数理解析研究所講究録
第 754 巻 1991 年 274-283

On Confluent PCE grammars

Kunio Aizawa[1]
會澤　邦夫
and
Akira Nakamura[2]
中村　　昭

[1]Department of Applied Mathematics, Hiroshima University, Higashi-Hiroshima, 724 Japan
[2]Department of Information Science, Meiji University, Kawasaki, Kanagawa, 214 Japan

Abstract: In this paper, a path controlled embedding graph grammar (PCE graph grammar) having the confluent property is proposed. Then the relationships between confluent PCE grammars and array languages are investigated.

1. Introduction

In the recent years, many models for context-free graph grammars have been proposed (see, e.g., Ehrig, et al. [2]). Some grammars are node rewriting and others are edge (or hyper-edge) rewriting. These grammars are context-free in the sense that one node is replaced without considering any other part of the rewrited graph. However, they may still be context-sensitive in the sense that generated graph depends on the order in which the production rules are applied. A graph grammar that does not suffer from this context-sensitivity is said to be confluent (see, e.g., Engelfriet [3]).

In general, graph grammars are less powerful to describe structures in contrast of their generative power. In Aizawa and Nakamura [1], we introduced a graph grammar called node-replacement graph grammar with path controlled embedding (nPCE grammars) which use a sequence of edges instead of the single edge to embedding a newly replaced graph into the host graph. It has been shown that there exists a subclass of nPCE grammars generating context-free array languages. In this paper, a path controlled embedding graph grammar (PCE graph grammar) having the confluent property is proposed. Then the relationships between confluent PCE grammars and array languages are investigated.

We assume for readers to be familiar with the theories of two-dimensional grammars and graph grammars (see e.g., Nagl [5] and Rosenfeld [6]). The following notations will be used in the rest of this paper.

(1) Let X be set. By 2^X we denote the set of subsets of X and if X is finite, then #X denotes

the cardinality of X.

(2) Let $\pi=\{c_1c_2...c_i\}$ be a string. By π^R we denote the reverse string of π, i.e., $\pi^R=\{c_i...c_2c_1\}$, and $|\pi|$ denotes the length of π, i.e., i.

(3) A *graph* is a system $H = (V, E, \Sigma_V, \Sigma_E, \phi_V, \phi_E)$, where V is a finite nonempty set of natural numbers called the *set of nodes*, E is a set of pairs of two elements from V called the *set of edges*, Σ_V is a finite nonempty set called the *set of node labels*, Σ_E is a finite nonempty set called the *set of edge labels*, ϕ_V is a mapping from V into Σ_V called the *nodes labelling function*, and ϕ_E is a mapping from E into Σ_E called the *edges labelling function*. H is called a graph over (Σ_V, Σ_E). Throughout of this paper, V(H) and E(H) denote the set of nodes and the set of edges of H, respectively.

(4) Let $H = (V, E, \Sigma_V, \Sigma_E, \phi_V, \phi_E)$ be a graph and let (x,y) is an edge of H. We say that the edge (x,y) is *incident with* the nodes x and y, and the nodes x, y are *neighbors*.

(5) Let $H = (V, E, \Sigma_V, \Sigma_E, \phi_V, \phi_E)$ be a graph and let x be a node of H. Then the *degree of x*, denoted as deg(x), is the number of edges incident with x.

(6) Let $A = (V, E, \Sigma_V, \Sigma_E, \phi_V, \phi_E)$ and $B = (V', E', \Sigma_V', \Sigma_E', \phi_V', \phi_E')$ be graphs. A is a *subgraph of B* if $V'\supseteq V$, $E'\cap\{(x, y)|\ x, y\in V\}\supseteq E$, $\Sigma_V'\supseteq\Sigma_V$, $\Sigma_E'\supseteq\Sigma_E$, $\phi_V'(x)= \phi_V(x)$ for $\forall x\in V$, and $\phi_E'((x, y))= \phi_E((x, y))$ for $\forall(x, y)\in E$. In this case, we call A the subgraph *spanned by V in B*. By B-A we denote the subgraph spanned by $V'\backslash V$ in B.

(7) Let $A = (V, E, \Sigma_V, \Sigma_E, \phi_V, \phi_E)$ and $B = (V', E', \Sigma_V, \Sigma_E, \phi_V', \phi_E')$ over (Σ_V, Σ_E). An *isomorphism from A into B* is a bijective mapping h from V into V' such that $\phi_V'\ h= \phi_V$ and $E'=\{(h(x),h(y))\ |\ (x, y)\in E\}$. We say that *A is isomorphic to B*.

(8) A graph $A = (V, E, \Sigma_V, \Sigma_E, \phi_V, \phi_E)$ is connected if for every x, y in V, there exists a sequence $x_1, x_2, \dots, x_n$ of nodes in V such that $x_1=x$, $x_n=y$ and for $1\le i\le n-1$, x_i is a neighbor of x_{i+1}.

2. Path controlled embedding

In this section, we define a kind of graph grammars called U-nPCE graph grammars and languages. They are restricted version of nPCE graph grammars proposed in Aizawa and Nakamura [1].

Definition 2.1. For any given graph H, its node P, and a string $\pi=\{c_1c_2...c_i\}$ of its edge labels, $P\pi$ is *realizable on H* if and only if there exists a set of nodes $\{P_0, P_1, \dots, P_i\}$ such that $P_0=P$ and P_j is a neighbor of P_{j-1} joined by an edge labelled with c_j $(1\le j\le i)$.

Definition 2.2. *A node-replacement graph grammar with path controlled embedding,* denoted as nPCE grammar, is a construction

$G = < \Sigma_N, \Sigma_E, P, Z, \Delta_N, \Delta_E >$, where

Σ_N is a finite nonempty set of node labels,

Σ_E is a finite nonempty set of edge labels,

Δ_N is a finite nonempty subset of Σ_N, called terminal node labels,

Δ_E is a finite nonempty subset of Σ_E, called terminal edge labels,

P is a finite set of productions of form (a, β, ψ), where a is a node, $\beta = (V, E, \Sigma_V, \Sigma_E, \phi_V, \phi_E)$ is a connected graph, ψ is a mapping from $\Sigma_E{}^+$ into $V(\beta)\times\Sigma_E$. ψ is called *embedding function.*

Z is a connected graph over (Σ_N, Σ_E) called *the axiom.*

A direct derivation step in a nPCE grammar is performed as follows:

Let $H = (V, E, \Sigma_V, \Sigma_E, \phi_V, \phi_E)$ be a graph. Let p = (a, β) be a production in P and ψ be the embedding function. Let β' be isomorphic to β (with h an isomorphism from β' into β), where β' and H–a have no common nodes. Then the result of the application of p to H (by using h) is obtained by first removing a from H, then replacing a with β' and finally adding edges (u,v) between every nodes u in β' and every v in H-a such that there exists a path $p \in$ domain(ψ) with $v = a\pi$ in H. The production p is applicable to the graph H if following conditions hold.

(1) If a has at least one neighbor in H, there exist at least one realizable path π, and
(2) if a node v of H-a is adjacent to the node labelled with a in H, v must be adjacent to the node 1 of β'.

Note here that the embedding function ψ bring no significant context into the generation procedure of an nPCE grammar since no node labels are referred in any place of embedding steps except the one of the newly replaced graph.

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

Definition 2.3. Let $G = < \Sigma_N, \Sigma_E, P, Z, \Delta_N, \Delta_E >$ be a nPCE grammar and let H, H' be graphs over (Σ_V, Σ_E).

(1) H *directly derives* H' in G, denoted as $H \Rightarrow_G H'$, if there exists a production p = (a, β, ψ) in P, a graph β' with $V(\beta') \cap V(H\text{-}a) = \varnothing$ and an isomorphism h from β' into β such that H' is isomorphic to the graph X constructed as follows:

$X = (V, E, \Sigma_V, \Sigma_E, \phi_V, \phi_E)$, where

$V = V(H\text{-}a) \cup V(\beta')$,

$E = \{ (x,y) \mid x,y \in V(H\text{-}a) \text{ and } (x,y) \in E(H) \}$
$\quad \cup \{ (x,y) \mid x,y \in V(\beta') \text{ and } (h(x),h(y)) \in E(\beta)\}$

$\cup$ { (x,y) | x $\in$ V(β'), y $\in$ V(H-a), there exists a path $\pi \in$ domain(ψ) with y = aπ and <h(x), e> $\in$ $\psi(\pi)$ for some e in Σ_E },

ϕ_V is equal to the node labelling function of H for nodes in V(H-a), equal to the node labelling function of β' for nodes in V(β'),

ϕ_E is equal to the edge labelling function of H for edges between the nodes of H, is equal to the edge labelling function of β' for nodes in V(β'),

We also say that H' is derived from H by replacing a using the production p.

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

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

We here present an example of the derivations of nPCE grammars.

Example 2.1. The application of the production in Fig. 1a to the graph in Fig. 1b results the graph in Fig. 1c.

By making use of the path controlled embedding mechanism defined above, it is possible to construct nPCE grammars generating array languages under some way to regard a graph as a two-dimensional array . In Aizawa and Nakamura [1], it is shown that, for any given context-free array grammar G, there exists an nPCE grammar G' such that L(G') is regarded as a set of two-dimensional array, L(G). However, describing the nPCE grammars generating array languages needs somewhat clumsy production rules especially in path descriptions. So, nPCE grammars using partial path group to describe paths in the embedding functions seems to be more suitable for describing patters having some geometrical structures. We review here the definitions of the partial path groups. For more precise definitions of the partial path groups, see Rosenfeld [7].

Definition 2.4. Let H be a connected graph of degree d, i.e., no more than d edges emanate from any node. By an *edge coloring* of H we mean an assignment of colors to the edges of H such that the edges emanating from any given node all have different colors.

Definition 2.5. Let p be a node of a graph H and let π be a string of colors. Then pπ is defined as the terminal node of the path defined by π starting from p, provided this path is realizable. For convenience, let us define a fictitious "blank" color representing "no move". It is obvious that the structure defined above resembles a group structure called "partial group". From now on we shall refer to this partial group as the *partial path group* of H defined by given coloring, and denote it by Π(H).

Note that the partial path groups can be defined on a graph generated by a graph grammar by regarding the edge labels of the generated graph to the colors mentioned in above definitions.

Definition 2.6. Let $\alpha = c_{i_1} c_{i_2} \ldots c_{i_k}$ be a string of colors. The string α' is called an *elementary reduction* of α if one of following statements is true:

a) $k>1$; for some $1 \le j \le k$ we have $c_{i_j} = c_0$; and $\alpha' = c_{i_1} \ldots c_{i_{j-1}} c_{i_{j+1}} \ldots c_{i_k}$.

b) $k>2$; for some $1 \le j < k$ we have $c_{i_j} = c_{i_{j+1}}$; and $\alpha' = c_{i_1} \ldots c_{i_{j-1}} c_{i_{j+2}} \ldots c_{i_k}$.

c) $k=2$; $c_{i_1} = c_{i_2}$; and $\alpha' = c_0$.

The string α' is called a *reduction* of α if there exists strings $\alpha=\alpha_0, \alpha_1, \ldots, \alpha_m=\alpha'$ such that α_i is an elementary reduction of α_{i-1}, $1 \le i \le m$. α is called *fully reducible* if $\alpha=c_0$, or if c_0 is a reduction of α. $\Pi(G)$ is called *free* if $P\alpha=P$ implies that α is fully reducible.

Definition 2.7. A partial path group $\Pi(H)$ is called *near-abelian* if each color except blank color commutes with all but one of the other colors, and does not commute with the remaining one.

It is shown in Rosenfeld [7] that the free near-abelian partial path groups with four colors correspond to a subgraph of a two-dimensional array.

We define nPCE grammars using embedding functions with near-abelian partial path groups to describe paths.

Definition 2.8. A *nPCE grammar with 4 colors free near-abelian partial path groups*, denoted as $\text{nPCE}_{\Pi 4}$ grammar, is a construction

$G = <\Sigma_N, \Sigma_E, P, Z, \Delta_N, \Delta_E>$, where $\Sigma_N, \Sigma_E, Z, \Delta_N, \Delta_E$ are same as in the definition of nPCE grammars provided that $\Sigma_E=\Delta_E$ and $|\Delta_E|=4$. P is a finite set of production rules of form $(a, \beta, \psi_{\Pi 4})$, where $\psi_{\Pi 4}$ is a mapping from Σ_E^+ into $(V(\beta)\times\Sigma_E)$ provided that if $\psi_{\Pi 4}$ maps π into (i, c) for some $c \in \Delta_E$, then c is the reduction of $\sigma\pi$, where σ is the path between node 1 to i in β; $\psi_{\Pi 4}$ is called embedding function with 4 colors near-abelian partial path groups. For a path π, $\psi_{\Pi 4}(\pi)$ implies the shortest realizable path equivalent to π under near-abelian partial path groups instead of π itself.

In the $\text{nPCE}_{\Pi 4}$ grammars, we use $\psi_{\Pi 4}$ instead of the embedding function ψ of nPCE grammars. In this case, arbitrarily long paths can be used to embed the newly replaced graphs. Since the derivation steps of our $\text{nPCE}_{\Pi 4}$ grammars are proceeded in the context-free node-replacement style, there are no insurance that the degree of each node is always at most 4 and the edges emanating from any given node have always different colors. In such cases, even if α is fully reducible, $P\alpha$ may not be P. However, the embedding mechanism of nPCE grammars still works correctly for such cases as far as using the shortest realizable paths.

One of the advantages to introduce the partial path groups into the path controlled embedding mechanism is flexibility of describing various structures. As mentioned above, two-dimensional square arrays correspond to free near-abelian groups with 4 colors, and other types of arrays (triangular, hexagonal) correspond to other types of abelian groups. The partial path groups allow us to treat other classes of graphs (such as trees and hypercubes etc.) in addition to arrays, and they have very simple characterizations. For detail discussions of those structures, see Rosenfeld [7]. The partial path groups for more complicated structures are discussed in Melter [4].

Definition 2.9. An $\mathrm{nPCE}_{\Pi 4}$ grammar is called having *uniform embedding function*, denoted as U-$\mathrm{nPCE}_{\Pi 4}$ grammar, if the following conditions for embedding functions hold.

(1) For each pair of production rules $p_i=(a_i, \beta_i, \psi_{\Pi 4_i})$, $p_j=(a_j, \beta_j, \psi_{\Pi 4_j})$ and a path $\pi \in \Sigma_E{}^+$, if both of $\psi_{\Pi 4_i}(\pi)$ and $\psi_{\Pi 4_j}(\pi)$ are defined, then $\psi_{\Pi 4_i}(\pi)=\psi_{\Pi 4_j}(\pi)$.

(2) For each production rule $p=(a, \beta, \psi_{\Pi 4})$, if $\psi_{\Pi 4}(\pi)=(v, c)$ is defined for some $\pi \in \Sigma_E{}^+$ and $v=1$, then $\psi_{\Pi 4}(\pi^R)$ is also defined and $\psi_{\Pi 4}(\pi)=\psi_{\Pi 4}(\pi^R)$.

(3) For each production rule $p=(a, \beta, \psi_{\Pi 4})$, if $\psi_{\Pi 4}(\pi)=(v, c)$ and $v\neq 1$, then there exist ρ and σ such that $\pi=\rho\sigma^R$, $\psi_{\Pi 4}(\rho)=\psi_{\Pi 4}(\rho^R)=(1, c)$ and $\psi_{\Pi 4}(\rho^R\sigma)=\psi_{\Pi 4}(\pi)$ are defined, and σ is a shortest realizable path between node 1 and v in β.

So we can describe a U-$\mathrm{nPCE}_{\Pi 4}$ grammar G as a construction G = $<\Sigma_N, \Sigma_E, P, \psi_{\Pi 4}, Z, \Delta_N, \Delta_E>$, where $\psi_{\Pi 4}=\bigcup_{p_i \in P} \psi_{\Pi 4_i}$ and each production rule has no exclusive embedding function.

3. Confluent derivations on PCE embedding

As defined in the last section, nPCE grammars are context-free in the sense that one node is replaced without considering any other part of the rewrited graph. However, the grammar may still be context-sensitive in the sense that generated graph depends on the order in which the production rules are applied. A graph grammar is said to be *confluent*, if derivation steps on distinct vertices can be done in any order.

Definition 3.1. Let G be an nPCE grammar. G is *confluent*, denoted as C-nPCE grammar, if the following condition holds for every sentential form H of G: Let v_1 and v_2 be distinct nodes of H labelled with nonterminal labels, and let p_1 and p_2 be production rules applicable to v_1 and v_2, respectively. If H_{12} is derived from H by applying p_1 at first then p_2 and H_{21} is derived by applying p_2 at first then p_1, then $H_{12}=H_{21}$.

The class of all graph languages generated by U-nPCE grammars are included in the class of all graph languages generated by C-nPCE grammars.

Theorem 3.1. For any given $nPCE_{\Pi 4}$ grammar G, G is confluent if G has uniform embedding function.

Proof: Let $G=<\Sigma_N, \Sigma_E, P, \psi_{\Pi 4}, Z, \Delta_N, \Delta_E>$ be U-$nPCE_{\Pi 4}$ grammar. For any given sentential form H of G, let v_1 and v_2 be distinct nodes of H labelled with nonterminal labels, and let $p_1=(a_{v_1}, \beta_1)$ and $p_2=(a_{v_2}, \beta_2)$ be production rules applicable to v_1 and v_2, respectively. If v_1 and v_2 are not neighbors of H and any pair of nodes from β_1 and β_2 do not become neighbors by applying both rules, then obviously $H_{12}=H_{21}$. If v_1 and v_2 are neighbors of H and the node 1s of β_1 and β_2 are not neighbors, then at least one of the production rules is not applicable. This is a contradiction. Assume here that v_1 and v_2 are not neighbors in H and become neighbors after applying p_1 and p_2. There exist four possible cases.

Case 1: Node 1s of β_1 and β_2 become neighbors.
From the definition of uniform embedding, there exists a path π from v_1 to v_2 and $\psi_{\Pi 4}(\pi)=\psi_{\Pi 4}(\pi^R)=(1, c)$. Thus obviously $H_{12}=H_{21}$.

Case 2: Node $i\neq 1$ of β_1 and node 1 of β_2 become neighbors.
From the definition of uniform embedding, there exists a path π from v_1 to v_2 such that $\psi_{\Pi 4}(\pi)=(i, c)$ and $\pi=\rho\sigma$ where σ is the path between node 1 to i in β_1 and c is the reduction of ρ. $\psi_{\Pi 4}(\rho^R)=(1, c)$ is also defined. Since c is an element of Δ_E, c is also the reduction of ρ^R. Then $H_{12}=H_{21}$.

Case 3: Node 1 of β_1 and node $i\neq 1$ of β_2 become neighbors.
Same as in the Case 2.

Case 4: Node $i\neq 1$ of β_1 and node $j\neq 1$ of β_2 become neighbors.
In this case, the path σ_i from node 1 to i in β_1 is equal to the path σ_j from 1 to j in β_2 under $\Pi 4$ since c of $\psi_{\Pi 4}(\pi\sigma_j)=(i, c)$ is the reduction of π, and also the reduction of $\sigma_i^R\pi\sigma_j$. Then from the fact $\psi_{\Pi 4}(\pi\sigma_j)=\psi_{\Pi 4}(\pi^R\sigma_j^R)$, $H_{12}=H_{21}$.

The case in which v_1 and v_2 are neighbors in H and are also neighbors after applying p_1 and p_2 is proved in the almost same way as in the above case.

4. Array languages defined by U-nPCE grammars

In this section, we introduce two mappings which map graph languages generated by U-nPCE$_{\Pi 4}$ grammar into the set of array languages. Then, we investigate the generative powers of U-nPCE$_{\Pi 4}$ grammars as the array patterns generators.

Definition 4.1. A mapping k from a graph generated by U-nPCE$_{\Pi 4}$ grammars into two-dimensional arrays is such that
(1) The horizontal neighborhood on array is defined as the set of edges labelled with two colors, say h and h', which do not commute each other.
(2) The vertical neighborhood on array is defined as the set of edges labelled with remaining two colors, say v and v', which also do not commute each other.
(3) The label of each node is the symbol in the corresponding position of array.
(4) For a graph H which has tmore than one node mapped to a position, k(H) is undefined.

The mapping k can be extended to the set of graphs in two different ways.

Definition 4.2. For any given graph language L(G) generated by a U-nPCE$_{\Pi 4}$ grammar,

(1) $K(L(G)) = \begin{cases} K=\{k(g) \mid g \in L(G)\} & \text{if } k(g) \text{ is defined for all elements of } L(G) \\ \text{undefined} & \text{otherwise} \end{cases}$

(2) $K'(L(G))=\{k(g) \mid g \in L(G) \text{ and } k(g) \text{ is defined}\}$.

Both K and K' can be extended to the families of languages in the natural way, i.e., $K(X)=\{K(L(G)) \mid G \text{ is a grammar in } X\}$ and $K'=\{K'(L(G)) \mid G \text{ is a grammar in } X\}$.

From the definitions of nPCE$_{\Pi 4}$ grammars and uniform embedding, it is not so difficult to see the following lemma holds:

Lemma 4.1. $\mathcal{F}$(K'(U-nPCE$_{\Pi 4}$)) = $\mathcal{F}$(CFAG).

The same result is obtained in Aizawa and Nakamura [1] but the concept of uniform embedding is not used in it. If all production rules of a U-nPCE$_{\Pi 4}$ grammar is restricted to be *strongly linear*, i.e., the right hand side of each rule has at most one nonterminals and there exists a single-stroke path covering the whole of the right hand side (see Yamamoto, et al. [8] for more detail definition), denoted as U-nPCE$_{\Pi 4}$-SLAG grammar, then following corollary is obtained:

Corollary 4.1. $\mathcal{F}$(K'(U-nPCE$_{\Pi 4}$-SLAG)) = $\mathcal{F}$(RAG).

Once we use the mapping K instead of K', the situation is entirely changed. In fact, $\mathcal{F}$(K(U-nPCE$_{\Pi 4}$) is no longer equal to $\mathcal{F}$(CFAG).

Lemma 4.2. $\mathcal{F}$(K(U-nPCE$_{\Pi 4}$)) $\subset$ $\mathcal{F}$(CFAG).
$\mathcal{F}$(K(U-nPCE$_{\Pi 4}$)) is incomparable with $\mathcal{F}$(RAG).

Proof: It is easy to see that $\mathcal{F}$(K(U-nPCE$_{\Pi 4}$) $\subseteq$ $\mathcal{F}$(CFAG). Thus, to prove $\mathcal{F}$(RAG) is not included in $\mathcal{F}$(K(U-nPCE$_{\Pi 4}$)), assume that there exists a U-nPCE$_{\Pi 4}$ grammar G such that K(G) is the set R of all rectangles. As shown in Yamamoto, et al. [8], R is in $\mathcal{F}$(RAG). If such grammar G exist, at least one production rule P is applied to a nonterminal node which is generated from a nonteminal node of the right hand side of P itself. Since otherwise arbitrary large pattern cannot be generated by application of rules whose right hand sides have constant size. If so, we can remove the array pattern generated from the first application of P and then connect the array pattern generated from the second application of P without any shearing effects since K is defined for G. Such a removed array pattern is finitely large unless there exists another repeated rule P'. So G is not the set of all rectangles. If such P' exists, then repeated application process of these rules proceed independently. Again G is not the set of all rectangles. This is a contradiction.

The results of this section are summarized in the following theorem:

Theorem 4.1. The diagram in Fig. 2 holds.

References

[1] Aizawa, K. and A. Nakamura: Graph grammars with path controlled embedding, *Theoretical Computer Science* (in printing), 1991.

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

[3] Engelfriet, J.: Context-free NCE graph grammars, *Lecture Notes in Computer Science*, 380, pp. 148-161, 1989.

[4] Melter, R. A.: Tessellation graph characterization using rosettas, *Pattern Recognition Letters*, 4, pp. 79-85, 1986.

[5] Nagl, M.: A tutorial and bibliographical survey on graph grammars, *Lecture Notes in Computer Science*, 73, pp. 70-126, 1979.

[6] Rosenfeld, A.: *Picture Languages*, Academic Press, New York, 1979.

[7] Rosenfeld, A.: Partial path groups and parallel group constructions, in G. Rozenberg and A. Salomaa (ed.), *The Book of L*, Springer-Verlag, Berlin, pp. 369-382, 1986.

[8] Yamamoto, Y., K. Morita and K. Sugata: Context-sensitivity of two-dimensional regular array grammars, *International Journal of Pattern Recognition and Artificial Intelligence*, 3, pp. 295-319, 1989.

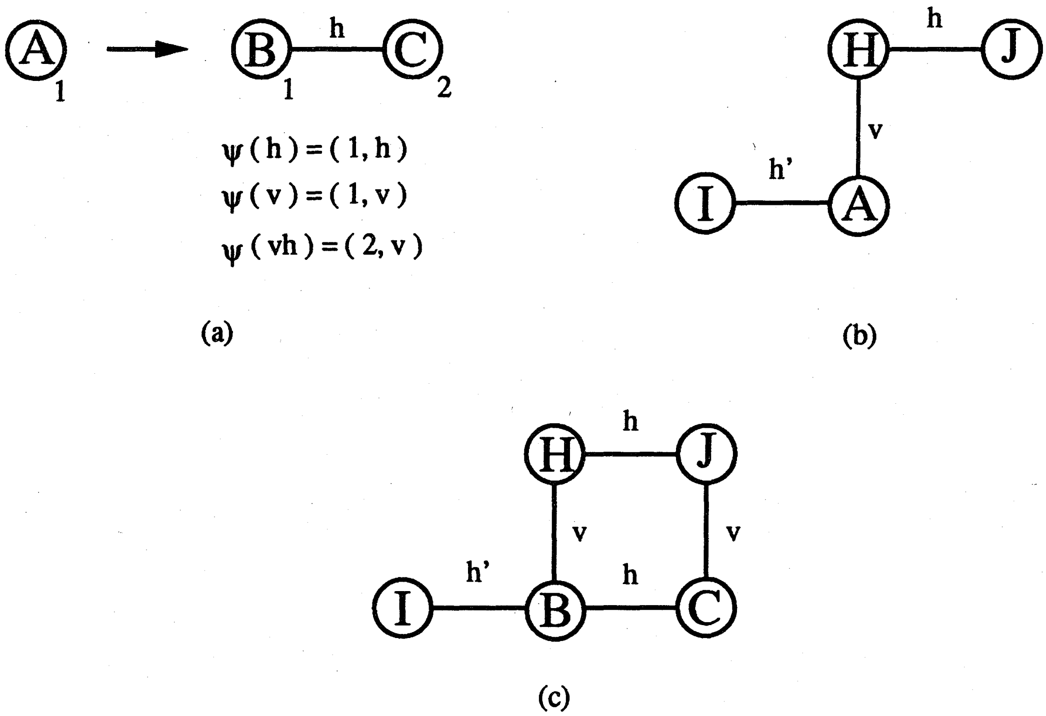

Fig. 1. An example of derivations of nPCE grammar.

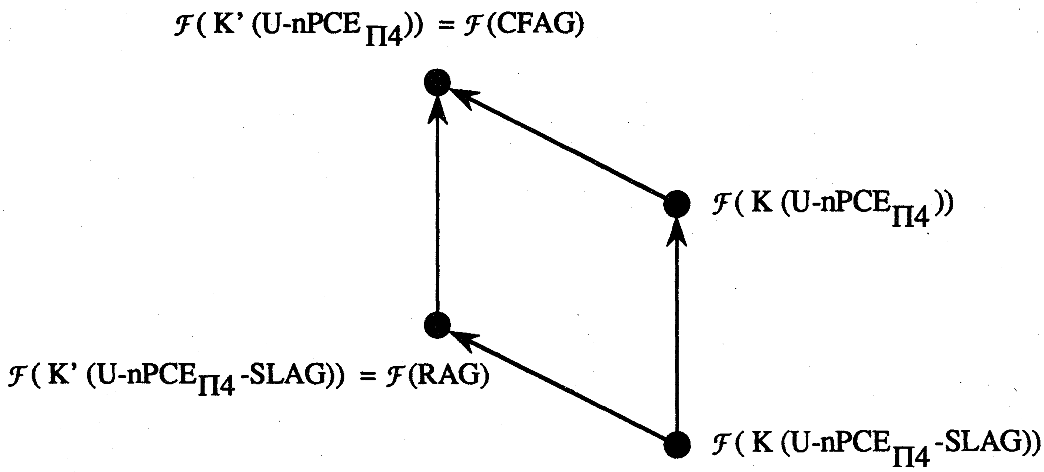

Fig. 2. Hierarchical results.