GRAPH GRAMMARS WITH NEIGHBOURHOOD-CONTROLLED EMBEDDING

D. JANSSENS
Department of Mathematics, University of Antwerp (U.I.A.), B-2610 Wilrijk, Belgium

G. ROZENBERG
Institute of Applied Mathematics and Computer Science, University of Leiden, Leiden, The Netherlands

Communicated by A. Salomaa
Received January 1981
Revised May 1981

Abstract. A central feature that distinguishes graph grammars (we consider grammars generating sets of node-labelled undirected graphs only) from string grammars is that in the former one has to provide a mechanism by which a daughter graph (the right-hand side of a production) can be embedded in the rest of the mother graph, while in the latter this embedding is provided automatically by the structure that all strings possess (left-to-right orientation). In this paper we consider a possible classification of embedding mechanisms for (node-rewriting) graph grammars. This classification originates from the basic ideas of [9]. On the one hand it allows one to fit a number of existing notions of a graph grammar into a common framework and on the other hand it points out new "natural" possibilities for defining the embedding mechanism in a graph grammar. The relationship between the graph-language generating power of graph grammars using various embedding mechanisms is established.

Introduction

In this paper we will consider sequential graph grammars, that is, graph grammars in which in a direct derivation step one production is applied to one occurrence of its left-hand side only.

Considering various definitions of sequential graph grammars in the literature, one realizes that in almost all of them a direct derivation step consists of the following phases. In order to rewrite a graph \( H \), (we consider undirected, node-labelled graphs only), we select a production of the form \( \alpha \rightarrow \beta \) (where \( \alpha \) and \( \beta \) are graphs), we replace an occurrence of \( \alpha \) in \( H \) by a copy, say \( \hat{\beta} \), of \( \beta \), and then we embed \( \hat{\beta} \) into the rest of \( H \); that is, we establish edges between the nodes of \( \hat{\beta} \) and the remaining nodes of \( H \). The main difference between the various existing models lies in the definition of this embedding mechanism, and hence the embedding mechanism forms the "heart" of the definition of a graph grammar.

The aim of this paper is to introduce a systematic framework for discussing the notion of embedding in a graph grammar. Our framework generalizes various ideas...
originating from a number of existing definitions of the embedding mechanism. In particular it starts with the basic idea of the embedding mechanism used in [9]. Although we do not pretend to provide a framework encompassing all types of graph grammars known in the literature we believe that our framework singles out in a systematic way the basic features of a number of existing embedding mechanisms. In this way the generating power of various kinds of graph grammars can be compared and moreover new embedding mechanisms, very natural in our framework, become apparent.

The paper is organized as follows:

In Section 1 basic notions and notations concerning graphs are established. In Section 2 the notion of a graph grammar with neighbourhood-controlled embedding is introduced and illustrated by examples. In Section 3 we compare the generative power of different types of graph grammars defined in Section 2. In Section 4 we discuss how several notions of a graph grammar from the literature fit into our framework.

1. Preliminaries

In this section we settle basic terminology and notation (in particular concerning graphs) to be used in this paper.

(1) Let $X$ and $Y$ be sets. By $\text{Id}_X$ we denote the identity relation on $X$, by $2^X$ we denote the set of subsets of $X$ and by $X \setminus Y$ we denote the set $\{x \mid x \in X, x \notin Y\}$. If $X$ is finite, then $\neq X$ denotes the cardinality of $X$.

(2) Let $X$, $Y$, $Z$ be sets, let $f$ be a function from $X$ into $Y$ and let $g$ be a function from $Y$ into $Z$. By $g \circ f$ we denote the composition of $f$ and $g$ (first $f$, then $g$).

(3) A (undirected node-labelled) graph is a system $H = (V, E, \Sigma, \phi)$ where $V$ is a finite nonempty set, called the set of nodes, $E$ is a set of multisets of two elements from $V$, called the set of edges, $\Sigma$ is a finite nonempty set, called the set of labels (or the alphabet) and $\phi$ is a function from $V$ into $\Sigma$, called the labelling function. $H$ is called a graph over $\Sigma$. Throughout this paper the set of nodes, the set of edges, the set of labels and the labelling function of a graph $H$ will be denoted by $V_H$, $E_H$, $\Sigma_H$ and $\phi_H$ respectively.

(4) Let $H$ be a graph and let \{x, y\} \in E_H. We say that the edge \{x, y\} is incident with the nodes $x$ and $y$ and the nodes $x$ and $y$ are neighbours.

(5) Let $H$ be a graph and let $x \in V_H$. Then the degree of $x$, denoted by $\text{deg}(x)$, is the number of edges incident with $x$.

(6) Let $A$ and $B$ be graphs. $A$ is a subgraph of $B$ if $V_A \subseteq V_B$; $E_A \subseteq E_B \cap \{(x, y) \mid x, y \in V_A\}$, $\Sigma_A \subseteq \Sigma_B$ and for $x \in V_A$, $\phi_A(x) = \phi_B(x)$. $A$ is a full subgraph of $B$ if $A$ is a subgraph of $B$ and $E_A = E_B \cap \{(x, y) \mid x, y \in V_A\}$. In this case we call $A$ the subgraph spanned by $V_A$ in $B$. By $B - A$ we denote the subgraph spanned by $V_B \setminus V_A$ in $B$. 
(7) Let $H$ and $\tilde{H}$ be graphs over an alphabet $\Sigma$. An isomorphism from $H$ into $\tilde{H}$ is a bijective function $h$ from $V_H$ into $V_{\tilde{H}}$ such that $\phi_{\tilde{H}} \circ h = \phi_H$ and $E_H = \{ (h(x), h(y)) \mid (x, y) \in E_H \}$. We say that $H$ is isomorphic to $\tilde{H}$.

(8) A graph is complete if $E_H = \{ (x, y) \mid x, y \in V_H \}$.

(9) A graph $H$ is discrete if $E_H = \emptyset$.

(10) A graph $H$ is connected if for every $x, y \in V_H$ there exists a sequence $x_1, x_2, \ldots, x_n$ of nodes in $V_H$ such that $x_1 = x$, $x_n = y$ and for $1 \leq i \leq n - 1$, $x_i$ is a neighbour of $x_{i+1}$.

2. Basic definitions

**Definition 1.** A graph grammar with neighbourhood-controlled embedding, abbreviated NCE grammar, is a system $G = (\Sigma, \Delta, P, Z)$ where $\Sigma$ is a finite nonempty set, called the total alphabet, $\Delta$ is a subset of $\Sigma$, called the terminal alphabet, $P$ is a finite set of productions of the form $(\alpha, \beta, \psi)$ where $\alpha$ is a connected graph, $\beta$ is a graph and $\psi$ is a function from $V_\alpha \times V_\beta \times \Sigma$ into $\{0, 1\}$; $\psi$ is called the embedding function of the production, and $Z$ is a graph over $\Sigma$, called the axiom.

A direct derivation step in a NCE grammar is performed as follows. Let $H$ be a graph. Let $\pi = (\alpha, \beta, \psi)$ be a production of $P$, let $\tilde{\alpha}$ be a full subgraph of $H$ such that $\phi_{\tilde{\alpha}} \circ h = \phi_H$ and let $\tilde{\beta}$ be isomorphic to $\beta$ (with $g$ being an isomorphism from $\tilde{\beta}$ into $\beta$) where $V_{\tilde{\beta}} \cap V_{H - \tilde{\alpha}} = \emptyset$. Then the result of the application of $\pi$ to $\tilde{\alpha}$ (using $h, g$) is obtained by first removing $\tilde{\alpha}$ from $H$, then replacing $\tilde{\alpha}$ by $\tilde{\beta}$ and finally adding edges $\{n, v\}$ between every $n \in V_{\tilde{\beta}}$ and every $v \in V_H \setminus V_{\tilde{\alpha}}$ such that

1. there exists a node $m \in V_{\alpha}$ with $(h(m), v) \in E_H$ and
2. $\psi(m, g(n), \phi_H(v)) = 1$.

In this situation $\tilde{\alpha}$ is called the mother graph, $\tilde{\beta}$ is called the daughter graph and $H$ is called the host graph.

Note that the embedding function $\psi$ explicitly specifies which nodes of $\tilde{\beta}$ can be connected to nodes of $H - \tilde{\alpha}$ that are neighbours of nodes in $\tilde{\alpha}$. Also, $\psi$ explicitly species nodes in $\tilde{\alpha}$ the neighbours of which can be connected to nodes in $\tilde{\beta}$. However $\psi$ cannot explicitly specify which neighbours of $\tilde{\alpha}$ can be connected to nodes in $\tilde{\beta}$ for the simple reason that, in general, the number of such neighbours cannot be a priori limited, while the specification of a NCE grammar must remain finite. Hence $\psi$ is a function from $V_\alpha \times V_{\tilde{\beta}} \times \Sigma$; thus the only way we can specify which neighbours of $\tilde{\alpha}$ can be connected to nodes of $\tilde{\beta}$ is by specifying them by their labels.

Formally the notion of a direct derivation step is defined as follows.

**Definition 2.** Let $G = (\Sigma, \Delta, P, Z)$ be a NCE grammar.

(1) Let $H$ and $\tilde{H}$ be graphs over $\Sigma$. Then $H$ directly derives $\tilde{H}$ in $G$, denoted $H \Rightarrow_\pi \tilde{H}$, if there exists a production $\pi = (\alpha, \beta, \psi)$ in $P$, an isomorphism $h$ from $\alpha$
into a full subgraph $\hat{\alpha}$ of $H$, a graph $\hat{\beta}$ with $V_{\hat{\beta}} \cap V_H = \emptyset$ and an isomorphism $g$ from $\hat{\beta}$ into $\beta$ such that $\tilde{H}$ is isomorphic to the graph $X$ constructed as follows:

$$X = (V_X, E_X, \Sigma, \phi_X)$$

where

$$V_X = (V_H \setminus V_{\hat{\alpha}}) \cup V_{\hat{\beta}},$$

$$E_X = \{\{x, y\} | x, y \in V_H \setminus V_{\hat{\alpha}} \text{ and } \{x, y\} \in E_H\}$$

$$\cup \{\{x, y\} | x, y \in V_{\hat{\beta}} \text{ and } \{g(x), g(y)\} \in E_{\hat{\beta}}\}$$

$$\cup \{\{x, y\} | x \in V_{\hat{\beta}}, y \in V_H \setminus V_{\hat{\alpha}} \text{ and there exists a node } m \in V_{\alpha} \text{ such that } \{h(m), y\} \in E_H \text{ and } \psi(m, \check{g}(x), \phi_{\hat{H}}(y)) = 1\}$$

and $\phi_X$ is defined by

$$\begin{cases} 
\phi_X(y) = \phi_H(y) & \text{if } y \in V_H \setminus V_{\hat{\alpha}}, \\
\phi_X(y) = \phi_{\hat{\beta}}(y) & \text{if } y \in V_{\hat{\beta}}.
\end{cases}$$

We also say that $\tilde{H}$ is derived from $H$ by replacing $\hat{\alpha}$ using the production $\pi$.

(2) We will denote the reflexive and transitive closure of $\Rightarrow_G$ by $\Rightarrow_G^*$ and the transitive closure of $\Rightarrow_G$ by $\Rightarrow_G^+.$

(3) The language of $G$, denoted by $L(G)$, is defined by $L(G) = \{H | H \text{ is a graph over } \Delta \text{ and } 2 \Rightarrow_G^* H\}.$

In the sequel we will use $\max r G$ to denote $\max \{|\# V_H | \text{there exists a production } (\alpha, \beta, \psi) \text{ in } P\}.$

**Example 1.** The following NCE grammar $G$ generates the set of all "rectangular grids" of the form depicted in Fig. 1:

$$G = (\Sigma, \Delta, P, Z)$$
where

\[ \Sigma = \{a, b, c, d\}, \quad \Delta = \{a, b, d\}, \quad Z = \]

\[ \psi(1, 3, l) = \psi(2, 4, l) = 1 \]

for each \( i \in \Sigma \) and \( \psi(x, y, z) = 0 \) in all other cases,

\[ \psi(1, 3, l) = \psi(2, 4, l) = 1 \]

for each \( i \in \Sigma \) and \( \psi(x, y, z) = 0 \) in all other cases,

\[ \psi(1, 3, l) = \psi(2, 4, l) = 1 \]

for each \( i \in \Sigma \) and \( \psi(x, y, z) = 0 \) in all other cases,
for each $l \in \Sigma$ and $\psi(x, y, z) = 0$ in all other cases).

An example of a derivation step is given in Fig. 2.

In general, a production of a NCE grammar is of the form $(\alpha, \beta, \psi)$. In this paper we will restrict our attention to systems in which $\alpha$ is a discrete graph with one node only (so we restrict ourselves to node-rewriting systems).

**Definition 3.** A $1$-$NCE$ grammar is a NCE grammar $(\Sigma, \Delta, P, Z)$ such that each production in $P$ is of the form $(\alpha, \beta, \psi)$ with $\# V_\alpha = 1$ and $E_\alpha = \emptyset$.

Let $(\alpha, \beta, \psi)$ be a production in a $1$-$NCE$ grammar. Then clearly $\psi$ corresponds in a natural way to a function from $V_\beta \times \Sigma$ into $\{0, 1\}$. Hence in the sequel, we assume that the productions of a $1$-$NCE$ grammar are given in the form $(\alpha, \beta, \psi)$ where $\psi$ is a function from $V_\beta \times \Sigma$ into $\{0, 1\}$. 
For 1-NCE grammars the notions of a derivation, a descendant and an ancestor are defined as follows.

**Definition 4.** Let $G = (\Sigma, \Delta, P, \mathcal{Z})$ be a 1-NCE grammar. A derivation sequence $D$ in $G$ is a sequence of graphs $H_0$, $H_1$, ..., $H_r$ such that $r \geq 1$ and for $0 \leq i \leq r - 1$, $H_i$ directly derives $H_{i+1}$, together with a sequence of functions $F_1, F_2, \ldots, F_r$ where for $1 \leq j \leq r$, $F_j$ is a function from $V_{H_j}$ into $V_{H_{j-1}}$. The $F_j$'s are related to the $H_j$'s as follows: let $H_{j-1}$ directly derive $H_j$ by replacing a node $v$ using the production $(\alpha, \beta, \psi)$. Let $X, \beta$ be defined as in Definition 2 (where we set $H = H_{j-1}$ and $iJ = H_j$) and let $f$ be the isomorphism from $H_j$ into $X$. Then

$$F_j(x) = \begin{cases} \alpha & \text{if } f(x) \in V_{\beta} \\ \beta & \text{if } f(x) \in V_{H_j} \setminus \{v\} \end{cases}$$

For $0 \leq i < j \leq r$ we define the function $F_{ij}$ from $V_{H_j}$ into $V_{H_i}$ by $F_{ij} = F_{i+1} \circ F_{i+2} \cdots \circ F_j$; also for $1 \leq l \leq r$ $F_{il} = id_{V_{H_l}}$. For $x \in V_{H_i}$ we call $F_{ij}(x)$ the ancestor of $x$ in $H_i$ and $x$ is called a descendant of $F_{ij}(x)$ in $H_j$. $D$ is called a derivation if $H_0 = \mathcal{Z}$ and $H_r$ is a graph over $\Delta$.

In a production $(\alpha, \beta, \psi)$ of a 1-NCE grammar, $\psi$ is a function of two arguments. Depending on the fact whether or not, for a given argument, $\psi$ depends on this argument (or, in the case of the first argument $V_\beta$, whether or not $\psi$ depends only on the label of the argument) we get the following "natural" subclasses of the class of 1-NCE grammars.

**Definition 5.** Let $G = (\Sigma, \Delta, P, \mathcal{Z})$ be a 1-NCE grammar. Then $G$ is a $(X, Y)$ grammar for each $X \in \{0, 1, 2 \}$ and $Y \in \{0, 1\}$ that satisfy the following conditions:

(a) If there exists a production $(\alpha, \beta, \psi)$ in $P$, nodes $x, y$ in $V_\beta$ and a label $l \in \Sigma$ such that $\psi(x, l) \neq \psi(y, l)$, then $X \geq 1$.

(b) If there exists a production $(\alpha, \beta, \psi)$ in $P$, nodes $x, y$ in $V_\beta$ and a label $l \in \Sigma$ such that $\phi_\beta(x) = \phi_\beta(y)$ and $\psi(x, l) \neq \psi(y, l)$, then $X = 2$.

(c) If there exists a production $(\alpha, \beta, \psi)$ in $P$, a node $x \in V_\beta$ and labels $l_1, l_2 \in \Sigma$ such that $\psi(x, l_1) \neq \psi(x, l_2)$ then $Y = 1$.

Thus, intuitively, $X = 0$ implies that $\psi$ is not dependent on the nodes of the daughter graph $B$ and $X = 1$ implies that, although $\psi$ can distinguish between different nodes of the daughter graph, $\psi$ cannot distinguish two nodes of the daughter graph labelled in the same way. $Y = 0$ implies that $\psi$ cannot distinguish between any two neighbours of the rewritten node (even if they have different labels). Observe that a "$(2, 1)$ grammar" is in this way a synonym for a "1-NCE grammar".

**Example 2.** The following $(1, 1)$ grammar generates the set of all graphs of the form of Fig. 3.
\[ G = (\Sigma, \Delta, P, Z) \]

where

\[ \Sigma = \{ S, a, b, c \}, \quad \Delta = \{ a, b, c \}, \quad Z = \emptyset \]

and

\[ P = \{ (s, \psi_1), (s, \psi_2) \} \]

where

\[ \psi_1(1, a) = 1, \quad \psi_1(1, b) = 1, \]
\[ \psi_1(2, b) = 1, \quad \psi_1(2, c) = 1, \]
\[ \psi_1(3, c) = 1, \quad \psi_1(3, c) = 1, \]
\[ \psi_1(x, y, z) = 0 \text{ in the other cases and } \psi_2 \text{ is defined in exactly the same way as } \psi_1. \]

It is obvious that an embedding function in a \((1, 1)\) grammar \((\Sigma, \Delta, P, Z)\) can be considered as a function on \(\Sigma \times \Sigma\) rather than on \(V_P \times \Sigma\). Similar observations for \((2, 0)\), \((1, 0)\), \((0, 1)\) and \((0, 0)\) grammars give rise to the following definitions.

**Definition 6.** (1) Let \((\Sigma, \Delta, P, Z)\) be a \((2, 0)\) grammar and let \(\pi = (\alpha, \beta, \psi) \in P.\)
Then the \((2,0)\)-reduced embedding function of \(\pi\), denoted by \(\psi^{(2,0)}\), is the function from \(V_\beta\) into \{0, 1\} defined by \(\psi^{(2,0)}(x) = 1\) if and only if there exists a \(l \in \Sigma\) with \(\phi(x, l) = 1\).

(2) Let \((\Sigma, \Delta, P, Z)\) be a \((1,1)\) grammar and let \(\pi = (\alpha, \beta, \psi) \in P\). Then the \((1,1)\)-reduced embedding function of \(\pi\), denoted by \(\psi^{(1,1)}\), is the function from \(\Sigma \times \Sigma\) into \{0, 1\} defined by \(\psi^{(1,1)}(l_1, l_2) = 1\) if and only if there exists an \(x \in V_\beta\) where \(\phi(x) = 1\) and \(\psi(x, l_2) = 1\).

(3) Let \((\Sigma, \Delta, P, Z)\) be a \((1,0)\) grammar and let \(\pi = (\alpha, \beta, \psi) \in P\). Then the \((1,0)\)-reduced embedding function of \(\pi\), denoted by \(\psi^{(1,0)}\), is the function from \(\Sigma\) into \{0, 1\} defined by \(\psi^{(1,0)}(l) = 1\) if and only if there exists a \(x \in V_\beta\) with \(\phi(x) = 1\) and \(\bar{l} \in \Sigma\) where \(\psi(x, l) = 1\).

(4) Let \((\Sigma, \Delta, P, Z)\) be a \((0,1)\) grammar and let \(\pi = (\alpha, \beta, \psi) \in P\). Then the \((0,1)\)-reduced embedding function of \(\pi\), denoted by \(\psi^{(0,1)}\), is the function from \(\Sigma\) into \{0, 1\} defined by \(\psi^{(0,1)}(l) = 1\) if and only if there exists a \(x \in V_\beta\) such that \(\psi(x, l) = 1\).

(5) Let \((\Sigma, \Delta, P, Z)\) be a \((0,0)\) grammar and let \(\pi = (\alpha, \beta, \psi) \in P\). Then the embedding constant of \(\pi\), denoted by \(\psi^{(0,0)}\), is defined by \(\psi^{(0,0)}(l) = 1\) if and only if there exists a \(x \in V_\beta\) and \(l \in \Sigma\) with \(\psi(x, l) = 1\) and \(\psi^{(0,0)} = 0\) in the other case.

**Remark 1.** Let \(G = (\Sigma, \Delta, P, Z)\) be a \((1,1)\) grammar. Let \(P'\) be obtained from \(P\) by replacing in every element \((\alpha, \beta, \psi)\) of \(P\) the embedding function by the \((1,1)\)-reduced embedding function. Then clearly \(P'\) uniquely defines \(P\). Similar observation hold for \((2,0), (1,0), (0,2)\) and \((0,0)\) grammars. Hence in the sequel we will also use reduced embedding functions in specifications of 1-NCE grammars.

In the case of a \((X, Y)\) grammar \(G\) with \(X \in \{0, 1\}\) it may happen that the \((X, Y)\)-reduced embedding functions of all productions in \(G\) are identical. In other words, in the specification of \(G\) one can provide one embedding function common to all productions. In this case we deal with a global specification of the embedding mechanism. This situation is formally described as follows.

**Definition 7.** Let \(X, Y \in \{0, 1\}\) and let \(G = (\Sigma, \Delta, P, Z)\) be a \((X, Y)\) grammar. Then \(G\) is a global \((X, Y)\) grammar, abbreviated \((X, Y)_{\kappa}\) grammar, if for each pair of productions \((\alpha, \beta, \psi), (\bar{\alpha}, \bar{\beta}, \bar{\psi}) \in P\) we have \(\psi^{(X,Y)} = \bar{\psi}^{(X,Y)}\).

In specifying a global \((X, Y)\) grammar we will provide one (reduced) embedding function (common to all productions) and productions themselves will be given in the form \((\alpha, \beta)\).

**Example 3.** The following \((0,1)_{\kappa}\) grammar generates the set of all graphs over the one-letter alphabet \(\{a\}\):

\[G = (\Sigma, \Delta, P, Z)\]
\[ \Sigma = \{A, a\}, \quad \Delta = \{a\}, \quad Z = \bullet, \]

and

\[ P = \{(\bullet, \bullet) \rightarrow (\bullet, \bullet), (\bullet, \bullet), (\bullet, \bullet)\}, \]

\[ \psi^{(0,1)}(a) = 1 \quad \text{and} \quad \psi^{(0,1)}(A) = 0. \]

**Remark 2.** (1) Observe that our definition of a \( (1, 1) \) grammar is equivalent to that of a NLC grammar in [5]. Furthermore, our definitions of \( (1, 1) \) grammars, \( (1, 0) \) grammars, and \( (1, 0) \) grammars correspond to the definitions of RNLC grammars, CFRNLC grammars and CFNLC grammars (from [5]) respectively.

(2) From Theorem 3 of [5] it follows that \( (1, 0) \) grammars have the property that the order in which the steps of a derivation are performed does not affect the resulting graph.

3. Comparing the generative power

In this section we will compare the generative power of various types of graph grammars defined in Section 2. If \( Z \) denotes a type of graph grammars then we will use \( \mathcal{L}(Z) \) to denote the class of all graph languages generated by \( Z \)-grammars. Thus, e.g., \( \mathcal{L}(2, 0) \) denotes the class of graph languages generated by all \( (2, 0) \) grammars.

In the proofs we will often use properties of derivations. To simplify the formalism the functions \( F_1, F_2, \ldots, F \) (from Definition 4) will not be used explicitly in specifying a derivation. However this should not lead to a confusion, because whenever the notions of an ancestor and a descendant are used, it will be clear from the context with respect to which derivation they are defined.

**Lemma 1.** \( \mathcal{L}(1-NCE) = \mathcal{L}(1, 1) = \mathcal{L}(1, 1)_c. \)

**Proof.** Since the equality \( \mathcal{L}(1, 1) = \mathcal{L}(1, 1)_c \) follows from (1) of Remark 2 and Theorem 6 of [5], it is sufficient to show that \( \mathcal{L}(1-NCE) = \mathcal{L}(1, 1) \). The inclusion \( \mathcal{L}(1, 1) \subseteq \mathcal{L}(1-NCE) \) follows directly from the definition. To show that \( \mathcal{L}(1-NCE) \subseteq \mathcal{L}(1, 1) \) we proceed as follows. Let \( G = (\Sigma, \Delta, P, Z) \) be an arbitrary 1-NCE system.

Let \( \pi = (\alpha, \beta, \psi) \) be a production in \( P \) not satisfying the restrictions of the definition of a \( (1, 1) \) grammar, that is, there exist \( x, y \in V_{\beta} \) and \( \ell \in \Sigma \) with \( \phi_{\beta}(x) = \psi(y, \ell) \) and \( \psi(x, \ell) \neq \psi(y, \ell) \) We construct the 1-NCE system \( \tilde{G} = (\tilde{\Sigma}, \tilde{\Delta}, \tilde{P}, Z) \) in the following way: \( \tilde{\Sigma} = \Sigma \cup \{ (\phi_{\beta}(x), x) | x \in V_{\beta} \} \) and \( \tilde{P} \) is the union of the following three sets of productions:
(1) The production \((\alpha, \tilde{\beta}, \tilde{\psi})\) where \(\tilde{\beta}\) is obtained from \(\beta\) by replacing the label \(\phi_\beta(x)\) of each note \(x\) by the pair \((\phi_\beta(x), x)\) and \(\tilde{\psi}\) is defined by
\[
\begin{align*}
\tilde{\psi}(x, l) &= \psi(x, l) & \text{if } l \in \Sigma, \\
\tilde{\psi}(x, l) &= \psi(x, f) & \text{if } l \in \tilde{\Sigma} \setminus \Sigma \text{ and } l = (f, y).
\end{align*}
\]

(2) The set of productions \((\gamma, \delta, \chi)\) where \((\gamma, \delta, \chi) \in P \setminus \{(\alpha, \beta, \psi)\}\) and such that \(\chi\) is defined by
\[
\begin{align*}
\chi(x, l) &= (x, l) & \text{if } l \in \Gamma, \\
\chi(x, l) &= (x, f) & \text{if } l \in \tilde{\Sigma} \setminus \Sigma \text{ and } l = (f, y).
\end{align*}
\]

(3) The set of productions
\[
(\tilde{\beta}, \gamma) \rightarrow (\tilde{\beta}, \chi)
\]

where \((f, y) \in \tilde{\Sigma} \setminus \Sigma\) and \(\chi(x, l) = 1\) for each \(l \in \tilde{\Sigma}\).

It is straightforward to verify that \(L(G) = L(G)\). Clearly the number of productions in \(\tilde{P}\) not satisfying the restrictions of the definition of a \((1, 1)\) grammar is smaller than in \(P\). By iterating the construction we obtain a \((1, 1)\) grammar \(M\) with \(L(G) = L(M)\). This proves the lemma. \(\Box\)

**Lemma 2.** \(L(2, 0) = L(1, 0) = L(1, 0)\).

**Proof.** \(L(1, 0) = L(1, 0)\) is shown in the remark following Theorem 6 in [5]. The inclusion \(L(1, 0) \subseteq L(2, 0)\) follows directly from the definition. To see that \(L(2, 0) \subseteq L(1, 0)\), observe that the grammar \(M\) from the proof of Lemma 1 is a \((1, 0)\) grammar if \(G\) is a \((2, 0)\) grammar. Hence, given an arbitrary \((2, 0)\) grammar \(G\) the same construction can be used to obtain a \((1, 0)\) grammar \(M\) with \(L(G) = L(M)\). This proves the lemma. \(\Box\)

**Lemma 3.** \(L(0, 1) \subseteq L(1, 0)\).

**Proof.** Consider the \((0, 1)\) grammar
\[G = (\Sigma, A, P, Z)\]
where
\[
\Sigma = \{a, \bar{a}, b, \bar{b}, S\}, \quad \Delta = \{a, b\},
\]
\[
Z = \begin{array}{c}
S \\
\bar{S} \\
a \\
\bar{a} \\
b \\
\bar{b} \\
A \\
\Delta \\
\end{array}
\]
\[
P = \{(a, \bullet, \bullet, \bullet), (a, \bullet, a), (a, a), (\bar{a}, \bar{a}), (\bar{a}, a), (a, \bar{a}), (a, \bar{a}), (\bar{b}, b), (\bar{b}, b), (b, b), (b, b), (\bullet, \bullet), (\bullet, \bullet), (\bullet, \bullet)\}.
\]
and the \((0, 1)\)-reduced embedding function \(\psi^{(0,1)}\) is defined by \(\psi^{(0,1)}(a) = \psi^{(0,1)}(b) = \psi^{(0,1)}(S) = 1\) and \(\psi^{(0,1)}(\bar{a}) = \psi^{(1,0)}(\bar{b}) = 0\). It is obvious that in every graph \(H\) of \(L(G)\) the number of \(a\)-labelled nodes equals the number of \(b\)-labelled nodes. It is also clear that \(L(G)\) contains the set of graphs of the form of Fig. 4: an example of a derivation of such a graph is given in Fig. 5.

Fig. 4.

Fig. 5.
Graph grammars with neighbourhood-controlled embedding

Fig. 5 (cont.)
We show that $L(G)$ is not in $\mathcal{L}(1,0)_k$. Indeed, assume that $G = (\Sigma, \Delta, P, Z)$ is a $(1,0)$ grammar with $L(\tilde{G}) = L(G)$. Without loss of generality we may assume that $\# V_x = 1$. Let $H$ be a graph of the form of Fig. 4 with $n > \text{max} G^\Sigma$ and let $\alpha = H_0 \Rightarrow C H_1 \Rightarrow C H_2 \Rightarrow C \ldots \Rightarrow C H_r = H$ be a derivation of $H$ in $\tilde{G}$. Let $i: 0 \leq i \leq r - 1$ be the maximal index with the property that there exists a $j, i < j \leq r$ and nodes $x, y$ such that

(i) $y$ is a descendant of $x$, $\phi_{H_i}(x) = \phi_{H_i}(y)$, $x$ is the node rewritten in the derivation step $H_i \Rightarrow C H_{i+1}$ and either $j = r$ or $y$ is the node rewritten in the derivation step $H_i \Rightarrow C H_{i+1}$, and

(ii) $x$ has at least 2 descendants in $H_j$.

From the assumption that $n > \text{(max} G^\Sigma$ it easily follows that there exists such an index $i$. Furthermore from (2) of Remark 2 it follows that we may assume that all nodes rewritten in the derivation steps $H_i \Rightarrow C H_{i+1} \Rightarrow \ldots \Rightarrow C H_r$ are descendants of $y$. From (i) it now follows that we can construct other derivations in $\tilde{G}$ by iterating an arbitrary number of times the part of $\alpha$ in which descendants of $x$ (but not of $y$) are rewritten, leaving the remaining part of $\alpha$ unchanged. Since in every graph of $L(\tilde{G})$ the number of nodes labelled by $a$ equals the number of nodes labelled by $b$, (ii) implies that there exists $p_1, p_2 \in V_{H_r}$ with $\phi_{G_r}(p_1) = a$, $\phi_{G_r}(p_2) = b$ and both $p_1, p_2$ are descendants of $x$.

From our assumptions on $\alpha$ it also follows that for each $v \in V_{H_r \setminus \{x\}}$ we have $\phi_{H_r}(v) \in \{a, b\}$. Moreover from the fact that $i$ is maximal and from $n > \text{(max} G^\Sigma$ it follows that there exist nodes $v_1, v_2 \in V_{H_r}$ such that $\phi_{H_r}(v_1) = a$ and $\phi_{H_r}(v_2) = b$. Because the subgraphs of $H$ spanned by the $a$-labelled nodes and the $b$-labelled nodes respectively are connected, it now follows that there exist neighbours $q_1, q_2$ of $x$ in $H_i$ and descendants $\tilde{p}_1, \tilde{p}_2$ of $x$ in $H$ such that $\phi_{H_i}(q_1) = a$, $\phi_{H_i}(q_2) = b$, $\phi_{H_i}(\tilde{p}_1) = a$, $\phi_{H_i}(\tilde{p}_2) = b$. $\tilde{p}_1$ is a neighbour of the (unique) descendant $\tilde{q}_1$ of $q_1$ in $H$ and $\tilde{p}_2$ is a neighbour of the (unique) descendant $\tilde{q}_2$ of $q_2$ in $H$. Clearly we have $\phi_H(\tilde{q}_1) = a$ and $\phi_H(\tilde{q}_2) = b$. However, from the definition of a $(1,0)_k$ system and from the assumptions about $\alpha$ it follows easily that this implies that $\{\tilde{p}_1, \tilde{q}_2\}$ is a contradiction because no $a$-labelled node of $H$ is a neighbour of a $b$-labelled node of $H$. $\square$

**Lemma 4.** $\mathcal{L}(1,0)_k \mathcal{L}(0,1) \neq \emptyset$.

**Proof.** Consider the $(1,0)_k$ grammar $G = (\Sigma, \Delta, P, Z)$ where $\Sigma = \{a, S\}$, $\Delta = \{a\}$.

\[
Z = S, \quad P = \{(S \rightarrow a, S \rightarrow S, S \rightarrow a)\}
\]

and the $(1,0)$-reduced embedding function $\psi^{(1,0)}$ is defined by $\psi^{(1,0)}(a) = \psi^{(1,0)}(a) = 1$ and $\psi^{(1,0)}(S) = \psi^{1,0}(S) = 0$. It is easy to verify that $L(G)$ is the set of graphs of the form

\[
\bullet \rightarrow a \rightarrow a \rightarrow c \rightarrow \bullet \rightarrow \bullet \rightarrow \bullet \rightarrow \ldots \bullet.
\]
We show that \( L(G) \) is not \( \mathcal{L}(0, 1) \). Assume that \( \tilde{G} = (\tilde{S}, \tilde{A}, \tilde{P}, \tilde{S}) \) is a \( (0, 1) \) grammar with \( L(\tilde{G}) = L(G) \). Clearly, one can construct a derivation \( \tilde{D} : \tilde{Z} = H_0 \Rightarrow \tilde{G} H_1 \Rightarrow \tilde{G} \cdots \Rightarrow \tilde{G} H_r \) in \( \tilde{G} \) for which there exists an index \( i \) with \( 0 \leq i \leq r - 1 \) such that

(i) \( H_r \in L(\tilde{G}) \),

(ii) let \( x \) be the node rewritten in \( H_i \Rightarrow \tilde{G} H_{i+1} \); then the nodes rewritten in \( H_{i+1} \Rightarrow \tilde{G} H_{i+2} \Rightarrow \tilde{G} \cdots \Rightarrow \tilde{G} H_r \) are descendents of \( x \),

(iii) \( x \) has at least 3 descendents in \( H_r \), and

(iv) \( \# V_{H_i} \geq 2 \).

Clearly (ii) implies that for every \( v \in V_{H_i} \setminus \{ x \} \) we have \( \phi_{H_i}(v) = a \) and hence for every production \( (\alpha, \beta, \psi) \) used in \( H_i \Rightarrow \tilde{G} H_{i+1} \Rightarrow \tilde{G} \cdots \Rightarrow \tilde{G} H_r \) we have \( \psi^{(0,1)}(a) = 1 \), because in the case that \( \psi^{(0,1)}(a) = 0 \) we can change the order of the derivation steps in \( \tilde{D} \) in such a way that the resulting graph is disconnected; a contradiction, because every element of \( L(\tilde{G}) \) is connected. It is easily seen that (i), (ii) and (iv) imply that \( x \) has at least one \( a \)-labelled neighbour in \( H_r \). It follows from (iii) and from the above that \( H_r \) has at least one node \( v \) with \( \deg(v) \geq 3 \); a contradiction.

Lemma 5. \( \mathcal{L}(0, 0) \setminus \mathcal{L}(0, 1)_k \neq \emptyset \).

Proof. Consider the \( (0, 0) \) grammar \( G = (\Sigma, \Delta, P, Z) \) where \( \Sigma = \{ a, B, c, S \} \), \( \Delta = \{ a, c \} \),

\[
Z = S \quad \text{and} \quad P = \{ (S, a), (S, B), (S, a, B), (S, B, c), (S, a), (S, B, c), (S, a), (S, B, c) \}
\]

where \( \psi_1^{(0,0)} = \psi_2^{(0,0)} = 1 \) and \( \psi_3^{(0,0)} = 0 \).

The language \( L(G) \) is the set of graphs of the form of Fig. 6 with \( n \geq 1 \).

![Fig. 6](image)

We show that \( L(G) \) is not in \( \mathcal{L}(0, 1)_k \). Indeed, let \( \tilde{G} = (\tilde{S}, \tilde{A}, \tilde{P}, \tilde{Z}) \) be a \( (0, 1)_k \) grammar with \( L(\tilde{G}) = L(G) \). We assume without loss of generality that \( \# V_{\tilde{Z}} = 1 \).
Clearly for the \((0, 1)\)-reduced embedding function \(\psi^{(0,1)}(a) = 1\), because in every graph \(X \in L(\bar{G})\) the subgraph \(X_a\) spanned by the \(a\)-labelled nodes of \(X\) is connected and \(\# V_X\) can be chosen arbitrarily large.

Now let \(H\) be a graph of \(L(G)\) (see Fig. 6) with \(n > (\maxr G)^\#\Sigma\). Let \(D = H_0 \Rightarrow \cdots \Rightarrow H_r = H\) be derivation of \(H\) in \(\bar{G}\). Let \(i, 0 \leq i \leq r - 1\) be the maximal index with the property that there exists a \(j, i < j \leq r\) and nodes \(x, y\) such that

(i) \(y\) is a descendant of \(x\), \(\phi_{H_i}(x) = \phi_{H_j}(y)\), \(x\) is the node rewritten in the derivation step \(H_i \Rightarrow H_{i+1}\) and either \(j = r\) or \(y\) is the node rewritten in the derivation step \(H_i \Rightarrow H_{i+1}\),

(ii) \(x\) has at least 2 descendants in \(H_r\).

From the assumption that \(n > (\maxr G)^\#\Sigma\) it easily follows that there exists such an index \(i\). Since for each \(X, Y \in L(\bar{G})\), \(\# V_X = \# V_Y\) implies that \(X\) is isomorphic to \(Y\), we know that the order in which the derivation steps of \(D\) are performed does not affect the resulting graph. Hence we may assume that all nodes rewritten in the derivation steps \(H_i \Rightarrow H_{i+1} \Rightarrow \cdots \Rightarrow H_r\) are descendants of \(x\) and, if \(j \neq r\), that all nodes rewritten in \(H_j \Rightarrow H_{j+1} \Rightarrow \cdots \Rightarrow H_r\) are descendants of \(y\). From (i) it now follows that we can construct other derivations in \(\bar{G}\) by iterating an arbitrary number of times the part of \(D\) in which descendants of \(x\) (but not of \(y\)) are rewritten, leaving the remaining part of \(D\) unchanged. Since in every graph of \(L(\bar{G})\) the number of nodes labelled by \(a\) equals the number of nodes labelled by \(c\), (ii) implies that

\[
\begin{align*}
&\text{there exist } p_1, p_2 \in V_{H_i} \text{ with } \phi_{H_i}(p_1) = a, \; \phi_{H_i}(p_2) = b \text{ and both } p_1, p_2 \text{ are descendants of } x
\end{align*}
\]

From our assumptions with respect to \(D\) it also follows that for each \(v \in V_{H_i}\backslash\{x\}\) we have \(\phi_{H_i}(v) \in \{a, c\}\).

The fact that \(i\) is maximal and \(n > (\maxr G)^\#\Sigma\) implies that there exists a neighbour \(q\) of \(x\) in \(H_i\) such that \(\phi_{H_i}(q) = a\). From \(\psi^{(0,1)}(a) = 1\) and from (i) it now easily follows that there exist two nodes \(v_1, v_2\) in \(H\) such that \(\{v_1, v_2\} \in E_H\), \(\phi_H(v_1) = a\) and \(\phi_H(v_2) = c\); a contradiction. \(\Box\)

**Theorem.** The diagram of Fig. 7 holds, where we denote \(A \rightarrow B\) if \(A \subseteq B\) and \(A \rightarrow \nrightarrow B\) if \(A \backslash B \neq \emptyset\) and \(B \backslash A \neq \emptyset\).

**Proof.** The theorem follows from the Lemmas 1 through 5 and from the definitions. \(\Box\)

4. Fitting various models in our framework

In the theory of (sequential) graph grammars quite a number of different notions of a graph grammar were investigated (see, e.g., [7] and [4]). We believe that at this stage of the development of the theory of graph grammars it is important to
have a framework (or a number of them) in which various models can be compared and new "natural" (not ad hoc) models become apparent. Any such a framework should single out what are really the basic features of the models considered.

In the framework that we propose in this paper we concentrate on the role of the embedding mechanism which we consider to be the heart of a graph grammar. Clearly, in any systematic approach one of the first questions that arise is: "where to start?" We have decided to start our investigations by considering node-labelled undirected graphs; that is we consider graph grammars generating graph languages consisting of such graphs only. Undoubtedly a node-labelled undirected graph is a very basic graph structure worth an investigation on its own and moreover a number of notions of a graph grammar generating languages consisting of such graphs were considered in the literature.

Although in our paper we have been concerned with the classification of 1-NCE grammars, a natural extension of this classification to the general case of NCE grammars is obvious. Now a NCE grammar $G$ will be classified as a $(Z, X, Y)$ grammar with $Z \in \{0, 1, 2\}$, $X \in \{0, 1, 2\}$ and $Y \in \{0, 1\}$ if the following holds. The interpretation of $X$ and $Y$ describing the ability of embedding functions to distinguish between the nodes of the daughter graph and between the nodes of the neighbourhood, respectively, remains the same. $Z = 0$ means that embedding functions cannot distinguish between the nodes of the mother graph while $Z = 1$ means that embedding functions cannot distinguish between different nodes of the mother graph having the same label. $Z = 2$ corresponds to the general case. We proceed now to analyse how several models known from the literature fit into our framework.
(1) In [9] two types of web-grammars are introduced. In a more general one a production is of the form $(\alpha, \beta, \Phi)$ where $\alpha$ and $\beta$ are graphs ($\alpha$ is assumed to be connected) and $\Phi$ is a function from $V_\alpha \times V_\beta$ into $2^\Sigma$ (where $\Sigma$ is the alphabet of node-labels). The application of a production $(\alpha, \beta, \Phi)$ to a graph $H$ in a direct derivation step is performed by replacing a subgraph $\check{\gamma}$ of $H$ which is isomorphic to $\alpha$ (let $h$ be an isomorphism from $\alpha$ onto $\check{\gamma}$) by a graph $\hat{\beta}$, isomorphic to $\beta$ (let $g$ be an isomorphism from $\hat{\beta}$ onto $\beta$). The embedding of $\hat{\beta}$ in $H - \check{\gamma}$ is specified by $\Phi$ as follows: an edge is established between a node $n$ of $V_\beta$ and a node $v$ of $H - \check{\gamma}$ if and only if there exists a node $m$ of $V_\alpha$ such that $v$ is a neighbour of $m$ in $H$ and the label of $v$ in $H$ belongs to $\Phi(h^{-1}(m), g(n))$. However in these grammars the possible derivation steps are restricted by the so-called application conditions. If we disregard these application conditions, then the definition is clearly equivalent to that of NCE grammars.

(2) In the same paper a more restricted class of web grammars is also considered; these grammars are called normal web grammars. They differ from grammars described under (1) in that for every production $(\alpha, \beta, \Phi)$ there exists an injective function $f$ from $V_\alpha$ into $V_\beta$ such that for each $v \in V_\alpha$, $\Phi(v, f(v)) = \Sigma$ and for each other element $x$ of $V_\alpha \times V_\beta$ we have $\Phi(x) = \emptyset$. Clearly, such a grammar can be described as a NCE grammar with the property that for each production $(\alpha, \beta, \psi)$ and for each $x \in V_\alpha, y \in V_\beta$, $l_1, l_2 \in \Sigma$ we have $\psi(x, y, l_1) = \psi(x, y, l_2)$. Hence normal web grammars form a special case of $(2, 2, 0)$ grammars.

(3) Web grammars more general than those discussed under (1) are introduced in [6]. They have productions of the form $(\alpha, C, \beta, E)$ where $C$ is a logical function specifying an application condition and $E$ is a set of logical functions specifying the embedding of the daughter graph in the host graph. As $C$ and $E$ are defined in a very general (and informal) way, these systems are more general than NCE grammars.

(4) In the same paper also more restricted classes of web-grammars, such as normal, monotonous and context-sensitive web grammars are considered. Almost all of the grammars discussed in the paper satisfy the following restrictions: the condition $C$ is always satisfied and $E$ is specified by giving a function $F$ from $V_\alpha$ into $V_\beta$. The interpretation of $F$ is the following: let a production $(\alpha, C, \beta, F)$ be applied to a graph $H$. Then a full subgraph $\check{\alpha}$ isomorphic to $\alpha$ (let $h$ be an isomorphism from $\alpha$ onto $\check{\alpha}$) is replaced by a graph $\hat{\beta}$ isomorphic to $\beta$ (let $g$ be an isomorphism from $\hat{\beta}$ onto $\beta$) and an edge is established between a node $n$ of $\hat{\beta}$ and a node $v$ of $H - \check{\alpha}$ if and only if there exists a node $m$ of $\check{\alpha}$ such that $m$ is a neighbour of $v$ in $H$ and

$$g(n) = F(h^{-1}(m)) \tag{*}$$

Clearly, if we assume that each $\alpha$ is connected then every such grammar can be described as a $(2, 2, 0)$ grammar.
(5) In [1] a similar definition is given, but in general $F$ is a function from $V_a$ into $2^{V_a}$, and the condition (*) is changed to \( g(n) \in F(h^{-1}(m)) \). Every such a grammar can be described as a $(2, 2, 0)$ grammar and every $(2, 2, 0)$ grammar is of this special kind.

(6) A further natural restriction of web grammars, introduced in [1], is the context-free restriction. A context-free web grammar is a web grammar such that for every production $(a, C, \beta, E)$, $a$ is a one-node graph. The embedding $E$ is again specified by a function $F$ from $V_a$ into $2^{V_a}$ and we still assume that $C$ is always satisfied. Every context-free web grammar can clearly be considered as a $(2, 0)$ grammar and every $(2, 0)$ grammar can be considered as a context-free web grammar.

(7) Combining the normal and the context-free restrictions together, we obtain web grammars that can be considered as "context-free graph grammars" from [3] (modified in the obvious way so that they generate undirected graphs). It is easily seen that the class of languages generated by those grammars is a proper subclass of the class of $(2, 0)$ grammars.

(8) Another well-known way of defining graph grammars is the "algebraic approach", where a direct derivation step is described by pushout and gluing constructions. The sequential graph grammars introduced in [4] are an example of this method. It is easy to see that, if we restrict ourselves to the case of node-labelled undirected graphs and to productions in which the left-hand side is connected, then every such grammar can be simulated by a $(2, 2, 0)$ grammar. If we add the additional condition that we replace only one node at a time, then we obtain the graph grammars discussed in (7) above.

(9) Although the general type of graph grammars introduced in [8] cannot be described in our framework, it is clear that NCE grammars and $(2, 2, 0)$ grammars can be considered as the counterparts (in the case of node-labelled undirected graphs) of Depth-1 graph grammars and an-graph grammars respectively.

5. Discussion

We would like to finish this paper by pointing out the following two problem areas:

(1) The most important problem in relation to the framework introduced in this paper is the construction of a diagram analogous to that of Theorem 1 for general NCE grammars. Although we have some partial results we are not able yet to settle all relationships needed to obtain such a diagram.

(2) Another research area that is natural from a mathematical viewpoint and that is also suggested by the existing literature on graph grammars is the extension of our approach to grammars generating directed graphs and graphs with edge-labels.
References


