string concept of homomorphism. Each tree homomorphism H is indeed a homomorphism with respect to tree concatenation, i.e., if $t \in T_\Sigma[X_k]$ and $t_1, \ldots, t_k \in T_\Sigma$ and H is extended (as we will always assume) to $T_\Sigma[X_k]$ by defining

$H_0(x_i) = x_i$, then $H(t[t_1,..., t_k]) = H(t)[H(t_1),..., H(t_k)]$. The straightforward proof of this elementary fact is left to the reader.

Let s be a subtree of t and let $t = u[s]$, where $u \in T_\Sigma[X_1]$ and u contains exactly one occurrence of x_1 (thus u is the part of t "outside" s). Then, for a tree homomorphism H, $H(t) = H(u)[H(s)]$ and consequently the number of occurrences of $H(s)$ in $H(t)$ is equal to the number of occurrences of x_1 in $H(u)$. This number will be called the *translation number* of s with respect to t (and H), denoted by $\mathrm{trn}_H(s, t)$.

In the following lemma we state the basic property of these translation numbers:

LEMMA 2.1. *For each* $H \in \mathrm{HOM}$ *there is a constant* N *such that if* s *is a subtree of* t, *then* $\mathrm{trn}_H(s, t) \in \Pi\{n \mid 0 \leqslant n \leqslant N\}$.

Proof. It is easy to see that for $t = \sigma(t_1 \cdots t_k)$

$$\begin{aligned} \mathrm{trn}_H(s, t) &= 1, && \text{if} \quad s = t, \\ &= \#_{x_i}(H_k(\sigma)) \cdot \mathrm{trn}_H(s, t_i), && \text{if} \quad s \text{ is a subtree of } t_i. \end{aligned}$$

Hence $\mathrm{trn}_H(s, t)$ is a product of numbers $\#_{x_i}(H_k(\sigma))$, $k \geqslant 1$, $\sigma \in \Sigma_k$, $0 \leqslant i \leqslant k$. Consequently, if N is the maximum of these numbers, then $\mathrm{trn}_H(s, t) \in \Pi\{n \mid 0 \leqslant n \leqslant N\}$. ∎

As a consequence of this lemma, the numbers $\mathrm{trn}_H(s, t)$, for fixed H, have only small (i.e., $\leqslant N$) prime factors, viz., the prime factors of all numbers $\#_{x_i}(H_k(\sigma))$.

2.3. *Copying Operations*

Let f be a function from $\mathbb{Z}$ into itself. For each language L over the alphabet Σ we define

$$c_f(L) = \{\$(w\$)^{f(n)} \mid f(n) \geqslant 1 \text{ and } w \in L\},$$

where \$ is not in Σ.

If $f(n) = k$ for all n, then we denote c_f by c_k; if f is the identity, then we denote c_f by c_*. Thus $c_k(L) = \{\$(w\$)^k \mid w \in L\}$ and $c_*(L) = \{\$(w\$)^n \mid n \geqslant 1,\ w \in L\}$. When $\mathrm{range}(f) \cap \mathbb{N}$ is finite (infinite), as in the case of c_k (c_*), we talk about finite (infinite) copying. If $f(n) = ab^n$ for $a, b \in \mathbb{N}$, then we talk about pure exponential copying. If f is a polynomial (with integer coefficients), then we talk about polynomial copying. Finally, if $\mathrm{range}(f) \cap \mathbb{N}$ contains numbers with arbitrarily large prime factors, then we talk about *prime copying*.

We also consider finite "coloured" copying, i.e., copying over a different alphabet. For each language L over the alphabet Σ we define $\bar{c}_2(L) = \{w\$\bar{w} \mid w \in L\}$, where $\bar{\Sigma} = \{\bar{\sigma} \mid \sigma \in \Sigma\}$ and \$ is not in $\Sigma \cup \bar{\Sigma}$.

3. AN INTERCALATION THEOREM FOR LEVEL LANGUAGES

In this section we consider the extended definable languages of [13], using the (restricted parallel) level grammars of [15] to generate them. This class of languages

will be denoted by RPLL. We first show that RPLL can be obtained from the context-free languages by applying tree homomorphisms to their derivation trees (as shown also in [12, 18]), and consequently RPLL also equals the class of bottom-up tree transformation languages (cf. [3]). We consider a few closure properties of RPLL, in particular finite and pure exponential copying. Then an intercalation theorem for RPLL is presented and it is used to show that RPLL is not closed under prime copying, in particular not under infinite polynomial copying. This result implies that RPLL is not adequate to express the properties of declarations in block-structured languages (as suggested in [13]).

DEFINITION 3.1. A *restricted level grammar* is a construct $G = (N, \Sigma, P, S)$, where N is the nonterminal alphabet, Σ is the terminal alphabet ($\Sigma \cap N = \emptyset$), P is a finite set of productions of the form $A \to v$ with $A \in N$ and $v \in (N \cup \Sigma)^*$, and $S \in N$ is the start symbol. In other words, a restricted level grammar is the same as a context-free grammar. The derivation relation, however, is defined differently. A sentential form of G is an element of $((N \times \mathbb{N}) \cup \Sigma)^*$. We define the derivation relation $\Rightarrow$ as follows (where $w_i \in ((N \times \mathbb{N}) \cup \Sigma)^*$ and $(A, n) \in N \times \mathbb{N}$): $w_0(A, n)\, w_1(A, n)\, w_2 \cdots (A, n)\, w_m \Rightarrow w_m w w_1 w w_2 \cdots w w_m$ if and only if (A, n) does not occur in w_i ($0 \leqslant i \leqslant m$) and there is a production $A \to y_0 A_1 y_1 A_2 y_2 \cdots A_k y_k$ in P with $A_i \in N$ and $y_i \in \Sigma^*$ such that $w = y_0(A_1, n+1)\, y_1(A_2, n+1)\, y_2 \cdots (A_k, n+1)\, y_k$. The relation $\overset{*}{\Rightarrow}$ is defined as usual, and the *language generated* by G is $L(G) = \{y \in \Sigma^* \mid (S, 1) \overset{*}{\Rightarrow} y\}$.

The class of languages generated by restricted level grammars is denoted by RPLL (the restricted parallel level languages). Level grammars were introduced in [15], in a slightly different way; using [15, Proposition 3], it is easy to see that the above definition is equivalent to the one in [15]. It was shown in [15] that RPLL equals the class ED of extended definable languages [13].

A restricted level grammar is a context-free grammar with level-numbers attached to the nonterminals (indicating the depth of the nonterminal in the derivation tree). The way in which the derivation relation is defined restricts the set of derivation trees of the context-free grammar to those that have the following property (let the "label sequence" of a node be the sequence of labels on the path from the root to the node): if two nodes have the same label sequence, then they are at the root of identical subtrees of the derivation tree; in other words, if two nodes have the same history, then they have the same future. In particular (and equivalently), if $A \to uBvBw$ is a production (with $A, B \in N$ and $u, v, w \in (\Sigma \cup N)^*$), then both occurrences of B have to derive the same string; it is exactly this property of level grammars which is formalized in an "Algol-like" way in the definition of extended definable language in [13], cf. the discussion at the end of [13]. A formal proof of these remarks is left to the reader.

EXAMPLES 3.2. Consider the restricted level grammar $G = (N, \Sigma, P, S)$ with $N = \{S, A\}$, $\Sigma = \{a, b\}$, and $P = \{S \to bAbAbAb,\ A \to aA,\ A \to \lambda\}$. It is easy to see that $L(G) = \{ba^n ba^n ba^n b \mid n \geqslant 0\}$.

The restricted level grammar $(\{S\}, \{a\}, P, S)$, where P consists of the rules $S \to SS$ and $S \to a$, generates the language $\{a^{2^n} \mid n \geqslant 0\}$.

In Lemma 3.3 we show how to remove λ-productions and single productions from level grammars.

LEMMA 3.3. *Let $G = (N, \Sigma, P, S)$ be a restricted level grammar. Then there exists an equivalent restricted level grammar $G' = (N, \Sigma, P', S)$ such that P' contains no productions of the form $A \to \lambda$ or $A \to B$ with $A, B \in N$.*

Proof. First we construct a grammar without λ-productions. The construction is a straightforward variant of the one for context-free grammars. Let $N_\lambda = \{A \in N \mid A \overset{*}{\Rightarrow} \lambda\}$. Construct $\bar{G} = (N, \Sigma, \bar{P}, S)$ such that if $A \to w_0 A_1 w_1 A_2 \cdots A_m w_m$ is in P, with $w_0 w_1 \cdots w_m \in ((N \cup \Sigma) - N_\lambda)^*$ and $A_i \in N_\lambda$ $(1 \leqslant i \leqslant m)$, then all productions $A \to w_0 B_1 w_1 B_2 \cdots B_m w_m$, except possibly $A \to \lambda$, are in $\bar{P}$, where $B_i \in \{A_i, \lambda\}$ and $B_i = B_j$ if $A_i = A_j$. It is left to the reader to show that $L(G) = L(\bar{G})$.

To obtain from $\bar{G}$ the required grammar G' without productions $A \to B$, exactly the same construction as for context-free grammars can be used: if $A \overset{*}{\Rightarrow} B$ in $\bar{G}$ and $B \to w$ in $\bar{P}$ (with $w \notin N$), then $A \to w$ is in P'. ∎

Let DCF denote the class of tree languages which are sets of derivation trees of a context-free grammar. We now show that the application of tree homomorphisms to DCF yields RPLL, cf. [12, 18].

THEOREM 3.4 [12, 18]. $\mathrm{RPLL} = y\mathrm{HOM}(\mathrm{DCF})$.

Proof. To show that $\mathrm{RPLL} \subseteq y\mathrm{HOM}(\mathrm{DCF})$, let $G = (N, \Sigma, P, S)$ be a restricted level grammar, which has no λ-productions. We define a context-free grammar $G' = (N', \Sigma', P', S')$ and a tree homomorphism H as follows: Let Δ be the ranked alphabet equal to $P \cup \{S'\}$, such that a production is in Δ_k if its right-hand side contains k different nonterminals (e.g., a production $A \to ABA$ has rank 2) and $S' \in \Delta_1$. Let Ω be the ranked alphabet with $\Omega_0 = \Sigma$ and $\Omega_2 = \{c\}$ for some new symbol c. Homomorphism H is a tree homomorphism from T_Δ to T_Ω, and we set $N' = \Delta - \Delta_0$ and $\Sigma' = \Delta_0$. If r is a production and the (distinct) nonterminals $A_1, \ldots, A_k$ $(k \geqslant 1)$ occur in its right-hand side v, then P' contains all rules $r \to r_1 \cdots r_k$, where r_i is a production with left-hand side A_i, and $H_k(r)$ is defined to be any tree $t \in T_\Omega(X_k)$ such that $\mathrm{yield}(t)[A_1, \ldots, A_k] = v$. Finally, P' contains all rules $S' \to r$, where r has left-hand side S, and $H_1(S') = x_1$. It is left to the reader to prove that $L(G) = yH(D)$, where D is the set of derivation trees of G'. Intuitively, whenever the restricted level grammar produces two brother nodes with the same label, the context-free grammar produces only one of them, and the tree homomorphism is used to produce its brother together with a copy of its subtree.

To show that $y\mathrm{HOM}(\mathrm{DCF}) \subseteq \mathrm{RPLL}$, let $G = (N, \Sigma, P, S)$ be a context-free grammar and let H be a tree homomorphism from T_Δ to T_Ω, where Δ is the ranked alphabet such that $\Delta_0 = \Sigma$ and Δ_k is the set of all nonterminals which have a production with right-hand side of length k. We may assume that no nonterminal

occurs more than once in the right-hand side of a production (if $A \to uBvBw$ is a production, then we introduce a new nonterminal $\bar{B}$ and a new production $\bar{B} \to B$, change the first production into $A \to uBv\bar{B}w$, and define $H_1(\bar{B}) = x_1$). We now define the restricted level grammar $G' = (N, \Omega_0, P', S)$ such that if $A \to \alpha_1 \cdots \alpha_k$ is a production in P with $\alpha_i \in N \cup \Sigma$, and $H_k(A) = t \in T_\Omega[X_k]$, then P' contains the production $A \to \text{yield}(t)[w_1, ..., w_k]$, where $w_i = \alpha_i$ if $\alpha_i \in N$ and $w_i = \text{yield}(H_0(\alpha_i))$ if $\alpha_i \in \Sigma$. It is again left to the reader to show that $L(G') = yH(D)$, where D is the set of derivation trees of G. ∎

Let REC denote the class of recognizable tree languages [19]. It is well known that DCF $\subseteq$ REC and that each recognizable tree language can be obtained as a (deterministic) relabeling of a tree language in DCF [19]. Since HOM is closed under composition, this shows that yHOM(DCF) = yHOM(REC). Let B denote the class of bottom-up tree transductions; yHOM(REC) equals the class yB(REC) of bottom-up tree transformation languages [3]. Thus we obtain

COROLLARY 3.5. RPLL = yHOM(REC) = yB(REC).

It follows from Corollary 3.5 (in particular RPLL = yB(REC)) that RPLL is closed under intersection with regular languages, cf. [3] (this was not noticed in [13]). Of the other AFL operations, it was shown in [13] that RPLL is closed under union, concatenation, Kleene star, and homomorphisms, but not under inverse homomorphisms (or finite substitutions). The class RPLL is clearly closed under finite copying [13] and pure exponential copying: if L is generated by the restricted level grammar $G = (N, \Sigma, P, S)$, then the grammar with new start symbol S' and additional production $S' \to \$S\$S\$$ generates $c_2(L)$, and the grammar with additional productions $S' \to \$T\$T\$T\$$, $T \to T\$T$ and $T \to S$ (where T is also new) generates $c_f(L)$ with $f(n) = 3 \cdot 2^n$ (and similarly for constants other than 2 and 3).

To show that RPLL is not closed under prime copying, we now prove an intercalation theorem for RPLL.

THEOREM 3.6. *Let L be a restricted parallel level language, i.e., $L \in$ RPLL, over the alphabet Σ. Then there exist positive integers M and N such that if $w \in L$ and $|w| > M$, then $w = w_0 v w_1 v w_2 \cdots v w_r$ (with $r \geqslant 1$ and $w_i, v \in \Sigma^*$) and the following hold:*

(1) $r \in \Pi\{p \mid 1 \leqslant p \leqslant N\}$;

(2) $0 < |v| \leqslant M$;

(3) $v = v_0 u v_1 u v_2 \cdots u v_s$ *(with $s \geqslant 1$ and $v_i, u \in \Sigma^*$) such that*

(a) $0 < |u| < |v|$ *and*

(b) *if y_n is defined for all $n \geqslant 0$ by $y_0 = u$ and $y_{n+1} = v_0 y_n v_1 y_n v_2 \cdots y_n v_s$, then for all $n \geqslant 0$*

$w_0 y_n w_1 y_n w_2 \cdots y_n w_r \in L$.

Proof. Intuitively, the theorem says that in each sufficiently large string of L one can find r nonoverlapping occurrences of a small substring such that r has only small prime factors and the substring can be "pumped" (in a special way) without leaving the language. Note that it follows from Condition (3)(a) that $|y_{n+1}| > |y_n|$ for all $n \geqslant 0$.

By Theorem 3.4 there exist a context-free grammar $G = (N, \Sigma, P, S)$ and a tree homomorphism H such that $L = yH(D)$, where D is the set of derivation trees of G. Moreover, by Lemma 3.3 and the construction used in Theorem 3.4 to show that RPLL $\subseteq y$HOM(DCF), we may assume that H is nondeleting and expanding, i.e., $H_k(\sigma)$ contains at least one occurrence of each x_i $(1 \leqslant i \leqslant k)$ and $yH_1(\sigma) \neq x_1$; consequently $|yH_k(\sigma)| \geqslant 2$. We will now apply H to the usual pumping lemma for context-free languages.

Let M be the maximum of all $|yH(t)|$, where t is any subtree of a tree in D and t has no path on which there are more than two occurrences of the same nonterminal. Consider a string $w \in L$ with $|w| > M$. Then $w = yH(t)$ for some tree $t \in D$ which has a path with a repetition of a nonterminal. Consider a lowest pair of nodes (on the same path) with the same label. Then (using x for x_1) there are trees $\bar{u} \in T_\Delta$ and $\bar{v}, \bar{w} \in T_\Delta[\{x\}]$, where Δ is the ranked alphabet of D, such that $t = \bar{w}[\bar{v}[\bar{u}]]$, $\bar{w}$ and $\bar{v}$ contain exactly one occurrence of x, and the roots of $\bar{v}[\bar{u}]$ and $\bar{u}$ constitute the above pair of nodes. Thus $|yH(\bar{v}[\bar{u}])| \leqslant M$, and $\bar{w}[\bar{v}^n[\bar{u}]] \in D$, where $\bar{v}^0[\bar{u}] = \bar{u}$ and $\bar{v}^{n+1}[\bar{u}] = \bar{v}[\bar{v}^n[\bar{u}]]$. Let $yH(\bar{w}) = w_0 x w_1 x w_2 \cdots x w_r$, $yH(\bar{v}) = v_0 x v_1 x v_2 \cdots x v_s$, $yH(\bar{u}) = u$, and $yH(\bar{v}[\bar{u}]) = v$. Then $v = yH(\bar{v})[yH(\bar{u})] = v_0 u v_1 u v_2 \cdots u v_s$ and $w = yH(\bar{w})[yH(\bar{v}[\bar{u}])] = w_0 v w_1 v w_2 \cdots v w_r$. Since H is nondeleting, both r and s are $\geqslant 1$. Let N be the number associated with H by Lemma 2.1. Then, by Lemma 2.1, $r = \mathrm{trn}_H(\bar{v}[\bar{u}], t) \in \Pi\{p \mid 0 \leqslant p \leqslant N\}$ and so $r \in \Pi\{p \mid 1 \leqslant p \leqslant N\}$. Clearly $|v| \leqslant M$ and both u and v are nonempty. From the fact that H is nondeleting and expanding, it follows that $yH(\bar{v}) \neq x$ and hence $|u| < |v|$. Finally, $yH(\bar{v}^n[\bar{u}]) = yH(\bar{v})^n[yH(\bar{u})] = y_n$ for all $n \geqslant 0$, and so $yH(\bar{w}[\bar{v}^n[\bar{u}]]) = yH(\bar{w})[y_n] = w_0 y_n w_1 y_n w_2 \cdots y_n w_r$ is in L for all $n \geqslant 0$. This proves the theorem. ∎

We now show that RPLL cannot handle prime copying of infinite languages.

THEOREM 3.7. *Let $L_0 \subseteq c_*(L)$, where L is an infinite language. If for each $z \in L$ and $k \in \mathbb{N}$ there exists $\$(z\$)^n \in L_0$ such that n has a prime factor $> k$, then $L_0 \notin$ RPLL.*

Proof. Assume that $L_0 \in$ RPLL and let M and N be the constants of the intercalation theorem for L_0. Consider a string $w = \$(z\$)^m \in L_0$ with $z \in L$ such that $|z| \geqslant M^2 + M$ and m has a prime factor $> \max\{N, M^2\}$. We will use Theorem 3.6 to derive a contradiction. By Theorem 3.6, $\$(z\$)^m = w_0 v w_1 v w_2 \cdots v w_r$ such that (1) – (3) hold. In particular $v = v_0 u v_1 u v_2 \cdots u v_s$ with $0 < |u| < |v| \leqslant M$. Since $|v| \leqslant M$ and $|z| > M$, v contains either zero or one \$. Let $w' = w_0 y_2 w_1 y_2 w_2 \cdots y_2 w_r = \$(z'\$)^{m'} \in L_0$ for some z' and m', where $y_2 = v_0 v v_1 v v_2 \cdots v v_s$. Note that $|v| < |y_2| \leqslant M^2$. We now consider two cases.

Case 1. $\#_{\$}(y_2) = \#_{\$}(v)$. Then $m' = m$. Compute the difference in length between w' and w: $|w'| - |w| = r(|y_2| - |v|) = m(|z'| - |z|)$. Since $|y_2| - |v| \leqslant M^2$ and m has a prime factor larger than both M^2 and N, it follows from the last equality that r has a prime factor larger than N, which contradicts Theorem 3.6(1).

Case 2. (Note that $\#_{\$}(v) = 0$ iff $\#_{\$}(y_2) = 0$.) Here $\#_{\$}(v) = 1$ and $\#_{\$}(y_2) \geqslant 2$. Since $\#_{\$}(y_2) \geqslant 2$, $|z'| \leqslant |y_2| \leqslant M^2$. Hence $z' \neq z$ and so $\#_{\$}(w_i) \leqslant 1$ for all i, $0 \leqslant i \leqslant r$. If $\#_{\$}(w_0) = \#_{\$}(w_r) = 0$, then $w_0 = w_r = \lambda$ and so $v = \$$, which contradicts Condition (3)(a) of Theorem 3.6. Hence either $\#_{\$}(w_0) = 1$ or $\#_{\$}(w_r) = 1$. Assume that $\#_{\$}(w_0) = 1$ (and the other case is symmetric). Then w_0 begins with \$. Since $|z| \geqslant M^2 + M$, $|v| \leqslant M$, and $w_0 v$ contains two occurrences of \$, we have that $|w_0| = |w_0 v| - |v| \geqslant |z| + 2 - M \geqslant M^2 + 2$. Consideration of w' then shows that $|z'| \geqslant |w_0| - 1 \geqslant M^2 + 1$, which contradicts $|z'| \leqslant M^2$.

This proves the theorem. ∎

The most intuitively obvious case is in fact the subcase of Case 1 where $\#_{\$}(v) = 0$. In that situation, it is clear that v has to occur the same number of times, say k, in each occurrence of z in w. Hence $r = km$ and so r would have a large prime factor. The case $\#_{\$}(v) = 1$ has been handled above by length arguments.

This theorem implies that $c_f(L)$ is not in RPLL for every infinite language and every function f such that range$(f) \cap \mathbb{N}$ contains numbers with arbitrarily large prime factors. Obvious examples of such functions are $f(n) = n$, $f(n)$ is the nth prime, and $f(n)$ is the nth nonprime. A more surprising example is $f(n) = 2^n - 1$: by Fermat's theorem, p is a factor of $2^{p-1} - 1$ for each prime p, see, e.g., [2]. This example shows that RPLL can handle only pure exponential copying (e.g., $f(n) = 2^n$), but not exponential copying in general. We now consider a large class of functions having the "prime property," viz., all infinite polynomials.

LEMMA 3.8. *Let f be a polynomial with integer coefficients. If the set* range$(f) \cap \mathbb{N}$ *is infinite, then it contains numbers with arbitrarily large prime factors.*

Proof. For some $k \in \mathbb{Z}$ and some polynomial g with integer coefficients, $f(n) = n \cdot g(n) + k$. If $k = 0$, the result is clearly true. If k is positive, then consider the polynomial $f'(n) = n \cdot g(kn) + 1$. If range$(f') \cap \mathbb{N}$ contains numbers with arbitrarily large prime factors, then so does range$(f) \cap \mathbb{N}$ (because $f(kn) = kng(kn) + k = k(ng(kn) + 1) = k \cdot f'(n)$ and hence $\{k \cdot f'(n) \mid f'(n) \in \mathbb{N}\} \subseteq$ range$(f) \cap \mathbb{N}$). If k is negative, then the same statement is true for $f'(n) = n \cdot g(-kn) - 1$: $f(-kn) = -k \cdot f'(n)$.

From the above argument it follows that it suffices to prove the lemma for polynomials of the form $f(n) = n \cdot g(n) \pm 1$. Assume now that $p_1, \ldots, p_s$ are all the prime factors of numbers in range$(f) \cap \mathbb{N}$. For any $m \in \mathbb{Z}$, take $n = m \cdot p_1 \cdot p_2 \cdots p_s$ and consider $f(n) = n \cdot g(n) \pm 1$. Then, if $f(n) \in \mathbb{N}$, the assumption implies that there is some p_i which divides both $f(n)$ and $n \cdot g(n)$. This is a contradiction. Note that, depending on the sign of the leading coefficient of f, $\lim_{n \to \infty} f(n) = +\infty$ or

$\lim_{n \to -\infty} f(n) = +\infty$. Hence taking m sufficiently large (positive or negative, respectively) will result in a positive $f(n)$. ∎

This shows that RPLL cannot handle infinite polynomial copying.

COROLLARY 3.9. *Let L be an infinite language. If p is a polynomial with integer coefficients such that* $\text{range}(p) \cap \mathbb{N}$ *is infinite, then* $c_p(L) \notin \text{RPLL}$. *In particular* $c_*(L) \notin \text{RPLL}$.

Proof. This result follows directly from Theorem 3.7 and Lemma 3.8. ∎

Thus, e.g., the language $\{\$(a^m\$)^{2n+1} \mid m \geqslant 1,\ n \geqslant 0\}$ is not in RPLL. From this it follows that the properties of declarations in a block-structured Algol-like language cannot be expressed in RPLL. Suppose that such a language is in RPLL. Then the intersection of this language with the regular language **begin integer** $a^*(; a^* := a^* + 1)^*$ **end** is also in RPLL. But this is the language {**begin integer** $a^m(; a^m := a^m + 1)^n$ **end** $\mid m \geqslant 1,\ n \geqslant 0\}$ because all identifiers have to be declared. Now application of the homomorphism h with $h(\textbf{begin}) = \lambda$, $h(\textbf{integer}) = \$$, $h(;) = \$$, $h(:=) = \$$, $h(+) = \lambda$, $h(1) = \lambda$ and $h(\textbf{end}) = \$$ yields the language $\{\$(a^m\$)^{2n+1} \mid m \geqslant 1,\ n \geqslant 0\} = c_p(a^+)$ with $p(n) = 2n + 1$. By Corollary 3.9 this language is not in RPLL. This is a contradiction.

We finally note that the infinity requirement on L in Corollary 3.9 is essential: if L is finite then $c_*(L)$ is regular and hence in RPLL.

4. Polynomial Copying in the Tree Transducer Hierarchy

In this section and the next, the reader is assumed to be familiar with the notation and terminology of the first three sections of [5]. With respect to the previous sections, we wish to remark the following: In [5] there is a special symbol e of rank 0 with $\text{yield}(e) = \lambda$; although we do not have such a symbol here, addition of it would make no difference with respect to all classes of languages discussed. We will use $c_{\exp}$ to denote c_f with $f(n) = 2^n$. The operations c_2, c_*, and $c_{\exp}$ are slightly different in [5], but that has no influence on our results. A tree homomorphism is the same as a one-state deterministic top-down tree transducer. The translation number of a subtree (with respect to some tree homomorphism) is the length of its derivation sequence. The intercalation theorem of Section 3 may be viewed as the one-state case of the intercalation theorem for deterministic top-down tree transducers of [11] as expressed in [6, Theorem 3.2.4], with the additional statement concerning the prime factors of the number of occurrences of the substring to be pumped (using a finite state relabeling, it is easy to see that $yDT_{(1)}(\text{REC}) = y\text{HOM}(\text{REC})$, in the notation of [6]). We recall that a tree trio is a class of tree languages closed under finite state relabeling (i.e., finite tree automata which relabel the nodes of the tree) and regular insertion (i.e., the insertion of a regular monadic language above each node of the tree).

In this section we investigate the class $y\mathrm{HOM}(K)$ for classes K of input tree languages other than REC. For a tree trio K, $y\mathrm{HOM}(K)$ is closed under finite and pure exponential copying ([5, Lemma 2.6]). Our main result in this section is that if $K = T(K')$ for some K' and $yDT(K') \subsetneq yT(K')$, then $y\mathrm{HOM}(K)$ is not closed under prime copying, and hence not closed under infinite polynomial copying (whereas $yDT(K)$ is). This holds in particular for $y\mathrm{HOM}(T^n(\mathrm{REC})) = yB^{n+1}(\mathrm{REC})$, where B is the class of bottom-up tree transductions [1, 3], and consequently bottom-up tree transducers cannot do infinite polynomial copying (note that the case $n = 0$ was treated in Section 3).

The main result of this section is stated in the next theorem.

THEOREM 4.1. *Let K be a tree trio and L a language. Let $L_0 \subseteq c_*(L)$ have the property that for each $w \in L$ and $k \in \mathbb{N}$ there exists $\$(w\$)^n \in L_0$ such that n has a prime factor $>k$. If $L_0 \in y\mathrm{HOM}(T(K))$, then $L \in yDT(K)$.*

Proof. The proof is similar to that of [5, Theorem 3.9]. Let $L_0 = yH(M(L_1))$ with $L_1 \in K$, $M \in T$, and $H \in \mathrm{HOM}$. We will construct $M' \in T$ such that $yH(M'_{\mathrm{un}}(L_1))$ contains some $\$(w\$)^n$ for each $w \in L$. Recall that "un" means that only uniform derivations are considered (see the definition of *uniform* in [5]). By [5, Lemma 3.5] and the closure of $DT(K)$ under HOM [4], this implies that $yH(M'_{\mathrm{un}}(L_1)) \in yDT(K)$ and hence, since $yDT(K)$ is closed under deterministic gsm mappings [4], $L \in yDT(K)$ which will complete the proof.

As in the proofs of [5, Theorems 3.9 and 3.16], M' is equivalent to M but keeps the following information in its finite control: when arriving at the root of an input subtree s in state q, M' will predict whether, for the tree s' such that $q(s) \overset{*}{\Rightarrow}_M s'$, $yH(s')$ contains 0, 1, or $\geqslant 2$ occurrences of \$. Since this property of s' is recognizable, M' can do this by simulating a top-down finite tree automaton on its output tree [5, Lemma 2.2]. It is left to the reader to provide the details. The states of M' are of the form $\langle q, d\rangle$, where q is a state of M and $d = 0$, 1, or 2 with the above meaning. Thus $L_0 = yH(M'(L_1))$. We now want to show that $yH(M'_{\mathrm{un}}(L_1))$ contains some $\$(w\$)^n$ for each $w \in L$.

Consider $\$(w\$)^n$ in L_0 such that n has a prime factor larger than N, where N is the number associated with H by Lemma 2.1. Let $q_0(t_1) \overset{*}{\Rightarrow} t_2$ be a derivation in M' such that $yH(t_2) = \$(w\$)^n$. We want to change this derivation into a uniform derivation $q_0(t_1) \overset{*}{\Rightarrow} t'_2$ such that $yH(t'_2) = \$(w\$)^{n_1}$ for some n_1. Let s be an arbitrary subtree of t_1 and let $\langle q, d\rangle$ occur in the state sequence of s. Let us try to make the derivation uniform (at the root of s) with respect to this state $\langle q, d\rangle$ (i.e., we want all rules starting with $\langle q, d\rangle$ in the rule sequence of s to have the same right-hand side). Let $\langle q, d\rangle(s) \overset{*}{\Rightarrow} s_1$ and $\langle q, d\rangle(s) \overset{*}{\Rightarrow} s_2$ occur in the derivation sequence of s. We want to replace one of these derivations by the other. Note that if either $yH(s_1)$ or $yH(s_2)$ does not occur in $yH(t_2)$, i.e., $\mathrm{trn}_H(s_1, t_2) = 0$ or $\mathrm{trn}_H(s_2, t_2) = 0$, then this replacement can easily be done. Assuming that this is not so, we consider three cases.

Case 1. ($d = 0$, i.e., $yH(s_1)$ and $yH(s_2)$ do not contain \$). All occurrences of $yH(s_1)$ in $\$(w\$)^n$ can be replaced by $yH(s_2)$ without leaving the language L_0. Assume

that $yH(s_1) \neq yH(s_2)$. Then $yH(s_1)$ has to occur the same number of times, say k, in each occurrence of w. Hence $yH(s_1)$ occurs kn times in $yH(t_2)$. By Lemma 2.1, $kn \in \Pi\{p \mid 0 \leqslant p \leqslant N\}$. This contradicts the fact that n has a prime factor $>N$. Hence $yH(s_1) = yH(s_2)$. Consequently the derivation $\langle q, d\rangle(s) \overset{*}{\Rightarrow} s_1$ can be replaced by $\langle q, d\rangle(s) \overset{*}{\Rightarrow} s_2$ without changing $yH(t_2) = \$(w\$)^n$.

Case 2 ($d = 1$, i.e., $yH(s_1)$ and $yH(s_2)$ contain exactly one occurrence of \$). Replacement of $yH(s_1)$ by $yH(s_2)$ results in a string with the same number of occurrences of \$ but a possibly different w. Suppose that there is an occurrence of $\$w\$$ which does not overlap with the occurrences of $yH(s_1)$. Then the resulting string is again $\$(w\$)^n$, and hence $yH(s_1) = yH(s_2)$; similarly for $yH(s_2)$. Now suppose that $yH(s_1)$ overlaps with all occurrences of $\$w\$$, and similarly for $yH(s_2)$. Then $yH(s_1)$ contains the first \$ in $\$(w\$)^n$ and $yH(s_2)$ the second (or vice versa). Thus $yH(s_1) \in \$\Sigma^*$ (where Σ is the alphabet of L) and so also $yH(s_2) \in \$\Sigma^*$. But then clearly $yH(s_1) = yH(s_2)$. Hence in this case we also have $yH(s_1) = yH(s_2)$ and $\langle q, d\rangle(s) \overset{*}{\Rightarrow} s_1$ can be replaced by $\langle q, d\rangle(s) \overset{*}{\Rightarrow} s_2$ without changing $yH(t_2) = \$(w\$)^n$.

Case 3 ($d = 2$, i.e., both $yH(s_1)$ and $yH(s_2)$ contain at least two occurrences of \$). Then $yH(s_2)$ contains an occurrence of $\$w\$$ and hence replacement of $yH(s_1)$ by $yH(s_2)$ in $\$(w\$)^n$ yields a string $\$(w\$)^{n_1}$ for some n_1.

We now know how to make the derivation uniform (at a node) with respect to one state. Let us try to make the whole derivation uniform. We cannot do this by making the derivation uniform at the root and then at its sons, et cetera (as discussed in the proof of [5, Theorem 3.9]), because in Case 3 we change n, whereas Case 1 depends on the fact that n has a large prime factor. Instead we proceed as follows: First, we make the derivation uniform with respect to all states $\langle q, d\rangle$ with $d = 0$ or 1 by processing all nodes of t_1 in the usual top-down fashion. This does not change $\$(w\$)^n$, cf. Cases 1 and 2. Then we make the resulting derivation uniform with respect to all states $\langle q, 2\rangle$, also in the usual top-down fashion (note that Case 3 does not depend on n having a large prime factor). This changes $\$(w\$)^n$ into some $\$(w\$)^{n_1}$. Although this process also changes rule sequences of nodes below the one processed, it is easy to see that such a change to a rule sequence does not introduce new rules, it only replaces certain subderivations by others. Hence the final derivation is still uniform with respect to states $\langle q, 0\rangle$ and $\langle q, 1\rangle$, and consequently it is uniform. ∎

COROLLARY 4.2. *Let K be a tree trio such that $yDT(K) \subsetneq yT(K)$. If p is a polynomial with integer coefficients such that* range$(p) \cap \mathbb{N}$ *is infinite, then* $y\mathrm{HOM}(T(K))$ *is not closed under* c_p *(in particular, not under* c_**).*

Proof. Let $L \in yT(K) - yDT(K)$. Clearly $L \in y\mathrm{HOM}(T(K))$. By Lemma 3.8, $L_0 = c_p(L)$ satisfies the conditions of Theorem 4.1. Hence, by Theorem 4.1, $c_p(L) \notin y\mathrm{HOM}(T(K))$. ∎

By [5, Theorems 3.12 and 3.14], this corollary holds in particular for

$yHOM(T^n(REC)) = yB^{n+1}(REC)$, $n \geqslant 1$, and hence (including the results of Section 3) for $n \geqslant 1$, $yB^n(REC)$ is not closed under infinite polynomial copying.

We now show that deterministic top-down tree transducers can do polynomial copying. Consequently infinite polynomial copying distinguishes multi-state from one-state (deterministic) top-down tree transducers.

We need a lemma. Let EDT0L denote $yDT(REG)$, where REG is viewed as a class of monadic tree languages, cf. [5, Sect. 4].

LEMMA 4.3. *If p is a polynomial with integer coefficients, then $\{a^{p(n)} \mid p(n) \geqslant 1\} \in$ EDT0L.*

Proof. It is shown in [17] that any polynomial $f(n)$ (with integer coefficients) which is positive and nondecreasing for $n \geqslant 0$, is the growth function of some D0L system (see [9, Sect. 15.3]), and hence $\{a^{f(n)} \mid n \geqslant 0\} \in$ EDT0L. It is easy to see that, for an arbitrary polynomial p with integer coefficients, $\{a^{p(n)} \mid p(n) \geqslant 1\}$ is the union of (at most) two such sets and a finite set, and consequently it is in EDT0L. ∎

THEOREM 4.4. *Let K be a tree trio. If p is a polynomial with integer coefficients, then $yDT(K)$ is closed under c_p.*

Proof. Let $L = yM(L_1)$ with $L_1 \in K$ and $M \in DT$. We have to show that $c_p(L) \in yDT(K)$. It is easy to see, using Lemma 4.3, that $\{\$(a\$)^{p(n)} \mid p(n) \geqslant 1\} \in$ EDT0L and hence it is $yN(L_2)$ for some regular monadic tree language L_2 and some $N \in DT$. We may assume that $\Sigma_0 \cap \Sigma_1 = \varnothing$ for the alphabet Σ of L_2 and that the alphabets of L_1 and L_2 are disjoint. Let L_3 be the result of inserting the regular language L_2 above the roots of the trees of L_1. Let M' be the deterministic top-down tree transducer which contains the rules of M, and also contains the rules of N modified as follows: left-hand sides $q(\sigma)$ are changed into $q(\sigma(x_1))$; each occurrence of a symbol a in any right-hand side is changed into $q_0(x_1)$, where q_0 is the initial state of M. Finally M' has all rules $q_0(\sigma(x_1)) \to q_0(x_1)$ for symbols σ in the alphabet of L_2. The initial state of M' is the one of N.

Then $yM'(L_3) = c_p(L)$. Intuitively, a monadic tree vx_1 with $v \in L_2$ is translated by M' into a tree with yield $\$(q_0(x_1)\$)^{p(n)}$ for some n; and hence a tree $v(t)$ with $v \in L_2$ and $t \in L_1$ is translated into $\$(w\$)^{p(n)}$, where $w = yM(t)$. ∎

We conclude this section by summarizing the results on infinite polynomial copying for the tree transducer hierarchy, cf. [5, Theorems 3.12 and 3.14].

COROLLARY 4.5. *Let p be a polynomial with integer coefficients such that* range$(p) \cap \mathbb{N}$ *is infinite. Let K_n denote $T^n(K_0)$ for some tree trio K_0, $n \geqslant 1$. If $L \in yK_n - yDT(K_{n-1})$, then $c_p(L) \in yDT(K_n)$, but $c_p(L) \notin yHOM(K_n)$ and $c_p(L) \notin yDT_{fc}(K_n)$.*

Proof. This follows directly from Theorem 4.4, the proof of Corollary 4.2, and [5, Theorem 3.9]. ∎

Note that, by [5, Theorem 3.12], languages in $yK_n - yDT(K_{n-1})$ exists if $yDT_{fc}(K_0) \subsetneq yT(K_0)$, in particular if $K_0 = \text{REC}$.

Corollary 4.5 also holds if c_p is replaced by c_f, where $f(n)$ is the nth nonprime, or c_g, where $g(n) = 2^n - 1$ (cf. the comments following Theorem 3.7).

5. Coloured Copying in the Tree Transducer Hierarchy

In this section we want to determine the place of the (nondeterministic) one-state top-down tree transducers in the tree transducer hierarchy, and to show that they form a proper hierarchy themselves. For related work see [20, Sect. 4]. Let NHOM (for "nondeterministic tree homomorphisms") denote the class of nondeterministic one-state top-down tree transducers. These transducers generalize the finite substitutions on strings.

Throughout this section, let K_0 be a tree trio such that $yDT_{fc}(K_0) \subsetneq yT(K_0)$, for instance $K_0 =$ REC. Let K_n denote $T^n(K_0)$ and let L_n be a language in $yT(K_{n-1}) - yDT(K_{n-1})$, cf. [5, Theorem 3.12]; recall that we may assume that $L_{n+1} = \text{rub}(c_{\exp}(L_n))$ or also that $L_{n+1} = \text{rub}(c_*(L_n))$. We want to show the correctness of Fig. 1 for $n \geqslant 1$ (for $n = 0$ and $K_0 = \text{REC}$, correctness was established in [6]). The correctness of this diagram with $y\text{NHOM}(K_n)$ omitted was shown in [5, Theorem 3.12]. It remains to show that $y\text{NHOM}(K_n) - yDT(K_n) \neq \emptyset$ and $yDT_{fc}(K_n) - y\text{NHOM}(K_n) \neq \emptyset$. These facts will be stated in Corollaries 5.2 and 5.5, respectively.

We need a technical lemma on closure properties of $y\text{HOM}(K)$ and $y\text{NHOM}(K)$ for certain classes K. Recall the notion of regular insertion from [5, Sect. 2.3]. We say that K is closed under regular insertion "at roots" if a regular monadic tree language can be inserted above the root of a tree (and nothing is inserted at nodes other than the root). We say that K is closed under insertion of $\#^*$ "below leaves," if each leaf σ (of a tree in some tree language in K) may be replaced by the monadic tree language $\sigma\#^*$ (where $\#$ is not in the alphabet of the language). It is straightforward to show that every tree trio is closed under regular insertion at roots and insertion of $\#^*$ below leaves.

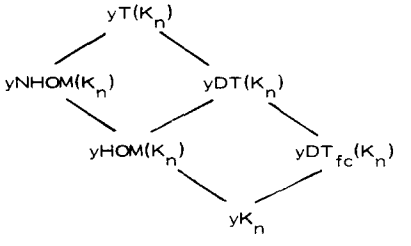

Figure 1

LEMMA 5.1. *Let K be a class of tree languages closed under regular insertion at roots and insertion of $\#^*$ below leaves.*

(1) *If $L \in y\text{HOM}(K)$, then $c_{\exp}(L) \in y\text{HOM}(K)$ and $c_*(L) \in y\text{NHOM}(K)$.*

(2) *$y\text{NHOM}(K)$ is closed under* rub.

Proof. The proof of (1) is similar to that of [5, Lemma 2.6]. Let $L = yH(L_1)$ with $H \in \text{HOM}$ and $L_1 \in K$. Insert $\not\#^*$ at the roots of the input trees. Extend H by $H_1(\not\#) = \not\#(\$x_1\$)$ and $H_1(\#) = \#(x_1\$x_1)$. Then $yH(\not\#^*(L_1)) = c_{\exp}(yH(L_1)) = c_{\exp}(L)$. Define a top-down tree transducer M with one state q and the following rules: $q(\not\#(x_1)) \to \not\#(\$q(x_1)\$)$, $q(\#(x_1)) \to \#(q(x_1)\,\$q(x_1))$, $q(\#(x_1)) \to q(x_1)$, and (to simulate H) all rules $q(\sigma(x_1 \cdots x_k)) \to H_k(\sigma)[q(x_1),\ldots, q(x_k)]$. Then $yM(\not\#^*(L_1)) = c_*(yH(L_1)) = c_*(L)$.

To prove that $y\text{NHOM}(K)$ is closed under rub, let $L = yM(L_1)$ with $M \in \text{NHOM}$ and $L_1 \in K$. Let L_2 be the result of inserting $\#^*$ below leaves of trees in L_1. We now define $M' \in \text{NHOM}$ such that M' simulates M and uses the tails of $\#$'s to produce arbitrary sequences of \$'s in the output of M. Then M' has the following rules: First, it has all rules of M. Second, if $q(\sigma) \to t$ is a rule of M (σ has rank 0), then $q(\sigma(x_1)) \to t'$ is a rule of M', where t' is the result of replacing in t each leaf δ by $\not\#(q(x_1)\,\delta q(x_1))$ or $\not\#(\delta q(x_1))$. Finally M' has the rules $q(\#(x_1)) \to \#(\$q(x_1))$, $q(\#(x_1)) \to q(x_1)$, and $q(\#) \to \$$. Then $yM'(L_2) = \text{rub}(yM(L_1)) = \text{rub}(L)$. ∎

Note that, by this lemma, NHOM has some infinite polynomial copying power, viz., c_* on input languages from $y\text{HOM}(K)$.

COROLLARY 5.2. $y\text{NHOM}(K_n) - yDT(K_n) \neq \emptyset$.

Proof. Since $L_n \in yK_n - yDT(K_{n-1})$ and $L_{n+1} = \text{rub}(c_{\exp}(L_n))$, Lemma 5.1 implies that $L_{n+1} \in y\text{NHOM}(K_n)$. Hence $L_{n+1} \in y\text{NHOM}(K_n) - yDT(K_n)$. The same holds in case $L_{n+1} = \text{rub}(c_*(L_n))$. ∎

We now show that the one-state top-down tree transducers form a proper hierarchy of classes of tree transformation languages. In [20] it is shown that $\{\text{NHOM}^n(\text{REC})\}$ is a proper hierarchy.

THEOREM 5.3. *$\{y\text{NHOM}^n(\text{REC})\}_{n \geq 0}$ is a proper hierarchy.*

Proof. Since $y\text{NHOM}^n(\text{REC}) \subseteq yT^n(\text{REC})$, it suffices to prove that $L_n \in y\text{NHOM}^n(\text{REC})$, for $n \geq 0$, where we can take $L_0 = \{a\}$ and $L_{n+1} = \text{rub}(c_{\exp}(L_n))$, cf. [5, proofs of Theorems 3.12 and 3.14]. This follows from Lemma 5.1 if $\text{NHOM}^n(\text{REC})$ satisfies the closure properties mentioned there. For REC this is obvious. Moreover, NHOM preserves these closure properties: for regular insertion at roots this is easy to prove and for insertion of $\#^*$ below leaves, the proof is similar to that of Lemma 5.1(2).

We note here that the closure properties in Lemma 5.1 were chosen such that they

would be preserved under NHOM (preservation of regular insertion in general is not clear). ∎

The incomparability of $yHOM(K_n)$ with $yDT(K_n)$ was proved in [5] as a consequence of the fact that $yHOM(K_n)$ is not closed under rub. Thus, by Lemma 5.1(2), the same argument cannot be used to show that $yDT_{fc}(K_n) - yNHOM(K_n) \neq \emptyset$. To prove this, we will show that $yNHOM(K_n)$ is not closed under coloured copying (see Section 2.3).

A "relabeling" is a one-state top-down tree transducer such that each rule has the form $q(\sigma(x_1 \cdots x_k)) \to \tau(q(x_1) \cdots q(x_k))$, where τ is an output symbol.

THEOREM 5.4. *Let K be a class of tree languages closed under relabeling and let L be a language. If $\bar{c}_2(L) \in yNHOM(T(K))$, then $\bar{c}_2(L) \in yHOM(DT(K))$. In particular, if $\bar{c}_2(L) \in yNHOM(NHOM(K))$, then $\bar{c}_2(L) \in yHOM(K)$.*

Proof. Clearly, if K is closed under relabeling, then so are $T(K)$ and $NHOM(K)$.

Let $L \subseteq \Sigma^*$ and $\bar{c}_2(L) \in yNHOM(T(K))$. It is easy to see that [5, Lemma 3.5] also holds for classes closed under relabeling and that the construction preserves the number of states. Hence the proof of [5, Theorem 3.6], which also applies to $\bar{c}_2$, yields that $\bar{c}_2(L) \in yHOM(T(K))$. Let $\bar{c}_2(L) = yH(M(L_1))$ with $L_1 \in K$, $M = (Q, \Sigma, \Delta, q_0, R)$ in T, and $H \in HOM$. By [5, Lemma 3.5], it suffices to show that $\bar{c}_2(L) = yH(M_{un}(L_1))$. Consider a derivation $q_0(t) \overset{*}{\Rightarrow} t'$ of M such that $\text{yield}(H(t')) = w\$\bar{w}$ with $w \in L$, and let s be a subtree of t with derivation sequence $\langle q_1(s) \overset{*}{\Rightarrow} s_1, \ldots, q_n(s) \overset{*}{\Rightarrow} s_n \rangle$. Suppose that $q_i = q_j$ with $i \neq j$. Then all occurrences of $yH(s_i)$ in $w\$\bar{w}$ may be replaced by $yH(s_j)$, and vice versa, without leaving the language $\bar{c}_2(L)$. Now if $yH(s_i) \in \Sigma^*$, then $yH(s_j)$ must also be in Σ^* and so $yH(s_i) = yH(s_j)$. Similarly for the case that $yH(s_i) \in \bar{\Sigma}^*$. If $yH(s_i)$ contains \$, then $yH(s_j)$ cannot occur in $yH(t')$. In all cases, the derivation $q_j(s) \overset{*}{\Rightarrow} s_j$ may be replaced by $q_i(s) \overset{*}{\Rightarrow} s_i$ without changing $yH(t') = w\$\bar{w}$. Hence $q_0(t) \overset{*}{\Rightarrow} t'$ can be changed into a uniform derivation $q_0(t) \overset{*}{\Rightarrow} t''$ such that $yH(t'') = w\$\bar{w}$, in the way indicated in the proof of [5, Theorem 3.6].

For the second statement of the theorem, we note that HOM is closed under composition. ∎

COROLLARY 5.5. $yDT_{fc}(K_n) - yNHOM(K_n) \neq \emptyset$.

Proof. Let L_n be in $yK_n - yDT(K_{n-1})$. Clearly $\bar{c}_2(L_n) \in yDT_{fc}(K_n)$. But $\bar{c}_2(L_n) \notin yNHOM(K_n)$. Indeed, $\bar{c}_2(L_n) \in yNHOM(T(K_{n-1}))$ implies $\bar{c}_2(L) \in yHOM(DT(K_{n-1})) = yDT(K_{n-1})$, by Theorem 5.4 and closure of $DT(K_{n-1})$ under HOM [4]. But then $L_n \in yDT(K_{n-1})$ because this class is obviously closed under (string) homomorphisms. ∎

This shows the correctness of Fig. 1.

The proof of Corollary 5.5 shows that $yHOM(K_n)$ is not closed under coloured copying. Since it is easy to see that $yDT(K_n)$ is closed under coloured copying, this

property distinguishes again the multi-state from the one-state (deterministic) top-down tree transducers.

We now show how to obtain languages outside the one-state hierarchy $\{yNHOM^n(REC)\}$.

COROLLARY 5.6. *If* $\bar{c}_2(L) \notin RPLL$, *then* $\bar{c}_2(L) \notin \bigcup_n yNHOM^n(REC)$.

Proof. Assume that $\bar{c}_2(L) \in \bigcup_n yNHOM^n(REC)$. Iterated application of Theorem 5.4 (in particular its last statement) yields that $\bar{c}_2(L) \in yHOM(REC)$, i.e., $\bar{c}_2(L) \in RPLL$ by Corollary 3.5. ∎

It was shown in [16, Theorem 6] that if L is not regular, then $\bar{c}_2(L) \notin RPLL$.

COROLLARY 5.7. $yDT_{fc}(REC) - \bigcup_n yNHOM^n(REC) \neq \emptyset$.

Proof. It follows from [16, Theorem 6] that $L = \{a^m b^m \$ c^m d^m \mid m \geqslant 1\} \notin RPLL$. Hence, by Corollary 5.6, $L \notin \bigcup_n yNHOM^n(REC)$. Clearly, $L \in yDT_{fc}(REC)$. ∎

This corollary and Theorem 5.3 together show that $yNHOM^n(REC)$ is a "small hierarchy" in $yT^n(REC)$, in the sense that languages which can be used to prove properness of $\{yT^n(REC)\}$ can already be found in $\{yNHOM^n(REC)\}$, and the smallest class of $\{yT^n(REC)\}$ already contains an element not in $\bigcup_n yNHOM^n(REC)$.

REFERENCES

1. B. S. BAKER, Tree Transductions and Families of Tree Languages, Ph. D. Thesis, Harvard University, Cambridge, Mass., Report TR-9-73, 1973 (see also "Proceedings, 5th ACM Symposium on Theory of Computing, pp. 200–206).
2. L. E. DICKSON, "Introduction to the Theory of Numbers," Dover, New York, 1957.
3. J. ENGELFRIET, Bottom-up and top-down tree transformations—a comparison, *Math. Systems Theory* **9** (1975), 198–231.
4. J. ENGELFRIET, Top-down tree transducers with regular look-ahead; *Math. Systems Theory* **10** (1977), 289–303.
5. J. ENGELFRIET, Three hierarchies of transducers, *Math. Systems Theory* **15** (1982), 95–125; see also [7].
6. J. ENGELFRIET, G. ROZENBERG, AND G. SLUTZKI, Tree transducers, L systems, and two-way machines, *J. Comput. System Sci.* **20** (1980), 150–202.
7. J. ENGELFRIET, G. ROZENBERG, AND G. SLUTZKI, Tree transducers, L systems, and two-way machines, *in* "Proceedings, 10 th ACM Symposium on Theory of Computing, San Diego, Calif. May 1978," pp. 66–74.
8. J. ENGELFRIET AND S. SKYUM, Copying theorems, *Inform. Process. Lett.* **4** (1976), 157–161.
9. G. T. HERMAN AND G. ROZENBERG, "Developmental Systems and Languages," North-Holland, Amsterdam, 1975.
10. J. E. HOPCROFT AND J. D. ULLMAN, "Formal Languages and Their Relation to Automata," Addison–Wesley, Reading, Mass., 1969.
11. C. R. PERRAULT, Intercalation lemmas for tree transducer languages, *J. Comput. System Sci.*, **13** (1976), 246–277.

12. L. PETRONE, Syntax directed mappings of context-free languages, *in* "IEEE Conf. Record of 9th Symposium on Switching and Automata Theory," pp. 160–175, 1968.
13. G. F. ROSE, An extension of ALGOL-like languages, *Comm. ACM* **7** (1964), 52–61.
14. A. SALOMAA, "Formal Languages," Academic Press, New York, 1973.
15. S. SKYUM, On extensions of ALGOL-like languages, *Inform. and Control* **26** (1974), 82–97.
16. S. SKYUM, Decomposition theorems for various kinds of languages parallel in nature, *SIAM J. Comput.* **5** (1976), 284–296.
17. A. L. SZILARD, Growth Functions of Lindenmayer Systems, Technical Report No. 4, University of Western Ontario, London, Ontario, Canada, 1971.
18. K. TANIGUCHI AND T. KASAMI, Macro expansions of context-free languages, *Electron. Comm. Japan*, **53-C** (7) (1970), 120–127.
19. J. W. THATCHER, Tree automata: an informal survey, *in* "Currents in the Theory of Computing" (A. V. Aho, Ed.), Prentice–Hall, Englewood Cliffs, N. J., 1973.
20. B. S. BAKER, Composition of top-down and bottom-up tree transductions, *Inform. and Control* **41** (1979), 186–213.