COPYING THEOREMS

Joost ENGELFRIET
Dept. of Applied Math., Technical University Twente, The Netherlands and Dept. of Computer Science, Aarhus University, Denmark

and

Sven SKYUM
Dept. of Computer Science, Aarhus University, Denmark

Received 7 August 1975

Formal languages

1. Introduction

In refs. [5,12] theorems of the following kind have been shown: for every language L, if $L_1 = \{w\#w|w\in L\}$ is in $\mathcal{F}_1$, then L_1 and L are in $\mathcal{F}_2$, where $\mathcal{F}_1$ is a class of "nondeterministic" languages and $\mathcal{F}_2 \subseteq \mathcal{F}_1$ is the corresponding class of "deterministic" languages, for instance $\mathcal{F}_1$ = ETOL and $\mathcal{F}_2$ = EDTOL (for these two classes of Lindenmayer languages, see [11,12]). Such a "copying theorem" shows that the copying power of $\mathcal{F}_1$ is fully contained in $\mathcal{F}_2$, i.e. languages not in $\mathcal{F}_2$ cannot be copied within $\mathcal{F}_1$. In this note we discuss some further theorems of this nature, in particular two results which generalize the case of ETOL and EDTOL. The first of these results concerns the indexed languages and was proved by Fischer [10]. Using this result together with the incomparability of EDTOL with the class of context-free languages (recently proved in [4]) and a copying theorem concerning bottom-up tree transformation languages, we show correctness of an inclusion diagram for ETOL, EDTOL, the context-free languages, the indexed languages and the top-down and bottom-up tree transformation languages. (However, the problem whether the indexed languages are contained in the top-down tree transformation languages remains open.) The second of these results concerns top-down tree transformation languages in general (the connection between such languages and Lindenmayer languages is shown in [3,7]).

2. Three properties of languages

Consider the following "deterministic" properties P1, P2, P3 of a language L over alphabet Σ.

(P1) For all $x, x', u, u', y, y' \in \Sigma^*$, if xuy, $xu'y$, $x'uy'$, and $x'u'y'$ are in L, then $u = u'$ or both $x = x'$ and $y = y'$.

(P2) For all $x, u, u', y, v, v', z \in \Sigma^*$, if $xuyvz$, $xu'yvz$, $xuyv'z$ and $xu'yv'z$ are in L, then $u = u'$ or $v = v'$.

(P3) For all $x, u, y, v, z \in \Sigma^*$, if $xuyvz$, $xuyuz$, $xyyuz$, and $xvyvz$ are in L, then $u = v$.

Roughly speaking, (P1) says that, in the generation of L by some rewriting system, there cannot be any "nested nondeterminism", (P2) says that there cannot be two nondeterministic symbols (or substrings) in one sentential form, and (P3) says that there cannot be two occurrences of the same nondeterministic symbol (or substring) in one sentential form. Thus these properties of languages force their grammars to be deterministic in a certain sense.

It can easily be seen that (P1) implies (P2) and (P2) implies (P3). We note that (P2) is equivalent to the following property (P2').

(P2') For all $x, x', y, y' \in \Sigma^*$, if xy, $x'y$, xy' and $x'y'$

are in L, then $x = x'$ or $y = y'$.

We now give some examples. The languages $\{w\#f_1(w)\#f_2(w)|w \in K\}$ and $\{wg_1(w)g_2(w)|w \in K\}$ have (P1), where K is a language over Σ_1, f_1 and f_2 are 1–1 functions $K \to \Sigma_1^*$, g_1 is a 1–1 function $K \to \Sigma_2^*$, g_2 is a 1–1 function $K \to \Sigma_3^*$, and Σ_1, Σ_2 and Σ_3 are disjoint alphabets. For instance, for any K, $\{w\#w\#w|w \in K\}$ has (P1).

The languages $\{w\#f(w)|w \in K\}$ and $\{wg(w)|w \in K\}$ have (P2), where K is a language over Σ_1, f a 1–1 function $K \to \Sigma_1^*$ and g a 1–1 function $K \to \Sigma_3^*$ with $\Sigma_1 \cap \Sigma_2 = \emptyset$. Note that languages of this form do not necessarily have (P1), for instance $\{w\#w^R|w \in \Sigma_1^*\}$, where w^R is the reverse of w, does not have (P1).

Note that the language $\{a^k\#b^m\#c^n|k, m, n \geq 1$, and $k = m = n$ or $k = m = n-1$ or $k = m-1 = n$ or $k = m-1 = n-1\}$ has (P3) but not (P2).

3. Copying theorem and an inclusion diagram

In [12] several results concerning ETOL and EDTOL are proved which can be summarized as follows.

Theorem 1 [12]. If L has (P3) and $L \in$ ETOL, then $L \in$ EDTOL. If, in particular, L is of the form $\{w\#f(w)|w \in K\}$ or $\{wg(w)|w \in K\}$ (as in 2), then also $K \in$ EDTOL. □

We now want to recall a generalization of this result to the indexed languages, proved in [10]. Let OI denote the class of outside-in macro languages, i.e. the class of indexed languages, and let LB denote the class of linear basic languages (see [10]). Note that LB = EDTOL and ETOL $\subseteq$ OI (see [2]).

Theorem 2 [10]. If L has (P1) and $L \in$ OI, then $L \in$ LB (= EDTOL). If, in particular, $L = \{w\#w\#w|w \in K\}$, then also $K \in$ EDTOL.

Proof. By the proof of Lemma 4.3.6 in [10], which is part of the proof that the language $\{a^m(ba^m)^n|m \geq 1, n = 2^m - 1\}$, which has property (P1), is not in OI. Property (P1) ensures that there is no nesting in the OI macro grammar, and that there cannot be more than one "non-deterministic" nonterminal in each sentential form. It is easy to see that if $L \in$ EDTOL then $K \in$ EDTOL by an argument similar to that in the proof of Theorem 2(I) in [12]. □

Consider the diagram (cf. [3]) depicted in fig. 1, where CF denotes the class of context-free languages,

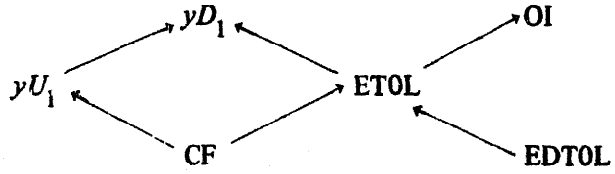

Fig. 1.

class of context-free languages, and yU_1 and yD_1 denote the classes of bottom-up and top-down tree transformation languages respectively (i.e. each L in yU_1 is of the form yield($B(R)$), where B is a bottom-up, finite state tree transformation and $R \in$ RECOG, the class of recognizable tree languages, and similarly for yD_1; see [1]). The inclusion ETOL $\subseteq yD_1$ is shown in [3,7]. Note that CF = yield(RECOG).

Open problem: is OI $\subseteq yD_1$?

Apart from this open problem we now show that all inclusions in the diagram are proper and that unrelated classes are incomparable. We need the following result, which solves a conjecture in [10].

Theorem 3 [4]. There is a context-free language which is not in EDTOL. □

Note that this implies incomparability of CF and EDTOL, since for instance $\{a^{2^n}|n \geq 0\}$ is in EDTOL–CF.

Let L_0 be in CF–EDTOL. Then [5]

(*) $\{w\#w^R|w \in L_0\}$ is in OI–ETOL,

where w^R is the reverse of w (Proof: obviously, if $L \in$ CF then $\{w\#w^R|w \in L\} \in$ OI. By Theorem 1, if $L \notin$ EDTOL then $\{w\#w^R|w \in L\} \notin$ ETOL). We note that $\{w\#w^R|w \in L_0\}$ is also in yD_1.

Next we obtain that

(**) $\{w\#w\#w|w \in L_0\}$ is in yU_1–OI

(Proof: obviously, if $L \in$ CF then $\{w\#w\#w|w \in L\} \in yU_1$. By Theorem 2, if $L \notin$ EDTOL then $\{w\#w\#w|w \in L\} \notin$ OI).

It now suffices to have an EDTOL language which is not in yU_1 (in [3] $\{a^n\#b^n\#c^n|n \geq 0\}$ is mentioned as such a language). To obtain such languages we show a "copying theorem" for yU_1.

Theorem 4. Let Σ_1, Σ_2 be disjoint alphabets and $K \subseteq \Sigma_1^*$. Let f be a 1–1 function $K \to \Sigma_2^*$ and let $L = \{wf(w)|w \in K\}$. If $L \in yU_1$, then $L \in$ CF.

Proof. We only sketch the proof. Let $L \in yU_1$. Then L = yield($h(R)$) where R is a recognizable tree

language and h is a tree homomorphism (see [1,6]). We may assume that h is nondeleting and nonerasing and that its target alphabet does not contain symbols of rank 1. Consider now all subtrees t of trees of R such that

$$\text{yield}(h(t)) \in \Sigma_1^* \cup \Sigma_2^*.$$

Suppose that there are infinitely many of such subtrees. Then they are arbitrarily high. Hence, by the pumping lemma for recognizable tree languages (Lemma 4 in [13]), there is such a subtree t which can be pumped up. The above assumptions on h then ensure that yield($h(t)$) is changed. This contradicts the form of L. Hence there is only a finite number of subtrees t such that

$$\text{yield}(h(t)) \in \Sigma_1^* \cup \Sigma_2^*.$$

Then we can change R into a recognizable tree language R' by a linear (bottom-up) tree transducer which removes these subtrees and puts them as (coded) information at their father nodes, and we can change h into a tree homomorphism h' which uses this information to simulate h, so that $h'(R') = h(R)$. It follows that for each subtree t of a tree of R' yield($h'(t)$) contains symbols from both Σ_1 and Σ_2. Hence h' cannot copy (is linear). Since the recognizable tree languages are closed under linear tree homomorphisms, $h'(R')$ is recognizable, and thus L = yield($h'(R')$) is context-free. □

We note that by (the proof of) Theorem 6 of [12] L is even a linear language and K is regular. Now we have that

(***) $\{a^{2^n}b^{2^n} \mid n \geq 0\}$ and $\{a^n b^n c^n \mid n \geq 1\}$ are in EDT0L $- yU_1$

(Proof: clearly both languages are in EDT0L. It follows from Theorem 4 that they are not in yU_1, since they are not context-free).

From (*), (**) and (***) correctness of the diagram follows. We finally mention that one could add the class IO of inside-out macro languages by drawing lines from yU_1 and EDT0L towards IO (EDT0L $\subseteq$ IO because EDT0L = LB [2], and $yU_1 \subseteq$ IO because the IO tree languages are closed under tree homomorphisms [9]). It was shown in [10] that the language $\{w \in \{a,b\}^* \mid$ the number of a's in w is a power of 2$\}$, which is in ET0L, is not in IO (and hence not in yU_1 and not in EDT0L). It then follows from (**) and (***) that the extended diagram is correct, except that it is open whether IO $\subseteq yU_1$.

4. A copying theorem for top-dwon tree transformation languages

In this section we show that Theorem 1 can be generalized to top-down tree transformation languages. Let, for any family $\mathcal{F}$ of tree languages, $T(\mathcal{F})$ denote the class of tree languages of the form $F(L)$ where F is a (nondeterministic) top-down tree transformation and $L \in \mathcal{F}$. Similarly for $DT(\mathcal{F})$, with F a deterministic top-down tree transformation. Let $y\mathcal{F}$ denote the class of languages yield(L) with $L \in \mathcal{F}$. A (nondeterministic) relabeling is a tree transformation which relabels nondeterministically the nodes of a tree by other symbols (depending on the symbol on that node; of course, different occurrences of the same symbol may be relabeled by different symbols). A finite state relabeling is a (nondeterministic) bottom-up or top-down finite state tree transformation which does not change the shape of the input tree. Thus each relabeling is a finite state relabeling. (In [6] the classes of relabelings and finite state relabelings are denoted by RELAB and QRELAB respectively).

Theorem 5. Let $\mathcal{F}$ be a class of tree languages closed under (nondeterministic) relabelings If L has (P3) and $L \in yT(\mathcal{F})$, then $L \in yDT(\mathcal{F})$. If $\mathcal{F}$ is closed under (nondeterministic) finite state relabelings and L is of the form $\{w\#f(w) \mid w \in K\}$ or $\{wg(w) \mid w \in K\}$ (as in 2), then also $K \in yDT(\mathcal{F})$.

Proof. The idea is essentially the same as that in the proof of Theorem 1. Let L = yield($F(M)$), where F is a (nondeterministic) top-down tree transducer and $M \in \mathcal{F}$. We want to simulate F (at least with respect to yields) by a relabeling R, which "guesses" the rules applied by F at each symbol of the input tree (for each state), followed by a deterministic top-down tree transducer F' which then applies these rules. In general this simulation does not work because F can copy an input subtree and process the copies nondeterministically, so that different rules may be applied at corresponding nodes (in the same state), but F' copies the input subtree "with the guesses" and is thereby forced to apply the same rule. However, from property (P3) it follows that, when-

ever F arrives at two different occurrences (copies) of the same input subtree in the same state, then all corresponding output subtrees have the same yield. Therefore, without changing yields, we may assume that F applies the same rule and gives the same output in such a situation. It follows that, in this case, the above simulation works. The construction is as follows. Let R be the (nondeterministic) relabeling such that the symbol a in the alphabet of M can be relabeled by any set S of rules of F (coded, of course, as symbols) such that all rules in S concern the symbol a and for every state q of F there is at most one rule in S "for q at a". Let F' be the deterministic top-down tree transducer with the same states as F, which, when arriving at symbol S in state q, applies the rule for q in S (if there is none, then the transducer does not accept). It is left to the reader to be convinced that

$$\text{yield}(F'(R(M))) = \text{yield}(F(M)).$$

Hence, since $\mathcal{F}$ is closed under relabelings, $L \in y\mathrm{DT}(\mathcal{F})$. Suppose now that L is of one of the indicated forms. Then K can obviously be obtained from L (on the tree level) by way of a deterministic bottom-up tree transducer, which removes # and everything to the right of # [all elements of the alphabet of $g(w)$, respectively]. It can be shown that, if $\mathcal{F}$ is closed under finite state relabelings, $\mathrm{DT}(\mathcal{F})$ is closed under deterministic bottom-up tree transformations (see [8]). Hence $K \in y\mathrm{DT}(\mathcal{F})$.

As shown in [3,7], ETOL $= yT(\mathcal{F})$ and EDTOL $= y\mathrm{DT}(\mathcal{F})$, where $\mathcal{F}$ is the class of recognizable monadic tree languages. Since this class is clearly closed under finite state relabelings, Theorem 1 is indeed a special case of Theorem 5.

Let D_n be the class $T^n(\mathrm{RECOG})$; D_n is closed under finite state relabelings [1]. Consequently, for every language K, if $K \in yD_n - y\mathrm{DT}(D_{n-1})$, then

$$\{w\#w|w \in K\} \in y\mathrm{DT}(D_n) - yD_n$$

(Proof. it is easy to see that if $K \in yD_n$ then $\{w\#w|w \in K\} \in y\mathrm{DT}(D_n)$; taking $\mathcal{F} = D_{n-1}$ in Theorem 5 proves the rest of the statement). This gives a "partial yield hierarchy result" in the spirit of that of Baker [1] for the tree hierarchy.

As a final example of the use of Theorem 5 we show that OI cannot be of the form $yT(\mathcal{F})$ for any class $\mathcal{F}$ of tree languages closed under finite state relabelings (this shows that Theorem 2 cannot be obtained as a special case of Theorem 5). In fact, suppose that OI $= yT(\mathcal{F})$. Let $L_0 \in$ CF–EDTOL (Theorem 3). Then $\{w\#w^R|w \in L_0\} \in$ OI. Consequently, by Theorem 5, $L_0 \in y\mathrm{DT}(\mathcal{F})$. Hence $\{w\#w\#w|w \in L_0\} \in y\mathrm{DT}(\mathrm{DT}(\mathcal{F}))$. It can be shown [8] that, if $\mathcal{F}$ is closed under finite state relabelings, $\mathrm{DT}(\mathcal{F})$ is closed under deterministic top-down tree transformations. Hence

$$\{w\#w\#w|w \in L_0\} \in y\mathrm{DT}(\mathcal{F}) \subseteq yT(\mathcal{F}) = \mathrm{OI}.$$

This contradicts (**). Note that OI is neither of the form $y\mathrm{DT}(\mathcal{F})$.

5. Conclusion

Copying theorems exist for the indexed languages and for several classes of tree transformation languages. In general it would be interesting to have theorems of the form: "if $\pi(L)$ is in $\mathcal{F}_1$, then $\pi(L)$ (or L) is in $\mathcal{F}_2$", where $\mathcal{F}_2 \subset \mathcal{F}_1$ and π is an operation other than copying, for instance an inverse homomorphism or a top-down tree transformation.

Acknowledgement

We thank Erik Meineche Schmidt for his comments on this paper.

References

[1] B.S. Baker, Tree transductions and families of tree languages, Report TR-9-73, Harvard Univ., 1973 (see also 5th Symp. on Theory of Computing, 200–206).

[2] P.J. Downey, Formal languages and recursion schemes, Report TR-16-74, Harvard Univ. (1974).

[3] P.J. Downey, Tree transducers and ETOL tree systems (abstract), Conf. Formal Languages, Automata and Development (Noordwijkerhout, Holland, 1975).

[4] A. Ehrenfeucht and G. Rozenberg, On some context-free languages which are not EDTOL languages, Techn. Rep. # CU-CS-048-74, Univ. Colorado, Dept. of Computer Science (1974).

[5] A. Ehrenfeucht, G. Rozenberg and S. Skyum, A relationship between ETOL and EDTOL languages, DAIMI Report PB-40, Aarhus Univ., Denmark, 1974 [to be published in Theoretical Computer Science 1 (1976) 325–330].

[6] J. Engelfriet, Bottom-up and top-down tree transformations – a comparison, Memorandum 19, Technical University Twente, Holland, 1971 (to be published in Math. Syst. Theory, vol. 9, no. 3).

[7] J. Engelfriet, Surface tree languages and parallel derivation trees, DAIMI Report PB-44, Aarhus Univ., Denmark, 1974 [to be published in Theoretical Computer Science Vol. 2, no. 1 (1976)].

[8] J. Engelfriet, Top-down tree transducers with regular look-ahead, in preparation (see Section 4 of Tree automata and tree grammars, Lecture Notes DAIMI FN-10, Aarhus Univ., Denmark, 1975).

[9] J. Engelfriet and E.M. Schmidt, IO and OI, DAIMI Report PB-47, Aarhus Univ., Denmark, 1975.

[10] M.J. Fischer, Grammars with macro-like productions, Doctoral Dissertation, Harvard Univ., 1968 (see also 9th Sw. & Aut. Theory, 131–142).

[11] G.T. Herman and G. Rozenberg, Developmental systems and languages (North-Holland Publ. Co., Amsterdam, 1975).

[12] S. Skyum, Decomposition theorems for various kinds of languages parallel in nature, to be published in SIAM J. on Computing (see also 7th Ann. ACM Symp. on Theory of Computing, May 1975).

[13] J.W. Thatcher, Tree automata: an informal survey, in: Currents in the Theory of Computing, ed. A.V. Aho (Prentice-Hall, 1973).