H.-J. Schneider

GRAPH-GRAMMARS: AN ALGEBRAIC APPROACH

H. Ehrig, Technische Universität Berlin FB 20
1 Berlin 10, Otto-Suhr-Allee 18-20, GERMANY

M. Pfender, Technische Universität Berlin FB 3

H.J. Schneider, Universität Erlangen-Nürnberg,
Lehrstuhl für Informatik II

0. Summary and Introduction.

Summary: The paper presents an algebraic theory of graph-grammars using homomorphisms and pushout-constructions to specify embeddings and direct derivations constructively. We consider the case of arbitrary directed graphs permitting loops and parallel edges and having labels on vertices and edges. The gluing of two arbitrary labeled graphs (pushout) is defined allowing a strictly symmetric definition of direct derivations and the embedding of derivations into a common frame. A two-dimensional hierarchy of graph-grammars is given including the classical case of Chomsky-grammars and several other graph-grammar constructions as special types. The use of well-known categorical constructions and results allows simplification of the proofs and a pregnant formulation of concepts like "parallel composition" and "translation of grammars".

Introduction: Trying to generalize the concept of Chomsky-grammars [1,2] to multi-dimensional symbol-structures the main problem is to specify the embedding and the exchange of substructures in order to get direct derivations of symbolstructures. This problem can be solved in a simple way by use of coordinates leading to the concept of "array"-grammars [7]. This concept however isn't adequate for a number of applications, especially: production and recognition of flow-charts, program-shemes and net-works, translation of programming-languages with respect to the syntax, some aspects of pattern recognition. In these cases it is less important to know the coordinates of the symbols than to have relations between the symbols like "left of", "above", "follows from", etc. More convenient for such cases are web-grammars [11,13], graph-grammars [8,12] and n-grammars in the sense of [14,15,9]. (Different relations correspond to different labelings of the edges). In [11,13] a direct derivation of webs (labeled graphs) from G to H is defined in the following way: Given a subweb 'B of G you get H as the union of G-'B and B' (being the right hand side of a production with left side 'B) together with certain edges between nodes in G-'B and B' which are specificed by an embedding rule. Similar constructions are given in [14,15] resp. [9] for n-diagrams where the embeddings are defined by relations resp. formal expressions. In [8,12] only context-free grammars are considered.

In this paper we give an algebraic definition of direct derivations using homomorphisms and pushouts, a well-known universal construction in category theory. In contrast to the above mentioned cases we are able to define embeddings (resp. the gluing of graphs) and direct derivations separately, e.g. for the case of unlabeled graphs:

1. Given arbitrary graphs D and B we define how to glue them together. Of course we don't want to get the disjoint union of D and B, so we have to specify all those vertices and edges of D and B we want to glue together. We get such a specification if we give an auxiliary graph K and two graph morphisms $p:K \longrightarrow B$ and $D:K \longrightarrow D$. Gluing together D and B along K via p and d (written $D \coprod_{d,p} B$) means just the construction of a pushout in the category Graph of graphs (cf. 1.1):

$$\begin{tikzcd} K \arrow[r, "p"] \arrow[d, "d"'] \arrow[dr, phantom, "p.o."] & B \arrow[d, "\tilde{d}"] \\ D \arrow[r, "\tilde{p}"'] & D \coprod_{d,p} B \end{tikzcd}$$

Thus in the disjoint union of D and B vertices $d(v)$ and $p(v)$ (v vertex in K) and edges $d(e)$ and $p(e)$ (e edge in K) are glued together to give $D \coprod_{d,p} B$.

2. A production $p = ('B \xleftarrow{'p} K \xrightarrow{p'} B')$ in our sense is not only a pair of graphs ('B, B') but also a specification of vertices and edges of 'B resp. B' given by an auxiliary graph K and graph morphisms $'p: K \longrightarrow 'B$ resp. $p': K \longrightarrow B'$.

Given a production $p = ('B \xleftarrow{'p} K \xrightarrow{p'} B')$ and an "enlargement" $e = (K \xrightarrow{d} D)$, i.e. a graph D with injective graph morphism d, with the same K we get a direct derivation

$$(p,e) \; : \; D \coprod_{d,'p} {'B} \longrightarrow D \coprod_{d,p'} B'$$

from $D \coprod_{d,'p} {'B}$ to $D \coprod_{d,p'} B'$ defined in 1.

Conversely a graph H can be directly derivated from a graph G if there is a production p and an enlargement e such that G is isomophic to $D \coprod_{d,'p} {'B}$ and H is isomorphic to $D \coprod_{d,p'} B'$.

With respect to the applications we don't want to distinguish isomorphic graphs. This fits in our concept because pushout-constructions are unique up to isomorphism. The general construction for labeled graphs will be discussed in § 1.

In contrast to other graph-grammar concepts let us point out:

a) The definition of productions and direct derivations is strictly symmetric. It is possible to generate or remove vertices and edges applying suitable productions. This is important for error-correction and recognition.

IEEE Computer Society

accepted for inclusion in the Conference Record SWAT 73

Proc. Conf. Switch. Automat. Theory 1973
p. 167-180

b) The application of productions isn't restricted to full subgraphs (cf.exemple 0.1)

c) An algorithmic description of derivations can use the construction of pushouts as a subroutine. A pushout of graphs can be constructed as a pushout of the vertices and the edges separately in the category of sets, i.e. as quotient sets of the disjoint unions.

d) The mathematical machinery of category-theory - especially the theory of pushouts - can be applied to get theoretical results concerning derivations in graph grammars.

Before we give two examples for our constructions let us summarize the contents of the present paper, which is an extension of a German version given in [5] : In § 1 we give the basic definitions and constructions for the gluing of labeled graphs being a generalization of the concatenation of strings in the classical case. Graph-grammars are defined in § 2 together with a discussion of existence and uniqueness of direct derivations. A two-dimensional hierarchy of graph-grammars including the classical case and several other constructions is given in § 3 and applications as well as specilizations are studied in § 4. Important for the applications are embeddings of derivations into "restgraphs" which are possible under certain coherence-conditions given in § 5. Finally in § 6 parallel composition of derivations is introduced leading to a derivation-category of a graph-grammar which is a generalization of the x-category in [6]. Functors between derivation-categories seem to be an adequate algebraic formulation of formal translations between grammars which have been studied in [12] .

<u>Example 0.1:</u> Given graph G in fig. 1 we want to replace the specified subgraph 'B by the graph B' specified as a subgraph of graph H in fig. 2

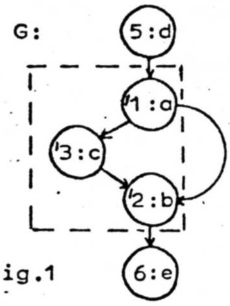


fig.1

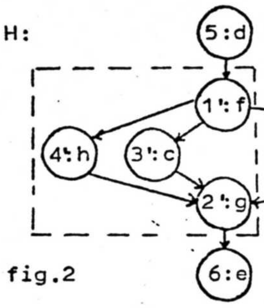


fig.2

using the production $p = ({}'B \xleftarrow{'p} K \xrightarrow{p'} B')$ given as follows: K is the discrete graph with vertices 1 and 2, the graph-homomorphisms 'p, p' are defined by $'p(i)={}'i$, $p'(i) = i'$ $(i=1,2)$. In our example we represent a vertex v with label x in the form v:x , the labels of the edges are erased. Now we chose a partial labeled graph D as given in fig. 3 with inclusion $d:K \longrightarrow D$ in order to show $G \cong D \sqcup_{d,'p} {}'B$ and $H \cong D \sqcup_{d,p'} B'$ and thus a direct derivation $G \xrightarrow{(p,e)} H$ with production p and enlargement $e = (K \xrightarrow{d} D)$.

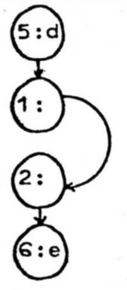


fig. 3

In fact we get graph G up to isomorphism as disjoint union of 'B and D with identified vertices $'1={}'p(1)$ and $1=d(1)$, resp. $'2={}'p(2)$ and $2=d(2)$. This construction is just the pushout-object $D \sqcup_{d,'p} {}'B$ and a similar construction yields, $D \sqcup_{d,p'} B'$ isomorphic to H. We have given labeled graphs 'B and B' but unlabeled K and only partially labeled D in order to get unique labels on G and H.

<u>Remark:</u> Note that 'B is no full subgraph of G because of the edge $'1 \longrightarrow {}'2$. This is in fact a new possibility in graph-grammar concepts.

<u>Example 0.2:</u> A more complicated example is studied now. Given a production $p = ({}'B \xleftarrow{'p} K \xrightarrow{p'} B')$ (first row in fig.11 on the last page)and an enlargement $e = (K \xrightarrow{d} D)$ middle column in fig. 11 we have the graphs $G = D \sqcup_{d,'p} {}'B$ and $H = D \sqcup_{d,p'} B'$ in the second row of fig.11 on the left and right hand side respectively.

The definition of the homomorphisms $'p$, p', d, $'\tilde{d}$, $\tilde{d}'$, $'\tilde{p}$ and $\tilde{p}'$ should be clear by the notation of the vertices, e.g. $'p(1) = {}'1$ and $'p(2) = {}'p(3) = {}'2 = {}'3$. d injective yields $'\tilde{d}$ and $\tilde{d}'$ injective and 'p, p' being not injective yields $'\tilde{p}(2) = {}'\tilde{p}(3) = 2$ and $\tilde{p}'(2) = \tilde{p}'(4) = 2$ respectively. For the label-convention confer example 0.1.

1. Basic definitions and constructions

<u>1.1 Definition :</u> A <u>directed graph</u> G consists of a set of <u>nodes</u> (or <u>vertices</u>) V, a set of <u>edges</u> E together with two maps $s:E \longrightarrow V$ (<u>source</u>) and $t:E \longrightarrow V$ (<u>target</u>):

$$G : E \overset{s}{\underset{t}{\rightrightarrows}} V$$

A <u>homomorphism</u> (or <u>morphism</u>) of graphs $f:G \longrightarrow G'$ is a pair $f = (f_E, f_V)$ of maps $f_E:E \longrightarrow E'$, $f_V:V \longrightarrow V'$ with $f_V \cdot s = s' \cdot f_E$ and $f_V \cdot t = t' \cdot f_E$:

$$\begin{tikzcd}
G: & E \arrow[r, shift left, "s"] \arrow[r, shift right, "t"'] \arrow[d, "f_E"'] \arrow[dr, phantom, "="] & V \arrow[d, "f_V"] \\
G': & E' \arrow[r, shift left, "s'"] \arrow[r, shift right, "t'"'] & V'
\end{tikzcd}$$

Remarks: a) Directed graphs together with graph-morphisms constitute a category Graph (cf.1.2). Composition of morphisms resp. identities are defined by $g\cdot f = (g_E\cdot f_E, g_V\cdot f_V)$ resp. $1_G = (1_E, 1_V)$ (cf. [4] 1.41)

b) With respect to the applications it seems convenient, to consider finite graphs only in order to get finite algorithms for the constructions. For the general theory, however, this restriction is not necessary and all the following constructions restricted to finite graphs lead to finite graphs.

1.2 Definition: A category $\underline{K}$ consists of a class $|\underline{K}|$ of objects together with a family $[\mathrm{Mor}\underline{K}(A,B)]_{A,B\in|\underline{K}|}$ of pairwise disjoint sets of morphisms, a composition $\cdot : \mathrm{Mor}\underline{K}(A,B) \times \mathrm{Mor}\underline{K}(B,C) \longrightarrow \mathrm{Mor}\underline{K}(A,C)$ $(A,B,C \in |\underline{K}|)$, satisfying $(h\cdot g)\cdot f = h\cdot(g\cdot f)$ provided that the compositions are defined and for each object $K\in|\underline{K}|$ an identity morphism 1_K satisfying $1_B\cdot f = f = f\cdot 1_A$ for $f\in \mathrm{Mor}\underline{K}(A,B)$, written $f:A\longrightarrow B$.

A functor $F:\underline{K}\longrightarrow \underline{L}$ between the categories $\underline{K}$ and $\underline{L}$ defines for each $K\in|\underline{K}|$ an object $F(K)\in|\underline{L}|$ and for each morphism $f\in \mathrm{Mor}\underline{K}(A,B)$ a morphism $F(f)\in \mathrm{Mor}\underline{L}(F(A),F(B))$ due to the following axioms:
$F(g\cdot f)=F(g)\cdot F(f)$ and $F(1_K)= 1_{F(K)}$ *)

Examples: a) The category Graph (cf.1.1a)

b) The category Sets resp. Sets^2 consists of all sets resp. all pairs of sets as objects and all mappings resp. pairs of mappings as morphisms. Composition is the usual one resp. usual composition in each component.

c) The forgetful functor U: Graph $\longrightarrow \mathrm{Sets}^2$ is defined by

$UG=(E,V)\in|\mathrm{Sets}^2|$ for $G:E \overset{s}{\underset{t}{\rightrightarrows}} V$ in Graph and

$Uf=(f_E,f_V)\in \mathrm{Sets}^2$ for $f:G\longrightarrow G'\in$ Graph

1.3 Definition: Let $\Omega = (\Omega_E,\Omega_V)\in|\mathrm{Sets}^2|$ be a pair of finite sets (alphabet for the labels of the edges resp. vertices),

$G:E \overset{s}{\underset{t}{\rightrightarrows}} V$ a graph, $m_E:E\rightarrow\Omega_E$, $m_V:V\rightarrow\Omega_V$ maps and $m:=(m_E,m_V)$ (map for labels)
Then (G,m) is called Ω-labeled graph or Ω-graph.

For Ω-graphs (G,m), (G',m') a graph-morphism $f:G\rightarrow G'$ is called Ω-graph-morphism, if $m'\cdot Uf=m$ in Sets^2.

$$\begin{tikzcd}
UG \arrow[d, "Uf"'] \arrow[dr, "m"] \arrow[dr, phantom, shift right=3, "="] & \\
UG' \arrow[r, "m'"'] & \Omega
\end{tikzcd}$$

Remarks: a) $m'\cdot Uf = m$ means $m'_E\cdot f_E = m_E$ and $m'_V\cdot f_V = m_V$

b) Ω-graphs together with Ω-graph-morphisms constitute a category Graph_Ω

c) Ω-graphs are equivalent in the sense of [9] if and only if they are isomorphic in Graph_Ω. *)

*) brackets may be erased, e.g. FK=F(K)

1.4 Definition: The following diagram in a category $\underline{K}$ is called a pushout, if $\tilde{d}\cdot f = d\cdot\tilde{f}$

$$\begin{tikzcd}
K \arrow[r, "f"] \arrow[d, "d"'] \arrow[dr, phantom, "\text{p.o.}"] & B \arrow[d, "\tilde{d}"] \\
D \arrow[r, "\tilde{f}"'] & G
\end{tikzcd}$$

and for all $X\in|\underline{K}|$ and morphisms $g_1 : D\longrightarrow X$, $g_2 : B\longrightarrow X$ with $g_1\cdot d = g_2\cdot f$ there is a unique morphism $g : G\longrightarrow X$ with $g\cdot\tilde{f} = g_1$ and $g\cdot\tilde{d} = g_2$. G is called pushout-object, written $G=D\coprod_{d,f}B$.

Remark: Given f and d the pushout-object G and $\tilde{f}$, $\tilde{d}$ are unique up to isomorphism (easy consequence of the definition).

Examples: a) In Sets the pushout-object $D\coprod_{d,f}B$ of two maps $d : K\longrightarrow D$ and $f : K\longrightarrow B$ is given by the gluing of D and B along K, more precisely

$$D\coprod_{d,f}B := D\dot{\cup}B/_{\bar{R}}$$

where $\bar{R}$ is the induced equivalence-relation of $R(d,f) := \{(d(k),f(k))\,|\,k\in K\}$ on the disjoint union $D\dot{\cup}B$.

The corresponding pushout-diagram is the outer diagram in

$$\begin{tikzcd}
K \arrow[rr, "f"] \arrow[d, "d"'] & & B \arrow[dl, "in_B"'] \arrow[d, "\tilde{d}"] \\
D \arrow[r, "in_D"] \arrow[rr, bend right, "\tilde{f}"'] & D\dot{\cup}B \arrow[r, "nat_{\bar{R}}"] \arrow[r, phantom, shift left=4, ":="] & D\dot{\cup}B/_{\bar{R}}
\end{tikzcd}$$

In example 0.1 we have for the sets of verties $K_V = \{1,2\}$, $D_V = \{1,2,5,6\}$, $B_V = \{'1,'2,'3\}$, d and f inclusions and get $D_V\dot{\cup}B_V/_{\bar{R}} = \{'1,'2,'3,5,6\}$ as vertices of G.

b) In Sets^2 the pushout-object G together with $\tilde{f}$ and $\tilde{d}$ can be constructed componentwise in Sets. The same is true for Graph: Since Graph can be regarded as the functor-category Funkt $(\cdot\rightrightarrows\cdot,$ Sets$)$ it follows that limits and colimits, especially pushouts can be constructed componentwise in Sets (cf. [4] 5.25 ff, [10] 2.7 Satz 1).

Thus given graph-morphisms $f:K\longrightarrow B$, $d:K\longrightarrow D$ we get the (canonical) pushout-object $D\coprod_{d,f}B$ by gluing the edges D_E and B_E along K_E and the nodes D_V and B_V along K_V separately. Source and target-functions of $D\coprod_{d,f}B$ are uniquely determined by those of D an B.

In examples 0.1 and 0.2 we have neglecting the labels:

$$G \cong D\coprod_{d,'p}{}'B \quad\text{and}\quad H \cong D\coprod_{d,p'}B'.$$

*) objects A and B in a category $\underline{K}$ are called isomorphic, written $A\cong B$, if there are morphisms $f:A\longrightarrow B$ and $g:B\longrightarrow A$ such that $g\cdot f=1_A$ and $f\cdot g=1_B$.

1.5 Lemma: Let

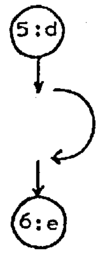
$$\begin{tikzcd} K \arrow[r, "f"] \arrow[d, "d"'] \arrow[dr, phantom, "\text{p.o.}"] & B \arrow[d, "\tilde d"] \\ D \arrow[r, "\tilde f"'] & G \end{tikzcd}$$

be a pushout in Graph with injective graph-morphism d (i.e. d_E, d_V are injective maps).

Then a) $\tilde d$ is injective and
b) $\tilde f$ is injective if and only if f is injective.

Proof: The assertion is true in Sets (easy exercise) and can be carried over to Graph by the construction in example 1.4 b).

In the case of labeled graphs $('B,m_{'B})$ $(B',m_{B'})$, (K,m_K) and Ω-graph-morphisms $'f:(K,m_K)\longrightarrow('B,m_{'B})$ and $f':(K,m_K)\longrightarrow(B',m_{B'})$ we would have for all $k \in K$ $m_{'B}{'f}(k)=m_K(k)=m_{B'}f'(k)$. Thus labels on those edges resp. vertices in $('B,m_{'B})$ and $(B',m_{B'})$ have to coincide which are corresponding gluing points. This would be a hard restriction for the applications which is not satisfied in examples 0.1 and 0.2. In order to avoid this difficulty we don't label the graph K and restrict the label-map m_D to the "restgraph" $\tilde D$ defined by all those edges and nodes in D which are not identified in G with edges and nodes of B. In fact $\tilde D$ is no longer a graph, but only an object in Sets^2. In example 0.1 $\tilde D$ is the set of edges and nodes in the following "restgraph":



1.6 Definition: Let

a) $K,D \in |Graph|$ and $d: K\longrightarrow D$ an injective graph-morphism

b) $\tilde D := (D_E - d_E[K_E],\ D_V - d_V[K_V]) \in$ Sets^2
$\tilde m : \tilde D \longrightarrow \Omega$ a morphism in Sets^2

c) $(B,m_B) \in |Graph_\Omega|$ and $f: K\longrightarrow B$ graph-morphism

d) $G \in |Graph|$ the cononical pushout-object in

$$\begin{tikzcd} K \arrow[r, "f"] \arrow[d, "d"'] \arrow[dr, phantom, "\text{p.o. in Graph}"] & B \arrow[d, "\tilde d"] \\ D \arrow[r, "\tilde f"'] & G \end{tikzcd}$$

Then (G,m_G) with m_G named below is called labeled gluing of (B,m_B) and $(D,\tilde m)$ along K, written: $(G,m_G) = D \underset{d,f}{\overset{\tilde m}{\sqcup}} (B,m_B)$

Construction of m_G:

$$\begin{tikzcd} UK \arrow[r, "Uf"] \arrow[d, "Ud"'] \arrow[dr, phantom, "\text{p.o. in Sets}^2"] & UB \arrow[d, "U\tilde d"] \arrow[ddr, "m_B"] & \\ UD \arrow[r, "U\tilde f"] \arrow[drr, dashed, "m_D"] & UG \arrow[dr, dashed, "m_G"] & \\ \tilde D \arrow[u, "\subseteq"] \arrow[rr, "\tilde m"'] & & \Omega \end{tikzcd}$$

fig. 4

According to the definition of $\tilde D$ and d being injective $UD = (D_E,D_V)$ can be regarded as the disjoint union of UK and $\tilde D$, i.e. coproduct in Sets^2. Thus there is a unique m_D in fig. 4 with $m_D \cdot \subseteq = \tilde m$ and $m_D \cdot Ud = m_B \cdot Uf$. On the other hand $U\tilde d \cdot Uf = U\tilde f \cdot Ud$ is a pushout in Sets^2 by example 1.4 b), thus we have by definition 1.4 a unique m_G in fig. 4 such that $m_G \cdot U\tilde f = m_D$ and $m_G \cdot U\tilde d = m_B$.

Corollary: (B,m_B) is up to isomorphism a sub-Ω-graph of (G,m_G), the same is true for (D,m_D) if and only if f is injective. The proof follows immediately by lemma 1.5.

Explicit construction of (G,m_G): Take the disjoint union of D und B in edges and nodes separately; then for all $v \in K_V$ identify $d_V(v) \in D_V$ with $f_V(v) \in B_V$ in $D_V \dot\cup B_V$ and for all $e \in K_E$ $d_E(e) \in D_E$ with $f_E(e) \in B_E$ in $D_E \dot\cup B_E$; define source s_G and target t_G in G by $s_G([e]) = s_D(e)$ for $e \in D_E$ and $s_G([e]) = s_B(e)$ for $e \in B_E$ (this is in fact well-defined!); take the labels from B on identified edges and nodes, and from D resp. B on the others. In example 0.1 with D in fig.3 and $K = \{1,2\}$ we get the Ω-graphs G and H in fig.1 and fig. 2 taking $B = {'B}$, $f = {'p}$ and $B=B'$, $f=p'$ respectively (d is inclusion, m and m_B like illustrated). The implementation of this construction will be discussed in a subsequent paper.

The construction of the label-map m_G has been done in the category Sets^2, but it turns out that — together with m_K and m_D — we get a pushout-diagram in $Graph_\Omega$ in fact. This is important for further theoretical investigations.

1.7 Lemma: Given the assumptions of 1.6 and

(1) $(G,m_G) = D \underset{d,f}{\overset{\tilde m}{\sqcup}} (B,m_B)$

Then there are unique morphisms $m_K: UK \longrightarrow \Omega$ $(m_K = m_B \cdot Uf)$ and $m_D: UD \rightarrow \Omega$ (induced by m_K and $\tilde m$) such that (G,m_G) is pushout-object in $Graph_\Omega$ in the pushout

(2)

$$\begin{array}{ccc} (K,m_K) & \xrightarrow{\quad f \quad} & (B,m_B) \\ d\big\downarrow & \text{p.o. in } \underline{Graph} & \big\downarrow \tilde d \\ (D,m_D) & \xrightarrow{\quad \tilde f \quad} & (G,m_G) \end{array}$$

Vice versa for each pushout (2) there is a unique $\tilde m : UD - d[K] \to \mathcal{L}$ such that (1) is true.

Proof: By construction of m_D and m_G in 1.6 and definition of $m_K = m_B \cdot Uf$ the morphism $f, d, \tilde f$ and $\tilde d$ are in fact $\mathcal{L}$-graph-morphisms with $\tilde d \cdot f = \tilde f \cdot d$. The universal pushout-property (cf. 1.4) of (2) in $\underline{Graph}_{\mathcal{L}}$ follows from the corresponding (unlabeled) diagram in Graph yielding a unique $\mathcal{L}$-graph-morphism $g: G \to X$ and it remains to show that g is an $\mathcal{L}$-graph-morphism. This can be done by using the pushout-properties of the corresponding diagram in $\underline{Sets}^2$ (cf.1.6). The explicit verification is left to the reader.

2. Graph - Grammars

The gluing of labeled graphs in 1.6 can be regarded as a generalization of the concatenation of strings in the classical case of formal languages. Now we are able to generalize the concepts of direct derivations and formal grammars to the case of labeled graphs; Note that in other graph-grammar concepts only derivations are generalized but not the concatenation of strings.

2.1 Definition: Let $\mathcal{L} = (\mathcal{L}_E, \mathcal{L}_V) \in |\underline{Sets}^2|$ be a label alphabet and $T = (T_E, T_V) \subseteq (\mathcal{L}_E, \mathcal{L}_V)$. Then $(\mathcal{L}, T)$ is called graph-alphabet, T_E and T_V are called terminal alphabet for the edges and the nodes respectively. $\underline{Graph}_T$ can be regarded as a full subcategory of $\underline{Graph}_{\mathcal{L}}$.

2.2 Definition: A graph-grammar is a 4-tuple $Q = (\mathcal{L}, T, S, P)$ with $(\mathcal{L}, T)$ being a graph-alphabet, S a single noded discret graph (no edges) with nonterminal label (initial graph) and P a finite set of productions of the following form:
$p = (('B,'m), 'p, K, p', (B',m'))$ with $('B,'m), (B',m') \in |\underline{Graph}_{\mathcal{L}}|$, $K \in |\underline{Graph}|$, $('B,'m) \notin |\underline{Graph}_T|$, $'p: K \to {}'B$, $p': K \to B'$ graph-morphisms.

2.3 Definition: $(H,m_H) \in |\underline{Graph}_{\mathcal{L}}|$ is directly derivable from (G,m_G) via $p \in P$, written $(G,m_G) \xrightarrow{p} (H,m_H)$, when there is an enlargement $e = (d, (D, \tilde m))$, i.e. $D \in |\underline{Graph}|$, $d: K \to D$ injective graph-morphism and $\tilde m : \bar D \to \mathcal{L}$ as defined in 1.6 b), such that there are isomorphisms in $\underline{Graph}_{\mathcal{L}}$ satisfying

$(G,m_G) \cong D \frac{\tilde m}{d,\,'p} ('B,'m)$ and

$(H,m_H) \cong D \frac{\tilde m}{d,\,p'} (B',m')$

Remarks: a) In examples 0.1 and 0.2 H is derived from G via p in the above defined sense. An isomorphism of labeled graphs is no more than a new denomination of edges and nodes with unchanged graph-structure and labels, i.e. just the property which is appropriate to the applications (cf. introduction).

b) The definition of the productions and direct derivations is in contrast to the definitions in [8,9,11,12,13,14,15] strictly symmetric. For each direct derivation $p: (G,m_G) \to (H,m_H)$ you would get a direct derivation $p^{-1}: (H,m_H) \to (G,m_G)$ with inverse production p^{-1} (change of the components in p) and the same enlargement, provided $p^{-1} \in P$. This possibility is important for recognition and error-correction.

c) Each production p yields a direct derivation $p : ('B,'m) \to (B',m')$ with $D = K$ and $d = 1_K$.

d) Sometimes it is important to note not only the production p but also the enlargement $e = (d, (D, \tilde m))$ for a given direct derivation, written $(p,e) : (G,m_G) \to (H,m_H)$.

2.4 Definition: $(H,m_H) \in |\underline{Graph}_{\mathcal{L}}|$ is called directly derivable from (G,m_G) in Q, written $(G,m_G) \xrightarrow[Q]{} (H,m_H)$, if there is a $p \in P$ with $p: (G,m_G) \to (H,m_H)$. (H,m_H) is called derivable from (G,m_G) in Q, written $(G,m_G) \xrightarrow[Q]{*} (H,m_H)$, if there is a chain of direct derivations

$$(G,m_G) \cong (G_o,m_o) \xrightarrow[Q]{} (G_1,m_1) \xrightarrow[Q]{} \cdots \xrightarrow[Q]{} (G_n,m_n) \cong (H,m_H) .$$

$L(Q) := \{(G,m_G) \mid S \xrightarrow[Q]{*} (G,m_G) \wedge (G,m_G) \in |\underline{Graph}_T|\}$ is called graph-language of Q, i.e. the set of all terminal labeled graphs derivable from the initial graph S in Q.

An immediate consequence of 2.3 remark b) is

2.5 Corollary: The set of terminal labeled graphs that can be parsed in Q, i.e. S is derivable with productions p^{-1} for $p \in P$, is exactly L(Q).

For the applications it is important to know wether a given production p is applicable to a graph (G,m_G) or not and to construct an enlargement $e = (d, (D, \tilde m))$ such that $(G,m_G) \cong D \frac{\tilde m}{d,\,'p} ('B,'m)$. This problem is trivial in the classical case of Chomsky-grammars but most important and nontrivial in the graph-grammar case. It turns out that the existence of such an enlargement can be shown if and only if $('B,'m)$ is a sub-$\mathcal{L}$-graph of (G,m_G) such that a natural "gluing-condition" is satisfied. In this case e is unique up to isomorphism, provided that 'p is injective, otherwise they are "similar relative 'p ".

2.6 Proposition: Let $(G,m_G) \in |\underline{Graph}_{\Omega}|$, $p \in P$ a production $p = (('B,'m),\ 'p,K,p',\ (B',m'))$ and $g : ('B,'m) \longrightarrow (G,m_G)$ be an injective Ω-graph-morphism, then

a) there is an enlargement $e = (d,(D,\tilde{m}))$ such that $(G,m_G) = D \frac{|\tilde{m}|}{d,'p} ('B,'m)$ and $'\tilde{d} = g$ (up to isomorphism) if and only if for $\tilde{G}_E := G_E - g_E\,['B_E]$ and $\tilde{G}_V := G_V - g_V\,['B_V]$ we have:

$s_G[\tilde{G}_E] \cup t_G[\tilde{G}_E] \subseteq \tilde{G}_V \cup g_V{}'p_V\ [K_V]$ (gluing-condition) i.e. sources and targets of edges in G having no origin in 'B under g, belong to the "restgraph - nodes" $\tilde{G}_V$ or they are "gluing-points", i.e. they will be sticked together with vertices of D in the pushout-object.

b) Enlargements $e_i = (d_i,(D_i,\tilde{m}_i))$ $(i=1,2)$ satisfying a) are "$\underline{\text{similar relative 'p}}$", i.e. there is a pair of bijections $b = (b_E,b_V) : UD_1 \longrightarrow UD_2$ satisfying $b \cdot Ud_1 = Ud_2$ and $\tilde{m}_1 = \tilde{m}_2 \cdot b|\tilde{D}_1$ such that

b is a graph-morphism up to 'p-equivalence of vertices; $v,\bar{v} \in D_{2V}$ are called 'p-equivalent, if $v = \bar{v}$ or there are $k,\bar{k} \in K_V$ with $'p_V(k) = {}'p_V(\bar{k})$ and $v = d_{2V}(k)$, $\bar{v} = d_{2V}(\bar{k})$.

c) If 'p is injective then b in b) is an isomorphism in $\underline{\text{Graph}}$, i.e. the enlargement e is unique up to isomorohism.

$\underline{\text{Examples:}}$ a) Take $K = \{1\}$ instead of $K = \{1,2\}$ in example 0.1 then '2 being the source of the edge $'2 \to 6$ in graph G fig.1 doesn't satisfy the gluing-condition. Of course this condition is natural and satisfied in all other examples, but only difficult to formalize.

b) In example 0.2 the graph D can be changed by erasing the edge from 3 to 7 and adding an edge from 2 to 7 s.t. G remains the pushout-object on the left-hand side (of course H must be changed in this case by erasing edge $3 \to 7$ and adding $2 \to 7$).

$\underline{\text{Proof of Prop. 2.6:}}$ a) Consider the diagram in fig.5 with $D_E = \tilde{G}_E \dot{\cup} K_E$, $D_V = \tilde{G}_V \dot{\cup} K_V$, inclusions $i_E, i_V, j_E, j_V, d_E, d_V$,

$'\tilde{p}_E : D_E \longrightarrow G_E$ being induced by $g_E{}'p_E$ and i_E and $'p_V$ by $g_V{}'p_V$ and i_V. In order to construct $s_D : D_E \to D_V$ let $r : K_V \longrightarrow g_V{}'p_V[K]$ be the restriction of $g_V{}'p_V$ and

$c : g_V{}'p_V[K] \longrightarrow K_V$ an arbitrary coretraction of r, i.e. $r \cdot c = id$. We now have a map

$s := (\tilde{G}_E \xrightarrow{\tilde{s}} \tilde{G}_V \dot{\cup} g_V{}'p_V[K_V] \xrightarrow{\tilde{G}_V \dot{\cup} c} \tilde{G}_V \dot{\cup} K_V = D_V)$

with $\tilde{s}$ restriction of s_G (gluing-condition) satisfying $'\tilde{p}_V s = s_G i_E$. We now define

$s_D : \tilde{G}_E \dot{\cup} K_E \longrightarrow D_V$ to be induced by s and $d_V s_K$ yielding $s_D \cdot d_E = d_V \cdot s_K$ and $'\tilde{p}_V s_D = s_G{}'\tilde{p}_E$ by definition of s_D and diagram-chasing in fig. 5 respectively.

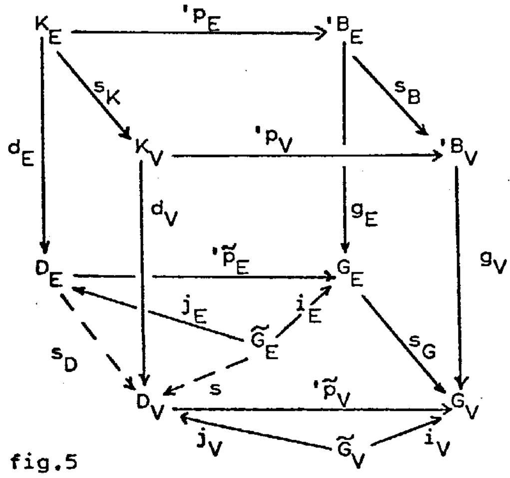


fig.5

Together with a similar construction for $t_D : D_E \longrightarrow D_V$ we get a graph $D = (D_E \overset{s_D}{\underset{t_D}{\rightrightarrows}} D_V)$ such that $d = (d_E,d_V) : K \longrightarrow D$ and $'\tilde{p} = ('\tilde{p}_E, '\tilde{p}_V) : D \longrightarrow G$ are graph-morphisms. Remark that s_D and t_D are not uniquely determined because of the arbitrary choice of the coretraction c, if $'p_V$ is not injective. It now will be shown, that fig. 6 is a pushout in $\underline{\text{Graph}}$.

$$\begin{tikzcd} K \arrow[r, "'p"] \arrow[d, "d"'] & 'B \arrow[d, "g"] \\ D \arrow[r, "'\tilde{p}"'] & G \end{tikzcd}$$

fig. 6

According to 1.4 it is sufficient to show the pushout-property for the V- and E-components respectively in the category $\underline{\text{Sets}}$. Given $h_1 : D_V \longrightarrow X$, $h_2 : {}'B_V \longrightarrow X$ with $h_1 \cdot d_V = h_2 \cdot {}'p_V$ let $\tilde{h}_1 := h_1 j_V : \tilde{G}_V \longrightarrow X$ inducing a unique $h : G_V \to X$ with $h \cdot i_V = \tilde{h}_1$ and $h \cdot g_V = h_2$ (note that $G_V \cong \tilde{G}_V \dot{\cup} {}'B_V$ with injections i_V and g_V). Moreover it is easy to show that h is the unique map satisfying $h \cdot {}'\tilde{p}_V = h_1$ and $h \cdot g_V = h_2$ yielding the pushout-property of fig. 6. Finally according to 1.6 b) we have $\tilde{D} = (\tilde{G}_E, \tilde{G}_V)$ such that $\tilde{m} : \tilde{D} \to \Omega$ can be defined by $\tilde{m} = m_G \cdot i$ yielding $(G,m_G) = D \frac{|\tilde{m}|}{d,'p} ('B,'m)$ and $'\tilde{d} = g$ up to isomorphism. It remains to show the gluing-condition to be necessary, this is straight foreward and will be left for the reader.

b) Let $(G,m_G) = D_1 \frac{|\tilde{m}_1|}{d_1,'p} ('B,'m) = D_2 \frac{|\tilde{m}_2|}{d_2,'p} ('B,'m)$ with $'\tilde{d}_1 = {}'\tilde{d}_2 = g$. Then by construction of $\tilde{G} = (\tilde{G}_E, \tilde{G}_V)$ and of the pushout-object $UG = (G_E, G_V)$ in $\underline{\text{Sets}}^2$ for n=1,2 we have injections $j_n : \tilde{G} \to UD_n$ with $U\tilde{p}_n \cdot j_n = i : \tilde{G} \to UG$ and $UD_1 \cong UK \dot{\cup} G = UD_2$. We thus have a $\underline{\text{Sets}}^2$-isomorphism $b = (b_E, b_V): UD_1 \longrightarrow UD_2$ satisfying $b \cdot j_1 = j_2$, $b \cdot Ud_1 = Ud_2$ and by easy verification $U\tilde{p}_2 \cdot b = U\tilde{p}_1$. For the source-maps $s_n: D_{n_E} \longrightarrow D_{n_V}$ (n=1,2) we thus get by use of the graph-morphisms $\tilde{p}_1$ and $\tilde{p}_2$:

$\tilde{p}_{2V} \cdot b_V \cdot s_1 = \tilde{p}_{1V} \cdot s_1 = s_G \cdot \tilde{p}_{1E} = s_G \tilde{p}_{2E} \cdot b_E = \tilde{p}_{2V} \cdot s_2 \cdot b_E$ and in the same way

$\tilde{p}_{2V} \cdot b_V \cdot t_1 = \tilde{p}_{2V} \cdot t_2 \cdot b_E$ for the target-maps.

In order to show that b is a graph-morphism up to 'p-equivalence of vertices it remains to show that for $v, \bar{v} \in D_{2V}$ with $\tilde{p}_{2V}(v) = \tilde{p}_{2V}(\bar{v})$ we have $v = \bar{v}$ or there are $k, \bar{k} \in K_V$ with $'p_V(k) = {'p_V}(\bar{k})$ and $v = d_{2V}(k)$, $\bar{v} = d_{2V}(\bar{k})$. But this is an easy consequence of $\tilde{p}_{2V} \cdot d_{2V} = g_V \cdot {'p_V} : K_V \longrightarrow G_V$ being a pushout-diagram in Sets with injective d_{2V} (cf. 1.4 a)).

Finally $b[\tilde{D}_1] = bj_1[\tilde{G}] = j_2[\tilde{G}] = [\tilde{D}_2]$ yields (cf.1.6) $\tilde{m}_2 \cdot b|_{\tilde{D}_1} = m_G \cdot U\tilde{p}_2 \cdot i_2 \cdot b|_{\tilde{D}_1} = m_G \cdot U\tilde{p}_2 \cdot b \cdot i_1 = m_G \cdot U\tilde{p}_1 \cdot i_1 = \tilde{m}_1$.

c) $\tilde{p}_2$ is injective by lemma 1.5 b) so that in b) we have $b_V \cdot s_1 = s_2 \cdot b_E$ and $b_V \cdot t_1 = t_2 \cdot b_E$, i.e. b is a morphism and hence an isomorphism in Graph.

3. Hierarchy of Graph-Grammars

In the classical theory, grammars are classified by the shape of the left and right-hand sides of the productions (Chomsky-hierarchy [1,2]). In addition to the type of the productions graph-grammars can be classified by the type of the alphabet and the occuring graphs in productions and enlargements. The dimension of the graph-grammar can be defined by alphabet-type. A new possibility is to regard graph-grammars with enlargement, i.e. for each production $p \in P$ the admissable enlargements in the direct derivations are restricted to belong to a given (infinite) set $\mathcal{E}_p$ of enlargements.

3.1 Definition: A graph-grammar $Q = (\mathcal{L}, T, S, P)$ is of vertex-type if $\mathcal{L}_E = T_E$. In this case the cardinality of $\mathcal{L}_E$ is called dimension of Q.

Motivation: In several applications the edges of the graphs are used to specify geometrical or logical relations between the vertices (cf. 3). Different specifications can be expressed by different labels on the edges. It doesn't seem useful in such applications to distinguish between terminal and nonterminal edges.

3.2 Definition: A graph-grammar $Q = (\mathcal{L}, T, S, P)$ is called to be of graph-type G0, G1, G2, G3 respectively, if for all $p = (('B,'m), 'p, K, p', (B',m')) \in P$ we have:

G0 (general graph-grammar): no restriction

G1 (K-discrete graph-grammar): K is discrete, i.e. without edges

G2 (faithful graph-grammar): $V_K = \{v_1, v_2\}$, $E_K = \emptyset$ and ('B,'m), (B',m') are faithful, i.e. edges with same sources and targets have different labels. In other words: The map $E \longrightarrow V \times V \times \mathcal{L}_E$ induced by the source, target and edge-label maps is injective.

G3 (totally-ordered graph-grammar): G2 is satisfied, Q is of vertex-type, there are no parallel edges, 'B and B' are strictly totaly ordered by the set of edges and $'p(v_1) = \min('B)$, $'p(v_2) = \max('B)$, $p'(v_1) = \min(B')$, $p'(v_2) = \max(B')$.

3.3 Proposition: Graph-grammars of type G0 and G1 are equivalent, i.e. they define the same class of graph-languages.

Proof: Clearly each G1-grammar is G0. Now let Q be G0 and $\dot{Q}$ be the grammar Q with removed edges in the auxiliary graphs K of the productions, such that $\dot{Q}$ is of type G1. $L(Q) = L(\dot{Q})$ is a consequence of the following fact: For each enlargement $d: K \longrightarrow (D, \tilde{m})$ and each $p: K \longrightarrow (B, m)$ we have

$$(G, m_G) = D \frac{|\tilde{m}|}{d,p} (B,m) \cong \dot{D} \frac{|\dot{\tilde{m}}|}{\dot{d},\dot{p}} (B,m) =: (\dot{G}, m_{\dot{G}})$$

with $\dot{D} = D - d[K_E, \emptyset]$ (hence $\tilde{\dot{D}} = \tilde{D}$), $\dot{\tilde{m}} := \tilde{m}$ and $\dot{d}, \dot{p}$ being the restrictions of d,p respectively. The isomorphism is a consequence of the following pushout-construction in Graph:

$$\begin{tikzcd} \dot{K} \arrow[r, "i_K"] \arrow[d, "\dot{d}"'] \arrow[dr, phantom, "(1)"] & K \arrow[r, "p"] \arrow[d, "d"] \arrow[dr, phantom, "(2)"] & B \arrow[d, "\tilde{d}"] \\ \dot{D} \arrow[r, "i_D"'] & D \arrow[r, "\tilde{p}"'] & G \end{tikzcd}$$

fig.7

Diagramm (2) in fig.7 is a pushout by definition, but the same is true for (1) with inclusions i_K and i_D: The V-component is clearly a pushout in Sets and the E-component too, because we have $\dot{K}_E = \emptyset$, $D_E \cong \dot{D}_E \dot{\cup} K_E$ by definition of $\dot{D}$ and d being injective and thus a pushout in Sets by 1.4 a). According to 1.4 b) (1) is a pushout in Graph and hence the exterior diagram in fig.7 is a pushout in Graph by the following lemma 3.4. We thus can choose $\dot{G} = G$ and $m_G = m_{\dot{G}}$ follows from $\tilde{m} = \dot{\tilde{m}}$.

3.4 Lemma: Let

$$\begin{tikzcd} A \arrow[r, "f"] \arrow[d, "a"'] \arrow[dr, phantom, "(1)"] & B \arrow[r, "g"] \arrow[d, "b"] \arrow[dr, phantom, "(2)"] & C \arrow[d, "c"] \\ A' \arrow[r, "f'"'] & B' \arrow[r, "g'"'] & C' \end{tikzcd}$$

be a commutative diagram in an arbitrary category K and (1) a pushout-diagram. Then the exterior diagram is a pushout if and only if (2) is a pushout.

Proof: This is the dual assertion of lemma 3 in [10] § 2.6 .

In the definition of the graph-types G2 and G3 in 3.2 there are only conditions concerning the graphs in the productions. It remains to investigate whether these conditions can be carried over to all derivable Ω-graphs. In fact each Ω-graph derivable in a G3-grammar is totally ordered because this is true for the initial graph S and the availability of the following lemma:

3.5 Lemma: Let p satisfy the assumptions in G3, then we have with the definitions in 2.3: $D\frac{\tilde{m}}{d,p}(B,m)$ is totally ordered if and only if D consists of exactly two totally ordered components D_1 and D_2 satisfying $d(v_1)=\max(D_1)$ $d(v_2)=\min(D_2)$.

The proof is a consequence of the construction in 1.6 and d being injective.

3.6 Corollary: Let Q be of type G3, then each Ω-graph in L(Q) is totally ordered (in the sense of 3.2).

A corresponding lemma for type G2 is not valid. In fact $D\frac{\tilde{m}}{d,p}(B,m)$ faithful yields the same for (B,m) and $(D,\tilde{m})$ but not vice versa. But a similar assertion is valid if we restrict the admissable enlargements in the grammar.

3.7 Definition: A graph-grammar with enlargements $Q = (\Omega,T,S,P,\mathcal{E})$ consists of a graph-grammar (Ω,T,S,P) and a family $\mathcal{E}=[\mathcal{E}_p]_{p\in P}$ of enlargements $e = (d,(D,\tilde{m}))$ (cf.2.3). Direct derivations $(p,e) : (G,m_G) \longrightarrow (H,m_H)$ are assumed to satisfy $e \in \mathcal{E}_p$.

Remark: In contrast to the set of productions P the sets $\mathcal{E}_p$ are not assumed to be finite.

3.8 Definition: A G2 E-grammar is a G2 grammar together with a family of enlargements $[\mathcal{E}_p]_{p\in P}$ such that for each $e=(d,(D,\tilde{m}))\in\mathcal{E}_p$ we have:

a) $(D,\tilde{m})$ is faithful
b) $d(v_1)$ isn't a source and $d(v_2)$ isn't a target of an edge in D
c) No edge from $d(v_2)$ to $d(v_1)$ in D has the same label as an edge from $p'(v_2)$ to $p'(v_1)$ in (B',m') or from $'p(v_2)$ to $'p(v_1)$ in $('B,'m)$.

3.9 Lemma: Let Q be of type G2 E then each Ω-graph in L(Q) is faithful.
Proof: By 3.8 b,c it is impossible that source and target of an edge in 'B or B' is sticked together respectively with source and target of an edge in D with same label.

Lemma 3.5 leads to a similar definition of graph-type G3 E such that graph-grammars of type G3 and G3 E are equivalent:

3.10 Definition: A G3 E-grammar is a G3 grammar together with enlargements such that for each $e \in \mathcal{E}_p$ $(p \in P)$:

a) D consists of two totally ordered components D_1 and D_2
b) $d(v_1) = \max(D_1)$, $d(v_2) = \min(D_2)$.

3.11 Definition: In a graph-grammar with enlargements $\mathcal{E}_p$ $(p\in P)$ is called unique if for all Ω-graphs (G,m_G) and all injections $'g:('B,'m) \longrightarrow (G,m_G)$ there is up to isomorphism at most one enlargement $e=(d,(D,\tilde{m}))\in\mathcal{E}_p$ satisfying $(G,m_G) \cong D\frac{\tilde{m}}{d,'p}('B,'m)$ and $'\tilde{d} \cong 'g$ and a corresponding condition for all injections $g' : (B',m') \longrightarrow (G,m_G)$.

3.12 Crollary: In graph-grammars with unique enlargements the derivation and parsing process is uniquely determined by specification of the productions and the injections of the sub-Ω-graphs which have to be replaced.

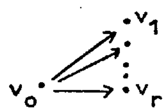
3.13 Proposition: A set of enlargements $\mathcal{E}_p$ is unique if and only if all enlargements similar relative 'p are isomorphic and the same for those being similar relative p'. The condition "$'p_V$ and p'_V injective" is sufficient.
The proof is evident using 2.6 and 3.11.

3.14 Theorem: Graph-grammars of type G2 E and G3 E have unique enlargements.
Proof: Since G3E is a special case of G2E it suffices to show it for type G2E. Given similar enlargements relative 'p (cf. 2.6 b) we have a bijection $b = (b_E,b_V) : UD_1 \longrightarrow UD_2$. Let $e : x \longrightarrow x'$ be an edge in D_1 and $b_E(e):y \longrightarrow y'$ in D_2 then by assumption y and y' is 'p-equivalent to $b_V(x)$ and $b_V(x')$ respectively. If $y' \neq b_V(x')$ then $y' = d_2(v_1)$ and $b_V(x')=d_2(v_2)$ ($y'=d_2(v_2)$ is impossible with respect to 3.8b) Now we have $x'=b_V^{-1}\cdot d_2(v_2)=d_1(v_2)$ yielding a contradiction to 3.8 b). Thus we have $y'=b_V(x')$ and in the same way $y=b_V(x)$ showing $b:D_1 \longrightarrow D_2$ to be a graph-isomorphism. Hence we have the assertion by proposition 3.13.

Finally we define the well-known Chomsky-classification [1,2] for graph-grammars:

3.15 Definition: A graph-grammar $Q=(\Omega,T,S,P)$ or $Q=(\Omega,T,S,P,\mathcal{E})$ is called to be of production-type P0, P1, P2, P3 respectively, if for all $p=(('B,'m),'p,K,p',(B',m'))\in P$ we have:

P0 (general graph-grammar): no restriction
P1 (monotone graph-grammar): card $('B_V) \leq$ card $(B'_V) \wedge$ card $('B_E) \leq$ card (B'_E)
P2 (context-free graph-grammar): card $('B_V) = 1 \wedge$ card $('B_E) = 0$
P3 (left-regular): contextfree and B' has shape



with nonterminal label on v_o exactly in the case $r > o$ and terminal labels on $v_1, \ldots, v_r$.

Of course there are many other possible generalizations of Chomsky-type P3 which need not to be equivalent in the graph-grammar case.

4. Special Graph-Grammars

It will be shown that formal grammars defined in [1,2], [12] and [13] are special cases of graph-grammars.

4.1 Theorem: Graph-grammars of vertex-type, graph-type G2E and production-type P2 coincide with those defined by Pratt in [12].

The proof is an immediate consequence of the definitions. In fact direct derivations in the sense of [12] turn out to be constructed by pushouts satisfying the conditions of G2E-grammars and vice versa. Moreover the grammars in [12] are context-free and the edges in the graphs are terminal labeled.

4.2 Theorem: Graph-grammars of vertex-type with dimension 1, graph-type G3 and production-type P0, P1, P2, P3 respectively coincide with ϵ-free Chomsky-grammars of type 0, 1, 2, 3 respectively, if each isomorphism class of labeled and totally ordered graphs will be identified with the corresponding string of labels over the alphabet Ω_V.

Proof: By lemma 3.5 it suffices to take graph-type G3E. The left and right hand side of a production p in the graph-grammar sense (are non-empty) can be identified with a pair of non-empty strings (x,y) and vice versa. The components of an enlargement $e \in \mathcal{E}_p$ correspond to strings u,w in the classial definition of direct derivations $uxw \longrightarrow uyw$. But note that the strings $u = u_1u_2 \ldots u_r$ and $w = w_1w_2 \ldots w_s$ $(u_i, w_j \in \Omega_V)$ correspond to a totally ordered but only partially labeled graph of shape

$$(D,\tilde{m}): \overset{u_1}{\bullet}\to\overset{u_2}{\bullet}\to\cdot\to\overset{u_r}{\bullet}\to\underset{d(v_1)}{\bullet}\quad\underset{d(v_2)}{\bullet}\to\overset{w_1}{\bullet}\to\overset{w_2}{\bullet}\to\cdot\to\overset{w_s}{\bullet}$$

The nonlabeled vertices are exactly $d(v_1)$ and $d(v_2)$ in 3.10. Since ('B,'m) in production p has the shape

$$('B,'m): \underset{'p(v_1)}{\overset{x_1}{\bullet}}\longrightarrow\overset{x_2}{\bullet}\longrightarrow\ \ldots\ \longrightarrow\underset{'p(v_2)}{\overset{x_q}{\bullet}}$$

the left hand side of the direct derivation $uxw \longrightarrow uyw$ can be identified with

$$D \underset{d,'p}{\overset{\tilde{m}}{\perp\!\!\perp}} ('B,'m): \overset{u_1}{\bullet}\to\cdot\cdot\to\overset{u_r}{\bullet}\to\overset{x_1}{\bullet}\to\cdot\to\cdot\to\overset{x_q}{\bullet}\to\overset{w_1}{\bullet}\to\cdot\cdot\to\overset{w_s}{\bullet}$$

In that way direct derivations in the Chomsky-case can be identified with those in the graph-grammar-case of vertex-type with dimension 1 and graph-type G3 and vice versa. Clearly the production-types P0,P1,P2,P3 correspond to Chomsky-types 0,1,2,3 respectively.

4.3 Proposition: Web-grammars and derivations in the sense of Rosenfeld and Milgram [13] can be simulated in the graph-grammar concept, but not vice versa.

Proof: Let (α,β,Φ) be a production in the sense of [13] yielding a direct derivation from web ω to ω' then there is a corresponding production and direct derivation in the graph-grammar sense defined in the following way:

$K = N_\beta \times N_\alpha$ (discrete graph with N_β, N_α being the set of nodes of β and α respectively), $p_\alpha: K \to \alpha$ and $p_\beta: K \to \beta$ defined by the projections. D consists of the nodes $(N_\omega - N_\alpha) \cup K$ together with all edges of ω between nodes in $N_\omega - N_\alpha$ and in addition there is an edge between $v \in (N_\omega - N_\alpha)$ and $(n',n) \in K$ in D if and only if

i) there is an edge between v and n in ω
ii) the label of v belongs to the set $\Phi(n',n)$

Further let $d: K \to D$ be the inclusion and $\tilde{m}$ be the restriction of the label-function belonging to ω (note that in [13] only the nodes are labeled corresponding to graph-grammars of vertex-type with dimension 1). Now we take the production $p = (\alpha, p_\alpha, K, p_\beta, \beta)$ and the enlargement $e = (d,(D,\tilde{m}))$ which yield in fact a direct derivation $(p,e): \omega \to \omega'$, provided the corresponding pushout-constructions are performed in the sense of undirected graphs without parallel edges as proposed in [13], i.e. parallel edges have to be identified.

Conversely in example 0.1 we have a direct derivation such that there is no corresponding production (with the same left and right hand side) and direct derivation in the sense of [13], because 'B is no full subgraph of G.

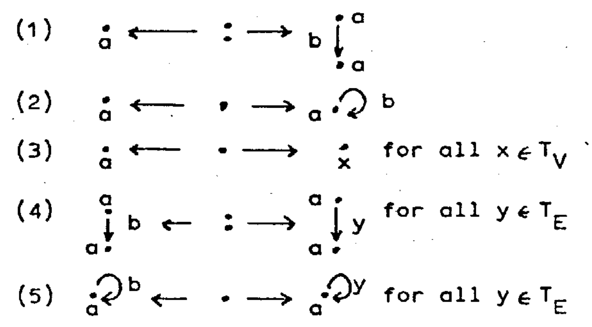
4.4 Remark: n-diagrams in the sense of [9,10,11] can be regarded as labeled graphs in an easy way, and with certain restrictions, which cannot be discussed here in detail, direct derivations can be carried over to graph-grammars of type G1, P0.

4.5 Proposition and example: a) Let Q be a graph-grammar such that for each $p \in P$ the graphs 'B and B' in the production p are connected then L(Q) consists only of connected graphs.

b) conversely the following graph-grammar $Q = (\Omega, T, S, P)$ yields a graph-language L(Q) consisting of all nonemty, finite, connected graphs with labels in $T = (T_E, T_V)$:

$\Omega_E = T_E \dot\cup \{b\}$, $\Omega_V = T_V \dot\cup \{a\}$ $S = \bullet a$

The productions are noted in the following way
$('B,'m) \xleftarrow{'p} K \xrightarrow{p'} (B',m')$:



Proof: a) It suffices to show $G = D \underset{d,'p}{\perp\!\!\perp} 'B$ and $H = D \underset{d,p'}{\perp\!\!\perp} B'$ to be connected in the case of connected graphs G,'B,B'. Suppose D has n components $D_1, \ldots, D_n$. Since G is connected there are vertices $k_1,\dots,k_n \in K_V$ satisfying $d_V(k_i) \in V_{D_i}$ $(i=1,\dots n)$. But $d_V(k_i)$ and $p'_V(k_i)$ are glued together respectively in the pushout yielding a connected graph H.

b) According to a) each graph in L(Q) is connected, finite and non-empty. Conversely it suffices to show that any connected Ω-graph with $\Omega = (\{b\}, \{a\})$ can be derived in Q, since productions (3), (4) and (5) can than be used to relabel this graph with arbitrary terminal labels. The proof will be shown by induction on the number of vertices $|G_V|$ in G. In case $|G_V| = 1$ rule (2) has to be applied n times $(n \geq 0)$. Now let $|G_V| = k+1$ and r,s vertices in G joined by an edge. The reserval of production (1) yields a graph G' with $|G'_V| = k$ which can be deduced by induction hypotheses. Since direct derivations are reversible (cf. 2.3 Remark b) G is derivable in Q too.

4.5 Remark: Graph-grammars of type G2, P3 can be used to derive trees. An example is given in [5] 3.4 showing that difficult transformations can be done in this concept.

5. Embedding of Derivations.

With respect to the applications it is most important to embed derivations into given "rest-graphs" in the sense of the gluing construction in 1.6 and to show that this again yields a derivation. This corresponds to following classical situation. For each (direct) derivation $uvw \longrightarrow u\bar{v}w$ in a Chomsky-grammar and all strings u',w' there is a (direct) derivation $u'uvww' \longrightarrow u'u\bar{v}ww'$. This problem is not trivial in the graph-grammar case and can in fact be solved only under certain natural coherence-conditions. The main tool in the proofs is lemma 3.4 showing the composition of two pushouts to be a pushout. We proceed in three steps and give an example in 5.4

1. Repeated gluings of Ω-graphs can be realized in one step (5.1)
2. The embedding of a direct derivation yields a direct derivation (5.2)
3. The embedding of an arbitrary derivation (in the sense of the gluing construction in 1.6) yields a derivation (5.3)

It should be noted that the exact formulation of these embedding problems is based on our concept of gluing graphs in 1.6 and there are no similar constructions in all the other graph-grammar concepts.

5.1 Lemma: Given a graph-morphism $p: K \longrightarrow (B,m_B)$ and an enlargement $e = (K \xrightarrow{d} (D,\tilde{m}))$ yielding a pushout in fig.8, a further graph-morphism $q: \bar{K} \longrightarrow D$ together with an enlargement $\bar{e} = (\xrightarrow{\bar{d}} (\bar{D},\tilde{\bar{m}}))$ then there is an enlargement $\hat{e} = (K \xrightarrow{\hat{d}} (\hat{D},\tilde{\hat{m}}))$ such that:

$$\bar{D}\frac{|\tilde{\bar{m}}|}{\bar{d},\tilde{p}\cdot q}\left(D\frac{|\tilde{m}|}{d,p}(B,m_B)\right) \cong D\frac{|\tilde{\hat{m}}|}{\hat{d},p}(B,m_B)$$

$$\begin{tikzcd}
K \arrow[r, "p"] \arrow[d, "d"'] & B \arrow[d, "\tilde{d}"] \\
D \arrow[r, "\tilde{p}"] & G
\end{tikzcd}$$

fig. 8

Proof: In the $\underline{Graph}_\Omega$-diagramm of fig. 9

$$\begin{tikzcd}
 & (K,m_K) \arrow[r, "p"] \arrow[d, "d"] \arrow[dr, phantom, "(1)"] & (B,m_B) \arrow[d, "\tilde{d}"] \\
(\bar{K},m_{\bar{K}}) \arrow[r, "q"] \arrow[d, "\bar{d}"] \arrow[dr, phantom, "(2)"] & (D,m_D) \arrow[r, "\tilde{p}"] \arrow[d, "\bar{\bar{d}}"] \arrow[dr, phantom, "(3)"] & (G,m_G) \arrow[d, "\tilde{\bar{\bar{d}}}"] \\
(\bar{D},m_{\bar{D}}) \arrow[r, "\tilde{q}"] & (\hat{D},m_{\hat{D}}) \arrow[r, "\tilde{\bar{p}}"] & (\hat{G},m_{\hat{G}})
\end{tikzcd}$$

fig. 9

diagrams (1), (2) and (3) are constructed one after another as pushout-diagrams according to definition 1.6 and lemma 1.7. We thus have:

$(G,m_G) \cong D\frac{|\tilde{m}|}{d,p}(B,m_B)$ by lemma 1.7

$(\hat{G},m_G) \cong \bar{D}\frac{|\tilde{\bar{m}}|}{\bar{d},\tilde{p}\cdot q}(G,m_G)$ by lemma 3.4

and taking $\hat{d}=\bar{\bar{d}}\cdot d$, $\tilde{\hat{m}}$ restriction of $m_{\hat{D}}$ to $U\hat{D} - \hat{d}[K]$

$(\hat{G},m_{\hat{G}}) \cong \hat{D}\frac{|\tilde{\hat{m}}|}{\hat{d},p}(B,m_B)$ by lemma 3.4

yielding the desired result.

5.2 Proposition: Let $Q = (\Omega,T,S,P)$ be a graph-grammar, $(p,e): (G,m_G) \longrightarrow (H,m_H)$ a direct derivation with enlargement $e = (K \xrightarrow{d} (D,\tilde{m}))$, $\bar{e} = (\bar{K} \xrightarrow{\bar{d}} (\bar{D},\tilde{\bar{m}}))$ a further enlargement and $'f: \bar{K} \longrightarrow G$, $f': \bar{K} \longrightarrow H$ graph-morphisms. Then there is an enlargement $\hat{e} = (K \xrightarrow{\hat{d}} (\hat{D},\tilde{\hat{m}}))$ yielding a direct derivation

$$(p,\hat{e}) : \bar{D}\frac{|\tilde{\bar{m}}|}{\bar{d},'f}(G,m_G) \longrightarrow \bar{D}\frac{|\tilde{\bar{m}}|}{\bar{d},f'}(H,m_H)$$

provided that the following coherence-condition is satisfied: There is a graph-morphism $q: \bar{K} \longrightarrow D$ such that $'f = '\tilde{p}\cdot q$ and $f' = \tilde{p}'\cdot q$ with $'\tilde{p},\tilde{p}'$ defined by $'p,p'$ respectively as given in fig.8 of lemma 5.1.

Interpretation: Given a direct derivation defined by gluing a production together with an enlargement D we again get a direct derivation by gluing both sides of the direct derivation together with another enlargement $\bar{D}$ where the gluing points have to belong to the same subgraph $q[\bar{K}]$ of D. The gluing points may in fact be coincide with those of the first construction and additional identifications are possible, since q is not supposed to be injective.

Proof: According to lemma 5.1 we have:

$$(\hat{G},m_{\hat{G}}) := \bar{D}\frac{|\tilde{\bar{m}}|}{\bar{d},'f}(G,m_G) = \bar{D}\frac{|\tilde{\bar{m}}|}{\bar{d},'\tilde{p}\cdot q}(G,m_G) \cong \hat{D}\frac{|\tilde{\hat{m}}|}{\hat{d},'p}('B,'m)$$

$$(\hat{H},m_{\hat{H}}) := \bar{D}\frac{|\tilde{\bar{m}}|}{\bar{d},f'}(H,m_H) \cong \hat{D}\frac{|\tilde{\hat{m}}|}{\hat{d},p'}(B',m')$$

The construction of pushout (2) in fig.9 regarded as graph-diagram only depends on q and $\bar{d}$; hence $\hat{D}$, $\tilde{q}$ and $\bar{d}$ are the same for both sides of the derivation. That is not true for $m_{\hat D}$ which is defined on $\hat{d}[K] = \bar{\bar{d}}\cdot d[K]$ by $'m\cdot{}'p$ and $m'\cdot p'$ respectively. In order to get the derivation $(p,\hat{e}) : (\hat{G},m_{\hat G}) \longrightarrow (\hat{H},m_{\hat H})$ it remains to show $'\tilde{\hat{m}} = \hat{\tilde{m}}'$. In fact the restriction $\tilde{\hat{m}} : U\hat{D} - \hat{d}[K] \longrightarrow \mathcal{R}$ of $m_{\hat D}$ in fig. 9 only depends on $\tilde{m}$, $\tilde{\bar{m}}$, $\hat{D}$, $\tilde{q}$, $\bar{\bar{d}}$, which will be shown for each $x \in U\hat{D} - \hat{d}[K]$ being an edge or a vertex in a lax notation:

1. case: $x \in U\hat{D} - \bar{\bar{d}}[D]$. We have $x \notin \bar{\bar{d}}\cdot q[\bar{K}] = \tilde{q}\cdot\bar{d}[\bar{K}]$ so there is a unique $\bar{x} \in U\bar{D} - \bar{d}[\bar{K}]$ satisfying $x = \tilde{q}(\bar{x})$. Hence $\tilde{\hat{m}}(x) = m_{\hat D}(x) = m_{\hat D}\cdot\tilde{q}(\bar{x}) = m_{\bar D}(\bar{x}) = \tilde{\bar{m}}(\bar{x})$ since $\bar{x} \notin \bar{d}[\bar{K}]$.
2. case: $x \in \bar{\bar{d}}[D] - \hat{d}[K] = \bar{\bar{d}}[D] - \bar{\bar{d}}\cdot d[K]$. We have $\tilde{\hat{m}}(x) = m_{\hat D}(x) = m_D(\bar{\bar{d}}^{-1}(x) = \tilde{m}(\bar{\bar{d}}^{-1}(x)$, since $\bar{\bar{d}}$ is injective and $\bar{\bar{d}}^{-1}(x) \notin d[K]$.

We thus have shown $'\tilde{\hat{m}} = \hat{\tilde{m}} = \hat{\tilde{m}}'$.

Proposition 5.2 is a special case of the following theorem:

<u>5.3</u> <u>Theorem</u>: Let

$$(G_o,m_{G_o}) \xrightarrow[(p_1,e_1)]{} (G_1,m_{G_1}) \longrightarrow \cdots \xrightarrow[(p_n,e_n)]{} (G_n,m_{G_n})$$

be a derivation in a graph-grammar $Q=(\mathcal{R},T,S,P)$ and let $'q:\bar{K}\longrightarrow G_o$, $q':\bar{K}\longrightarrow G_n$ be graph-morphisms and $\bar{e} = (\bar{K}\xrightarrow{\bar{d}}(\bar{D},\bar{m}))$ an enlargement. Then there is a derivation

$$\bar{D}\frac{|\tilde{\bar{m}}|}{\bar{d},{}'q}(G_o,m_{G_o}) \xrightarrow[Q]{*} \bar{D}\frac{|\tilde{\bar{m}}|}{\bar{d},q'}(G_n,m_{G_n})$$

provided $'q$, q' satisfy the following coherence-condition: There are morphisms $q_i : \bar{K}\longrightarrow D_i$ $(1\leq i\leq n)$ such that

(1) $\quad 'q = {}'\tilde{p}_1\cdot q_1$

(2) $\quad q' = \tilde{p}'_n\cdot q_n$

(3) $\quad \tilde{p}'_i\cdot q_i = {}'\tilde{p}_{i+1}\cdot q_{i+1} \quad (1\leq i\leq n)$

where $'\tilde{p}_i, \tilde{p}'_i$ are defined by the following pushout respectively

$$\begin{tikzcd}
{}'B_i \arrow[d] \arrow[dr, phantom, "p.o"] & K_i \arrow[l, "{}'p_i"'] \arrow[r, "p'_i"] \arrow[d, "d_i"] \arrow[dr, phantom, "p.o."] & B'_i \arrow[d] \\
G_i & D_i \arrow[l, "{}'\tilde{p}_i"] \arrow[r, "\tilde{p}'_i"'] & H_i
\end{tikzcd}$$

<u>Remark:</u> a) We use the notation $p_i=(('B_i,{}'m_i),{}'p_i,K_i,p'_i,(B'_i,m'_i))$ and $e_i = (K_i \xrightarrow{d_i} (D_i,\tilde{m}_i))$.

b) Conditions 1), 2) and 3) can be expressed by the commutativity of the following diagram:

$$\begin{tikzcd}
 & & G_o \\
 & D_1 \arrow[ur, "{}'\tilde{p}_1"] \arrow[dr, "\tilde{p}'_1"'] & \\
 & & G_1 \\
\bar{K} \arrow[uuurr, bend left, "{}'q"] \arrow[uur, "q_1"] \arrow[r, "q_2"] \arrow[dddr, "q_n"] \arrow[ddddrr, bend right, "q'"'] & D_2 \arrow[ur, "{}'\tilde{p}_2"] \arrow[dr, "\tilde{p}'_2"'] & \\
 & & G_2 \\
 & \vdots & \vdots \\
 & D_n \arrow[dr, "\tilde{p}'_n"] & \\
 & & G_n
\end{tikzcd}$$

<u>Interpretation:</u> The coherence-condition means the following: Each gluing point $'q(\bar{k})$ in G_o can be transmitted to all derivation-steps $(\tilde{p}'_i\cdot q_i(\bar{k}) = {}'\tilde{p}_{i+1}\cdot q_{i+1}(\bar{k}) \in G_i)$ leading to the gluing-point $q'(\bar{k}) \in G_n$. This will be discused in example 5.4 in more detail.

<u>Proof</u>: According to (3) and proposition 5.2 there are direct derivations

$$\bar{D}\,\frac{|\tilde{\bar{m}}|}{d,{}'\tilde{p}_i\cdot q_i}\,(G_{i-1},m_{G_{i-1}}) \longrightarrow \bar{D}\frac{|\tilde{\bar{m}}|}{\bar{d},\tilde{p}'_i\cdot q_i}(G_i,m_{G_i})$$

for $1\leq i\leq n$. Taking

$$(\bar{G}_i,m_{\bar{G}_i}) := \bar{D}\,\frac{|\tilde{\bar{m}}|}{\bar{d},\tilde{p}'_i\cdot q_i}(G_i,m_{G_i})$$

we have according to (3) for $1<i\leq n$ and define for $i = 1$

$$(\bar{G}_{i-1},m_{\bar{G}_{i-1}}) = \bar{D}\,\frac{|\tilde{\bar{m}}|}{\bar{d},{}'\tilde{p}_i\cdot q_i}(G_{i-1},m_{G_{i-1}})$$

We thus have a sequence of direct derivations

$$(\bar{G}_o,m_{\bar{G}_o}) \longrightarrow (\bar{G}_1,m_{\bar{G}_1}) \longrightarrow \cdots \longrightarrow (\bar{G}_n,m_{\bar{G}_n})$$

which coincides with the desired derivation by (1) and (2).

<u>5.4</u> <u>Example</u>: In fig.12 (last page) we have given a derivation of a flowchart . Subgraphs framed in the same way correspond to the left and right hand side of a production respectively. Now we want to glue these graphs together with the frame $\bar{D}$ (consisting of too components) given in fig.10 such that we get a direct derivation again. According to theorem 5.3 we are not free in the choice of $'q$ and q', since the gluing points indicated by the same numbers respectively must be compatible with the productions: Vertices not touched by a production don't change their numbers, otherwise the number is carried over by $\tilde{p}'_i$ to another vertex belonging to the right hand side of the production. For example $*$ in fig. 12 cannot be a gluing point with number 3.

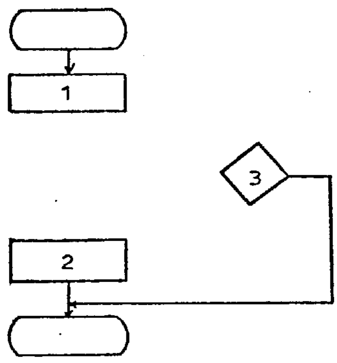


fig. 10

6. Derivationcategory and Formal Translations

In order to formalize series- and parallel-composition of direct derivations we define in analogy to the classial case in [6] a derivationcategory of a graph-grammar. This category can be used to deal with the following problems which will be discussed in a subsequent paper in more detail:

1. Recognition of Ω-graphs
2. Uniqueness of derivations
3. Decomposition of derivations in series- and parallel-composition of direct derivations
4. Minimization, reduction and normal forms of productions
5. Formalization of translations

6.1 Construction: Let $Q = (\Omega, T, S, P)$ be a graph-grammar. We first construct a graph $\underset{\sim}{Q}$ of direct derivations in Q: Vertices in $\underset{\sim}{Q}$ are all isomorphismclasses of Ω-graphs, edges from $[(G,m_G)]$ to $[(H,m_H)]$ are all direct derivations $(p,e):(G,m_G) \longrightarrow (H,m_H)$. Two such direct derivations (p,e) and (p',e') are assumed to be equal if and only if $p=p'$ and e and e' are isomorphic in the sense of 2.6,c. Now let $F\underset{\sim}{Q}$ be the free category of path's (i.e. derivations) over $\underset{\sim}{Q}$. Two derivations

$$(G_o,m_{G_o}) \xrightarrow[(p_1,e_1)]{} (G,m_{G_1}) \to \cdots \xrightarrow[(p_n,e_n)]{} (G_n,m_{G_n})$$

and

$$(\hat{G}_o,m_{\hat{G}_o}) \xrightarrow[(\hat{p}_1,\hat{e}_1)]{} (\hat{G}_1,m_{\hat{G}_1}) \to \cdots \xrightarrow[(\hat{p}_n,\hat{e}_n)]{} (\hat{G}_n,m_{\hat{G}_n})$$

are called equivalent, if

$(G_o,m_{G_o}) \cong (\hat{G}_o,m_{\hat{G}_o})$, $(G_n,m_{G_n}) \cong (\hat{G}_n,m_{\hat{G}_n})$,

$(\hat{p}_i)_{1\le i\le n}$ is a permutation of the family $(p_i)_{1\le i\le n}$ and if there are injective $\underline{Graph}_\Omega$ -morphisms

$$q_i : \left(\coprod_{1\le j\le i}(B'_j,m'_j)\right) \amalg \left(\coprod_{i\le k\le n}('B_k,'m_k)\right) \longrightarrow$$

$\to (G_i,m_{G_i})$ such that

$q_i \cdot u_i = \tilde{d}'_i \;:\; (B'_i,m'_i) \longrightarrow (G_i,m_{G_i})$ and

$q_i \cdot u_{i+1} = '\tilde{d}_{i+1} : ('B_{i+1},'m_{i+1}) \longrightarrow (G_i,m_{G_i})$

and similar morphisms $\hat{q}_i$ $(1\le i\le n)$. ($\amalg$ means the coproduct i.e. disjoint union of Ω-graphs with injections u_i); especially we have

$\coprod_{1\le k\le n}('B_k,'m_k)$ is sub-Ω-graph of (G_o,m_{G_o})

and

$\coprod_{1\le j\le n}(B'_j,m'_j)$ is sub-Ω-graph of (G_n,m_{G_n})

Finally let $\underline{Q}$ be the quotientcategory of $F\underset{\sim}{Q}$ relative to the above defined equivalence of derivations, $\underline{Q}$ is called the derivationcategory of Q.

Remark: According to this construction we do not distinguish derivations which have the same source and target and consist of the same set of productions applied to pairwise disjoint subgraphs (perhaps in a different succession). Such derivations can be called parallel-composition of the corresponding productions. The construction is a generalization of the free X-category of formal languages defined in [6].

6.2 Corollary: $(G,m_G) \in |\underline{Graph}_T|$ belongs to $L(Q)$ if and only if there is a morphism in the derivationcategory $\underline{Q}$ from S to (G,m_G).

6.3 Definition: $(H,m_H) \in |\underline{Graph}_\Omega|$ is called uniquely derivable from $(G,m_G) \in |\underline{Graph}_\Omega|$ if $|Mor\underline{Q}((G,m_G),(H,m_H))| = 1$. Q is called unique, if for all $(G,m_G) \in |\underline{Graph}_T|$ we have $|Mor\underline{Q}(S,(G,m_G))| \le 1$.

6.4 Definition: Let Q and Q' be graph-grammars with derivationcategories $\underline{Q}$ and $\underline{Q}'$ respectively. Then a functor $C : \underline{Q} \to \underline{Q}'$ is called formal translation from Q to Q'.

Remark: Formal translations preserve the structure of the derivations ("syntaxtree")

6.5 Examples: a) Pair-grammars in the sense of [12] define functors between the derivation-categories of the corresponding graph grammars, which are of type G2,P2 in our sense (cf. § 3). Those pair-grammars are introduced to define translations between different types of grammars, several examples are given in [12]. Thus we can use functors to define translations between graph grammars. In this way the series-composition of translations can be reduced to the composition of functors and we hope that related problems can be solved using the highly developed theory of categories and functors.

b) Reductions of Chomsky-grammars can be regarded as functors between the corresponding free X-categories (cf. [6] §3). But graph grammars and derivation-categories are generalizations of Chomsky-grammars and corresponding free X-categories, so that it is reasonable to generalize the reduction and minimization too.

6.6 Conclusion: Of course the ideas given up to now don't state a complete theory of graph grammars, but only some fundamental concepts. Further developments of these concepts are in preparation. On the other hand we are quite sure that our approach will allow practical applications (algorithms are in work) and still more theoretical results concerning the problems stated in the introduction of § 6.

References

[1] N. Chomsky, On certain formal properties of grammars
Inf. Contr. 2,2 (1959), p.137-167

[2] N. Chomsky, A note on phrase structure grammars
Inf.Contr. 2,4 (1959), p.393-395

[3] V. Claus, Ein Vollständigkeitssatz für Programme und Schaltkreise
Acta Informatica 1,1 (1971), p.64-78

[4] H. Ehrig / M. Pfender, Kategorien und Automaten
Vlg. W. de Gruyter, Berlin-New York 1972

[5] H. Ehrig / M. Pfender / H.J. Schneider, Kategorielle Konstruktionen in der Theorie der Graph-Grammatiken, Fachgespräch über mehrdimensionale formale Sprachen, Erlangen Juni 1973
Arb.berichte Inst.Math.Masch.Dat.verarb. vol.6, No.3 (1973), p. 30-55

[6] G. Hotz, Reduktionssätze über eine Klasse formaler Sprachen mit endlich vielen Zuständen
Math. Zeitschrift 1o4 (1968),p.2o5-221

[7] D. Milgram / A. Rosenfeld, Array automata and array grammars
Proc. IFIP Congr. 1971, Booklet TA-2, p. 166-173 (Preprints)

[8] J. Mylopoulos, On the relation of graph automata and graph grammars.
Univ.Toronto, Dept.Computer Science, Technical Report 34 (1971)

[9] M. Nagl, Eine Präzisierung des Pfaltz-Rosenfeldschen Produktionsbegriffs bei mehrdimensionalen Grammatiken
Fachgespräch über mehrdimensionale formale Sprachen, Erlangen Juni 1973
Arb.berichte Inst.Math.Masch.Dat.verarb. vol. 6, No. 3 (1973)

[10] B. Pareigis, Kategorien und Funktoren
Teubner Vlg., Stuttgart 1969

[11] J.L. Pfaltz / A. Rosenfeld, Web grammars
Proc. Int. Joint Conf. Artif. Intell., Washington, 1969, p. 6o9-619

[12] T.W. Pratt, Pair grammars, graph languages and string-to-graph translations
J.Comp.Syst.Sciences 5 (1971),p.56o-595

[13] A. Rosenfeld / D.C.Milgram, Web automata and Web Grammars
Machine Intelligence 7,p.3o7-324 (1972)

[14] H.J. Schneider, Chomsky-Systeme für partielle Ordnungen
Arbeitsber. Inst. Math.Masch. Datenverarb. Univ. Erlangen, vo.III,No.3, 197o

[15] H.J. Schneider, A necessary and sufficient condition for Chomsky-production over partially ordered symbol sets
Gesellsch. f. Informatik, 2. Jahrestagung (1972), p. 9o-98. Lect. Notes Eco Math. Syst., vo. 78, Springer Vlg., Berlin 1973

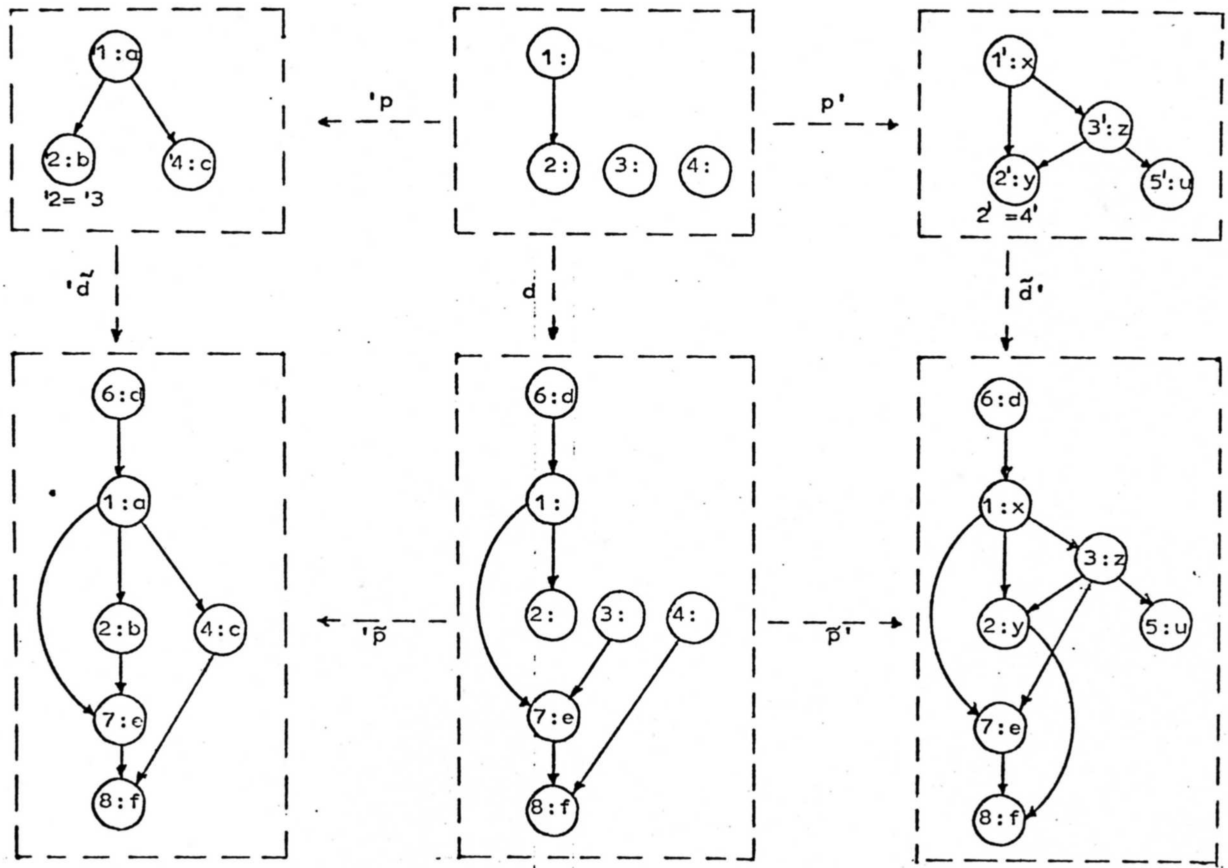


fig. 11: Direct derivation of labeled graphs

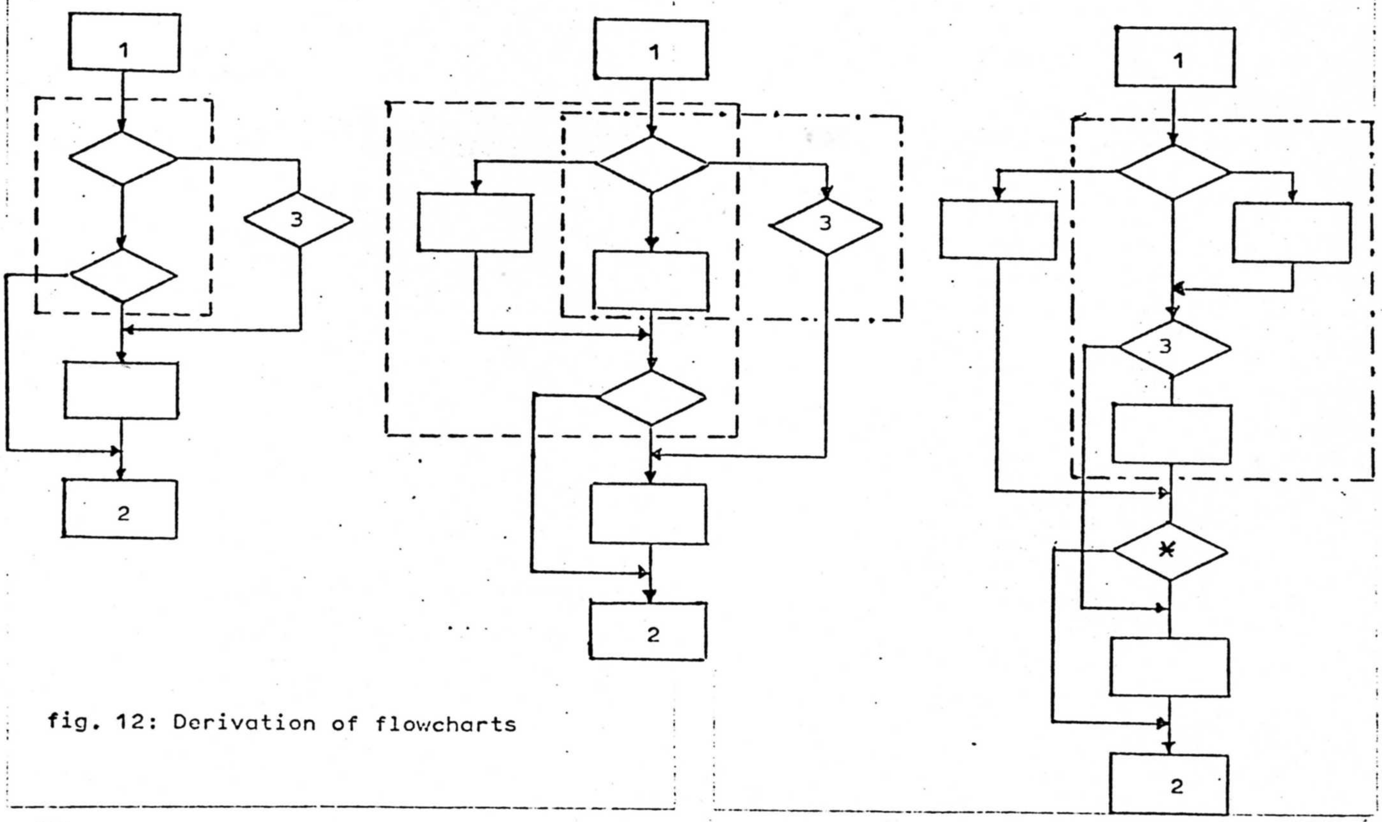


fig. 12: Derivation of flowcharts