Theoretical Computer Science 82 (1991) 131-139
Elsevier

Note

Decidability of structural equivalence of E0L grammars*

Kai Salomaa

Department of Mathematics, University of Turku, SF-20500 Turku, Finland

Sheng Yu

Department of Computer Science, University of Western Ontario, London, Ontario, Canada N6A 5B7

Communicated by G. Rozenberg
Received March 1990
Revised July 1990

Abstract

Salomaa, K., and S. Yu, Decidability of structural equivalence of E0L grammars, Theoretical Computer Science, 82 (1991) 131-139.

We introduce height-counting tree automata that are able to recognize the syntax trees of E0L grammars. The equivalence problem of height-counting tree automata is shown to be decidable and using this result we solve an open problem raised by Ottmann and Wood [5, 6], i.e., the decidability of structural equivalence of E0L grammars.

1. Introduction

An E0L grammar can be seen as a context-free grammar with a parallel rewriting mechanism. In each step of an E0L derivation every nonterminal is rewritten by the productions of the grammar. Similarly to a context-free derivation, an E0L

* Research supported by the Natural Sciences and Engineering Research Council of Canada grants OGPIN007.

derivation can be represented by a syntax tree. Due to the parallelism of the derivation such a syntax tree is always balanced, i.e., every path from the root to a leaf is of equal length.

When considering language generating devices such as context-free or EOL grammars, the first question one may ask is whether one can decide the equivalence of such grammars, i.e., whether two given grammars generate the same language. Unfortunately, the equivalence problems for both context-free and EOL grammars are undecidable (cf. [8, 9, 11]).

It is possible that grammars G_1 and G_2 generate the same language even though the structures of derivations of a given terminal word using G_1 and G_2, respectively, are completely different, i.e., the syntax trees are not the same even modulo renaming of the nonterminal symbols. This leads one to consider the structural equivalence of grammars, which is also called strong equivalence. Grammars G_1 and G_2 are said to be structurally equivalent if they are equivalent and the structures of the syntax trees (disregarding the labels of the internal nodes) corresponding to any terminal word are the same. The decidability of structural equivalence of context-free grammars has been established in [4, 7]. The problem of deciding whether the sets of syntax trees generated by two context-free grammars are equal is shown to be decidable in [3].

It is observed in [10] that the decidability of structural equivalence of context-free grammars follows from the decidability of equivalence of finite tree automata. The equivalence problem of finite tree automata is seen to be decidable essentially as the equivalence of finite automata on words. The original decidability proofs for structural equivalence in [4, 7] are much more complicated. For instance, the proof given in [4] is based on showing that two reduced context-free grammars are structurally equivalent if and only if they are isomorphic. In the tree automaton terminology this corresponds to showing that two automata with a minimized set of states recognize the same tree language exactly when one is obtained from the other by renaming some states. Here we will use a tree automaton approach for the structural equivalence problem of EOL grammars.

The structural equivalence of EOL grammars is considered in [5, 6]. There partial decidability results have been obtained for certain restricted grammars, but the decidability of the general problem has remained open. Here we prove that the structural equivalence of EOL grammars is decidable. We introduce a tree automaton model called height-counting tree automaton. A height-counting tree automaton is essentially a finite bottom-up tree automaton that, additionally, remembers the height of the subtree processed so far. Using the finite-state control the automaton checks that the input tree corresponds locally to a correct syntax tree of the EOL grammar while the height counting mechanism enables this automaton to check that the tree is balanced and thus corresponds to a parallel derivation. Although a height-counting automaton is clearly more powerful than a finite tree automaton, it retains the desired feature of having a decidable equivalence problem. The intuitive reason for this fact is that the height-counting mechanism operates in all automata

identically and thus the decidability of equivalence can be proved by a slight modification of the proof for the corresponding result for finite tree automata.

2. Definitions and notations

The reader is assumed to be familiar with the basics of formal languages (cf. e.g. [9, 11]). In the following we briefly recall the definition of an E0L grammar (cf. [8]) and some related concepts. We will also introduce a slightly modified version of a finite tree automaton (cf. [2]).

Let A be a finite set. The set of nonempty words over A is denoted A^+. The length of a word $w \in A^+$ is denoted by $|w|$. By $\#A$ we mean the cardinality of A. An *E0L grammar* is a four-tuple $G=(V, \Sigma, S, P)$ where V is a finite set of nonterminal symbols; Σ is a finite set of terminal symbols ($\Sigma \cap V = \emptyset$); $S \subseteq V$ is the set of initial nonterminals and P is a finite set of productions of the form $X \to w$ where $X \in V$ and $w \in (V \cup \Sigma)^+$. Note that we do not allow productions in which the left-hand side is a terminal symbol or the right-hand side is empty. We allow more than one initial symbol. It is well known that these features do not change the family of generated languages except by excluding the empty word.

The set of *syntax trees* $S(G)$ of an E0L grammar $G=(V, \Sigma, S, P)$ is defined inductively as follows. If $X \in V \cup \Sigma$, then the tree with one node labeled by X belongs to $S(G)$. Let $t \in S(G)$ where t has n leaves labeled by nonterminals $X_1, \ldots, X_n$, respectively. Let $X_i \to a_1^i \ldots a_{k_i}^i$, $a_j^i \in V \cup \Sigma$, $i=1, \ldots, k_i$, $i=1, \ldots, n$, be productions of P. Denote by t' the tree that is obtained from t by attaching to the ith leaf k_i successors labeled by $a_1^i, \ldots, a_{k_i}^i$. Then $t' \in S(G)$.

A syntax tree t is said to be *terminal* if its root is labeled by an element of S and all leaves of t are labeled by elements of Σ. The set of terminal syntax trees of G is denoted $\mathrm{TS}(G)$. The *yield* of a tree $t \in S(G)$, denoted $\mathrm{yd}(t)$, is the word over $V \cup \Sigma$ that is obtained by concatenating the labels of the leaves of t. The *language generated* by the E0L grammar G is defined as

$$L(G) = \{w \in \Sigma^+ \mid \exists t \in \mathrm{TS}(G) \quad \mathrm{yd}(t) = w\}.$$

Since the syntax trees of G are always expanded in parallel at all leaves, it is clear that $L(G)$ as defined above is equal to the language generated by the E0L grammar G using the standard definition (cf. [8]).

We say that E0L grammars G_1 and G_2 are *equivalent* if $L(G_1) = L(G_2)$. It is well known that equivalence is undecidable even for context-free grammars. Grammars G_1 and G_2 are said to be structurally equivalent if they generate the same sets of terminal syntax trees when one disregards the nonterminals labeling the internal nodes of the trees. This is defined formally in the following.

Let $G=(V, \Sigma, S, P)$ be an E0L grammar and denote

$$M_G = \max\{|w| \mid X \to w \in P \text{ for some } X \in V\}. \tag{1}$$

Let $\{c_1, \ldots, c_{M_G}\}$ be a set of symbols disjoint with V and Σ. Let $t \in S(G)$. Then the *structure of the syntax tree* corresponding to t, $\mathrm{str}(t)$, is exactly the same as t except that the labels of the internal nodes are changed as follows. Let n be a nonleaf node of t with i successors, $1 \leq i \leq M_G$. Then in $\mathrm{str}(t)$ the node corresponding to n is labeled c_i. The set of structures of terminal syntax trees of G is defined as

$$\mathrm{STS}(G) = \{\mathrm{str}(t) \mid t \in \mathrm{TS}(G)\}.$$

E0L grammars G_1 and G_2 are said to be *structurally equivalent* if $\mathrm{STS}(G_1) = \mathrm{STS}(G_2)$ (cf. [6]).

Our main result is that structural equivalence of E0L grammars is decidable. To this end we define a tree automaton model that is able to recognize the structures of syntax trees of an E0L grammar.

Above we have considered labeled trees quite informally. When considering tree automata it is more convenient to assume that the labels belong to a ranked alphabet and each node having m successors is labeled by an m-ary symbol. In the following Ω is a ranked alphabet and the set of m-ary symbols is denoted by Ω_m, $m \geq 0$. The set of Ω-trees F_Ω consists of finite trees labeled by elements of Ω where each node labeled by $\omega \in \Omega_m$ has exactly m successors. Formally, the set F_Ω is the smallest set satisfying the conditions:

(i) $\Omega_0 \subseteq F_\Omega$.

(ii) If $\omega \in \Omega_m$ and $t_1, \ldots, t_m \in F_\Omega$ for $m > 0$, then $\omega(t_1, \ldots, t_m) \in F_\Omega$.

A deterministic height-counting tree automaton, DHC tree automaton, is a five-tuple

$$\mathscr{A} = (\Omega, A \times N, A' \times N, d, R)$$

where

(i) Ω is a finite ranked alphabet;

(ii) A is a finite set of states;

(iii) N is the set of nonnegative integers;

(iv) $A' \subseteq A$ is the set of final states;

(v) $d \in A - A'$ is a designated dead state;

(vi) R assigns to each m-ary symbol $\omega \in \Omega_m$ a mapping

$$\omega_R : (A \times N)^m \to (A \times N)$$

that satisfies the following conditions:

(a) If $m = 0$, then $\omega_R \in (A - \{d\}) \times \{0\}$ (or equivalently, $\omega_R : (A \times N)^m \to (A - \{d\}) \times \{0\}$).

(b) If $m > 0$, then

$$\omega_R((a_1, n_1), \ldots, (a_m, n_m)) = \begin{cases} (a, n+1) & \text{for } a \in A - \{d\} \text{ if } n_1 = \cdots = n_m = n \\ & \text{and } d \notin \{a_1, \ldots, a_m\}, \\ (d, 0) & \text{otherwise,} \end{cases} \tag{2}$$

where $(a_1, n_1), \ldots, (a_m, n_m) \in A \times N$.

Note that above we said A to be the set of states although strictly speaking the states of the automaton are pairs in $A \times N$. The designated dead state d has the

property that the automaton reaches the root of a tree t in a state with d as the first component if and only if t is not balanced. Note also that the rules (2) are completely deterministic.

The mapping R assigns inductively to each tree $t \in F_\Omega$ an element $t_R \in A \times N$ as follows. Let $t = \omega(t_1, \ldots, t_m)$, $\omega \in \Omega_m$, $m \geq 0$, and $t_1, \ldots, t_m \in F_\Omega$. Then

$$t_R = \omega_R((t_1)_R, \ldots, (t_m)_R).$$

The tree language recognized by $\mathscr{A}$ is

$$T(\mathscr{A}) = \{t \in F_\Omega \mid t_R \in A' \times N\},$$

i.e., $T(\mathscr{A})$ consists of exactly those trees where $\mathscr{A}$ reaches the root in a state of $A' \times N$.

If one ignores the integers appearing as the second components of states, a DHC tree automaton is essentially a deterministic finite bottom-up (or frontier-to-root) tree automaton (cf. [2]). The second components of the states are needed to enable the automata to recognize the syntax trees of E0L grammars. It is well known that the set of balanced trees cannot be recognized by a finite tree automaton. However, the DHC tree automaton model is useful for our purpose because although the state set is not finite the equivalence problem of DHC tree automata remains decidable.

3. Decidability of structural equivalence

Our main result states that the structural equivalence problem of E0L grammars is decidable. It is proved by showing that the structures of terminal syntax trees of an E0L grammar can be recognized by a DHC tree automaton and then showing that equivalence is decidable for DHC tree automata.

An E0L grammar $G = (V, \Sigma, S, P)$ is said to be *invertible* if no two productions of P have the same right-hand side, i.e., if $X_1 \to w \in P$ and $X_2 \to w \in P$, $X_1, X_2 \in V$, $w \in (V \cup \Sigma)^+$, then $X_1 = X_2$.

The result of Lemma 3.1 is from [5, 6]. Its proof uses a subset construction that is similar to the one that is used in [4] to show that every context-free grammar is structurally equivalent to an invertible context-free grammar. Note that Lemma 3.1 does not hold if one allows only a single start symbol.

Lemma 3.1. *Let G be an E0L grammar. Then one can effectively construct an invertible E0L grammar G' such that G and G' are structurally equivalent, i.e.,* $\mathrm{STS}(G) = \mathrm{STS}(G')$.

Lemma 3.2. *Let $G = (V, \Sigma, S, P)$ be an invertible E0L grammar. Then one can effectively construct a DHC tree automaton $\mathscr{A} = (\Omega, A \times N, A' \times N, d, R)$ such that* $T(\mathscr{A}) = \mathrm{STS}(G)$.

Proof. The ranked alphabet Ω is defined by setting $\Omega_0 = \Sigma$ and $\Omega_i = \{c_i\}$, $i = 1, \ldots, M_G$, where M_G is defined as in (1). By the choice of Ω, STS(G) is a subset of F_Ω. Let $A = V \cup \Sigma \cup \{d, f\}$, $d, f \notin \Sigma \cup V$ (d is the designated dead state), and $A' = S$. The function R is determined by the following:

(i) $a_R = (a, 0)$ if $a \in \Omega_0 (= \Sigma)$.

(ii) Let $m \in \{1, \ldots, M_G\}$, $a_1, \ldots, a_m \in A$, and $n_1, \ldots, n_m \in N$. Then

$$(c_m)_R((a_1, n_1), \ldots, (a_m, n_m)) = \begin{cases} (a, n+1) & \text{if } n_1 = \cdots = n_m = n \text{ and } a \to a_1 \ldots a_m \in P; \\ (f, n+1) & \text{if } n_1 = \cdots = n_m = n,\ d \notin \{a_1, \ldots, a_m\} \text{ and} \\ & \quad \nexists a \in V \text{ such that } a \to a_1 \ldots a_m \in P; \\ (d, 0) & \text{otherwise.} \end{cases}$$

The symbol d indicates that the subtree processed so far is not balanced whereas the failure symbol f is reached if the tree is balanced but does not correspond to a correct syntax tree. The use of two failure symbols is necessary since in the proof of the decidability of equivalence of DHC tree automata we need the property that the designated dead state d is reached only if the corresponding subtree is not balanced.

Since G is invertible, the value $(c_m)_R((a_1, n_1), \ldots, (a_m, n_m))$ is always uniquely determined and $\mathcal{A}$ as defined above is a DHC tree automaton.

Let $t \in F_\Omega$. We claim that $t_R = (a, n)$, $a \in V \cup \Sigma$, $n \geq 0$, if and only if there exists a syntax tree $t' \in S(G)$ such that $\mathrm{str}(t') = t$ and the root of t' is labeled by a. We denote the height of a tree $t \in F_\Omega$ by $\mathrm{hg}(t)$ and proceed by induction on the height:

(i) If $\mathrm{hg}(t) = 0$, then $t = a \in \Sigma$, $t_R = (a, 0)$ and $t = t'$.

(ii) Let $t = c_m(t_1, \ldots, t_m)$ and suppose that the claim holds for $t_1, \ldots, t_m$. If $(t_i)_R = (d, 0)$ for some $i \in \{1, \ldots, m\}$, then $t_R = (d, 0)$ and there does not exist $t' \in S(G)$ such that $\mathrm{str}(t') = t$. (Since t_i is not balanced, also t is not balanced.) The same is true if, for some $i, j \in \{1, \ldots, m\}$, $(t_i)_R = (a_1, n_1)$ and $(t_j)_R = (a_2, n_2)$, where $n_1 \neq n_2$. Assume that $(t_i)_R = (a_i, n)$, $a_i \in V \cup \Sigma \cup \{f\}$, $i = 1, \ldots, m$, $n \in N$. If there exists $a \in V$ such that $a \to a_1 \ldots a_m \in P$, then $t_R = (a, n+1)$ and by choosing $t' = a(t'_1, \ldots, t'_m) \in S(G)$ we have $\mathrm{str}(t') = t$. If $a_1 \ldots a_m$ is not the right-hand side of any production of P, then $t_R = (f, n+1)$ and from the induction hypothesis it follows that there does not exist $t' \in S(G)$ such that $\mathrm{str}(t') = t$.

Now the equality $T(\mathcal{A}) = \mathrm{STS}(G)$ is seen as follows. Let $t \in F_\Omega$. Then $t_R = (a, n)$, $a \in S$, $n \geq 0$ if and only if there exists $t' \in S(G)$ such that $\mathrm{str}(t') = t$ and the root of t' is labeled by an element of S if and only if $t \in \mathrm{STS}(G)$. □

It remains to be shown that one can effectively decide whether two DHC tree automata recognize the same tree language.

Lemma 3.3. *Let $\mathcal{A}_i = (\Omega, A_i \times N, A'_i \times N, d_i, R_i)$, $i = 1, 2$, be DHC tree automata. Then one can construct a DHC tree automaton $\mathcal{A} = (\Omega, A \times N, A' \times N, d, R)$ such that*

$$T(\mathcal{A}) = T(\mathcal{A}_1) - T(\mathcal{A}_2).$$

Proof. Choose $A=(A_1-\{d_1\})\times(A_2-\{d_2\})\cup\{(d_1,d_2)\}$, $A'=A_1'\times(A_2-(A_2'\cup\{d_2\}))$ and $d=(d_1,d_2)$. The mapping R is determined by the following:

Let $m\geqslant 0$, $\omega\in\Omega_m$, $n_1,\ldots,n_m\in N$ and $(a_1,b_1),\ldots,(a_m,b_m)\in A$ (i.e., $a_i\in A_1$, $b_i\in A_2$ and $a_i=d_1$ if and only if $b_i=d_2$, $i=1,\ldots,m$). Then

$$\omega_R((a_1,b_1,n_1),\ldots,(a_m,b_m,n_m))$$

$$=(\Pi_1(\omega_{R_1}((a_1,n_1),\ldots,(a_m,n_m))),\Pi_2(\omega_{R_2}((b_1,n_1),\ldots,(b_m,n_m))),h) \quad (3)$$

where Π_i, $i=1,2$, is the projection $A_i\times N\to A_i$ and $h=n+1$ if $n_1=\cdots=n_m=n$, $d_1\notin\{a_1,\ldots,a_m\}$ and $d_2\notin\{b_1,\ldots,b_m\}$ and $h=0$ otherwise. Note that since the DHC tree automaton $\mathscr{A}_i$, $i=1,2$, enters the designated dead state d_i exactly when the corresponding subtree is not balanced, we have

$$\Pi_1(\omega_{R_1}((a_1,n_1),\ldots,(a_m,n_m)))=d_1 \text{ iff } \Pi_2(\omega_{R_2}((b_1,n_1),\ldots,(b_m,n_m)))=d_2.$$

Hence the right-hand side of (3) belongs always to the state set of $\mathscr{A}$ and the element (d_1,d_2) is the designated dead state of $\mathscr{A}$.

Let $t\in F_\Omega$. Suppose that t is not balanced. Then clearly $t_R=(d_1,d_2,0)$ and $t\notin T(\mathscr{A})$. Suppose then that t is balanced. Then $t_R=(a_1,a_2,n)$, $t_{R_1}=(a_1,n)$, $t_{R_2}=(a_2,n)$, where $a_1\in A_1-\{d_1\}$, $a_2\in A_2-\{d_2\}$. By the choice of A',

$$t\in T(\mathscr{A}) \text{ iff } t\in T(\mathscr{A}_1) \text{ and } t\notin T(\mathscr{A}_2).$$

Hence we have shown that $T(\mathscr{A})=T(\mathscr{A}_1)-T(\mathscr{A}_2)$. □

Lemma 3.4. *Suppose we are given a DHC tree automaton*

$$\mathscr{A}=(\Omega,A\times N,A'\times N,d,R).$$

Then we can effectively decide whether $T(\mathscr{A})=\emptyset$.

Proof. Denote $k=\#A-1$. We claim that $T(\mathscr{A})\neq\emptyset$ if and only if there exists $t\in T(\mathscr{A})$ such that $\mathrm{hg}(t)<2^k$.

Let $t\in T(\mathscr{A})$ be such that

$$\mathrm{hg}(t)\geqslant 2^k. \quad (4)$$

We show that there exists $t'\in T(\mathscr{A})$ such that $\mathrm{hg}(t')<\mathrm{hg}(t)$. Denote by $\mathrm{sub}_i(t)$ the set of subtrees of t of height i, $i=0,\ldots,\mathrm{hg}(t)$. Denote

$$B(i)=\{a\in A\mid\exists r\in\mathrm{sub}_i(t)\ r_R=(a,i)\}.$$

There exist 2^k-1 nonempty subsets of $A-\{d\}$ and hence by (4) there exist $i,j\in\{0,\ldots,\mathrm{hg}(t)\}$, $i<j$ such that $B(i)=B(j)$. Thus for every $r\in\mathrm{sub}_j(t)$ there exists $r'\in\mathrm{sub}_i(t)$ such that $r_R=(a,j)$ and $r'_R=(a,i)$, $a\in A$. Choose t' to be the tree that is obtained from t by replacing every subtree $r\in\mathrm{sub}_j(t)$ by r'. Suppose we have $t_R=(a,\mathrm{hg}(t))$ for some $a\in A'$. Then $t'_R=(a,\mathrm{hg}(t)-j+i)$ and $t'\in T(\mathscr{A})$ with $\mathrm{hg}(t')<\mathrm{hg}(t)$. □

We obtain the main result as a direct consequence of the above lemmas.

Theorem 3.5. *Structural equivalence of EOL grammars is decidable.*

Proof. Let EOL grammars G_1 and G_2 be given. By Lemma 3.1 we can construct invertible grammars G_i', $i=1,2$, such that G_i' and G_i are structurally equivalent. By Lemma 3.2 there exists effectively DHC tree automata $\mathscr{A}_i$ such that $T(\mathscr{A}_i) = \mathrm{STS}(G_i')$, $i=1,2$. By Lemmas 3.3 and 3.4 we can decide whether $T(\mathscr{A}_1) = T(\mathscr{A}_2)$ and hence also whether $\mathrm{STS}(G_1) = \mathrm{STS}(G_2)$. □

We conclude with a discussion on the possibilities of extending the result of Theorem 3.5 for more general classes of grammars. The most natural generalization is to consider the structural equivalence of ETOL grammars (cf. [8]). It is not difficult to see that the syntax trees of ETOL grammars can be recognized by an extension of height-counting automata, where the second components of the states store a word representing the sequence of tables used in the subtree processed so far. However, due to the presence of several tables these automata are inherently nondeterministic and a proof for the decidability of the equivalence problem seems much more difficult than for height-counting automata. Thus the decidability of structural equivalence of ETOL grammars remains open.

As a next step one may consider indexed grammars (cf. [1, 9]). A derivation of an indexed grammar can be represented by a syntax tree where each node is labeled by a nonterminal followed by some sequence of index symbols. (Thus the set of labels of nodes of the trees is not finite and the notion of structural equivalence may not be as natural as for context-free or EOL grammars.) Every context-free language can be generated by a right-linear indexed grammar. Using this observation and the fact that language equivalence of context-free grammars is undecidable, it is easy to see that structural equivalence of indexed grammars is undecidable.

In fact one can show that structural equivalence is undecidable even for a restricted class of indexed grammars, where all index generating productions are of the form $S \to Sf$, where f is an index symbol, S is the initial nonterminal and S does not appear in the right-hand side of any production except the above index generating productions. (That is, the grammar always first generates an arbitrary sequence of indices for the initial nonterminal S and after this the grammar uses only the context-free and index consuming productions.) These grammars are still sufficiently general to simulate the derivations of ETOL grammars.

References

[1] A. Aho, Indexed grammars—An extension of context-free grammars, *J. Assoc. Comput. Mach.* **15** (1968) 647–671.

[2] F. Gécseg and M. Steinby, *Tree Automata* (Akadémiai Kiadó, Budapest, 1984).

[3] S. Ginsburg and M. Harrison, Bracketed context-free languages, *J. Comput. System Sci.* **1** (1967) 1–23.

[4] R. McNaughton, Parenthesis grammars, *J. Assoc. Comput. Mach.* **14** (1967) 490–500.
[5] T. Ottmann and D. Wood, Defining families of trees with E0L grammars, *Research Report* CS-89-39, University of Waterloo, 1989.
[6] T. Ottmann and D. Wood, Structural equivalence of E0L grammars, *Research Report* CS-89-40, University of Waterloo, 1989.
[7] M. Paull and S. Unger, Structural equivalence of context-free grammars, *J. Comput. System Sci.* **2** (1968) 427–463.
[8] G. Rozenberg and A. Salomaa, *The Mathematical Theory of L Systems* (Academic Press, New York, 1980).
[9] A. Salomaa, *Formal Languages* (Academic Press, New York, 1973).
[10] J.W. Thatcher, Tree automata: an informal survey, in: A.V. Aho, ed., *Currents in the Theory of Computing* (Prentice Hall, Englewood Cliffs, NJ, 1973) 143–172.
[11] D. Wood, *Theory of Computation* (Wiley, New York, 1987).