

A. RUDNICKI

## **An economical transformation of graph grammars**

*Informatique théorique et applications*, tome 27, n° 4 (1993), p. 311-325.

[http://www.numdam.org/item?id=ITA\\_1993\\_\\_27\\_4\\_311\\_0](http://www.numdam.org/item?id=ITA_1993__27_4_311_0)

© AFCET, 1993, tous droits réservés.

L'accès aux archives de la revue « Informatique théorique et applications » implique l'accord avec les conditions générales d'utilisation (<http://www.numdam.org/legal.php>). Toute utilisation commerciale ou impression systématique est constitutive d'une infraction pénale. Toute copie ou impression de ce fichier doit contenir la présente mention de copyright.

NUMDAM

Article numérisé dans le cadre du programme  
Numérisation de documents anciens mathématiques  
<http://www.numdam.org/>

## AN ECONOMICAL TRANSFORMATION OF GRAPH GRAMMARS (\*)

by A. RUDNICKI <sup>(1)</sup> <sup>(2)</sup>

Communicated by W. RYTTER

---

*Abstract.* – *The main result is a technical contribution in the theory of so called labeled graphs (l-graphs). We present a more economical transformation of a certain type of monotonic graph grammar  $G$  to equivalent context-free graph grammar  $G'$ . The best result previously known was quadratic size of the part of  $G'$  corresponding to a single production of  $G$ . Our main result is an improvement which gives linear size reductions.*

*Résumé.* – *Le résultat principal est une contribution technique à la théorie des graphes étiquetés. Nous présentons une transformation plus économique d'un certain type de grammaires de graphes monotones  $G$  dans des grammaires équivalentes  $G'$  libres de contexte. Le meilleur résultat jusque là obtenu était quadratique en la taille de la partie de  $G'$  correspondant à une production donnée de  $G$ . Notre résultat principal est une amélioration qui donne des réductions linéaires.*

### 1. INTRODUCTION

Graph grammars have been defined in various ways as generalizations of the usual string grammar. Theory of graph grammars can be applied in the following areas: semantics of recursively defined functions, record handling, data base systems, compiler techniques, development and evolution in biology, pattern synthesis and recognition and many others. Efficiency aspects of graph language recognition require that the corresponding graph grammar be in the simplest possible form. Therefore transformations among various classes of graph grammars are of a great concern. Obviously one should expect that such a transformation does not change the generative power of the grammar.

---

(\*) Received December 1991, accepted December 1992.

(<sup>1</sup>) Supported by grant KBN 2-1190-91-01.

(<sup>2</sup>) Institute of Informatics, Warsaw University.

In this paper we follow [3] and consider the problem of effective transformation of monotonic ( $M$ ) graph grammar  $G$  to an equivalent context-free ( $CF$ ) graph grammar  $G'$ . The main idea deals with a direct proof of  $M=CF$  of [3] which has been shown in two steps in [3], namely  $M=CS$  and  $CS=CF$  ( $CS$ , context-sensitive grammar). Our main result is an improvement which gives linear size reductions.

Specifically, if  $t$  and  $s$  denote the size of the left and right side, respectively, of a production of a given monotonic grammar, then the upper bound on the number of productions in the corresponding context-free grammar has been shown in [3] to be  $2(2s+t)s$ , while our method gives only  $2s+t$  productions.

In the next chapter we recall basic and most general definitions concerning graph grammars. Then we formulate and prove the main result. We provide a representative example of the construction used in the proof.

## 2. DEFINITIONS

Let us consider two finite sets  $\Sigma_V$  and  $\Sigma_E$ , which we call alphabets of labels of nodes and edges, respectively.

DEFINITION 1:  $\{l\text{-graph}\}$

A labeled graph ( $l$ -graph) over the alphabets  $\Sigma_E$  and  $\Sigma_V$  is a tuple  $d=(K, (\rho_a)_{a \in \Sigma_E}, \beta)$ , where:

1.  $K$  is a finite set of nodes.
2. for each  $a \in \Sigma_E$ ,  $\rho_a \in K \times K$  is a binary relation over  $K$ .
3.  $\beta: K \rightarrow \Sigma_V$  is a labeling function of nodes.

Let  $\alpha$  and  $\alpha' \in K$ . Each pair  $(\alpha; \alpha') \in \rho_a$ , where  $a \in \Sigma_E$ , can be viewed as a directed edge from  $\alpha$  to  $\alpha'$ , labeled with symbol " $a$ ". Each node  $k$  is labeled with  $\beta(k) \in \Sigma_V$ . By  $d(\Sigma_V, \Sigma_E)$  we denote the set of all  $l$ -graphs over the alphabets  $\Sigma_V$  and  $\Sigma_E$ ,  $d_\epsilon$  denotes the empty graph and  $d^1(\Sigma_V, \Sigma_E)$  is the class of all single-node graphs from  $d(\Sigma_V, \Sigma_E)$ .

An example of an  $l$ -graph is shown in figure 1. Assume that  $\Sigma_V = \{v, m, t, f\}$  and  $\Sigma_E = \{i, j, k, l\}$ .

DEFINITION 2:  $\{subgraph\}$

An  $l$ -graph  $d'=(K', \rho'_a, \beta') \in d(\Sigma_E, \Sigma_V)$  is an induced subgraph of  $d=(K, \rho_a, \beta) \in d(\Sigma_E, \Sigma_V)$  (denoted:  $d' \subseteq d$ ) if  $K' \subseteq K$  and  $\rho'_a, \beta'$  are  $\rho, \beta$  restricted to the set  $K'$ .

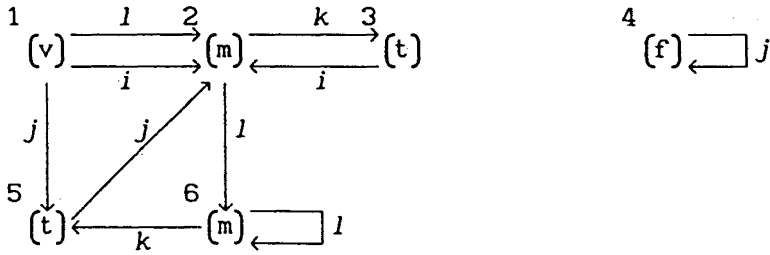


Figure 1

DEFINITION 3: {operator}

Nonempty strings over an alphabet

$$W = \Sigma_V \cup \{L_a \mid a \in \Sigma_E\} \cup \{R_a \mid a \in \Sigma_E\} \cup \{C, I, \cup, \cap, \cdot, \emptyset\}$$

are called operators if they are formed according to the following rules:

- a)  $L_a$  and  $R_a$  are operators for each  $a \in \Sigma_E$ .
- b) If  $A$  is an operator, then  $(CA)$  and  $(vA)$  are also operators, for each  $v \in \Sigma_V^+$ .
- c) If  $A$  and  $B$  are operators, then  $AB$ ,  $(A \cup B)$  and  $(A \cap B)$  are also operators.

Let  $d = (K, \rho_a, \beta)$  and  $d' = (K', \rho'_a, \beta')$  be  $l$ -graphs over  $\Sigma_V, \Sigma_E$  and  $d' \subseteq d$ . For each  $x \in K'$  any operator  $A$  specifies a subset of node set  $K$ , denoted  $A(x)$ , according to the following recursive definitions:

a)  $L_i(x) = \{y \in K - K' : (y, x) \in \rho_i\}$ ;  $R_i(x) = \{y \in K - K' : (x, y) \in \rho_i\}$ .

In other words:

$L_i(x)$  specifies the set of source nodes, not contained in  $K'$ , of  $i$ -labeled edges which end at  $x$ .

$R_i(x)$  denotes the set of target nodes, not contained in  $K'$ , of  $i$ -labeled edges which start at  $x$ .

b)  $(CA)(x) = \{y \mid y \in K - K' \text{ and } y \notin A(x)\}$  – the complement of  $A(x)$  in  $K - K'$ .

$(v_1 v_2 \dots v_m A)(x) = \{y \mid y \in A(x) \text{ and } \beta(y) \in \{v_1, v_2, \dots, v_m\}\}$ , determines the subset of  $A(x)$  the nodes of which are labeled with one of the symbols  $v_1, v_2, \dots, v_m$ .

c)  $I(x) = \{y \mid y \in K - K'\}$  – the set of all nodes outside  $K'$ .

d)  $AB(x)$  – composition of two operators.

$$(A \cap B)(x) = A(x) \cap B(x)$$

$$(A \cup B)(x) = A(x) \cup B(x)$$

DEFINITION 4: { *graph production* }

Graph production is a triple  $p = (d_l, d_r, E)$ , where graph  $d_l$  is the left side of the production,  $d_r$  is the right side,  $d_l, d_r \in d(\Sigma_V, \Sigma_E)$  and  $E = ((l_a, r_a))_{a \in \Sigma_E}$  is an embedding transformation,  $l_a = \bigcup_{j=1}^m A_j(y_j) \times \{z_j\}$ ;  $r_a = \bigcup_{j=1}^m \{z_j\} \times A_j(y_j)$ ;  $m \geq 1$ ,  $y_j \in K_l$ ,  $z_j \in K_r$ ,  $A_j$  is an operator;  $K_l, K_r$  are sets of nodes of  $d_l$  and  $d_r$  respectively.

Informally, components  $l_a$  and  $r_a$  assign  $a$ -labeled edges which connect  $d_r$  and  $d - d_l$ .  $A_j(y_j)$  determines a subset of  $K - K_l$  nodes of which are to be adjacent to  $z_j$  after the application of the production.

DEFINITION 5: { *direct derivation* }

Graph  $d' \in d(\Sigma_V, \Sigma_E)$  is directly derivable from  $d \in d(\Sigma_V, \Sigma_E)$  by means of production  $p = (d_l, d_r, E)$ , which is denoted  $d \xrightarrow{p} d'$ , if

a)  $d_l \subseteq d$ ,  $d_r \subseteq d'$ .

b)  $d - d_l = d' - d_r$ .

c)  $\text{In}_a(d_r, d') = l_a$  and  $\text{Out}_a(d_r, d') = r_a$ , for each  $a \in \Sigma_E$ , where  $\text{In}_a(d_r, d')$  is the set of all  $a$ -labeled edges, originating in  $d' - d_r$  and terminating in  $d_r$ ,  $\text{Out}_a(d_r, d')$  is the set of all  $a$ -labeled edges, originating in  $d_r$  and terminating in  $d' - d_r$ .

Informally the application of production  $p$  for graph  $d$  modulo renaming of nodes, looks as follows: we find  $d_l$  in  $d$ , remove it with all its edges and replace it with  $d_r$ . Next we link  $d_r$  to  $d$  with the help of  $E$ . Observe that  $E$  depends on  $d_l$ .

DEFINITION 6: { *graph grammar* }

A labeled graph grammar (LGG in short) is a tuple

$$G = (\Sigma_V, \Sigma_E, \Delta_V, \Delta_E, d_0, P, -s \rightarrow),$$

where:

a)  $\Sigma_V, \Sigma_E$  are nonempty finite alphabets for labeling of nodes and edges.

b)  $\Delta_V \subset \Sigma_V, \Delta_E \subset \Sigma_E$  are terminal alphabets.

c)  $d_0 \in d(\Sigma_V, \Sigma_E) - d(\Delta_V, \Sigma_E) \cup \{d_\epsilon\}$  is an initial graph.

d)  $P$  is a finite set of productions.

d)  $-s \rightarrow$  is the transitive closure of the direct derivation described above.

Let  $D$  denote the class of all equivalent  $l$ -graphs over  $\Sigma_V, \Sigma_E$ , i.e. all structural equivalent graphs  $d$  which only have different node labels, then:

DEFINITION 7: { *graph language* }

$L(G) := \{ D \mid D \in d(\Delta_V, \Delta_E) \text{ and } D_0 -s \rightarrow D \}$  is called the language of graph grammar  $G$ .

This means that graphs belonging to the language have only terminal nodes and edges, and are derivable from  $d_0$ .

DEFINITION 8: { *equivalence* }

Two grammars  $G_1$  and  $G_2$  are called equivalent, denoted by  $G_1 \equiv G_2$ , if  $L(G_1) = L(G_2)$ ;

DEFINITION 9: { *separable grammar* }

A LGC is called separable if for each  $p \in P$  either (a) or (b) take place:

(a)  $d_l, d_r \in d(\Sigma_V - \Delta_V, \Sigma_E) - \{d_\epsilon\}$ .

(b)  $d_l \in d^1(\Sigma_V - \Delta_V, \Sigma_E), d_r \in d^1(\Delta_V, \Sigma_E) \cup \{d_\epsilon\}$ .

DEFINITION 10: { *types of productions* }

Graph production  $p = (d_l, d_r, E)$  is called:

1. monotonic if  $|K_l| \leq |K_r|$ .

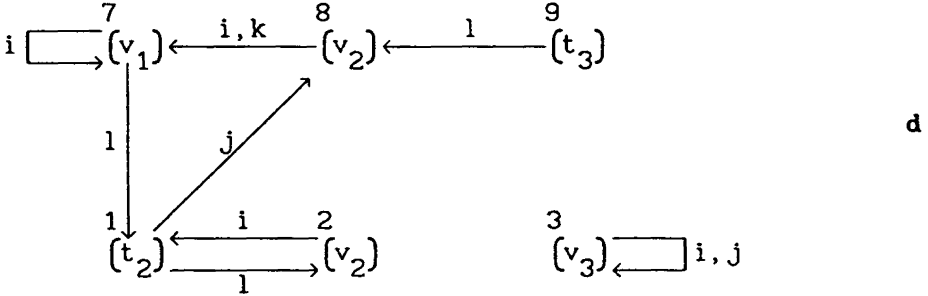
2. context-sensitive if for some  $d'_l \subseteq d_l, d'_r \subseteq d_r$  where  $d'_l \in d^1(\Sigma_V - \Delta_V, \Sigma_E)$   $d'_r \in d(\Sigma_V, \Sigma_E) - \{d_\epsilon\}$  we have  $d_l - d'_l = d_r - d'_r$  and  $E$  on the context  $d_r - d'_r$  is constant, i.e.:

$$\bigcup_{x \in K_l - K'_l} (L_a(x); x) \subseteq l_a, \quad \bigcup_{x \in K_l - K'_l} (x; R_a(x)) \subseteq r_a.$$

3. Context-free if  $p$  is context-sensitive and  $|K_l| = 1$ .

We shall consider monotonic, context-sensitive and context-free LGG, according to the set of productions  $P$ .

The following example demonstrates the application of the monotonic production. Consider graph  $d$  (taken from [3]):



and monotonic graph production  $p=(d_1, d_r, E)$  :

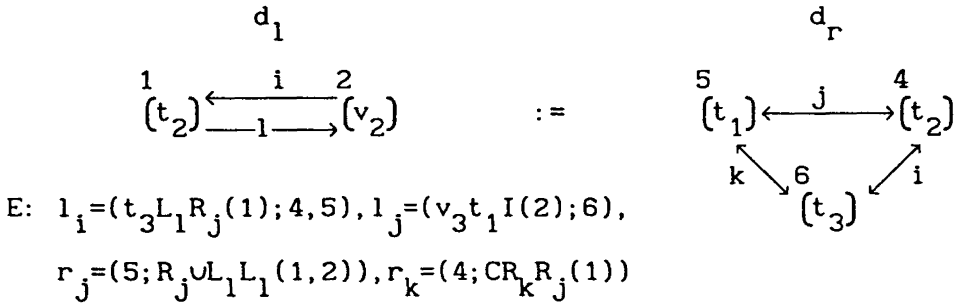


Figure 2

Replaced  $d_i$  by  $d_r$  and carefully applied all operators of  $E$  we obtain:

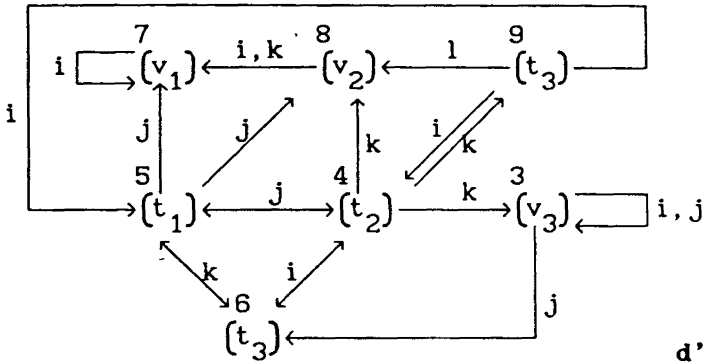


Figure 3

3. TRANSFORMATION OF MONOTONIC LGG TO CONTEXT-FREE ONE

Assume that for  $i$ -th monotonic production  $|K_l|=s_i$  and  $|K_r|=t_i$ . The structure of the transformation is the following:

Every monotonic production which is not context-free is decomposed into a sequence of context-free productions which, when applied one after the other, give the same result as the monotonic production. Such a sequence consists of three subsequences with the total length of  $2s_i+t_i$  context-free productions. First subsequence (part *A*) identifies a subgraph of  $d$  which is isomorphic to  $d_l$  ( $c$ -edges), assigns new unique labels to its nodes and creates auxiliary  $\tilde{a}$ -edges. Part *A* is a preparing part of the transformation. Next part *B* is the main. With the help of the second subsequence we replace graph isomorphic to  $d_l$  by  $d_r$  and construct the embedding of  $d_r$  into  $d'=d-d_l$ . Last subsequence of context-free productions (part *C*) replaces node labels and deletes all unnecessary edges. Derivation steps are controlled by creation blocking  $f$ -edges.

**THEOREM:** *For any monotonic graph grammar  $G$  with  $n$  productions there exists an equivalent context-free grammar  $G'$  with  $\sum_{i=1}^n (2s_i+t_i)$  productions.*

*Proof:* Let  $G=(\Sigma_V, \Sigma_E, \Delta_V, \Delta_E, d_0, P, -s \rightarrow)$  be a monotonic LGG. Without loss of generality we can assume that it is separable [3]. Let  $P_m \subset P$  be the subset of productions of the grammar  $G$  which are monotonic and not context-free. Let:

$$\alpha = \sum_{i=1}^n |K_{l_i}|; \quad \beta = \sum_{i=1}^n |K_{r_i}|; \quad \gamma = \alpha + \beta$$

$$\Pi_{mv} := (N_1, N_2, \dots, N_\gamma), \quad \Pi'_{mv} = \Pi_{mv} \cup (N'_1, N'_2, \dots, N'_\gamma), \quad \Sigma_V \cap \Pi'_{mv} = \emptyset.$$

This means that for each monotonic production  $p_i$  we introduce  $2(|K_{l_i}|+|K_{r_i}|)$  nonterminal labels of nodes.

Denote:

$$\tilde{\Sigma}_E = \{ \tilde{a} \mid a \in \Sigma_E \}, \quad \tau = \max_{i \in P_m} |K_{l_i}|.$$

We put

$$\Sigma'_E = \Sigma_E \cup \tilde{\Sigma}_E \cup c \cup f \cup \{ \text{elab}_1, \text{elab}_2, \dots, \text{elab}_\tau \},$$

thus introducing  $|\Sigma_E|+\tau+2$  new nonterminal edge labels. Similarly,  $\Sigma'_V = \Sigma_V \cup \Pi'_{mv}$ , i. e.  $2\gamma$  new nonterminal node labels are specified.



*I.* Let  $p = (d_l, d_r, E) \in P_m$  – be a monotonic production with  $|K_l| = s, |K_r| = t$ , clearly  $|K_l| \leq |K_r|$  and, without loss of generality:

$$K_l = \{1, 2, \dots, s\} \quad K_r = \{s+2, s+3, \dots, s+t+1\}.$$

Next, let  $(N_1, N_2, \dots, N_s) \cup (N'_2, N'_3, \dots, N'_{s+t+1})$  – be the set of nonterminal labels of nodes, used for the decomposition of this production. Finally, let  $K'_l = \{1\}$ ,  $K'_r = \{2s+1, 2s+2, \dots, s+t+1\}$ . The remaining nodes in  $d_l - d'_l = \{2, 3, \dots, s\}$  and  $d_r - d'_r = \{s+2, s+3, \dots, 2s\}$  constitute a fixed context and are paired with the help of the function  $h: i \rightarrow s+i, i=2, \dots, s$ . We will decompose  $p$  into a sequence of productions  $p^1, p^2, \dots, p^{2s+t}$ .

*A:* With the help of  $p^1, p^2, \dots, p^s$ , which form the first part of the sequence of context-free productions we identify a subgraph of  $d$  which is isomorphic to  $d_l$  and assign unique labels to its nodes.

*A1:* Production  $p^1 = (d_l^1, d_r^1, E^1)$  is specified as follows:  $d_l^1 = \{1\}$ ,  $d_r^1 = \{1\}$ , labeled  $N_1$ ,  $E^1$  consists of  $E_{id}^f(1|1)$  – which means, that all edges are removed except  $f$ -edges (“id” – identical). Other  $E^1$  components are the following:  $l_a^- = (L_a(1); 1)$  and  $r_a^- = (1; R_a(1))$ ,  $a \in \Sigma_E$ , which replaces  $a$  – by  $\bar{a}$ -edge.  $r_{elab_1} = (1; I(1))$ , creating  $elab_1$ -edges to all nodes of the graph  $d - \{1\}$ .  $l_f = (\Pi'_{mv} I(1); 1)$  which creates  $f$ -edges if in the initial graph  $d$  there are node labels from  $\Pi'_{mv}$  in  $d$  (blocking edges). The existence of such blocking  $f$ -edges leads to a blind branch in derivation and, as such edges are preserved, the resulting graph does not belong to the graph language.

*A2:* Let  $1 < j \leq s$ . Productions  $p^j = (d_l^j, d_r^j, E^j)$  are constructed as follows:  $d_l^j = \{j\}$ ,  $d_r^j = \{j\}$ , labeled  $N_j$ ,  $E^j$  consist of  $E_{id}^{\alpha \in \Sigma_E - \{\bar{E} \cup \Sigma_E\}}(j|j)$ , complemented by the following components:  $l_a^- = (L_a(j); j)$ ,  $r_a^- = (j; R_a(j))$ , where  $a \in \Sigma_E$ .  $r_{elab_j} = (j; I(j))$  which create  $elab_j$ -edges incident to all other nodes of the graph  $d$ .

$$l_f = (C(L_{elab_{j-1}} \cup R_{elab_{j-1}} L_{elab_{j-1}})(j); j),$$

thus not allowing to omit the application of  $p^{j-1}$  before the application of  $p^j$ . Furthermore,  $l_f = (N_j \dots N_\gamma I(j); j)$  creates blocking  $f$ -edges, in case when labels of nodes  $N_j \dots N_\gamma$  appear in the given graph. This means that a production which comes after the current one or belonging to another sequence has already been applied.

The next component of the embedding transformation of  $p^j$  generates  $c$ -edges and is defined in general as follows:

Let  $1 \leq i < j$ . If  $\mu \in \Sigma_E, \tilde{\mu} \in \tilde{\Sigma}_E, v \in \Sigma_E, \tilde{v} \in \tilde{\Sigma}_E, (i, j) \in \rho_{l_\mu}, (j, i) \in \rho_{l_v}$  then  $l_c = (N_i(\bigcap_{\tilde{\mu}} L_{\tilde{\mu}} \cap \bigcap_{\tilde{v}} R_{\tilde{v}})(j); j)$ . In the case  $(i, j) \notin \rho_{l_\mu}, \bigcap_{\tilde{\mu}} L_{\tilde{\mu}}$  in  $l_c$  is replaced by  $I$ . Analogously, if  $(j, i) \notin \rho_{l_v}, \bigcap_{\tilde{v}} R_{\tilde{v}}$  is replaced by  $I$ . Finally, when

there are no edges between  $i$  and  $j$  in  $d_i, l_c$  takes the form:  $l_c = (N_i I(j); j)$ . Hence, an incoming  $c$ -edge to node  $j$  is created in two cases:

1. If there is an edge connection structure between  $i$  and  $j$  in  $d$ , which corresponds to the analogous one in  $d_i$ .
2. If in  $d_i$  and  $d$  there is no edge between  $i$  and  $j$ .

In this way we check whether  $d$  contains all the edges between  $j$  and  $1, \dots, j-1$ . The last component checks for superfluous edges in  $d$ , in comparison to  $d_i$ :

If  $\mu, v \in \Sigma_E, \tilde{\mu}, \tilde{v} \in \tilde{\Sigma}_E, (i, j) \in \rho_{l_\mu}, (j, i) \in \rho_{l_v}$ , then

$$l_f = (N_i(L_{\tilde{\mu}} \cup R_{\tilde{v}})(j); j), \quad \text{where} \quad L_{\tilde{\mu}} = \bigcup_{\tilde{a} \neq \tilde{\mu}} L_{\tilde{a}}, \quad R_{\tilde{v}} = \bigcup_{\tilde{a} \neq \tilde{v}} R_{\tilde{a}}$$

If:

$$(i, j) \notin \rho_{l_\mu}, \quad \forall \mu \in \Sigma_E \quad \text{then} \quad L_{\tilde{\mu}} = \bigcup_{\tilde{a} \in \tilde{\Sigma}_E} L_{\tilde{a}}$$

$$(j, i) \notin \rho_{l_v}, \quad \forall v \in \Sigma_E \quad \text{then} \quad R_{\tilde{v}} = \bigcup_{\tilde{a} \in \tilde{\Sigma}_E} R_{\tilde{a}}$$

This component creates blocking  $f$ -edges if there are edges between  $i$  and  $j$  in  $d$  which do not occur in  $d_i$ .

**B:** The subsequence of context-free productions  $p^{s+1}, p^{s+2}, \dots, p^{2s}$  causes that the graph isomorphic to  $d_i$  is replaced by  $d_r$ , labeled with  $N'_{j+1}$ , for  $1 \leq j \leq t$ . Furthermore, we construct the embedding of  $d_r$  into  $d' = d - d_i$ . First we will embed remaining part  $d'_r$ :

**B1:** Production  $p^{s+1} = (d_i^{s+1}, d_r^{s+1}, E^{s+1})$  is constructed as follows:  $d_i^{s+1} = \{1\}$ , labeled  $N_1$

$$d_r^{s+1} = d_r \{2s+1, \dots, s+t+1\},$$

labeled  $(N'_{s+1}, \dots, N'_{t+1})$ , i.e.  $d'_r. E^{s+1}$  consists of:

$$l_\mu = (N_j I(1); q) \quad \text{if} \quad \exists q, j | q \in \{2s+1, \dots, s+t+1\},$$

$$2 \leq j \leq s \quad \text{with} \quad (s+j; q) \in \rho_{r_\mu} \quad \mu \in \Sigma_E$$

$$r_\mu = (q; N_j I(1)) \quad \text{if } \exists q, j | q \in \{2s+1, \dots, s+t+1\}, \\ 2 \leq j \leq s \quad \text{with } (q; s+j) \in \rho_{r_\mu}$$

These components connect  $d'_r$  and  $d_r - d'_r$ . It remains to embed  $d'_r$  into  $d$ . Consider  $(A(j); k) = l_a \in E_i$  of  $p_i$  labeled with  $a \in \Sigma_E$ ,  $1 < j \leq s$ ,  $k \in K'_r$ . Observe that:

1. Embedding of  $d'_r$  may depend on embedding of nodes  $d_r - d'_r$ .
2. Operators may assign nodes from  $d_r$  which, for a context-free production, are adjacent to the given node.

These problems can be omitted if we replace this component by  $l_a = (\tilde{A}(N_j R_{\text{elab}_1})(1); k)$ . Operator  $\tilde{A}$  determines nodes from  $d - d_i$  using complement of  $\Pi_{mv}$  (i.e.  $\Sigma_V$ ) and edges  $\tilde{a}$ . Analogously for  $r_a: (k; A(j)) = r_a$  is replaced by  $r_a = (k; \tilde{A}(N_j R_{\text{elab}_1})(1))$ .

All components of the type  $(A(1); k) = l_a$  with  $k \in K'_r$  are replaced by  $l_a = (\tilde{A}(1); k)$ . Similarly, for  $r_a$  we obtain  $r_a = (k; \tilde{A}(1))$ .

$E^{s+1}$  also contains  $E_{\text{id}}^{a \in \tilde{\Sigma}_E \cup f \cup c} (1 | 2s+1)$ , and is complemented by the following components:

$$l_f = (C(L_{\text{elab}_s} \cup R_{\text{elab}_s} L_{\text{elab}_s})(1); 2s+1)$$

– creates an  $f$ -edge if production  $p^{2s}$  has been applied before.

$$l_f = (N_2 \dots N_s (CR_c)(1); 2s+1)$$

– creates blocking  $f$ -edges in the case when at least one node of  $N_2, \dots, N_s$  is not connected with  $\{1\}$  with a  $c$ -edge.

**B2:** Let  $1 < j \leq s$ . Productions  $p^{s+j} = (d_i^{s+j}, d_r^{s+j}, E^{s+j})$  look as follows:

$$d_i^{s+j} = \{j\}, \text{ labeled } N_j \\ d_r^{s+j} = \{s+j\}, \text{ labeled } N'_j$$

$E^{s+j}$  consists of  $E_{\text{id}}^{a \in \tilde{\Sigma}_E \cup f \cup c} (j | s+j)$ , and the following components:  $l_f = (N'_{j+1} \dots N'_s (CR_c)(j); s+j)$  – creates blocking  $f$ -edges, if there is no  $c$ -edge between  $i$  and  $j$  ( $j < i \leq s$ ), i.e. the edge which connects these nodes has no counterpart in  $d_i$ .

$l_f = (N_{j-1} I(j); s+j)$  – forces the application of production  $p^{s+j-1}$  earlier.

Let  $1 < z < j$

$$l_\mu = (N_z I(j); j) \quad \text{if in } d_r \exists (s+j; s+z) \in \rho_{r_\mu}, \quad \text{where } 2 \leq j \leq s \\ r_\mu = (j; N_z I(j)) \quad \text{if in } d_r \exists (s+z; s+j) \in \rho_{r_\mu}, \quad \text{where } 2 \leq j \leq s$$

These components correlate  $d_i - d'_i$  with  $d_r - d'_r$ . Then it is necessary to embed  $d_r - d'_r$  into the host graph. For this operation the same remarks as in point B1 should be made.

Let  $(A(z); s+j) = l_a \in E_i$  of  $p_i$ ,  $1 < j \leq s$ ,  $z \leq s (z \neq j)$ . This component is replaced by  $l_a = (\tilde{A}(\bar{N}_z I(j)); s+j)$ , where  $a \in \Sigma_E$  and  $\bar{N}_z \neq \begin{cases} N'_z, & 1 < z < j \\ N_z, & j < z \leq s \\ N'_{s+1}, & z = 1 \end{cases}$

Where operator  $\tilde{A}$  determines nodes from  $d - d_i$  using complement of  $\Pi'_{mv}$  (i.e.  $\Sigma_V$ ) and edges  $\tilde{a}$ . Analogously for components  $r_a$  we obtain:

$$r_a = (s+j; \tilde{A}(\bar{N}_z I(j))).$$

All components of the type  $(A(j); s+j) = l_a$  are replaced by  $l_a = (\tilde{A}(j); s+j)$  and for  $r_a$  we receive:  $r_a = (s+j; \tilde{A}(j))$ .

C: The last subsequence of productions  $p^{2s+1}, p^{2s+2}, \dots, p^{2s+t}$  replaces nodes in  $d'$  labeled  $N'_{j+1}$   $1 \leq j \leq t$ , by the same nodes with labels from  $d_r$ . Here also all unnecessary edges which were applied in the construction are deleted.

Productions  $p^{2s+j} = (d_i^{2s+j}, d_r^{2s+j}, E^{2s+j})$  where  $1 \leq j \leq t$  are defined as follows:

$$d_i^{2s+j} = \{s+j+1\}, \text{ labeled } N'_{j+1}$$

$$d_r^{2s+j} = \{s+j+1\}, \text{ labeled } \beta \{s+j+1\}$$

$E^{2s+j}$  consists of  $E_{id}^a \in \Sigma_E \cup f$ .

II. The decomposition into the sequence of productions is performed for all  $p \in P_m \subset P$ . Let  $P_1$  be the set of all productions received in the above way. Left and right sides ( $d_i$  and  $d_r$ ) or productions from  $P - P_m$  are constant and embedding transformations  $E$  are modified in such a way that they preserve blocking edges. Let  $P_2$  be the set of all productions obtained from  $P - P_m$  by this modification. Then if  $P' = P_1 \cup P_2$  we have  $G' = (\Sigma'_V, \Sigma'_E, \Delta'_V, \Delta'_E, d_0, P', -s \rightarrow) \equiv G$ . This is because:

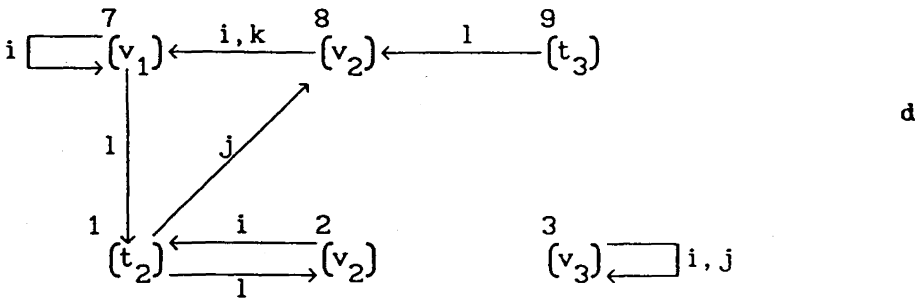
a) For each derivation in  $G$  we obtain the corresponding derivation in  $G'$ , because each step for  $p \in P_m$  is replaced by a sequence of steps of the derivation with productions from  $I$ , and each step for  $p \in P - P_m$  is replaced by a step with a modified production from  $P_2$ .

b) For the opposite, assume that the derivation of the graph  $D \in L(G')$  is defined. This means that graph  $D$  does not contain  $f$ -labeled blocking edges. Therefore the sequence of productions which are related to the production in  $G$  has been applied according to the order defined in  $I$ . It is easy to see

that this sequence does not intersect itself and the others, because embedding transformations  $E$  are modified in the sense, that they preserve blocking edges and  $D$  does not have such edges. We obtain the same derivation in  $G$  by replacing the part of derivation sequence by corresponding derivation in  $G$ , i.e.  $G' \equiv G$ . ■

4. EXAMPLE

The following example demonstrates the technique of decomposing a monotonic production into a sequence of  $2s + t$  context-free productions. Consider graph  $d$  (Taken from page 316):



and monotonic graph production  $p = (d_l, d_r, E) \in P_m :$

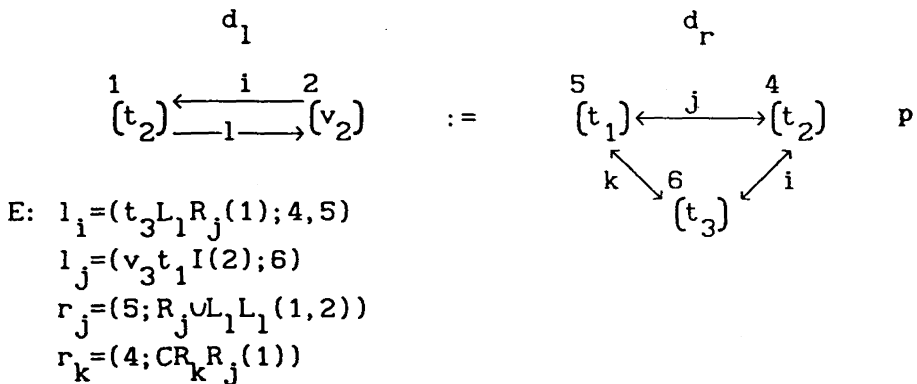


Figure 4

According to the algorithm of the proof denote: ( $s = 2, t = 3$ )

$$K_l = \{1, 2\}, \quad K_r = \{4, 5, 6\}, \quad K'_l = \{1\}, \quad K'_r = \{5, 6\}$$

$$h : 2 \rightarrow 4, \quad d_l - d'_l = \{2\}, \quad d_r - d'_r = \{4\}.$$

We are decomposing  $p$  into a sequence  $p^1, p^2, \dots, p^7$  of context-free productions.

A1.  $p^1 = (d_l^1, d_r^1, E^1)$ ,  $d_l^1 = \{1\}$ , labeled  $t_2$ ;  $d_r^1 = \{1\}$ , labeled  $N_1$ ,  $E^1$  consist of:  $E_{id}^f(1|1)$  complemented by:

$$\begin{aligned} l_{\tilde{l}} &= (L_l(1); 1), & r_{\tilde{l}} &= (1; R_l(1)), & l_{\tilde{r}} &= (L_r(1); 1), \\ r_{\tilde{r}} &= (1; R_r(1)), & r_{elab_1} &= (1; I(1)), & l_f &= (\Pi'_{mu} I(1); 1) \end{aligned}$$

A2.  $p^2 = (d_l^2, d_r^2, E^2)$ ,  $d_l^2 = \{2\}$ , labeled  $v_2$ ;  $d_r^2 = \{2\}$ , labeled  $N_2$ ,  $E^2 : E_{id}^a \in \Sigma_E - (\Sigma_E \cup \Sigma_E)(2|2)$ , thus preserving  $c$ -,  $f$ -,  $elab_1$ - and  $elab_2$ -edges, complemented by:

$$\begin{aligned} l_{\tilde{l}} &= (L_l(2); 2), & r_{\tilde{l}} &= (2; R_l(2)), & r_{elab_2} &= (2; I(2)), \\ l_f &= (C(L_{elab_1} \cup R_{elab_1} L_{elab_1})(2); 2), & l_f &= (N_2 \dots N_\gamma N_\gamma I(2); 2), \\ l_c &= (N_1(L_{\tilde{l}} \cap R_{\tilde{l}})(2); 2), & l_f &= (N_1(L_{\tilde{l}} \cup R_{\tilde{l}})(2); 2). \end{aligned}$$

Now we have:

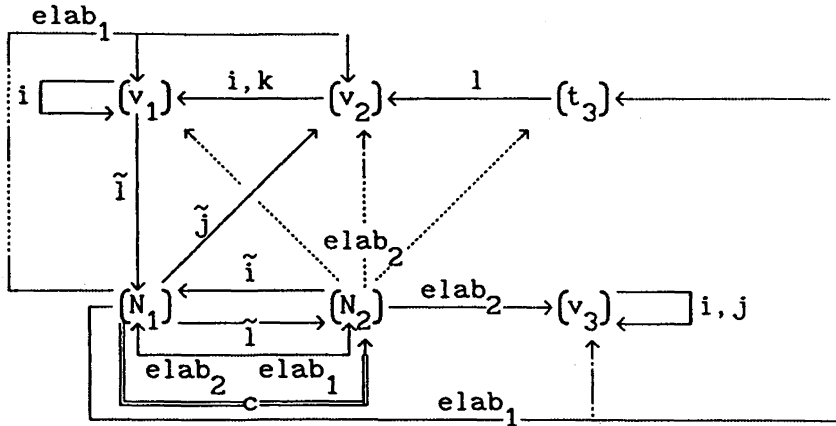


Figure 5

B1.  $p^3 = (d_l^3, d_r^3, E^3)$ ,  $d_l^3 = \{1\}$ , labeled  $N_1$ ;  $d_r^3 = \{5, 6\}$ , labeled  $N'_3, N'_4$ ,  $E^3 : E_{id}^a \in \Sigma_E \cup f \cup c(1|5)$ , and also:

$$\begin{aligned} l_j &= (N_2 I(1); 5), & l_i &= (N_2 I(1); 6), & r_j &= (5; N_2 I(1)), & r_i &= (6; N_2 I(1)), \\ l_j &= (v_3 t_1 I(N_2 R_{elab_1})(1); 6), & r_j &= (5; R_{\tilde{r}} \cup L_{\tilde{l}} L_{\tilde{l}} (N_2 R_{elab_1}(1))), \end{aligned}$$

$$l_i = (t_3 L_1 R_{\bar{7}}(1); 5), \quad r_j = (5; R_{\bar{7}} \cup L_1 L_{\bar{7}}(1)),$$

$$l_f = (C(L_{\text{elab}_2} \cup R_{\text{elab}_2} L_{\text{elab}_2})(1); 5), \quad l_f = (N_2(CR_c)(1); 5).$$

B2.  $p^4 = (d_i^4, d_r^4, E^4)$ ,  $d_i^4 = \{2\}$ , labeled  $N_2$ ;  $d_r^4 = \{4\}$ , labeled  $N'_2$ .  
 $E^4 : E_{\text{id}}^{a \in \Sigma_E \cup f} (2|4)$ , and also:  $l_f = (N_1 I(2); 4)$ ,

$$l_i = (t_3 L_1 R_{\bar{7}}(N'_3 I(2)); 4), \quad r_k = (4; v_1 v_2 v_3 t_1 t_2 t_3 CR_k R_{\bar{7}}(N'_3 I(2))).$$

Now looks at the intermediate graph:

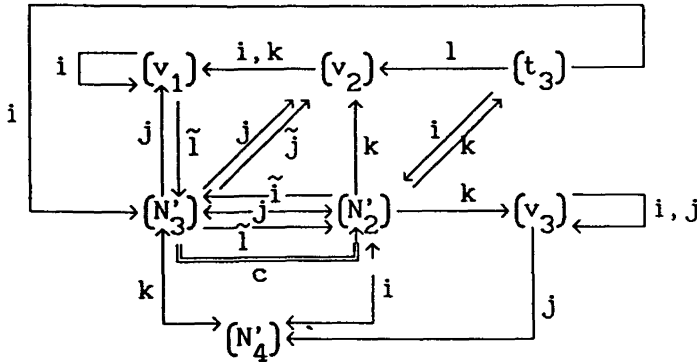


Figure 6

C.  $p^5 = (d_i^5, d_r^5, E^5)$ ,  $d_i^5 = \{4\}$ , labeled  $N'_2$ ;  $d_r^5 = \{4\}$ , labeled  $t_2$

$$E^5 : E_{\text{id}}^{a \in \Sigma_E \cup f} (4|4).$$

$p^6 = (d_i^6, d_r^6, E^6)$ ,  $d_i^6 = \{5\}$ , labeled  $N'_3$ ;  $d_r^6 = \{5\}$ , labeled  $t_1$

$$E^6 : E_{\text{id}}^{a \in \Sigma_E \cup f} (5|5).$$

$p^7 = (d_i^7, d_r^7, E^7)$ ,  $d_i^7 = \{6\}$ , labeled  $N'_4$ ;  $d_r^7 = \{6\}$ , labeled  $t_3$

$$E^7 : E_{\text{id}}^{a \in \Sigma_E \cup f} (6|6).$$

It is clear that if we take nodes distinct from  $\{1\}$  and  $\{2\}$  as  $d_i$  in  $d$  then in the process of decomposition we obtain blocking  $f$ -edges. Branches of derivation tree which correspond to these cases are blind. Continuing derivation along such a branch we shall never obtain a graph labeled only by terminal labels. Only the branch on which  $d_i$  consists of  $\{1\}$  and  $\{2\}$  does not lead to the creation of  $f$ -edges.

Now, using all 7 context-free productions one after the other, we get the following result (graph  $d'$ ):

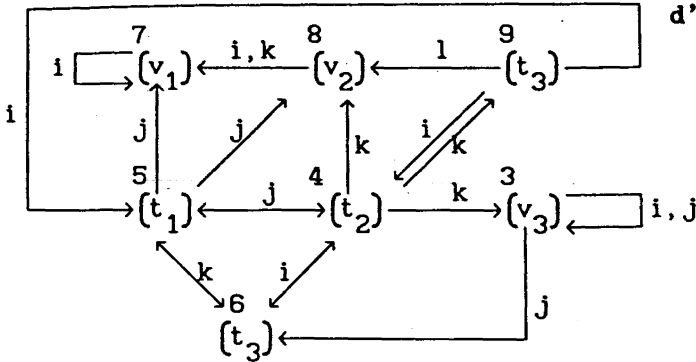


Figure 7

Resulting graph  $d'$  is identical to the graph received as the result of applying initial monotonic production (see *fig. 3* on page 316).

#### REFERENCES

1. E. GROTSCH and M. NAGL, *Explicit Versus Implicit Parallel Rewriting on Graphs*, LNCS Springer-Verlag, N73, 1979, pp. 237-254.
2. L. LEVY and YUEH KANG, On Labeled Graph Grammars, *Computing*, 20, 1978, pp. 109-125.
3. M. NAGL, *Graph-Grammatiken Theorie, Anwendungen, Implementierung*, Wiesbaden, 1979.
4. M. NAGL, A tutorial and bibliographical survey on graph grammars, *LCNS*, Springer-Verlag, N73, 1979, pp. 70-126.
5. M. NAGL, Set Theoretic Approaches to Graph Grammars. *NLCS*, Springer-Verlag, N291, 1987, pp. 41-54.