Theoretical Computer Science 82 (1991) 95-111
Elsevier

Nonterminal separation in graph grammars

Joost Engelfriet, George Leih* and Grzegorz Rozenberg

Department of Computer Science, Leiden University, P.O. Box 9512, 2300 RA Leiden, The Netherlands

Communicated by A. Salomaa
Received December 1988

Abstract

Engelfriet, J., G. Leih and G. Rozenberg, Nonterminal separation in graph grammars, Theoretical Computer Science 82 (1991) 95-111.

The notion of nonterminal separation in eNCE graph grammars is introduced: such a graph grammar is k-separated (with $k \geqslant 1$) if the distance between any two nonterminal nodes in any of its sentential forms is at least k (2-separated eNCE grammars are known as boundary eNCE graph grammars). There exists a 2-separated eNCE graph language that is not 3-separated. Apex eNCE graph grammars can be arbitrarily separated: every apex eNCE graph language can be generated by a k-separated apex eNCE grammar, for every k.

0. Introduction

Node label controlled (NLC) graph grammars have become well known since their introduction in 1980 [13, 14]. This may be due to the fact that, in comparison with some other types of graph grammars, they are rather easy to handle. Some of the main features of the NLC grammar are the following (we assume the reader to be familiar with the idea of rewriting in a graph grammar; see [4] and [5] for an overview). An NLC grammar generates undirected node-labeled graphs. The left-hand side of a production consists of a single labeled node. The embedding mechanism is such that edges can only be added between newly generated nodes and former neighbors of the replaced node. Moreover, which edges are added depends solely on the labels of the nodes involved.

Recently several variations of the NLC model have been investigated. One of these is the eNCE model (see [2, 6, 7, 10, 11, 16, 17]). It differs from the NLC grammar in a number of aspects: (1) the embedding mechanism makes use of the identity (rather than the label) of the nodes in the right-hand side of the productions

* The work of this author was conducted as part of the PRISMA project, a joint effort with Philips Research, partially supported by the Dutch "Stimuleringsprojectteam informaticaonderzoek" (SPIN).

(this change of the NLC embedding mechanism is called neighborhood controlled embedding (NCE) [15]; (2) the generated graphs have also edge labels (the "e" in "eNCE"); (3) the embedding mechanism also uses the labels of the edges incident with the replaced node. It is shown in [10] that eNCE grammars are more powerful than NLC grammars, whereas it seems that all the nice features of the NLC grammar also hold for the eNCE grammar. In this paper we consider eNCE grammars, and in particular we investigate the notion of separation in these grammars.

In [19, 20, 21, 10] it is demonstrated that the so-called boundary restriction (to be explained) on graph grammars yields a number of additional nice features and properties. One of the reasons for this is that boundary NLC and eNCE grammars are confluent (cf. [3]), i.e., when two nodes in a graph can be replaced, they may be replaced in either order (without influencing the result); arbitrary NLC or eNCE grammars do not have to be confluent. An eNCE grammar is a boundary (eNCE) grammar if there are no edges between nonterminal nodes in its sentential forms. In other words, the distance between nonterminal nodes in any sentential form of a boundary grammar is at least two, whereas it may (of course) be one in the case of an arbitrary eNCE grammar.

In this paper we study the general notion of distance between nonterminal nodes in eNCE grammars. In particular we investigate whether "small" distances between nonterminal nodes can be avoided. For $k \geq 1$ we say that a (eNCE) grammar is k-separated if the distance between any two nonterminal nodes in its sentential forms is $\geq k$. Thus, all eNCE grammars are 1-separated, and the 2-separated eNCE grammars are precisely the boundary grammars. It is well known that 1-separated grammars are more powerful than 2-separated grammars. We prove that 2-separated grammars are more powerful than 3-separated grammars. Thus, "small" distances cannot be avoided in (boundary) eNCE graph grammars. This is in contrast to the case for context-free string grammars, where it is shown that small distances between nonterminals *can* be avoided (see [1, 12, 18]). We conjecture that for $k \geq 3$, k-separated grammars are more powerful than $(k+1)$-separated grammars.

Another natural restriction on graph grammars is the apex restriction (introduced in [8, 9]). An eNCE grammar is an apex (eNCE) grammar if the embedding mechanism can only establish edges between terminal nodes. It is easy to see (cf. [8, 6]) that each apex language can be generated by a 2-separated apex grammar. We show that this can be generalized as follows: each apex language can be generated by a k-separated apex grammar for each $k \geq 1$. Hence, demanding apex grammars to be k-separated, for some k, has no influence on their generating power. Roughly speaking, the idea behind this result can be explained as follows. After the application of a production in an apex grammar, the distance between the newly generated nonterminal nodes and the old nonterminal nodes is larger than the distance between these old nonterminal nodes and the replaced node. Thus, "small" distances are entirely due to small distances between the new nonterminal nodes. Consequently, another grammar can be constructed in which these new nodes are "contracted" into one node; this new grammar is more separated than the original one.

The organization of the paper is as follows. The preliminary definitions concerning graphs can be found in Section 1, and the definition of an eNCE grammar in Section 2. In Section 3, k-separation is introduced, and the generating power of k-separated and $(k+1)$-separated eNCE grammars is compared. Finally, in Section 4 apex eNCE grammars are defined, and it is shown that these grammars can be arbitrarily separated.

This paper is part of a series of five papers on eNCE grammars (the other four are [6, 7, 10, 11]). Altogether these five papers (together with [2, 16, 17]) form a rather complete picture of basic properties of eNCE grammars and languages. In our opinion they demonstrate that this class is natural (as an extension of NLC) and interesting (as illustrated by the various results of this series of papers). We would also like to stress that the class of eNCE grammars is technically attractive in the sense that many of the constructions used, either in proving properties or in designing grammars for specific languages, can be done easily.

The results established in this paper were first presented (without full proofs) in [9], where directed graphs were considered.

1. Preliminaries

In this section we discuss some notation and terminology used in this paper.

For a function $f: A \to B$ and a set $C \subseteq A$, $f(C)$ denotes the set $\{f(c) \mid c \in C\}$.

A *node- and edge-labeled graph*, or just a *graph*, is a system $H = (V, E, \Sigma, \Gamma, \varphi)$, where V is the finite set of nodes, Σ is the alphabet of node labels, Γ is the alphabet of edge labels, $E \subseteq \{(\{v, w\}, \lambda) \mid v, w \in V, v \neq w, \lambda \in \Gamma\}$ is the set of (labeled) edges, and $\varphi: V \to \Sigma$ is the node labeling function. Thus we consider undirected graphs without loops; multiple edges between the same pair of nodes are allowed if they are labeled differently. We use V_H, E_H, Σ_H, Γ_H, and φ_H to denote the different components of H. For better readability, an edge $(\{v, w\}, \lambda)$ will be denoted (v, λ, w) or (w, λ, v) in the sequel; so (v, λ, w) and (w, λ, v) denote the same edge; λ is said to be the label of (v, λ, w).

Let H be a graph. If $V_H = \emptyset$ then H is the *empty graph* (denoted Λ), and if $E_H = \emptyset$ then H is *discrete*. A graph H with just one node, say v, with $\varphi_H(v) = X$ for some $X \in \Sigma_H$, will also be denoted X.

A graph $H = (V, E, \Sigma, \Gamma, \varphi)$ is called a *graph over Σ and Γ*. For alphabets Σ and Γ, the set of all graphs over Σ and Γ is denoted $\mathrm{GR}_{\Sigma,\Gamma}$. A *graph language* is any subset of $\mathrm{GR}_{\Sigma,\Gamma}$.

Let H and K be graphs over Σ and Γ. H and K are *isomorphic* if there is a bijection $\beta: V_H \to V_K$ such that $E_K = \{(\beta(v), \lambda, \beta(w)) \mid (v, \lambda, w) \in E_H\}$ and, for all $v \in V_H$, $\varphi_K(\beta(v)) = \varphi_H(v)$. The *union* of H and K, which is only defined if $V_H \cap V_K = \emptyset$, is the graph $H \cup K = (V_H \cup V_K, E_H \cup E_K, \Sigma_H \cup \Sigma_K, \Gamma_H \cup \Gamma_K, \varphi_H \cup \varphi_K)$.

Let H be a graph. Whenever $(v, \lambda, w) \in E_H$, we say that v and w are *neighbors*. A sequence $v_1, v_2, \ldots, v_r$ of nodes in V_H, with $r \geq 1$, is a *path* between v_1 and v_r

in H if v_i, v_{i+1} are neighbors for all $1 \leqslant i \leqslant r-1$. The *length* of this path is $r-1$. For two nodes x and y in V_H, the *distance* between x and y in H, denoted $\text{dist}_H(x, y)$, is the length of a shortest path between x and y, if such a path exists; otherwise $\text{dist}_H(x, y) = \infty$, where ∞ is such that $k < \infty$ for all $k \geqslant 0$.

For a graph H and $V \subseteq V_H$, the *subgraph of H induced by V* is the graph $H[V] = (V, \bar{E}, \Sigma_H, \Gamma_H, \bar{\varphi})$, where $\bar{E} = \{(v, \lambda, w) \in E_H \mid v, w \in V\}$, and $\bar{\varphi}$ equals φ restricted to V.

2. Basic definitions on graph grammars

In this paper we use eNCE graph grammars, defined as follows.

Definition 2.1. A *graph grammar with neighborhood controlled embedding and dynamic edge relabeling*, for short eNCE grammar, is a system $G = (\Sigma, \Delta, \Gamma, \Omega, P, S)$, where Σ is the *total alphabet of node labels*, $\Delta \subseteq \Sigma$ is the *alphabet of terminal node labels*, Γ is the *total alphabet of edge labels*, $\Omega \subseteq \Gamma$ is the *alphabet of final edge labels*, P is the finite set of *productions*, and $S \in \Sigma - \Delta$ is the *initial nonterminal*. A production $\pi \in P$ is of the form $\pi = (X, D, B)$, with $X \in \Sigma - \Delta$, $D \in \text{GR}_{\Sigma,\Gamma}$, and $B \subseteq V_D \times \Gamma \times \Gamma \times \Sigma$.

Before we proceed with the definition of the behavior of a graph grammar we introduce some terminology concerning the concepts defined above. First, elements of Δ are called *terminals*, and elements of $\Sigma - \Delta$ are called *nonterminals*. Then, for a graph $H \in \text{GR}_{\Sigma,\Gamma}$, a node $v \in V_H$ is called *terminal* if $\varphi_H(v) \in \Delta$, and *nonterminal* otherwise. For a production $\pi = (X, D, B)$ we use $\text{apex}(\pi)$ to denote the set $\{v \in V_D \mid \varphi_D(v) \in \Sigma - \Delta\}$ of nonterminal nodes of D. Moreover, X is called the *left-hand side* of π, D is called the *right-hand side* of π, and B is called the *embedding relation* of π. We write $\text{lhs}(\pi) = X$, $\text{rhs}(\pi) = D$, and $\text{emb}(\pi) = B$. Finally, π is a *Λ-production* if $\text{rhs}(\pi) = \Lambda$.

Definition 2.2. Let $G = (\Sigma, \Delta, \Gamma, \Omega, P, S)$ be an eNCE grammar. Let H and K be graphs over Σ and Γ, let $v \in V_H$, let $\pi = (X, D, B) \in P$, and let $\beta : V_D \to V_K$ be a total injective function such that $V_H \cap \beta(V_D) = \emptyset$. Then we write $H \Rightarrow_{(v,\pi,\beta)} K$, or just $H \Rightarrow K$, if $\varphi_H(v) = X$ and K is the following graph:

$$
\begin{aligned}
V_K &= (V_H - \{v\}) \cup \beta(V_D),\\
E_K &= \{(x, \mu, y) \in E_H \mid x \neq v, y \neq v\}\\
&\quad \cup \{(\beta(x), \mu, \beta(y)) \mid (x, \mu, y) \in E_D\}\\
&\quad \cup \{(\beta(x), \mu, y) \mid x \in V_D, y \in V_H - \{v\}, \mu \in \Gamma,\\
&\qquad \exists \lambda \in \Gamma : (v, \lambda, y) \in E_H, (x, \lambda, \mu, \varphi_H(y)) \in B\},\\
\Sigma_K &= \Sigma,\\
\Gamma_K &= \Gamma,\\
\varphi_K(x) &= \varphi_H(x) \text{ if } x \in V_H \quad \text{and} \quad \varphi_K(\beta(x)) = \varphi_D(x) \text{ if } x \in V_D.
\end{aligned}
$$

As usual, $\Rightarrow^*$ is the transitive-reflexive closure of $\Rightarrow$. A graph $H \in \mathrm{GR}_{\Sigma,\Gamma}$ such that $S \Rightarrow^* H$ is called a *sentential form* of G. The *language generated by* G is $L(G) = \{H \in \mathrm{GR}_{\Delta,\Omega} \mid S \Rightarrow^* H\}$.

As usual, somewhat informally, $H \Rightarrow K$ is called a *derivation step*, and a sequence of such derivation steps is called a *derivation*.

Thus, the set of all graphs with only terminal node labels and final edge labels which can be derived from S (more precisely, from any graph with just one node labeled S) forms the language of an eNCE graph grammar. The class of all languages generated by eNCE grammars is denoted **eNCE**.

Note that we define a derivation step between "concrete" graphs while in [6, 7, 10] "abstract" derivation steps have been considered, i.e., (in the notation of above) steps of which the result may be any graph isomorphic with K. The "concrete" derivation steps are more convenient for technical reasons. However, we do not lose any generating power, as it is easy to see that the set of all sentential forms of an eNCE grammar G is closed under isomorphisms (and, in particular, $L(G)$ is closed under isomorphisms).

Next, two examples of an eNCE graph grammar are given. The same examples, and some more, can be found in [6] and [10].

Example 2.3. Let $G_1 = (\Sigma, \Delta, \Gamma, \Omega, P, S)$ be the eNCE grammar defined by $\Sigma = \{a, S\}$, $\Delta = \{a\}$, $\Gamma = \{\lambda, \mu\}$, $\Omega = \{\mu\}$, and with P consisting of the productions $\pi_1 = (S, D, B)$ and $\pi_2 = (S, \Lambda, \emptyset)$, where $V_D = \{x, y, z\}$, $E_D = \{(x, \mu, y), (x, \lambda, z), (y, \mu, z)\}$, $\varphi_D(x) = \varphi_D(y) = a$, $\varphi_D(z) = S$, and $B = \{(x, \lambda, \mu, a), (y, \mu, \mu, a)\}$. In Fig. 1(a) a picture of production π_1 is given. Nonterminal nodes are drawn as boxes and terminal nodes as circles. The graph inside the big box (i.e., the nonterminal node that is rewritten) represents $\mathrm{rhs}(\pi_1)$, the symbol in the left upper corner represents $\mathrm{lhs}(\pi_1)$, and the edges crossing the big box represent $\mathrm{emb}(\pi_1)$. These latter edges have two labels: the one outside the box is the "old" label and the one inside the box is the "new" label. It is not difficult to see now that $L(G_1)$ is the set of all "ladders" as depicted in Fig. 1(b).

Let G_2 be defined as G_1 above, but with the productions as depicted in Fig. 2. It is easily seen that $L(G_2)$ is the set of all binary trees over $\{a\}$ and $\{\mu\}$.

Finally we introduce some more terminology with respect to derivations. Consider an eNCE grammar $G = (\Sigma, \Delta, \Gamma, \Omega, P, S)$ and a derivation $\delta : S \Rightarrow_{(v_1,\pi_1,\beta_1)} H_1 \Rightarrow \cdots \Rightarrow_{(v_r,\pi_r,\beta_r)} H_r$ in G, with $H_i \in \mathrm{GR}_{\Sigma,\Gamma}$ and $r \geq 1$. Then H_r is called the *result of* δ, and δ is said to be *a derivation of* H_r. Moreover, δ is said to be *consistent* if $v_i \notin V_{H_j}$ for all $1 \leq i \leq j \leq r$. Thus, in a consistent derivation, no node is "used twice", i.e., when a node is rewritten it cannot be introduced anymore later on. It is not difficult to see that for every derivation there is a consistent derivation with the same result. Hence, from now on we will consider consistent derivations only.

The following notion will play an important role in Section 4.

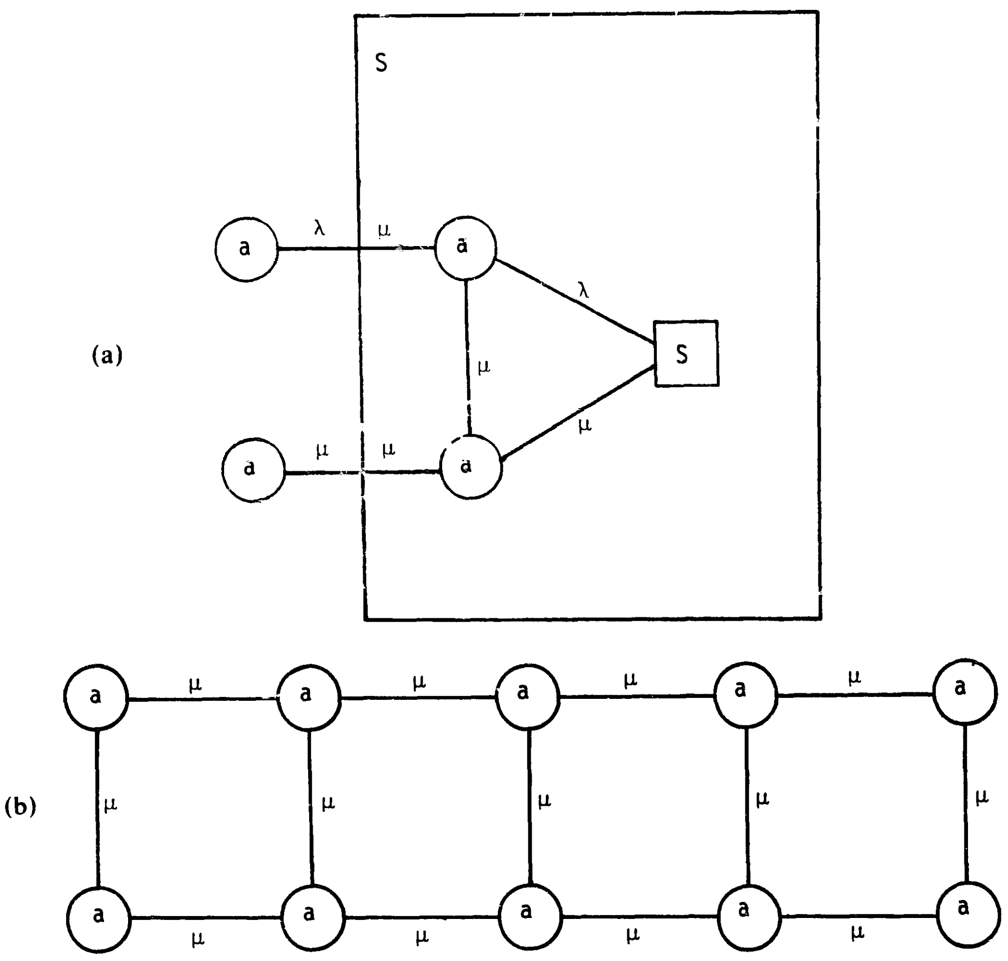

Fig. 1.

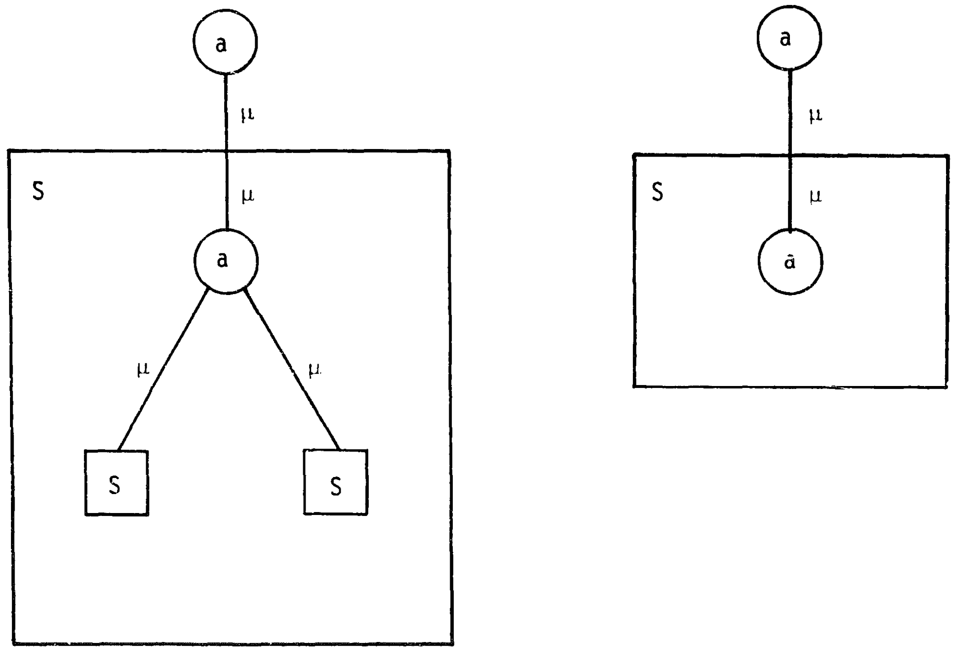

Fig. 2.

Definition 2.4. Let $G=(\Sigma, \Delta, \Gamma, \Omega, P, S)$ be an eNCE grammar, let $\delta: S \Rightarrow_{(v_1,\pi_1,\beta_1)} H_1 \Rightarrow \cdots \Rightarrow_{(v_r,\pi_r,\beta_r)} H_r$ be a consistent derivation in G with $r \geq 1$, and let x be a node in H_r. Then the *producer of x in* δ, denoted $\text{producer}_\delta(x)$, equals the number $i \in \{1, 2, \ldots, r\}$ such that $x \in \beta_i(V_{\text{rhs}(\pi_i)})$.

Intuitively, the fact that $\text{producer}_\delta(x) = i$ means that x is generated in the ith derivation step $H_{i-1} \Rightarrow H_i$ of δ (with $H_0 = S$).

3. Separation

We now introduce the notion of nonterminal separation in graph grammars.

Definition 3.1. Let G be an eNCE grammar, and let $k \geq 1$. G is *k-separated* if for every sentential form H of G and every two distinct nonterminal nodes x and y of H, $\text{dist}_H(x, y) \geq k$.

For $k \geq 1$ we use $\mathbf{SEP}_k$ to denote the set of all eNCE languages that can be generated by a k-separated eNCE grammar. Trivially, the eNCE grammar G_1 of Example 2.3 is k-separated for each $k \geq 1$, and so $L(G_1) \in \mathbf{SEP}_k$. Moreover, it is easy to see that G_2 is 2-separated, but not 3-separated. However, we will prove in Section 4 that for every k $L(G_2) \in \mathbf{SEP}_k$.

Let us now make some easy observations. First, $\mathbf{SEP}_1 = \mathbf{eNCE}$, and second, $\mathbf{SEP}_{k+1} \subseteq \mathbf{SEP}_k$ for all $k \geq 1$. Furthermore, as mentioned in the Introduction, it is not difficult to show (cf. Lemma 2.1 of [19]) that the class $\mathbf{SEP}_2$ coincides with the class of so-called boundary eNCE languages, which have been investigated in [10]. Since it is proved in that paper that the class of boundary eNCE languages is a proper subclass of eNCE, it follows that $\mathbf{SEP}_2$ is properly included in $\mathbf{SEP}_1$.

The main question concerning separation is whether $\mathbf{SEP}_{k+1}$ is a proper subset of $\mathbf{SEP}_k$ for all $k \geq 1$. Unfortunately, we do not know the answer. However, we do know that $\mathbf{SEP}_3$ is properly included in $\mathbf{SEP}_2$, which shows that "small" distances cannot be avoided in boundary eNCE grammars. In order to prove this we first need the notion of a linear grammar (cf. [6]).

Definition 3.2. An eNCE grammar G is *linear* if the right-hand side of each production of G contains at most one nonterminal node.

We will use the notation **LIN** for the class of all languages that can be generated by a linear eNCE grammar. Clearly, G_1 is linear, but G_2 is not.

Now we are ready to give the proof of the proper inclusion of $\mathbf{SEP}_3$ in $\mathbf{SEP}_2$. It is based on the (nontrivial) fact, shown in [6], that **LIN** is properly included in $\mathbf{SEP}_2$; note that $\mathbf{LIN} \subseteq \mathbf{SEP}_3 \subseteq \mathbf{SEP}_2$ (in fact, a linear grammar is k-separated for all $k \geq 1$ because the sentential forms of a linear grammar have no more than one nonterminal node).

Theorem 3.3. $\mathbf{SEP}_3$ *is properly included in* $\mathbf{SEP}_2$.

Proof. Consider any graph language L such that L is in $\mathbf{SEP}_2$ but not in **LIN**. An example of such a language is the set of all binary trees of Example 2.3 (cf. Theorem 16 of [6]). Define now for each $H \in L$ the graph $\text{dot}(H)$ by $\text{dot}(H) = (V, E, \Sigma, \Gamma, \varphi)$, where $V = V_H \cup \{\xi\}$ with ξ a new node, $\Sigma = \Sigma_H \cup \{\$\}$ with \$ a new node label, $\Gamma = \Gamma_H$, $\varphi(v) = \varphi_H(v)$ for each $v \in V_H$, $\varphi(\xi) = \$$, and $E = E_H \cup \{(v, \lambda, \xi) \mid v \in V_H, \lambda \in \Gamma\}$. Hence, $\text{dot}(H)$ is obtained from H by adding one node labeled \$ which is connected to all other nodes by a λ-labeled edge, for all $\lambda \in \Gamma_H$. Let $\text{dot}(L)$ be the set $\{\text{dot}(H) \mid H \in L\}$. It is not difficult to see that $\text{dot}(L) \in \mathbf{SEP}_2$, using $L \in \mathbf{SEP}_2$: first generate the \$-labeled node, and then simulate a 2-separated grammar generating L; it is easy to take care that the edges incident with the \$-labeled node are established.

Assume now that there exists an eNCE grammar G such that $L(G) = \text{dot}(L)$ and G is 3-separated. A contradiction will be derived from this. We may assume that G contains no Λ-productions (cf. Theorem 10 in [10]). If G would be linear, then L could also have been generated by a linear eNCE grammar (just remove the \$-labeled node from the right-hand sides of the productions of G). Since $L \notin$ **LIN**, it follows that G is not linear. Hence, there exists a derivation $\delta : S \Rightarrow^* H_1 \Rightarrow^* H_2$ in G such that $H_2 \in L(G)$ and H_1 contains (at least) two different nonterminal nodes. There are now two possibilities.

(1) The \$-labeled node is generated by one of the nonterminal nodes in H_1, say x. Let y be one of the other nonterminal nodes in H_1. Since G contains no Λ-productions, y generates at least one terminal node, which has to get connected to the \$-labeled node. Hence, there is an edge between x and y in H_1, contradicting the 3-separatedness of G.

(2) The \$-labeled node already appears in H_1. Consider now two different nonterminal nodes x and y in H_1. Since G contains no Λ-productions, both x and y generate at least one terminal node, each of which has to get connected to the \$-labeled node. Hence there is an edge between x (y, resp.) and the \$-labeled node, which means that the distance between x and y in H_1 is at most two. This again contradicts the 3-separatedness of G.

Hence, $\text{dot}(L)$ is in $\mathbf{SEP}_2$ but not in $\mathbf{SEP}_3$, which proves our theorem. □

As mentioned above we do not know whether the same result can be obtained for arbitrary k. But we conjecture that $\mathbf{SEP}_{k+1}$ is properly included in $\mathbf{SEP}_k$ for all $k \geqslant 3$, and we propose the following counter examples: the language generated by the eNCE grammar G_k defined in Fig. 3 is in $\mathbf{SEP}_k$ but does not seem to be in $\mathbf{SEP}_{k+1}$. In Fig. 4 a typical graph in $L(G_k)$ is drawn (for $k = 4$). We have removed the only edge label from the pictures. Note that $L(G_3)$ is precisely the set of all trees.

We will now argue that there do not exist "simpler" counter examples. In fact, all counter examples must be languages that cannot be generated by a nonterminal bounded eNCE grammar. An eNCE grammar G is *nonterminal bounded* if there is

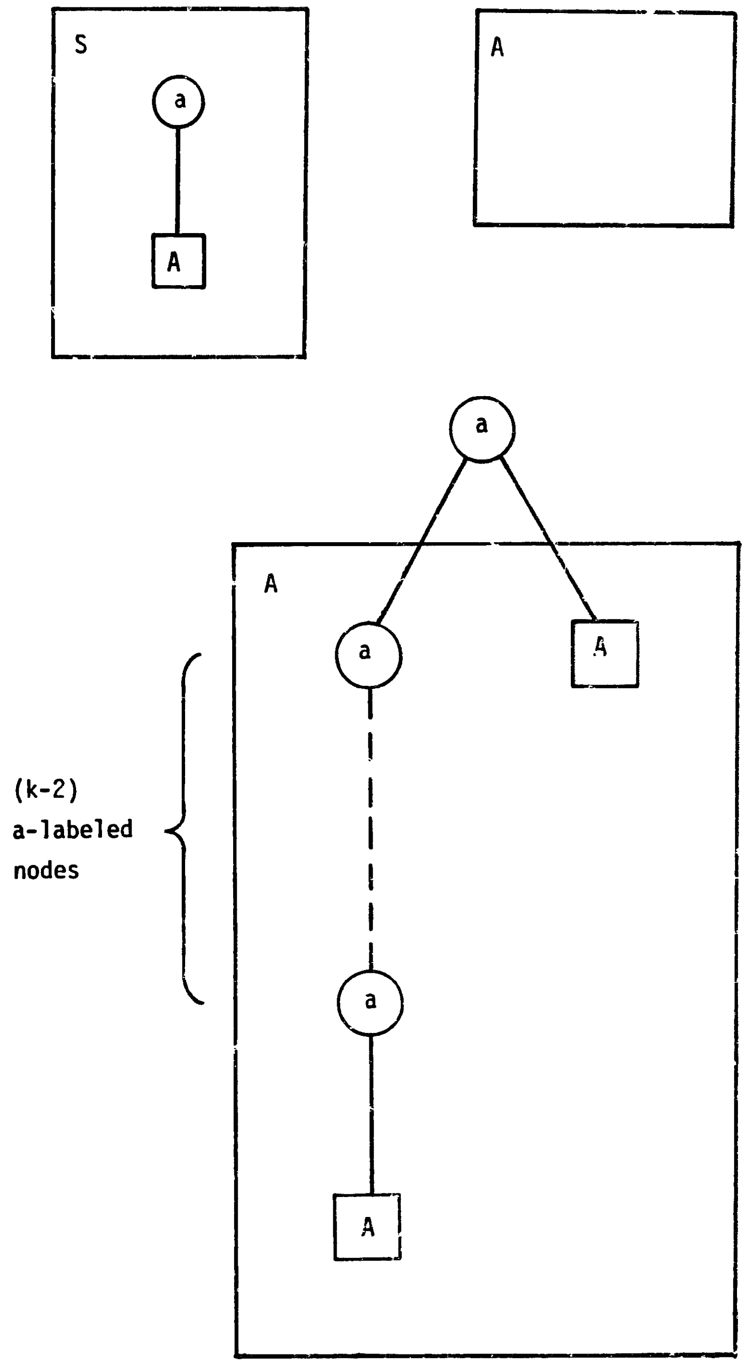

Fig. 3.

a number $n \geq 1$ such that each sentential form of G has at most n nonterminal nodes. Hence, a graph grammar which is *not* nonterminal bounded has for each $n \geq 1$ a sentential form with more than n nonterminal nodes. It is proved in [6] that each nonterminal bounded eNCE grammar is equivalent to a linear one. So, since $\mathbf{LIN} \subseteq \mathbf{SEP}_{k+1}$, any grammar generating a language in $\mathbf{SEP}_k$ but not in $\mathbf{SEP}_{k+1}$ is not nonterminal bounded. The next section shows moreover that such a grammar cannot be apex either.

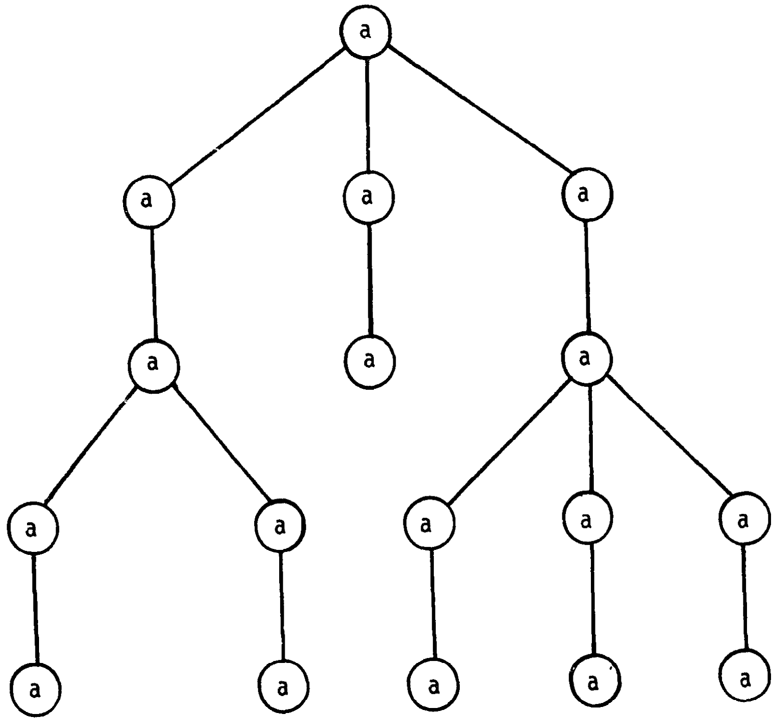

Fig. 4.

Finally we note that the above counter examples, and the one in the proof of Theorem 3.3, are all graph languages of unbounded degree (i.e., there is no bound on the degree of the nodes of the graphs in the language). We conjecture that $\mathbf{SEP}_2$ languages of bounded degree can be arbitrarily separated.

4. Apex graph grammars and separation

In this final section we consider apex eNCE graph grammars, and we show that demanding apex grammars to be k-separated has no influence on their generating power, in the sense that every apex eNCE language can be generated by a k-separated apex eNCE grammar for each $k \geq 1$. First we give the precise definition of an apex grammar (see also [8, 9]).

Definition 4.1. An eNCE grammar $G = (\Sigma, \Delta, \Gamma, \Omega, P, S)$ is an *apex eNCE grammar*, for short A-eNCE grammar, if for every production $\pi = (X, D, B) \in P$ the embedding relation B is a subset of $\{(x, \lambda, \mu, a) \in V_D \times \Gamma \times \Gamma \times \Sigma \mid \varphi_D(x) \in \Delta \text{ and } a \in \Delta\}$.

It is not difficult to see that both G_1 and G_2 of Example 2.3 are apex grammars. The class of all languages generated by A-eNCE grammars is denoted **A-eNCE**. Moreover, $\mathbf{A\text{-}SEP}_k$ (for $k \geq 1$) is used to denote the class of all languages that can be generated by an apex grammar which is k-separated. By definition $\mathbf{A\text{-}SEP}_1 =$ **A-eNCE**. Moreover, it is easy to see that each A-eNCE language can be generated by an A-eNCE grammar which is 2-separated (cf. Lemma 5 of [6]); hence, $\mathbf{A\text{-}SEP}_2 = \mathbf{A\text{-}eNCE}$. This section is devoted to proving that $\mathbf{A\text{-}SEP}_k = \mathbf{A\text{-}eNCE}$ for all $k \geq 1$. To prove this we need the following lemma. It states the essential fact that whenever two nonterminal nodes in a sentential form of a k-separated A-eNCE grammar have distance k, they have been generated in the same derivation step.

Lemma 4.2. *Let G be a k-separated A-eNCE grammar, for some $k \geq 2$, and let $\delta : S \Rightarrow_{(v_1,\pi_1,\beta_1)} H_1 \Rightarrow \cdots \Rightarrow_{(v_r,\pi_r,\beta_r)} H_r$ be a consistent derivation in G, with $r \geq 1$. Consider two distinct nonterminal nodes x and y in H_r such that the distance between x and y in H_r is k. Then* $\text{producer}_\delta(x) = \text{producer}_\delta(y)$.

Proof. We use induction on r to prove this statement.

$r = 1$. It is clear that in this case each node in H_1 is in the set $\beta_1(V_{\text{rhs}(\pi_1)})$, and so $\text{producer}_\delta(x) = \text{producer}_\delta(y) = 1$.

$r > 1$. Let δ' be the consistent derivation in G consisting of the first $r-1$ steps of δ. Hence, H_{r-1} is the result of δ'. We make the induction hypothesis that the lemma holds for δ'. Consider now two distinct nonterminal nodes x and y in H_r at distance k, and a path $w_0, w_1, \ldots, w_{k-1}, w_k$ in H_r of length k such that $w_0 = x$ and $w_k = y$. Four cases can be distinguished.

Case 1. x and y are both nodes in $\beta_r(V_{\text{rhs}(\pi_r)})$. In this case $\text{producer}_\delta(x) = \text{producer}_\delta(y) = r$.

Case 2. x and y are both in H_{r-1}. We argue that $x, w_1, \ldots, w_{k-1}, y$ is also a path in H_{r-1}. It clearly suffices to show that all nodes w_i on the path are in H_{r-1}. Assume, to the contrary, that i is the smallest number with $1 \leq i \leq k-1$ such that w_i is not a node in H_{r-1}. From the way the embedding mechanism of an eNCE grammar works it immediately follows that $x, w_1, \ldots, w_{i-1}, v_r$ is a path in H_{r-1} of length $i < k$; this contradicts the k-separatedness of G. Hence, indeed $x, w_1, \ldots, w_{k-1}, y$ is a path of length k in H_{r-1}, and so the induction hypothesis states that $\text{producer}_{\delta'}(x) = \text{producer}_{\delta'}(y)$. The lemma now follows from the easy observation that $\text{producer}_\delta(z) = \text{producer}_{\delta'}(z)$ for all nodes z which are both in H_r and in H_{r-1}.

Case 3. x is in H_{r-1}, but y is in $\beta_r(V_{\text{rhs}(\pi_r)})$. We will show that this situation cannot occur. Assume that i is the smallest number with $1 \leq i \leq k$ such that w_i is in $\beta_r(V_{\text{rhs}(\pi_r)})$. As above, it follows from the way the embedding mechanism of an eNCE grammar works, that $x, w_1, \ldots, w_{i-1}, v_r$ is a path in H_{r-1}. Since only terminal nodes can get connected by the embedding mechanism of an apex grammar, it follows moreover that w_i (and w_{i-1}) is a terminal node. Hence, since $w_k = y$ is nonterminal, $i \neq k$, and so there is a path of length $< k$ between the nonterminal nodes x and v_r in H_{r-1}. This is a contradiction, and so Case 3 cannot occur.

Case 4. y is in H_{r-1}, but x is in $\beta_r(V_{\text{rhs}(\pi_r)})$. This leads to a contradiction in the same way as in Case 3.

Hence, we have seen that Cases 3 and 4 cannot occur, whereas in Cases 1 and 2 indeed $\text{producer}_\delta(x) = \text{producer}_\delta(y)$. This proves the lemma. □

This lemma lays the foundation for the proof of the following theorem. The idea is that a k-separated apex grammar G can be turned into a $(k+1)$-separated grammar $\bar{G}$ by "contracting" the nonterminal nodes in the right-hand side of the productions of G into one new nonterminal node, which represents all the contracted nodes. One derivation step of $\bar{G}$ consists of applying a production of G to each of the contracted nodes. Lemma 4.2 shows that in this way there will remain no nonterminal nodes at distance k, since all such nodes are in the right-hand side of one production of G (and are thus contracted in $\bar{G}$). Similar constructions have been used in the proofs of Theorems 12 and 14 of [6].

Theorem 4.3. A-SEP$_k$ = A-SEP$_{k+1}$ *for all* $k \geq 1$.

Proof. Let $G = (\Sigma, \Delta, \Gamma, \Omega, P, S)$ be an A-eNCE grammar that is k-separated, for some k. According to Lemma 5 of [6] we may assume that there are no edges between nonterminal nodes in rhs(π), for all $\pi \in P$ (hence, $k \geq 2$). Assume that for distinct productions π and $\bar{\pi}$ in P, $V_{\text{rhs}(\pi)} \cap V_{\text{rhs}(\bar{\pi})} = \emptyset$. Assume, moreover, that for each $\pi \in P$ all nodes in apex(π) have distinct labels (this property is easy to obtain). Recall that apex(π) is the set of nonterminal nodes in rhs(π).

We will construct an A-eNCE grammar $\bar{G} = (\bar{\Sigma}, \Delta, \bar{\Gamma}, \Omega, \bar{P}, S)$ with $L(\bar{G}) = L(G)$ such that $\bar{G}$ is $(k+1)$-separated.

First, let $\bar{\Sigma} = \Delta \cup \{S\} \cup \{K \in \mathrm{GR}_{\Sigma-\Delta,\emptyset} \mid K = \mathrm{rhs}(\pi)[\mathrm{apex}(\pi)]$ for some $\pi \in P\}$, and $\bar{\Gamma} = \Gamma \cup \{\langle\lambda, x\rangle \mid \lambda \in \Gamma, x \in \mathrm{apex}(\pi)$ for some $\pi \in P\}$. Thus, each new symbol in $\bar{\Sigma}$ is the subgraph of the right-hand side of some production of G induced by the nonterminal nodes. According to the assumptions we made, these are all discrete graphs, of which the nodes are labeled by (different) elements of $\Sigma - \Delta$.

Second, we associate with each production $\pi \in P$ a graph contract(π) $\in \mathrm{GR}_{\bar{\Sigma},\bar{\Gamma}}$ which is constructed from rhs(π) as follows. If apex(π) $= \emptyset$ then contract(π) $=$ rhs(π), and otherwise contract(π) $= (V, E, \bar{\Sigma}, \bar{\Gamma}, \varphi)$, where

$$V = (V_{\text{rhs}(\pi)} - \mathrm{apex}(\pi)) \cup \{\xi_\pi\},$$

where ξ_π is for each production π a distinct new node,

$$E = \{(v, \lambda, w) \in E_{\text{rhs}(\pi)} \mid v, w \notin \mathrm{apex}(\pi)\}$$
$$\cup \{(v, \langle\lambda, x\rangle, \xi_\pi) \mid (v, \lambda, x) \in E_{\text{rhs}(\pi)}, v \notin \mathrm{apex}(\pi), x \in \mathrm{apex}(\pi)\},$$

$$\varphi(v) = \varphi_{\text{rhs}(\pi)}(v) \text{ if } v \neq \xi_\pi \quad \text{and} \quad \varphi(\xi_\pi) = \mathrm{rhs}(\pi)[\mathrm{apex}(\pi)].$$

Then we are now ready to define the productions in $\bar{P}$. To start with, if $\pi \in P$ is such that lhs(π) $= S$, then $\bar{P}$ contains the production $(S, \mathrm{contract}(\pi), \emptyset)$. Furthermore, if K is one of the discrete graphs in $\bar{\Sigma} - \Delta - \{S\}$, and if π_x is, for each $x \in V_K$, a production in P with lhs(π_x) $= \varphi_K(x)$, then $\bar{P}$ contains the production (X, D, B), where

$$X = K,$$

$$D = \bigcup_{x \in V_K} \mathrm{contract}(\pi_x),$$

$$B = \{(v, \langle\lambda, x\rangle, \mu, a) \mid (v, \lambda, \mu, a) \in \mathrm{emb}(\pi_x)\}.$$

Note that the two assumptions we made and the distinctness of the ξ_π's assure that the graphs contract(π_x) are mutually disjoint. Hence, D is well defined.

This concludes the construction of $\bar{G}$. In order to show that $L(\bar{G}) = L(G)$ and that $\bar{G}$ is $(k+1)$-separated, we need the following definitions and claim.

Consider a derivation

$$\delta : S \Rightarrow_{(v_1,\pi_1,\beta_1)} H_1 \Rightarrow \cdots \Rightarrow_{(v_r,\pi_r,\beta_r)} H_r$$

in G with $r \geq 1$, and let $H = H_r$. δ is called *apex-complete* if δ is consistent and, for each $1 \leq i \leq r$, either $\beta_i(\text{apex}(\pi_i)) \subseteq V_H$ or $\beta_i(\text{apex}(\pi_i)) \cap V_H = \emptyset$. Intuitively, this means that all nonterminal nodes with the same producer either have all been rewritten or have all not yet been rewritten. Furthermore, if δ is apex-complete, then the graph contract(δ) is constructed from H as follows: contract(δ) = $(V, E, \bar{\Sigma}, \bar{\Gamma}, \varphi)$ with

$$V = \{v \in V_H \mid \varphi_H(v) \in \Delta\}$$
$$\cup \{\xi_i \mid 1 \leq i \leq r, \text{apex}(\pi_i) \neq \emptyset, \beta_i(\text{apex}(\pi_i)) \subseteq V_H\},$$
$$E = \{(v, \lambda, w) \in E_H \mid \varphi_H(v), \varphi_H(w) \in \Delta\}$$
$$\cup \{(\beta_i(v), \langle\lambda, x\rangle, \xi_i) \mid 1 \leq i \leq r, \xi_i \in V, (v, \lambda, x) \in E_{\text{rhs}(\pi_i)},$$
$$v \notin \text{apex}(\pi_i), x \in \text{apex}(\pi_i)\},$$
$$\varphi(v) = \varphi_H(v), \quad \text{for each } v \in V_H \text{ with } \varphi_H(v) \in \Delta,$$
$$\varphi(\xi_i) = \text{rhs}(\pi_i)[\text{apex}(\pi_i)], \quad \text{for each } \xi_i \in V.$$

With these definitions we can state the following claim.

Claim. *A graph $\bar{H}$ is a sentential form of $\bar{G}$ if and only if $\bar{H} = S$ or $\bar{H}$ is isomorphic with* contract(δ), *for some apex-complete derivation δ in G.*

Proof (sketch). The $\Rightarrow$ part of the claim is not difficult to show. With respect to the $\Leftarrow$ part, we make some comments. Let $\delta : S \Rightarrow_{(v_1,\pi_1,\beta_1)} H_1 \Rightarrow \cdots \Rightarrow_{(v_r,\pi_r,\beta_r)} H$ be an apex-complete derivation in G with $r \geq 1$. The claim is obvious for $r = 1$. For $r \geq 2$ it follows from the apex-completeness of δ that there exists an s, $1 \leq s < r$, such that

- apex(π_s) $\neq \emptyset$,
- $\beta_s(\text{apex}(\pi_s)) \cap V_H = \emptyset$, and
- for every t, if $v_t \in \beta_s(\text{apex}(\pi_s))$, then $\beta_t(\text{apex}(\pi_t)) \subseteq V_H$.

Hence, s is such that each nonterminal node v_t in rhs(π_s) has been rewritten, but none of the nonterminal nodes in rhs(π_t) has yet been rewritten. By the confluency of apex grammars (cf. the Introduction) the derivation can be reordered in such a way that the elements of $\beta_s(\text{apex}(\pi_s))$ are rewritten at the end of the derivation. So we can assume that $\{v_{r-p+1}, \ldots, v_r\} = \beta_s(\text{apex}(\pi_s))$ where $p = \#\text{apex}(\pi_s)$, $p \geq 1$. The first $r-p$ steps of this derivation form an apex-complete derivation, for which we assume that the claim holds (induction hypothesis). The last p steps can be simulated by a single production (X, D, B) of $\bar{G}$, with $X = \text{rhs}(\pi_s)[\text{apex}(\pi_s)]$ and $D = \bigcup_{t \in I} \text{contract}(\pi_t)$, where $I = \{r-p+1, \ldots, r\}$. It is left to the reader to formalize the proof of this claim.

Since trivially each $H \in L(G)$ can be generated by an apex-complete derivation δ, and since $\text{contract}(\delta) = H$ in this case, it follows directly from the claim that $L(\bar{G}) = L(G)$.

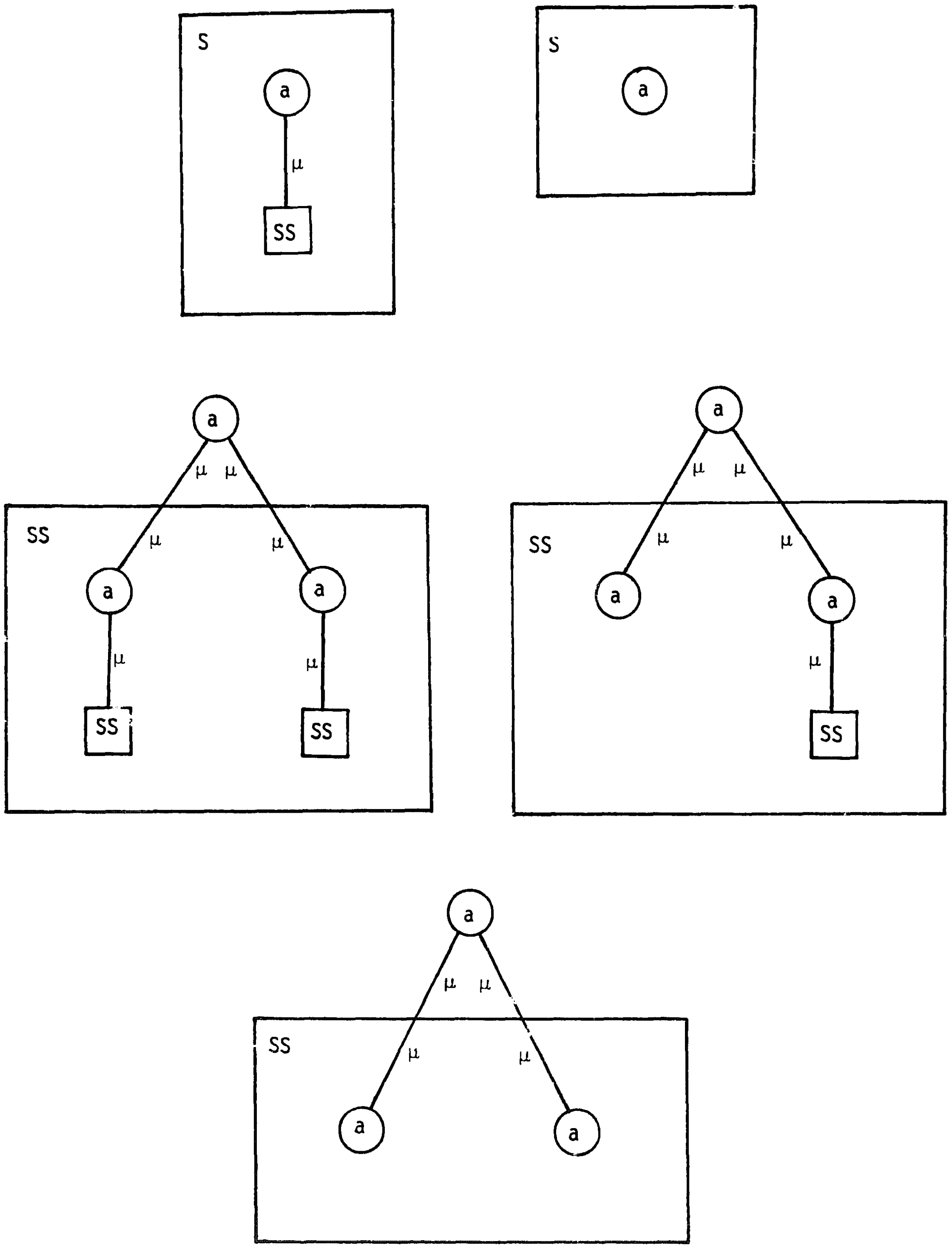

Fig. 5.

We next discuss the $(k+1)$-separability of $\bar{G}$. Consider a sentential form $\bar{H}$ of $\bar{G}$, and let $\delta : S \Rightarrow_{(v_1,\pi_1,\beta_1)} H_1 \Rightarrow \cdots \Rightarrow_{(v_r,\pi_r,\beta_r)} H_r$ be an apex-complete derivation in G such that $\bar{H}$ is isomorphic with contract(δ), cf. the claim above. Consider any path $w_0, w_1, \ldots, w_t$ in contract(δ) such that w_0 and w_t are distinct nonterminal nodes and w_1 to w_{t-1} are terminal nodes. We show that $t \geq k+1$. From the definition of contract(δ) it easily follows that there are nonterminal nodes x and y in H_r such that $x, w_1, \ldots, w_{t-1}, y$ is a path in H_r. Hence, since G is k-separated, $t \geq k$. Assume now that $t = k$. We will derive a contradiction from this. From Lemma 4.2 it follows that producer$_\delta(x)$ = producer$_\delta(y)$, and so x and y are generated in the same derivation step, say in the ith $(1 \leq i \leq r)$. Thus, $x, y \in \beta_i(\text{apex}(\pi_i))$, and $\beta_i(\text{apex}(\pi_i)) \subseteq V_{H_r}$. From the way in which contract(δ) is defined, it now directly follows that $w_0 = w_t = \xi_i$. This contradicts our assumption that w_0 and w_t are distinct nonterminal nodes, so

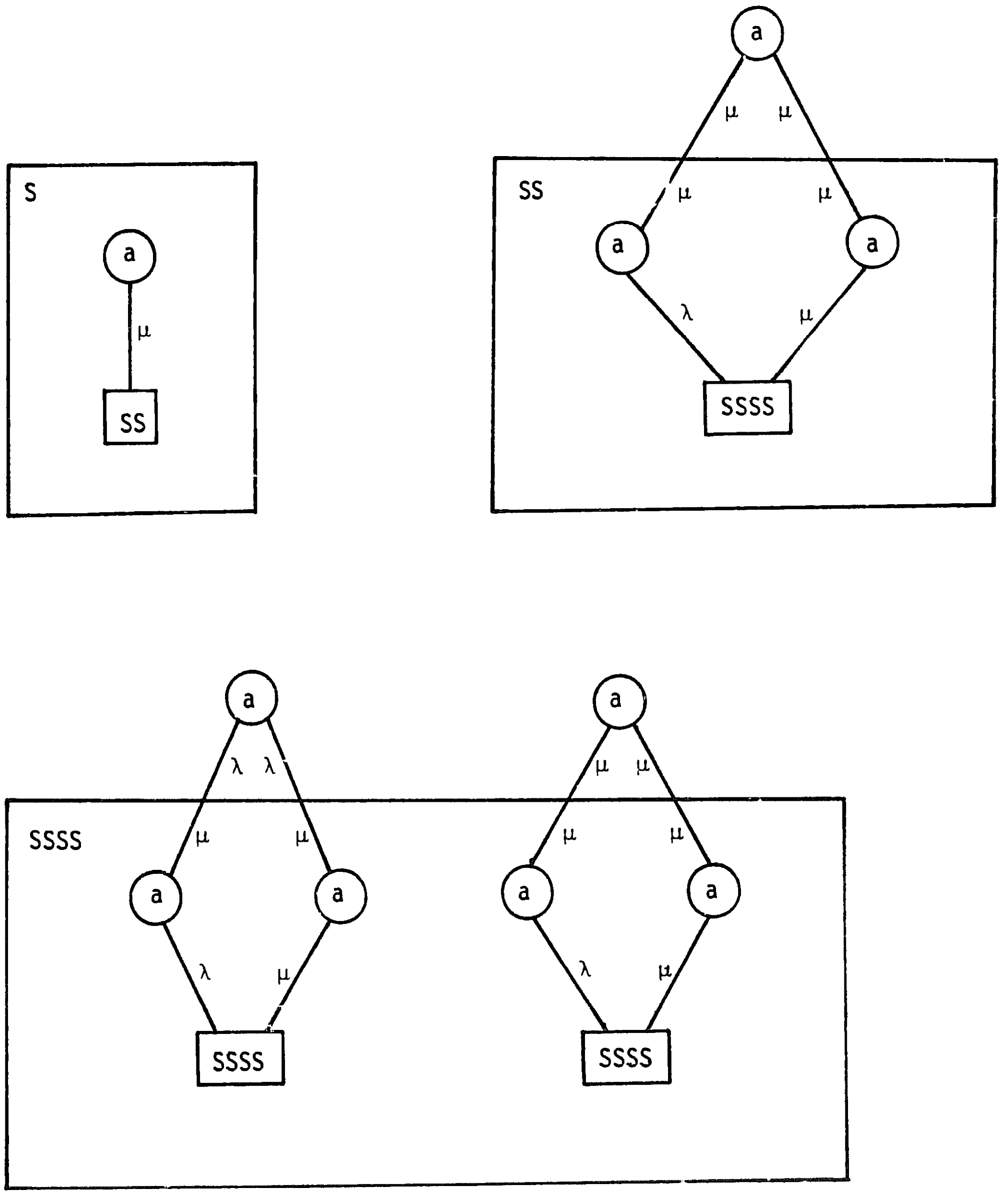

Fig. 6.

indeed $t \neq k$. Thus, $\bar{G}$ is $(k+1)$-separated, which proves the theorem. A more precise analysis would show that $\bar{G}$ is in fact $(k+2)$-separated (cf. the next example). □

Example 4.4. Consider the 2-separated apex grammar G_2 of Example 2.3, generating the set of all binary trees. A 4-separated apex grammar G_4 generating the same language is given in Fig. 5; this grammar is not $\bar{G}_2$ from the proof of Theorem 4.3, but it is close to it. Figure 6 shows some of the productions of a 6-separated apex grammar G_6 generating the binary trees. For each of the three grammars a derivation is sketched in Fig. 7. This figure illustrates clearly that a larger separation can be obtained by contracting nonterminal nodes with the same producer. It also shows that G_i is not $(i+1)$-separated, for $i = 2, 4, 6$.

From Theorem 4.3 it now follows directly that A-eNCE grammars can be arbitrarily separated.

Corollary 4.5. **A-eNCE = A-SEP$_k$** *for every* $k \geqslant 1$.

Finally, we observe that it has been shown in [6] that **LIN** and **A-eNCE** are incomparable, and so it follows that both **A-eNCE** and **LIN** are proper subsets of $\bigcap_{k\geqslant 1}$ **SEP**$_k$.

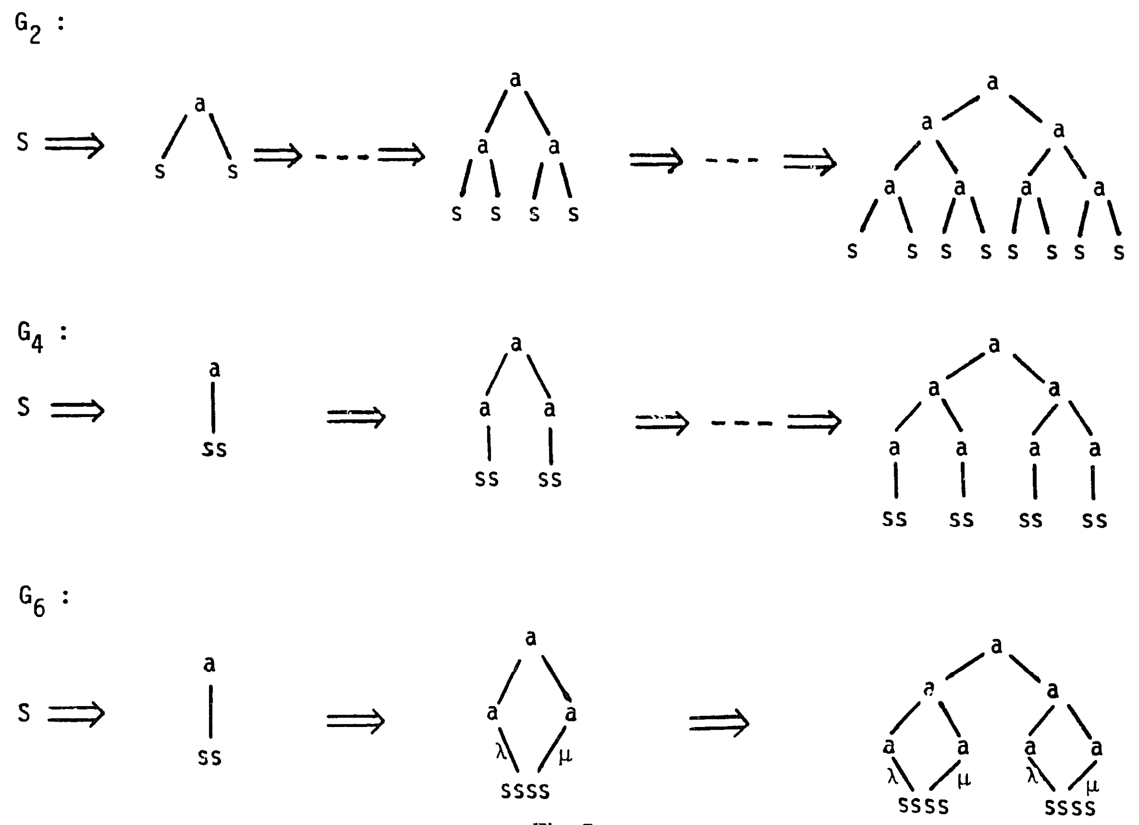

Fig. 7.

References

[1] M. Blattner and S. Ginsburg, Position-restricted grammar forms and grammars, *Theoret. Comput. Sci.* **17** (1982) 1-27.

[2] F.J. Brandenburg, On partially ordered graph-grammars, in: *Graph-Grammars and Their Application to Computer Science*, Lecture Notes in Computer Science **291** (Springer, Berlin, 1987) 99-111.

[3] B. Courcelle, An axiomatic definition of context-free rewriting and its application to NLC graph grammars, *Theoret. Comput. Sci.* **55** (1988) 141-182.

[4] H. Ehrig, M. Nagl and G. Rozenberg, eds., *Graph-Grammars and Their Application to Computer Science*, Lecture Notes in Computer Science **153** (Springer, Berlin, 1983).

[5] H. Ehrig, M. Nagl, G. Rozenberg and A. Rosenfeld, eds., *Graph-Grammars and Their Application to Computer Science*, Lecture Notes in Computer Science **291** (Springer, Berlin, 1987).

[6] J. Engelfriet and G. Leih, Linear graph grammars: power and complexity, *Inform. and Comput.* **81** (1989) 88-121.

[7] J. Engelfriet and G. Leith, Complexity of boundary graph languages, *RAIRO Inf. Théor. et Appl.* **24** (1990) 267-274.

[8] J. Engelfriet, G. Leih and G. Rozenberg, Apex graph grammars and attribute grammars, *Acta Inform.* **25** (1988) 537-571.

[9] J. Engelfriet, G. Leih and G. Rozenberg, Apex graph grammars, in: *Graph-Grammars and Their Application to Computer Science*, Lecture Notes in Computer Science **291** (Springer, Berlin, 1987) 167-185.

[10] J. Engelfriet, G. Leih and E. Welzl, Boundary graph grammars with dynamic edge relabeling, *J. Comput. System Sci.* **40** (1990) 307-345.

[11] J. Engelfriet and G. Rozenberg, A comparison of boundary graph grammars and context-free hypergraph grammars, *Inform. and Comput.* **84** (1990) 163-206.

[12] R.W. Floyd, Syntactic analysis and operator precedence, *J. ACM* **10**(3) (1963) 316-333.

[13] D. Janssens and G. Rozenberg, On the structure of node-label-controlled graph languages, *Inform. Sci.* **20** (1980) 191-216.

[14] D. Janssens and G. Rozenberg, Restrictions, extensions, and variations of NLC grammars, *Inform. Sci.* **20** (1980) 217-244.

[15] D. Janssens and G. Rozenberg, Graph grammars with neighbourhood-controlled embedding, *Theoret. Comput. Sci.* **21** (1982) 55-74.

[16] M. Kaul, Syntaxanalyse von Graphen bei Präzedenz-Graph-Grammatiken, Dissertation, Universität Osnabrück, 1985.

[17] M. Kaul, Practical applications of precedence graph grammars, in: *Graph-Grammars and Their Application to Computer Science*, Lecture Notes in Computer Science **291** (Springer, Berlin, 1987) 326-342.

[18] H.A. Maurer, A. Salomaa and D. Wood, On generators and generative capacity of E0L forms, *Acta Inform.* **13** (1980) 87-107.

[19] G. Rozenberg and E. Welzl, Boundary NLC graph grammars-basic definitions, normal forms, and complexity, *Inform. and Control* **69** (1986) 136-167.

[20] G. Rozenberg and E. Welzl, Graph theoretic closure properties of the family of boundary NLC graph languages, *Acta Inform.* **23** (1986) 289-309.

[21] G. Rozenberg and E. Welzl, Combinatorial properties of boundary NLC graph languages, *Discrete Appl. Math.* **16** (1987) 58-73.