Fundamental Study

On multiple context-free grammars*

Hiroyuki Seki
Department of Information and Computer Sciences, Faculty of Engineering Science, Osaka University, Toyonaka, Osaka 560, Japan

Takashi Matsumura
Nomura Research Institute, Ltd., Tyu-o-ku, Tokyo 104, Japan

Mamoru Fujii
College of General Education, Osaka University, Toyonaka, Osaka, 560, Japan

Tadao Kasami
Department of Information and Computer Sciences, Faculty of Engineering Science, Osaka University, Toyonaka, Osaka 560, Japan

Communicated by M. Takahashi
Received July 1989
Revised January 1990

Abstract

Seki, H., T. Matsumura, M. Fujii and T. Kasami, On multiple context-free grammars, Theoretical Computer Science 88 (1991) 191–229.

Multiple context-free grammars (*mcfg's*) is a subclass of generalized context-free grammars introduced by Pollard (1984) in order to describe the syntax of natural languages. The class of languages generated by mcfg's (called *multiple context-free languages* or, shortly, *mcfl's*) properly includes the class of context-free languages and is properly included in the class of context-sensitive languages. First, the paper presents results on the generative capacity of mcfg's and also on the properties of mcfl's such as formal language-theoretic closure properties. Next, it is shown that the time complexity of the membership problem for multiple context-free languages is $O(n^e)$, where n is the length of an input string and e is a constant called the degree of a given mcfg. *Head grammars* (*hg's*) introduced by Pollard and *tree adjoining grammars* (*tag's*) introduced by Joshi et al. (1975) are also grammatical formalisms to describe the syntax of natural languages. The paper also presents the following results on the generative capacities of hg's, tag's and 2-mcfg's, which are a subclass of mcfg's: (1) The class HL of languages generated by hg's is the same as the one generated by tag's; (2)

*This paper is partially based on the authors' previous reports [6, 11].

HL is the same as the one generated by left-wrapping hg's (or right-wrapping hg's) which is a proper subclass of hg's; (3) HL is properly included in the one generated by 2-mcfg's. As a corollary of (1), it is also shown that HL is a substitution-closed full AFL.

Contents

1. Introduction

Literature on generative grammars shows often a mention of inadequacy of context-free grammars (cfg's) for describing the structures involving discontinuous constituents in natural languages [13]. Context-sensitive grammars (csg's or Type 1 grammars), on the other hand, may not be an adequate model of grammars of natural languages because they are too powerful in generative capacity, and phrase structures which are "natural" extension of phrase structures in cfg's are not defined in Type 0 and Type 1 grammars. Although various types of formal grammars between cfg's and csg's were proposed and investigated in 1960s and early 1970s [2, 15], the main interest was the process of sentence derivation and generation, rather than the development of grammars suitable for defining phrase structures and formalizing their syntax analysis.

Generalized context-free grammars (*gcfg's*) introduced by Pollard [12] are an interesting formalization for defining phrase structures. However, since the generative capacity of gcfg's is readily shown to be the same as that of Type 0 grammars [6, 7], gcfg's themselves are also too powerful.

Multiple context-free grammars (*mcfg's*) were introduced as a subclass of gcfg's in [6]. Mcfg's deal with tuples of strings, and a rewriting rule of mcfg's has the following form:

$$A_0 \rightarrow f[A_1, A_2, \ldots, A_q],$$

where f is a function whose arguments and function values are tuples of strings and satisfies the following conditions:

(1) Each component of the value of f is a concatenation of some constant strings and some components of its arguments.

(2) Each component is not allowed to appear in the value of f more than once.

Vijay-Shanker et al. [23] introduced linear context-free rewriting systems (lcfrs'). Lcfrs' are essentially the same grammar formalism as mcfg's except that lcfrs' are required to satisfy the information-lossless condition (see condition (f3) of Lemma 2.2; this conditon is called nonerasing condition in [23]), while mcfg's need not satisfy this condition. However, it is shown that this condition does not weaken the generative capacity of mcfg's (see Lemma 2.2).

The class of languages generated by mcfg's, called *multiple context-free languages* (*mcfl's*), properly includes the class of context-free languages (cfl's) and is properly included in the class of context-sensitive languages. In mcfg's, it is possible to account for structures involving discontinuous constituents such as "respectively" sentences in a simple manner, and such concepts as phrase structures and derivation trees in cfg's can be extended naturally in mcfg's. Furthermore, the class of mcfl's enjoys the formal language-theoretic closure properties that the class of cfl's does. For example, the class of mcfl's is closed under union, concatenation, Kleene closure, ε-free Kleene closure, substitution and intersection with regular languages (see Theorem 3.9). Hence, the class of mcfl's is a substitution-closed full AFL [2]. Moreover, the time complexity of the membership problem for mcfl's was shown to be polynomial [6, 23].

If an mcfg G deals with only i-tuples of strings for $1 \leqslant i \leqslant m$, then G is called an m-mcfg. Let m-$MCFL$ denote the class of languages generated by m-mcfg's. Then the following inclusion relations hold (Theorem 3.4):

$$CFL = 1\text{-}MCFL \quad \text{and} \quad m\text{-}MCFL \subsetneq (m+1)\text{-}MCFL \quad \text{for} \quad m \geqslant 1.$$

In Section 3, we present results on the generative capacity of mcfg's, which include a pumping lemma for mcfl's and also on properties of mcfl's such as formal language-theoretic closure properties. Next, it is shown that the time complexity of the membership problem for mcfl's is $O(n^e)$, where n is the length of an input string and e is a constant called the degree of a given mcfg.

As other grammar formalisms to describe the syntax of natural languages, *tree adjoining grammars* and *head grammars* were developed. Tree adjoining grammars (*tag's*), introduced by Joshi et al. [5], deal with elementary trees which are composed by means of an operation called adjoining (see also [3, 20]). Head grammars (*hg's*), introduced by Pollard [12], deal with headed strings by means of head-wrapping operations besides concatenation operations. Vijay-Shanker et al. [22] use pairs of strings (α_1, α_2) (called split strings) instead of headed strings and introduce *modified head grammars* (*mhg's*) which deal with split strings. It was shown in [22] that the generative capacities of mhg's and tag's are equivalent and that the generative capacity of mhg's is not weaker than that of hg's. Moreover, Vijay-Shanker [20] showed that the generative capacity of tag's is equivalent to that of linear indexed grammars (lig's) introduced by Gazdar [1] as a subclass of indexed grammars [2]. Weir et al. [25] showed that the generative capacity of tag's is also equivalent to that of combinatory categorial grammars (ccg's) introduced by Steedman [17, 18] as an extension of categorial grammars. Let *CFL*, *HL*, *TAL*, *LIL*, *CCL* and *m-MCFL* ($m \geqslant 1$) denote the class of languages generated by cfg's, hg's, tag's, lig's, ccg's and m-mcfg's, respectively. Summarizing, it has already been known that the following inclusion relations hold between these classes of languages:

$$CFL = 1\text{-}MCFL \subsetneq HL \subseteq TAL = LIL = CCL \subseteq 2\text{-}MCFL.$$

Furthermore, $TAL\ (= LIL = CCL)$ and *m-MCFL* are shown to be substitution-closed full AFL's in [20] and [6], respectively (see Theorem 3.9 in this paper for the latter). Vijay-Shanker et al. [22] conjecture that $HL = TAL$, Weir [24] conjectures that $HL \subsetneq 2\text{-}MCFL$ and Roach [14] conjectures that *HL* is closed under substitution. However, these conjectures were not proved (it was shown that *HL* is closed under ε-free substitution in [9]).

In Section 4, we give affirmative answers to all of these open problems. It is also shown, as a corollary, that the generative capacity of hg's is not weakened even if the head-wrapping operations are restricted only to left-wrapping operations or only to right-wrapping operations (see Section 4.1).

2. Definitions

2.1. Generalized context-free grammars

A *generalized context-free grammar* (*gcfg*) [12] is a 5-tuple $G = (N, O, F, P, S)$ (we have slightly modified the definition of a gcfg in order to make it easier to compare it with a cfg, see [12, Appendix 2]), where

(G1) N is a finite set of *nonterminal symbols*;

(G2) O is a set of n-tuples of strings ($n \geqslant 1$) over a finite set of symbols;

(G3) F is a finite set of partial functions from finite dimensional direct products $O \times O \times \cdots \times O$ to O. Let us define F_q to be the set of partial mappings from O^q to O which are in F;

(G4) P is a finite subset of $\bigcup_q (F_q \times N^{q+1})$;

(G5) $S \in N$ is the *initial symbol*.

An element of P is called a *rewriting rule* (or simply *rule*) and written as

$$A_0 \to f[A_1, A_2, \ldots, A_q]$$

instead of $(f, A_0, A_1, A_2, \ldots, A_q)$ (there may be more than one occurrence of some nonterminal symbols in $A_1, A_2, \ldots, A_q$). For $A_0 \to f[A_1, A_2, \ldots, A_q]$, if $q = 0$, i.e. if f is an element in O, the rule is said to be a *terminating rule*; otherwise, it is said to be a *nonterminating rule*.

For $A \in N$, let us define $L_G(A)$ as the smallest set satisfying the following two conditions:

(L1) If a terminating rule $A \rightarrow \theta$ is in P, then $\theta \in L_G(A)$;

(L2) If $\theta_i \in L_G(A_i)$ $(1 \leqslant i \leqslant q)$, $A \rightarrow f[A_1, A_2, \ldots, A_q] \in P$ and $f[\theta_1, \theta_2, \ldots, \theta_q]$ is defined, then $f[\theta_1, \theta_2, \ldots, \theta_q] \in L_G(A)$.

If $\theta \in L_G(A)$, we say that *θ parses as an A in G* or *θ is derived from A in G*; θ is called an *A-phrase* (or simply *phrase*). We let $L(G) = L_G(S)$; $L(G)$ is called the *generalized context-free language* (*gcfl*) *generated by G*. For gcfg's G_1 and G_2, we say that G_1 is weakly equivalent to G_2 if $L(G_1) = L(G_2)$.

A concept which is an extension of derivation trees of cfg's can be defined for gcfg's. It is suited for formal definition of semantics of sentences.

A *derivation tree* in a gcfg G is defined as follows:

(T1) For a terminating rule $A \rightarrow \theta$, the tree whose root (labeled with A, or the applied rule instead of A, if necessary) has only one child (labeled with θ) is a derivation tree of θ.

(T2) If T_i is a derivation tree of θ_i whose root is labeled with A_i $(1 \leqslant i \leqslant q)$, $A \rightarrow f[A_1, A_2, \ldots, A_q]$ is in P and $f[\theta_1, \theta_2, \ldots, \theta_q]$ is defined, then a tree such that (1) the root is labeled with A, (2) the root has q children, and (3) the subtree rooted at the ith child is isomorphic to T_i $(1 \leqslant i \leqslant q)$, is a derivation tree of $f[\theta_1, \theta_2, \ldots, \theta_q]$.

(T3) There is no other derivation tree.

In a cfg $G = (N, T, P, S)$, the string obtained by concatenating the labels of leaves in a derivation tree of $\alpha \in T^*$ is equal to α. This is not always true in gcfg's.

For a derivation tree t and a node v in t, if v is labeled with a rule R, then we say that R is *applied at v* in t or the *applied rule* at v in t is R. If the subtree of t rooted at v is a derivation tree of θ, then we say that θ is *derived from v* in t. Assume that θ and θ' are derived from v and v' in t, respectively, and a node v' is an ancestor of a node v in t. We say that θ is a *subphrase* of θ'.

Pollard [12] showed that gcfg's and gcfl's are generalizations of cfg's and cfl's, respectively.

In (G3) of the definition of a gcfg, if arbitrary partial recursive functions are permitted as functions of F, it can easily be shown that any recursively enumerable set of strings (any language generated by a Type 0 grammar [2]) is a gcfl. Even if functions of F are restricted to be arbitrary compositions of elementrary functions, the same conclusion can be obtained (this result is not surprising since any partial recursive function can be defined by a Turing machine) [6]. Of course, the converse is also true. That is, if O is the set of all strings over a finite alphabet and all functions in F are partial recursive in a gcfg $G = (N, O, F, P, S)$, then the language generated by G is a recursively enumerable set.

If a function in a gcfg is defined without using the information of some arguments, then "unnatural" phrases, which do not reflect their subphrases, may be introduced. To avoid introducing such "unnatural" phrase structures, the following conditions for functions in F may be necessary:

(1) For given arguments, a predicate representing whether or not the value of the function is defined is given as a composition of elementary functions.

(2) The information in arguments of functions does not get lost by any applications of the functions, i.e. the values (strings) of arguments must be reconstructible from the value (string) of the function.

In Section 2.2, we introduce a subclass of gcfg's called multiple context-free grammars which deal with tuples of strings and use only functions which are defined as concatenations of constant strings and components of arguments.

2.2. Multiple context-free grammars

An *m-parallel multiple context-free grammar* for a positive integer m, abbreviated as *m-pmcfg* or *pmcfg*, is defined to be a gcfg $G=(N, O, F, P, S)$, which satisfies the following conditions (M1) through (M4). Let T be a finite set of terminal symbols which is disjoint with N. Then

(M1) $O=\bigcup_{i=1}^{m}(T^*)^i$.

(M2) Let $a(f)$ be the number of arguments of $f\in F$. For each $f\in F$, positive integers $r(f)$ and $d_i(f)$ $(1\leqslant i\leqslant a(f))$ which are not greater than m are given, and f is a function from $(T^*)^{d_1(f)}\times(T^*)^{d_2(f)}\times\cdots\times(T^*)^{d_{a(f)}(f)}$ to $(T^*)^{r(f)}$ which satisfies the following condition (f1). Let f^h $(1\leqslant h\leqslant r(f))$ denote the hth component of f. We define

$$\bar{x}_i=(x_{i1}, x_{i2}, \ldots, x_{id_i(f)})$$

and

$$X=\{x_{ij} \mid 1\leqslant i\leqslant a(f),\quad 1\leqslant j\leqslant d_i(f)\}.$$

(f1) Functions $f^h(\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_{a(f)})$ $(1\leqslant h\leqslant r(f))$ are represented by concatenation of some constant strings in T^* and some variables in X. That is,

$$f^h[\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_{a(f)}]=\alpha_{h0}z_{h1}\alpha_{h1}z_{h2}\cdots z_{hv_h(f)}\alpha_{hv_h(f)}, \tag{2.1}$$

where $\alpha_{hk}\in T^*$ $(0\leqslant k\leqslant v_h(f))$ and $z_{hk}\in X$ $(1\leqslant k\leqslant v_h(f))$.

(M3) A positive integer $d(A)$ is given for each nonterminal symbol $A\in N$. If a rule $A\to f[A_1, A_2, \ldots, A_{a(f)}]$ is in P, then $r(f)=d(A)$ and $d_i(f)=d(A_i)$ $(1\leqslant i\leqslant a(f))$.

(M4) For the initial symbol S, $d(S)=1$.

From (M3), $\bar{\alpha}\in L_G(A)$ implies $\bar{\alpha}\in(T^*)^{d(A)}$; especially, $\alpha\in L(G)$ implies $\alpha\in T^*$ from (M4).

If all the functions in F satisfy the next condition (f2) in addition to (f1), G is called an *m-multiple context-free grammar* and is abbreviated as *m-mcfg* or simply *mcfg*.

(f2) For each h $(1\leqslant h\leqslant r(f))$ and each variable x_{ij} in X, the total number of occurrences of x_{ij} in the right-hand sides of (2.1) is at most one.

If some variable occurs in the right-hand side of (2.1) more than once or some variable occurs in the right-hand sides of (2.1) for different h's, the string substituted for the variable will be copied more than once. (f2) is the condition for inhibiting the use of such copy operations to define f.

A language generated by an m-pmcfg and an m-mcfg is called an *m-parallel multiple context-free language* and an *m-multiple context-free language*, respectively, and is abbreviated as *m-pmcfl* (or simply *pmcfl*) and *m-mcfl* (or simply *mcfl*), respectively. The classes of m-pmcfl's pmcfl's m-mcfl's, mcfl's and cfl's are denoted by *m-PMCFL*, *PMCFL*, *m-MCFL*, *MCFL* and *CFL*, respectively. It was shown that the following relation holds ([6]; the original idea is from [12]):

$$CFL = 1\text{-}MCFL. \tag{2.2}$$

Example 2.1. (1) $\{\alpha^2 \mid \alpha \in \{0,1\}^+\} \in 1\text{-}PMCFL \cap 2\text{-}MCFL$,

(2) $\{\alpha^{2^n} \mid n \geqslant 0\} \in 1\text{-}PMCFL$,

(3) $\{a_1^n a_2^n \ldots a_{2m}^n \mid n \geqslant 0\} \in m\text{-}MCFL$, $m \geqslant 1$,

(4) $\{a^{n^2} \mid n > 0\} \in 2\text{-}PMCFL$.

These languages are generated by the following pmcfg's respectively:

(1) Let $L = \{\alpha^2 \mid \alpha \in \{0,1\}^+\}$ and let us define $T = \{0,1\}$, $N = \{A, S\}$, $P = \{A \to 0 \mid 1 \mid g_1[A, A],\ S \to g_2[A]\}$, where $g_1[x, y] = xy$, $g_2[x] = xx$. Then the 1-pmcfg $G = (N, T^*, \{g_1, g_2\}, P, S)$ generates the language L. Next, let us define $P = \{A \to (0,0) \mid (1,1) \mid g_{3,0}[A] \mid g_{3,1}[A], S \to g_4[A]\}$, where $g_{3,a}[(x, y)] = (ax, ay)$, $a = 0, 1$, $g_4[(x, y)] = xy$. Then the 2-mcfg $G = (N, T^*, \{g_{3,0}, g_{3,1}, g_4\}, P, S)$ also generates L.

(2) Let $T = \{a\}$, $N = \{S\}$ and $P = \{S \to a \mid g_2[S]\}$, where g_2 is the same as (1). Then the 1-pmcfg $G = (N, T^*, \{g_2\}, P, S)$ generates $\{a^{2^n} \mid n \geqslant 0\}$.

(3) Let $T = \{a_i \mid 1 \leqslant i \leqslant 2m\}$, $N = \{A, S\}$, $P = \{A \to (\varepsilon, \varepsilon, \ldots, \varepsilon) \mid g_5[A], S \to g_6[A]\}$, where $g_5[(x_1, x_2, \ldots, x_m)] = (a_1 x_1 a_2, a_3 x_2 a_4, \ldots, a_{2m-1} x_m a_{2m})$, $g_6[(x_1, x_2, \ldots, x_m)] = x_1 x_2 \ldots x_m$. Then the m-mcfg $G = (N, \bigcup_{i=1}^m (T^*)^i, \{g_5, g_6\}, P, S)$ generates the language $\{a_1^n a_2^n \ldots a_{2m}^n \mid n \geqslant 0\}$.

(4) Let $T = \{a\}$, $N = \{A, S\}$, $P = \{A \to (\varepsilon, \varepsilon) \mid g_7[A], S \to g_8[A]\}$, where $g_7[(x_1, x_2)] = (ax_1, ax_1^2 x_2)$, $g_8[(x_1, x_2)] = ax_1^2 x_2$. Then 2-pmcfg $G = (N, T^* \cup (T^*)^2, \{g_7, g_8\}, P, S)$ generates $\{a^{n^2} \mid n > 0\}$.

None of the sets given in Example 2.1 is in *CFL* [2, 15]. Hence,

$$CFL \subsetneqq 1\text{-}PMCFL \cap 2\text{-}MCFL \tag{2.3}$$

Lemma 2.2. *For a given m-pmcfg (m-mcfg) G, we can construct an m-pmcfg (m-mcfg) $G' = (N', O', F', P', S')$ which is weakly equivalent to G and satisfies the following information-lossless condition* (f3) *and* (N1) *through* (N5):

(f3) *For any f in F′, any variable x in X appears exactly once in the right-hand side of* (2.1) *for some h* $(1 \leqslant h \leqslant r(f))$.

(N1) *For any nonterminal symbol A in N′ which appears in the left-hand side of some terminating rule, $d(A) = 1$.*

(N2) *For any terminating rule in P′, the length of the right-hand side is not greater than* 1.

(N3) *For any nonterminal symbol A in N′ except the initial symbol S′, if $(\alpha_1, \alpha_2, \ldots, \alpha_{d(A)}) \in L_G(A)$, then $\alpha_i \neq \varepsilon$ $(1 \leqslant i \leqslant d(A))$, where ε denotes the empty string.*

(N4) *If the rule $S' \to \varepsilon$ exists in P', then S' does not appear in the right-hand side of any nonterminating rule in P'.*

(N5) *For any nonterminating rule in P', each constant string α_{hk} in the right-hand side of* (2.1) *is the empty string.*

Proof. For a given m-mcfg G, we first construct an m-mcfg G' which is weakly equivalent to G and satisfies the condition (f3) as follows. Assume that x_{11}, for example, does not appear in the right-hand side of (2.1) for any h ($1 \leqslant h \leqslant r(f)$). For each rule $A \to f[A_1, A_2, \ldots]$ whose function in the right-hand side is f, we introduce a new nonterminal symbol A_1' ($d(A_1') = d(A_1) - 1$) corresponding to $A_1 \in N$, and replace the rule by $A \to f'[A_1', A_2, \ldots]$, where f' is a function obtained from f by deleting the variable x_{11} and replacing the variables $x_{12}, \ldots, x_{1d_1(f)}$ by $x_{11}, \ldots, x_{1d_1(f)-1}$, respectively. Moreover, for each rule $A_1 \to g[B_1, B_2, \ldots]$ whose left-hand side is A_1, we construct a new rule $A_1' \to g'[B_1, B_2, \ldots]$, where g' is a new function whose first component is the second component of g, the second component is the third component of g, and so on. This operation should be repeated until condition (f3) is satisfied (this procedure terminates in a finite number of repetitions because the procedure introduces only nonterminal symbols which have smaller values of $d(\cdot)$).

Next, for each nonterminal symbol A and $\Psi \subseteq \{1, 2, \ldots, d(A)\}$, we introduce a nonterminal symbol $A[\Psi]$ ($d(A[\Psi]) = d(A) - |\Psi|$). We define $A[\emptyset] = A$. We reconstruct G by executing Procedure 1 which adds new rules and deletes unnecessary ones so that $L_{G'}(A[\Psi]) = \{\bar{\alpha}[\Psi] \mid \bar{\alpha} \in L_{G'}(A)$ and "$j \in \Psi$ iff jth component of $\bar{\alpha}$ is ε"$\}$ holds. $\bar{\alpha}[\Psi]$ is a tuple of strings obtained by deleting all jth components ($j \in \Psi$) from $\bar{\alpha}$.

Procedure 1. For each nonterminal symbol A in G', let $M(A)$ be a variable in which a subset of the power set of $\{1, 2, \ldots, d(A)\}$ is stored. The initial value is the empty set $\emptyset$. Execute the following steps (1) through (5) in this order. The resulting grammar is a desired one satisfying (f3) and (N1) through (N5).

(1) First, remove all rules of the form $A \to \varepsilon$. Then for each terminating rule R: $A \to (\alpha_1, \alpha_2, \ldots, \alpha_{d(A)})$, $\alpha_i \in T^*$, $d(A) > 1$, let $\alpha_{i_1}, \alpha_{i_2}, \ldots, \alpha_{i_p}$ be the nonempty (non-ε) elements of $\alpha_1, \alpha_2, \ldots, \alpha_{d(A)}$, in this order. Let us define the set $\Psi = \{1, 2, \ldots, d(A)\} - \{i_1, i_2, \ldots, i_p\}$. Remove the rule R and introduce new nonterminal symbols $\bar{\alpha}_{i_1}, \bar{\alpha}_{i_2}, \ldots, \bar{\alpha}_{i_p}, A[\Psi]$ and the following new rules:

(a) $\bar{\alpha}_{i_q} \to \alpha_{i_q}$, $1 \leqslant q \leqslant p$,

(b) $A[\Psi] \to (\bar{\alpha}_{i_1}, \bar{\alpha}_{i_2}, \ldots, \bar{\alpha}_{i_p})$ if $\Psi \neq \{1, 2, \ldots, d(A)\}$.

Add Ψ to $M(A)$ if $\Psi \notin M(A)$.

(2) For each nonterminating rule $A \to f[A_1, A_2, \ldots, A_{a(f)}]$ except those introduced by this procedure, and each $A_i[\Psi_i]$, $\Psi_i \in M(A_i)$, $1 \leqslant i \leqslant a(f)$, let $i_1, i_2, \ldots, i_p$ be the suffixes in increasing order such that $\Psi_{i_q} \neq \{1, 2, \ldots, d(A)\}$ ($1 \leqslant q \leqslant p$). Construct the following new rules. Let f'' be the function obtained from f by substituting ε for the variables x_{ij} ($j \in \Psi_i$) in the right-hand side of (2.1) in the definition of f. Let f' be the function obtained from f'' by deleting (a) the arguments $\bar{x}_i$ except $\bar{x}_{i_1}, \bar{x}_{i_2}, \ldots, \bar{x}_{i_p}$ and

(b) the components equal to ε. Let $h_1, h_2, \ldots$ be the components deleted. If $\Psi = \{h_1, h_2, \ldots\} \notin M(A)$, then add Ψ to $M(A)$. Generate the following new rule if $\Psi \neq \{1, 2, \ldots, d(A)\}$:

$$A[\Psi] \rightarrow f'[A_{i_1}[\Psi_{i_1}], A_{i_2}[\Psi_{i_2}], \ldots, A_{i_p}[\Psi_{i_p}]].$$

(3) Delete all of the old nonterminating rules.

(4) Let S' be the new initial symbol and add the rule $S' \rightarrow ID[S]$, where $ID[x] = x$. If $\varepsilon \in L_{G'}(S)$, then also add the rule $S' \rightarrow \varepsilon$.

(5) For each rule which does not satisfy the condition (N5), add new nonterminal symbols and terminating rules which are used only for each nonempty constant string α_{hk} in the right-hand side of the rule.

(6) For each terminating rule $A \rightarrow a_1 a_2 \ldots a_h$ with $h \geqslant 2$ and $a_k \in T$ $(1 \leqslant k \leqslant h)$, delete this rule and add rules $A \rightarrow f[A_{a_1}, \ldots, A_{a_h}]$ and $A_{a_k} \rightarrow a_k$ for $1 \leqslant k \leqslant h$, where $f[x_1, x_2, \ldots, x_n] = x_1 x_2 \ldots x_n$ and A_{a_k}'s are new nonterminal symbols.

Let G be an mcfg and $\bar{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_n)$ be an A-phrase. Let t be a derivation tree of $\bar{\alpha}$ and v_1, v_2 $(v_1 \neq v_2)$ be two internal nodes of t whose labels are A_1, A_2, respectively. Suppose that $\bar{\beta} = (\beta_1, \beta_2, \ldots, \beta_{k1})$ and $\bar{\gamma} = (\gamma_1, \gamma_2, \ldots, \gamma_{k2})$ are tuples of strings in $L_G(A_1)$ and $L_G(A_2)$, respectively, whose derivation trees are the subtrees of t rooted at v_1 and v_2, respectively. Each β_j appears in $\bar{\alpha}$ at most once (and only once when G satisfies the condition (f3) of Lemma 2.1) as a substring of some component of $\bar{\alpha}$, and even if β_i and β_j $(i \neq j)$ are contained in the same component of $\bar{\alpha}$, they do not overlap with each other (see the definition of $g_{v,v'}$ in Section 3.1). If there is no ancestor–descendant relation between v_1 and v_2, β_i and γ_j do not overlap with each other even if β_i and γ_j are contained in the same component of $\bar{\alpha}$. Thus, in an mcfg, for each nonterminal symbol A, an A-phrase is composed of $d(A)$ components and they unite together in the syntax and ancestor–descendant relations between phrases exist. This property distinguishes an mcfg from a scattered context-free grammar [15], a parallel context-free grammar [16, 19] and a matrix grammar [15] (abbreviated as an scfg, a pcfg and an mg, respectively).

In an scfg, each rule has the form $(A_1 \rightarrow u_1, A_2 \rightarrow u_2, \ldots, A_q \rightarrow u_q)$, where $A_i \in N$, $u_i \in (T \cup N)^*$ $(1 \leqslant i \leqslant q)$, and N and T are the sets of nonterminal and terminal symbols, respectively. When all of $A_1, A_2, \ldots, A_q$ appear in $\alpha \in (T \cup N)^*$, the string β obtained from α by substituting $u_1, u_2, \ldots, u_q$ for $A_1, A_2, \ldots, A_q$, respectively, is called a string derived from α by the rule. In the string β, these u_i's will not behave to unite together in further derivations. Thus, in an scfg, we may not be able to define meaningfully an ancestor–descendant relation between phrases. Similar situations exist in a pcfg and an mg. Although a pcfg has some resemblance to a pmcfg in that it has copy operations, they are not equivalent because the class of languages generated by pcfg's and CFL do not include each other while CFL is properly included in $MCFL$ (see the properties (2.2) and (2.3) of mcfg's).

3. Properties of multiple context-free grammars

In this section, we will present properties of pmcfg's and mcfg's. These results except for Theorem 3.7 are first obtained in [6].

3.1. Generative capactiy and closure properties

First, some results on the generative capacity of pmcfg's and mcfg's are presented [see also (2.2) and (2.3)].

Theorem 3.1. *PMCFL is properly included in the class CSL of context-sensitive languages.*

Now, some definitions will be introduced for the proof of the next lemma. Let t be a derivation tree of an mcfg G. Let v and v' be internal nodes in t labeled with A and A', respectively, where v' is an ancestor of v or v itself. A function $g_{v,v'}$ from $(T^*)^{d(A)}$ to $(T^*)^{d(A')}$ is defined as follows. Let $\bar{y}=(y_1, y_2,\dots, y_{d(A)})$ be a variable over $(T^*)^{d(A)}$:

(1) $g_{v,v}[\bar{y}]=\bar{y}$

(2) Assume that $v\neq v'$. Let $v_1, v_2,\dots, v_w$ (labeled with $A_1, A_2,\dots, A_w$, respectively) be the children of v', and v_i $(1\leqslant i\leqslant w)$ be the child of v' on the path from v' to v in t. Let $A'\rightarrow f[A_1, A_2,\dots, A_w]$ be the rule applied at v' in t, and s_j be the string derived from v_j $(j\neq i)$ in t. Then

$$g_{v,v'}[\bar{y}]=(s_1,\dots, s_{i-1}, g_{v,v_i}[\bar{y}], s_{i+1},\dots, s_w).$$

From the definition, for any $\bar{\alpha}\in L_G(A)$,

$$g_{v,v'}[\bar{\alpha}]\in L_G(A') \tag{3.1}$$

Each component of $g_{v,v'}(\bar{y})$ can be represented by the concatenation of some variables in $\{y_1, y_2,\dots,y_{d(A)}\}$ and constant strings. If G satisfies the condition (f3), each variable y_i is contained in one and only one component of $g_{v,v'}(\bar{y})$. Let us denote the sum of string lengths of components of $g_{v,v'}(\bar{y})$ by $|g_{v,v'}(\bar{y})|$. If G satisfies the conditions (f3), (N3) and (N4) of Lemma 2.1, and on the path from v' to v in t there exists a node which is not v and has two or more children, then the following inequality holds:

$$|g_{v,v'}(\bar{y})|>d(A). \tag{3.2}$$

The following lemma analogous to the pumping lemma for cfl's [2, 15] holds for mcfl's. For a string α, let $|\alpha|$ denote the length of α.

Lemma 3.2 (pumping lemma for mcfl's). *For any m-mcfl L, if L is an infinite set then there exist some* $u_j\in T^*$ $(1\leqslant j\leqslant m+1)$, v_j, w_j, $s_j\in T^*$ $(1\leqslant j\leqslant m)$ *which satisfy the following conditions*:

(1) $\sum_{j=1}^{m} |v_js_j|>0$, *and*

(2) *for any nonnegative integer i,*

$$z_i = u_1 v_1^i w_1 s_1^i u_2 v_2^i w_2 s_2^i u_3 \ldots u_m v_m^i w_m s_m^i u_{m+1} \in L.$$

Proof. Let $G(=(N, O, F, P, S))$ be an m-mcfg which generates L and satisfies the conditions (f3) and (N1) through (N5) of Lemma 2.1. Let us denote $\max\{a(f) \mid f \in F\}$ by q. Since L contains a string of length greater than two, $q \geqslant 2$ holds from the conditions (N1) and (N5). Let us consider a derivation tree t of $z \in L$ such that $|z| \geqslant q^{|N|+1}$. There exists a path p from the root r to a leaf in t such that the number of the nodes on p which has more than two children is at least $\log_q |z| = |N|+1$ by the assumption $|z| \geqslant q^{|N|+1}$.

Therefore, there exist distinct nodes v and v' on p with a same label (say, $A \in N$) which have at least two children. Assume that v is a descendant of v'. Let $k = d(A)$. Let us denote $g_{v,v'}$ by g for simplicity, and the function obtained by compositing g i times by g^i. Note that g^i is not a value obtained by concatenating the value of g i times. For a function g, let us denote the jth component of g by g_j.

Let $K = \{1, 2, \ldots, k\}$. We define a function μ from K to K such that if a variable $y_n (n \in K)$ is contained in g_j, then $\mu(n) = j$. Let $\bar{J}$ be the maximal nonempty subset K' of K which satisfies the condition: if we regard μ as a function from K' to K (by restricting the domain), μ is a permutation over K'. This subset $\bar{J}$ (called the *kernel*) can always be found.

From the definition of $\bar{J}$ and the fact that the number of components of g is k, for each variable y_n $(n \notin \bar{J})$, y_n is moved to one of the components in the kernel by compositing g at most $(k-1)$ times. Therefore, if we let $J^i = \{j \mid$ the jth component of g^i is a constant string$\}$, then $J^i = J^{k-1}$ holds for each $i (i \geqslant k)$. Let $v = \mu^{k-1}$. Since v is also a permutation over the kernel $\bar{J}$, there exists some integer p such that the permutation obtained by compositing v p times is the identity permutation. Let us denote $g^{p(k-1)}$ by $\bar{g}$ for simplicity. $\bar{g}_j(\bar{y})$ is a constant string of the form $\gamma_j \in T^+$ if $j \notin \bar{J}$ and $\gamma_{j1} y_j \gamma_{j2}$ if $j \in \bar{J}$, where γ_{j1} and γ_{j2} are strings over $T \cup \{y_j \mid j \notin \bar{J}\}$. Hence, for any $j \in \bar{J}$, $\bar{g}_j^2(\bar{y}) = \gamma'_{j1} \bar{g}_j(\bar{y}) \gamma'_{j2}$, where γ'_{j1} and γ'_{j2} are the strings over T obtained from γ_{j1} and γ_{j2}, respectively, by substituting γ_i for $y_i (i \notin \bar{J})$. For any positive integer i,

(1) if $j \in \bar{J}$, then

$$\bar{g}_j^i(\bar{y}) = (\gamma'_{j1})^{i-1} \bar{g}_j(\bar{y}) (\gamma'_{j2})^{i-1}; \tag{3.3}$$

(2) otherwise,

$$\bar{g}_j^i(\bar{y}) = \gamma_j. \tag{3.4}$$

Since $|g(\bar{y})| > k$ from (3.2) and $|\bar{g}^{i+1}(\bar{y})| > |\bar{g}^i(\bar{y})|$,

$$\sum_{j \in \bar{J}} |\gamma'_{j1} \gamma'_{j2}| > 0. \tag{3.5}$$

On the other hand, from the condition (f3),

$$g_{v',r}(\bar{y}) = u_0 y_{h_1} u_1 y_{h_2} \cdots u_{k-1} y_{h_k} u_k, \tag{3.6}$$

where r is the root of t, $u_h \in T^*$ $(0 \leqslant h \leqslant k)$ and $(h_1, h_2, \ldots, h_k)$ is a permutation of $(1, 2, \ldots, k)$. Let $\bar{\beta} \in L_G(A)$ be the string derived from v in t. Then, from (3.1), $\bar{g}^i(\bar{\beta}) \in L_G(A)$, $i \geqslant 0$. Again from (3.1),

$$g_{v',r}(\bar{g}^i(\bar{\beta})) \in L_G(A), \quad i \geqslant 0. \tag{3.7}$$

The lemma holds by (3.3) through (3.7) letting $z_i = g_{v',r}(\bar{g}^i(\bar{\beta}))$ for $i \geqslant 0$. □

The next lemma can be proved by using Lemma 3.2.

Lemma 3.3. *$L = \{a_1^n a_2^n \ldots a_{2m+1}^n \mid n \geqslant 1, a_i \in T, 1 \leqslant i \leqslant 2m+1\}$ is not an m-mcfl.*

Proof. Assume that L is an m-mcfl. By Lemma 3.2, let

$$\begin{aligned} z_i &= a_1^{n_i} a_2^{n_i} \ldots a_{2m+1}^{n_i} \\ &= u_1 v_1^i w_1 s_1^i u_2 v_2^i w_2 s_2^i u_3 \ldots u_m v_m^i w_m s_m^i u_{m+1}. \end{aligned}$$

For some j $(1 \leqslant j \leqslant m)$, if there exists some k such that v_j or s_j contains $a_k a_{k+1}$ as a substring, then z_2 contains $a_k a_{k+1}$ twice as its substring. This implies that $z_2 \notin L$; a contradiction. Hence, for each j $(1 \leqslant j \leqslant m)$, $k(j)$ and $h(j)$ $(1 \leqslant k(j) \leqslant h(j) \leqslant 2m+1)$ are uniquely determined, and $v_j \in a_{k(j)}^*$ and $s_j \in a_{h(j)}^*$. Therefore, for some q $(1 \leqslant q \leqslant 2m+1)$, one of $u_1, w_1, \ldots, u_m, w_m, u_{m+1}$ should contain $a_q^{n_i}$ as its substring. But for sufficiently large i, $n_i > |u_j|$ $(1 \leqslant j \leqslant m+1)$ and $n_i > |w_j|$ $(1 \leqslant j \leqslant m)$ hold. This is also a contradiction. □

By using an argument similar to the one used in Example 2.1(3), we can show that $\{a_1^n a_2^n \ldots a_{2m+1}^n \mid n \geqslant 1\} \in (m+1)$-*MCFL*. By this fact and Lemma 3.3, Theorem 3.4 follows.

Theorem 3.4. *m-MCFL$\subsetneqq(m+1)$-MCFL for any $m \geqslant 1$.*

Similarly, next lemma holds as a corollary of Lemma 3.2.

Lemma 3.5. *For any m, $L = \{a^{2^n} \mid n \geqslant 1\}$ does not belong to m-MCFL.*

Proof. Assume that L is an m-mcfl. By Lemma 3.2, let

$$z_i = a^{2^{n_i}} = u_1 v_1^i w_1 s_1^i u_2 v_2^i w_2 s_2^i u_3 \ldots u_m v_m^i w_m s_m^i u_{m+1}.$$

Let $k = \sum_{j=1}^{m} |v_j s_j| > 0$. Then, for any positive integer i, $2^{n_1} + (i-1)k$ is a power of 2. This is a contradiction. □

From Example 2.1(2) and Lemma 3.5, we obtain the next theorem.

Theorem 3.6. *1-PMCFL$\nsubseteq$MCFL.*

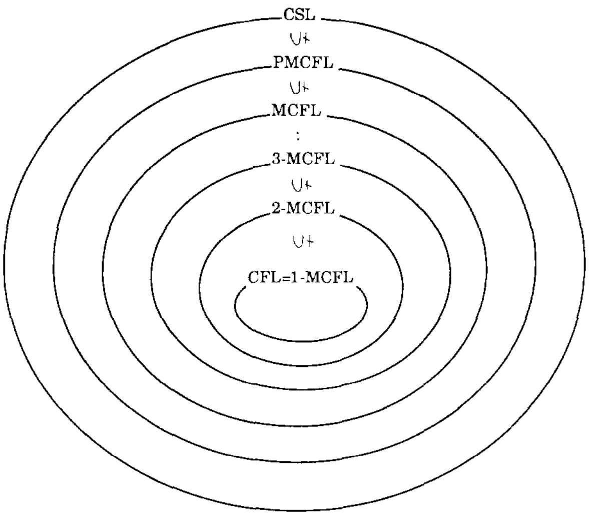


Fig. 1. Inclusion relations between subclasses of CSL.

Inclusion relations between the classes of languages mentioned above are summarized in Fig. 1.

As is the case in cfl's, we have the following positive result for pmcfl's and/or mcfl's.

Theorem 3.7 (Vijay-Shanker et al. [23]). *Any mcfl is semilinear.*

Theorem 3.8. *For a given pmcfg G, it is decidable whether or not $L(G)$ is empty.*

Note that Lemma 3.5 can be proved as a corollary of Theorem 3.7.

$PMCFL$ and $MCFL$ have the following closure properties which CFL has.

Theorem 3.9. (1) *The class m-$PMCFL$ and m-$MCFL$ are closed under substitution.*

(2) *The class m-$PMCFL$ and m-$MCFL$ are closed under union, concatenation, Kleene closure, ε-free Kleene closure.*

(3) *The class m-$PMCFL$ and m-$MCFL$ are closed under intersection with regular languages.*

(Hence, m-$PMCFL$ and m-$MCFL$ are substitution-closed full AFL's.)

Proof. (1) can be easily shown from definition. (2) can be shown from (1) and the fact that these operations can be expressed by regular expressions. For proof of (3) let $G=(N,O,F,P,S)$ be an m-pmcfg (or m-mcfg) which generates L. Let S_R, σ, s_0 and A_R denote the set of states, the state transition function, the initial state and the set of final states, respectively, of a deterministic finite automaton which accepts R. We construct an m-pmcfg (or m-mcfg) $G'=(N',O,F,P',S')$ as follows:

(1) $N'=\{S'\}\cup\{A[s_{10},s_{11},s_{20},s_{21},\dots,s_{d(A)0},s_{d(A)1}]\mid A\in N,\quad s_{ij}\in S_R,\quad 1\leqslant i\leqslant d(A),$ $j=0,1\}$.

(2.1) For each rule

$$A_0\to f[A_1,A_2,\dots,A_{a(f)}]$$

in P and

$$A_i'=A_i[s_{10}^{(i)},s_{11}^{(i)},s_{20}^{(i)},s_{21}^{(i)},\dots,s_{d(A_i)0}^{(i)},s_{d(A_i)1}^{(i)}],\quad 0\leqslant i\leqslant a(f),$$

which satisfy the following *connecting condition*, let

$$A_0'\to f[A_1',A_2',\dots,A_{a(f)}']\in P'.$$

The connecting condition: let each component f^h $(1\leqslant h\leqslant r(f)=d(A_0))$ of f be

$$f^h(\bar{x}_1,\bar{x}_2,\dots,\bar{x}_{a(f)})=\alpha_{h0}z_{h1}\alpha_{h1}z_{h2}\cdots z_{hv_h(f)}\alpha_{hv_h(f)},$$

where

$$\bar{x}_i=(x_{i1},x_{i2},\dots,x_{id_i(f)}),$$

$$\alpha_{hk}\in T^*,\quad 0\leqslant k\leqslant v_h(f),$$

$$z_{hk}=x_{i_{(h,k)}j_{(h,k)}},\quad 1\leqslant k\leqslant v_h(f),\ 1\leqslant i_{(h,k)}\leqslant a(f),\ 1\leqslant j_{(h,k)}\leqslant d_{i_{(h,k)}}(f).$$

Then we have the following:

(a) $s_{j_{(h,1)}0}^{(i_{(h,1)})}=\sigma(s_{h0}^{(0)},\alpha_{h0}),$

(b) $s_{j_{(h,k)}0}^{(i_{(h,k)})}=\sigma(s_{j_{(h,k-1)}1}^{(i_{(h,k-1)})},\alpha_{h(k-1)}), 2\leqslant k\leqslant v_h(f),$ and

(c) $s_{h1}^{(0)}=\sigma(s_{j_{(h,v_h(f))}1}^{(i_{(h,v_h(f))})},\alpha_{hv_h(f)}).$

(2.2) $S'\to S[s_0,s_F]\in P'$, $s_F\in A_R$.

(2.3) There is no rule except those mentioned above.

It is easily shown that

$$L_{G'}(A[s_{10},s_{11},s_{20},s_{21},\dots,s_{d(A)0},s_{d(A)1}])$$
$$=\{\bar{\alpha}=(\alpha_1,\alpha_2,\dots,\alpha_{d(A)})\mid\bar{\alpha}\in L_G(A),\ s_{i1}=\sigma(s_{i0},\alpha_i),\ 1\leqslant i\leqslant d(A)\}.$$

Hence, $L(G')=L\cap R$. □

Since it is undecidable whether or not the intersection of given two cfl's is empty [2, 15], it follows from Theorem 3.8 that both of *PMCFL* and *MCFL* are not closed under intersection. Hence, they are not closed under complement, because they are closed under union.

3.2. Membership problem for multiple context-free languages

Here we discuss the membership problem for mcfl's, i.e. the problem of deciding, for a given mcfg G and a string $\alpha\in T^*$, whether or not α is in $L(G)$. Hereafter, we assume that a given mcfg $G=(N,O,F,P,S)$ satisfies the condition (f3), (N1), (N3) and (N4) of Lemma 2.2.

For $\alpha \in T^*$, let $\bar{\alpha} = (\alpha(l_1, r_1), \alpha(l_2, r_2), \ldots, \alpha(l_k, r_k))$ be a k-tuple of nonoverlapping substrings of α. We define the *position vector* of $\bar{\alpha}$ to be $(l_1, r_1, l_2, r_2, \ldots, l_k, r_k)$. Since a position vector uniquely determines a tuple of substrings in a given string, we sometimes specify a phrase by giving its position vector. For a phrase $(l_1, r_1, l_2, r_2, \ldots, l_k, r_k)$, the greatest value among $r_1, r_2, \ldots, r_k$ is called the *right end* of the phrase.

Now let $\bar{\alpha}$ be an A-phrase and assume that

(1) $\alpha_i \in L_G(A_i)$ for $1 \leqslant i \leqslant m$,

(2) $A \to f[A_1, A_2, \ldots, A_{a(f)}]$, with $m = a(f)$, and

(3) $\bar{\alpha} = f(\alpha_1, \alpha_2, \ldots, \alpha_{a(f)})$ $(\in L_G(A))$.

Then we call α_i $(1 \leqslant i \leqslant a(f))$ the *ith daughter phrase* of $\bar{\alpha}$, or simply, a *daughter phrase* of $\bar{\alpha}$. We will write the position vector of the ith daughter phrase as

$$(l_1^{(i)}, r_1^{(i)}, \ldots, l_{d(A_i)}^{(i)}, r_{d(A_i)}^{(i)})$$

for $1 \leqslant i \leqslant a(f)$. The daughter phrase whose right end is greater than that of any other daughter phrase is called the *rightmost daughter phrase.*

Let $R: A \to f[A_1, A_2, \ldots, A_{a(f)}]$ be a rule in P. Suppose that, for any $\alpha \in L(G)$, the number of the subphrases $\bar{\alpha}$'s of α satisfying the conditions (1) through (3) above is not greater than $O(n^e)$, where $n = |\alpha|$. Then we say that the *degree of the rule R*, denoted by $D(R)$, *is not greater than e.* We also say that the *degree of a grammar* G, denoted by $D(G)$, *is not greater than e*, if the degree of every rule in G is not greater than e.

We evaluate $D(R)$ as follows:

(A1) The total number of components of the position vectors of the daughter phrases $\alpha_1, \alpha_2, \ldots, \alpha_{a(f)}$ is equal to

$$2 \sum_{i=1}^{a(f)} d(A_i).$$

(A2) In (2.1), let z_{hk} be $x_{i_{(h,k)} j_{(h,k)}}$ $(1 \leqslant i_{(h,k)} \leqslant a(f)$ and $1 \leqslant j_{(h,k)} \leqslant d_{i_{(h,k)}}(f) (= d(A_{i_{(h,k)}}))$ for $1 \leqslant h \leqslant r(f) (= d(A))$ and $1 \leqslant k \leqslant v_h(f))$. Then the constraints

$$l_{j_{(h,k)}}^{(i_{(h,k)})} = r_{j_{(h,k-1)}}^{(i_{(h,k-1)})} + |\alpha_{h(k-1)}| + 1$$

must be satisfied for $1 \leqslant h \leqslant r(f)$ and $2 \leqslant k \leqslant v_h(f)$. For example, for

$$f^h(\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_{a(f)}) = x_{21} b a x_{13} x_{12},$$

there are two constraints such that

$$l_3^{(1)} = r_1^{(2)} + 3,$$

and

$$l_2^{(1)} = r_3^{(1)} + 1.$$

For each h $(1 \leqslant h \leqslant r(f))$, the number of such constraints given above is equal to the number of variables appearing in the right-hand side of (2.1) minus 1. Hence, the total number of constraints is equal to

$$\{\text{the total number of variables appearing in the right-hand side of } (*1) \text{ for } 1 \leqslant h \leqslant r(f)\} - d(A).$$

By the conditions (f2) and (f3), this is equal to

$$\left\{\sum_{i=1}^{a(f)} d(A_i)\right\} - d(A).$$

(A3) The constraints mentioned in (A2) are linearly independent when they are considered as linear equations on variables $l_j^{(i)}, r_j^{(i)}$ $(1 \leqslant i \leqslant a(f), 1 \leqslant j \leqslant d_i(f)\,(=d(A_i)))$. Hence, by (A1) and (A2), the number of variables which are independent is not greater than

$$d = d(A) + \sum_{i=1}^{a(f)} d(A_i). \tag{3.8}$$

Therefore, $D(R) \leqslant d$ holds.

The above discussion (A1) through (A3) gives an upper bound on $D(R)$ in a general case. If we have more information about R, the upper bound may be improved. For example, if we know that the second component of the first daughter phrase is always less than or equal to some fixed value in (A3), then we can conclude that $D(R) \leqslant d-1$ holds. In the following, we assume for simplicity that G satisfies (N5) of Lemma 2.2, i.e. every α_{hk} in (2.1) is the empty string for $1 \leqslant h \leqslant r(f)$ and $0 \leqslant k \leqslant d_i(f)$. For a given mcfg G which does not satisfy (N5), we can construct an mcfg G' which satisfies (N5), $L(G) = L(G')$ and $D(G) = D(G')$, by using Lemma 2.2.

Let I_n denote the set $\{1, 2, \ldots, n, *\}$. For each $A \in N$, j $(1 \leqslant j \leqslant n)$ and $\bar{u} = (u_1, u_2, \ldots, u_{2d(A)}) \in I_n^{2d(A)}$, let us define $p(A, j, \bar{u})$ to be

$\{\bar{v} = (v_1, v_2, \ldots, v_{2d(A)}) \mid (1)\ \bar{v}$ is a position vector of some A-phrase, (2) the right end of $\bar{v}$ is not greater than j, and (3) $v_i = u_i$ or $u_i = *$ for $1 \leqslant i \leqslant 2d(A)\}$.

$\bar{u}$ is called a *constraint vector*. The value "$*$" in a component of $\bar{u}$ (if it exists) denotes that there is no constraint for that component. Let $\gamma(\bar{u})$ denote the number of $*$'s appearing in $\bar{u}$. Then

$$|p(A, j, \bar{u})| \leqslant j^{\gamma(\bar{u})} \leqslant n^{\gamma(\bar{u})}$$

by the definition of p.

In what follows, for an mcfg G, we describe the procedure MEMBER which decides in $O(|\alpha|^{D(G)})$-time whether or not a given string α is in $L(G)$. This procedure is an extension of an $O(n^3)$-algorithm for the membership problem for cfl's.

MEMBER has variables $P(A, \bar{u})$ for each $A \in N$, $\bar{u} \in I_n^{2d(A)}$, and $\Delta p(A)$ for each $A \in N$. $P(A, \bar{u})$ is used to store the set $p(A, j, \bar{u})$, and $\Delta p(A)$ is used to store temporarily the set of position vectors which are to be added to $P(A, \bar{u})$.

We assume that the arithmetic operations on integers not greater than n can be executed within a constant time (rigorously, within $O(p(\log n))$, where p is some polynomial function).

The data structures and the operations on them used in MEMBER are represented in the following way:

(1) For each $A \in N$ and $\bar{u} \in I_n^{2d(A)}$, $P(A, \bar{u})$ is represented by a linearly linked list l. l is allocated in such a way that the address of its header cell can be calculated within some constant time from A and $\bar{u}$. l has two pointers, one of which points the "current" cell (used in procedure E) and the other points the last cell.

(2) For each position vector $\bar{v}$ and $A \in N$, we use one bit memory, denoted by $b(\bar{v}, A)$, whose address can be calculated within some constant time from $\bar{v}$ and A. If $b(\bar{v}, A) = 1$, then it represents that $\bar{v} \in \Delta p(A)$; otherwise, $\bar{v} \notin \Delta p(A)$. The set $\Delta p(A)$ itself is represented by a linearly linked list l'. l' has two pointers, say p_1 and p_2. The cells preceding the cell pointed by p_1 are considered as "marked" and the other cells "unmarked". p_2 points the last cell.

```
Procedure MEMBER($\alpha$)
(* For a given string $\alpha$ in $T^*$, MEMBER answers "$\alpha \in L(G)$" or "$\alpha \notin L(G)$". Assume that
for each $A \in N$ and $\bar{u} \in I_n^{2d(A)}$, the initial value of the variable $P(A, \bar{u})$ is the empty set $\emptyset$. *)

begin
  for $j := 1$ to $n$ do B($j$); (* B($j$) sets the value of $P(A, \bar{u})$ to $p(A, j, \bar{u})$ *)
  if $P(S, (1, n)) \neq \emptyset$ then answer "$\alpha \in L(G)$" else "$\alpha \notin L(G)$";
end;
```

```
Procedure B($j$)
(* Assume that the value of the variable $P(A, \bar{u})$ is already set to $p(A, j-1, \bar{u})$ for each
$A$ and $\bar{u}$ when B($j$) is called. Then the value of $P(A, \bar{u})$ is set to $p(A, j, \bar{u})$ for each $A$ and
$\bar{u}$ by executing B($j$) *)

begin
  for each $A \in N$ do
    $\Delta p(A)$ := the set of all the position vectors of $A$-phrases whose right ends
            are $j$;                                                    (3.9)

  for each $A \in N$ do begin
    for each $\bar{v} = (v_1, v_2, \ldots, v_{2d(A)}) \in \Delta p(A)$ and
    each constraint vector $\bar{u} = (u_1, u_2, \ldots, u_{2d(A)})$ such that $v_i = u_i$ or $u_i = *$ holds for
    each $i$ $(1 \leqslant i \leqslant 2d(A))$
    do $P(A, \bar{u}) := P(A, \bar{u}) \cup \{\bar{v}\}$
  end
end;
```

For simplicity, in what follows, we may also use the term "A-phrase" to denote its position vector. Statement (3.9) is refined as follows:

```
for each $A \in N$ do
  $\Delta p(A)$ := the set of all (unmarked) $A$-phrases, each of which can be obtained by
        applying only a terminating rule;
for each $A \in N$ and each unmarked $\bar{v}_1 \in \Delta p(A)$ do begin
  mark $\bar{v}_1$;
  $C(A, \bar{v}_1, j)$
  (* $C(A, \bar{v}_1, j)$ adds every $A_0$-phrase whose rightmost daughter phrase is $\bar{v}_1$, to
  $\Delta p(A_0)$ *)
end;
```

Let

$$\varphi : A_0 \to f[A_1, A_2, \ldots, A_{a(f)}]$$

be a rule and $i_1, i_2, \ldots, i_s$ be distinct nonnegative integers not greater than $a(f)$. Let $\bar{v}_1$ be an A_{i_1}-phrase whose right end is j and $\bar{v}_t$ be an A_{i_t}-phrase whose right end is less than j for $2 \leqslant t \leqslant s$. Consider the following condition:

(P) There exists an A_0-phrase $\bar{v}$ which satisfies

(P1) the rightmost daughter phrase of $\bar{v}$ is $\bar{v}_1$,

(P2) the i_tth daughter phrase of $\bar{v}$ is $\bar{v}_t$ for $1 \leqslant t \leqslant s$, and

(P3) $\bar{v}$ is obtained by applying the rule φ to $\bar{v}_1, \bar{v}_2, \ldots,$ and $\bar{v}_s$.

A necessary condition (and also a sufficient condition if $s = a(f)$) for the condition (P) to hold is

(B1) Assume that $\bar{v}_1 = (l_1, r_1, \ldots, l_k, r_k, \ldots)$ and $r_k = j$. Then the variable $x_{i_1 k}$ appears at the right end of the right-hand side of (2.1) for some h $(1 \leqslant h \leqslant r(f))$, i.e.

$$z_{hv_h(f)} = x_{i_1 k},$$

(B2) $\bar{v}_1, \bar{v}_2, \ldots,$ and $\bar{v}_s$ are nonoverlapping, and

(B3) $\bar{v}_1, \bar{v}_2, \ldots,$ and $\bar{v}_s$ satisfy all the constraints in (2.1) (see (A2) at the beginning of this section).

Considering the arguments given above, we can define procedure $C(A, \bar{v}_1, j)$ as follows.

Procedure $C(A, \bar{v}_1, j)$

(* Assume that the right end of $\bar{v}_1$ is j. Then $C(A, \bar{v}_1, j)$ finds all the A_0-phrases whose rightmost daughter phrase is $\bar{v}_1$, and add them to $\Delta p(A_0)$ if they are not yet in $\Delta p(A_0)$. *)

```
begin
  for each rule $\varphi : A_0 \to f[A_1, A_2, \ldots, A_{a(f)}]$ and
  each nonnegative integer $i_1$ not greater than $a(f)$ satisfying "$A_{i_1} = A$ and the
  condition (B1) above holds"
  do $E(\varphi, (\bar{v}_1))$;
end;
```

(* For simplicity, we assume $i_1 = 1$ in the following. We can always make $i_1 = 1$ by introducing the function which "rotates" its arguments. *)

Procedure $\mathrm{E}(\varphi, (\bar{v}_1, \bar{v}_2, \ldots, \bar{v}_s))$

(* Let $\varphi : A_0 \to f[A_1, A_2, \ldots, A_{a(f)}]$ and assume that $s \leqslant a(f)$. Then $\mathrm{E}(\varphi, (\bar{v}_1, \bar{v}_2, \ldots, \bar{v}_s))$ tests whether all the conditions (B1), (B2) and (B3) are satisfied with $i_1 = 1$, $i_2 = 2, \ldots,$ and $i_s = s$. If so, procedure E finds all the A_0-phrases $\bar{v}$'s such that the rightmost daughter phrase of $\bar{v}$ is $\bar{v}_1$, the ith daughter phrase of $\bar{v}$ is $\bar{v}_i$ $(1 \leqslant i \leqslant s)$, and $\bar{v}$ is obtained by applying φ to $\bar{v}_1, \bar{v}_2, \ldots,$ and $\bar{v}_s$. Then E adds them to $\Delta p(A_0)$ if they are not in $\Delta p(A_0)$. *)

```
begin
  $\bar{u}$:= the constraint vector for the $(s+1)$th daughter phrase;
  (* $\bar{u}$ can be obtained from the constraints of (2.1) and $\bar{v}_1, \bar{v}_2, \ldots, \bar{v}_s$. *)
  if $P(A_{s+1}, \bar{u}) = \emptyset$ then return
  else
    for each $\bar{v}_{s+1}$ in $P(A_{s+1}, \bar{u})$
      such that $\bar{v}_1, \bar{v}_2, \ldots,$ and $\bar{v}_{s+1}$ are nonoverlapping do
      if $s+1 < a(f)$ then $\mathrm{E}(\varphi, (\bar{v}_1, \bar{v}_2, \ldots, \bar{v}_{s+1}))$
      else (* $s+1 = a(f)$ holds *) begin
        $\bar{v}$:= the $A_0$-phrase whose rightmost daughter phrase is $\bar{v}_1$, whose $t$th daugh-
            ter phrase is $\bar{v}_t$ $(1 \leqslant t \leqslant s+1)$, and which is obtained by applying $\varphi$ to
            $\bar{v}_1, \bar{v}_2, \ldots,$ and $\bar{v}_{s+1}$;
        $\Delta p(A_0) := \Delta p(A_0) \cup \{\bar{v}\}$
      end
end;
```

Suppose that $D(G)$ is not greater than e. Then

(1) for each j $(1 \leqslant j \leqslant n)$, the total number of tuples of daughter phrases to be tested in procedure $\mathrm{B}(j)$ is $\mathrm{O}(n^{e-1})$ (note that the right end of a rightmost daughter phrase is fixed to j), and,

(2) for each tuple of daughter phrases, the required time for testing is $\mathrm{O}(1)$. Hence, we conclude that the time complexity of $\mathrm{MEMBER}(\alpha)$ is $\mathrm{O}(n^e)$.

Theorem 3.10. *Let G be an mcfg which satisfies the conditions* (f3), (N1), (N3) *and* (N4) *of Lemma* 2.2, *and suppose that $D(G)$ is not greater than e. For a given $\alpha \in T^*$, we can decide whether or not α is in $L(G)$ within* $\mathrm{O}(|\alpha|^e)$ *time.*

Using similar methods described in this section, an $\mathrm{O}(n^{e+1})$-algorithm for the membership problem for pmcfl's can be obtained. The difference between the algorithms for mcfg's and pmcfg's is that in the case of pmcfg's, a variable occurring in (2.1) for some h may occur at other positions in (2.1) for the same h and/or occur in (2.1) for another h. For such a variable x, the additional test is necessary that examines whether all the strings corresponding to the occurrences of x are the same. This test takes $O(n)$ time.

Vijay-Shanker et al. [23] showed that the time complexity of the membership problem for mcfl's is polynomial order by showing that for a given mcfg G, an alternating Turing machine can be constructed which accepts $L(G)$ in $O(\log n)$ space. However, they did not give any upper bound for polynomial time complexity of the problem. Theorem 3.10 further gives an upper bound which depends on the degree of a grammar. It is an interesting problem to find, for a given G, an mcfg whose degree is not greater than some constant and is weakly equivalent to G. In cfg's, the degree of any Chomsky normal form is not greater than 3.

Joshi [4] pointed out that from Theorem 3.10 it follows that the class of languages generated by multicomponent tag's [23] whose tree sets have at most k trees in them can be recognized in time $O(n^{6k})$.

4. Head grammars, tree adjoining grammars and 2-mcfg's

4.1. Head grammars

In [12], Pollard defined a special subclass of gcfg's, called *head grammars* (*hg's*), which are intended for describing the syntax of natural languages. The hg's exceed the cfg's in generative capacity but retain most of their pleasant mathematical properties.

We will define head grammars and some related concepts according to [12] in the sequel. For a finite set T of terminal symbols, let $T\uparrow$ denote the set

$$\{(\alpha, i) \mid \alpha \in T^*,\ 1 \leqslant i \leqslant |\alpha|\} \cup \{(\varepsilon, 0)\}.$$

For $(\alpha, i) \in T\uparrow$, the ith symbol of α is called the *head* of (α, i) and each element of $T\uparrow$ is called a *headed string* over T.

Definition 4.1. An s-ary function $c:(T\uparrow)^s \to T\uparrow$ is called an (s-ary) *headed concatenation operation* if a nonnegative integer h ($h \leqslant s$) and terminal strings $\gamma_0, \gamma_1, \ldots, \gamma_s$ on T^* are specified, and for each headed string $(\alpha_j, i_j) \in T\uparrow$ ($1 \leqslant j \leqslant s$),

$$c[(\alpha_1, i_1), (\alpha_2, i_2), \ldots, (\alpha_s, i_s)]$$

is defined as follows:

Case 1: If $(\alpha_h, i_h) \neq (\varepsilon, 0)$, then the value of the function c is obtained by concatenating the argument strings and the terminal strings $\gamma_0, \gamma_1, \ldots, \gamma_n$, and the head of the resulting string is defined to be the head of the hth argument of c, i.e.

$$c[(\alpha_1, i_1), (\alpha_2, i_2), \ldots, (\alpha_s, i_s)] = \left(\gamma_0 \alpha_1 \gamma_1 \alpha_2 \ldots \gamma_{s-1} \alpha_s \gamma_s,\ \sum_{i=0}^{h-1} |\gamma_i| + \sum_{i=1}^{h-1} |\alpha_i| + i_h\right).$$

Case 2: If $(\alpha_j, i_j)=(\varepsilon, 0)$ for all j $(1 \leqslant j \leqslant s)$, then

$$c[(\varepsilon,0),(\varepsilon,0),\ldots,(\varepsilon,0)]=(\varepsilon,0).$$

Case 3: If $(\alpha_h, i_h)=(\varepsilon,0)$ and $(\alpha_p, i_p) \neq (\varepsilon, 0)$ for some p $(p \neq h,\ 1 \leqslant p \leqslant s)$, then

$$c[(\alpha_1,i_1),(\alpha_2,i_2),\ldots,(\alpha_s,i_s)] \text{ is undefined.}$$

For each string $\alpha=\alpha_1\alpha_2\ldots a_n \in T^*$, let us define $\alpha(i,j)$ to be the substring $a_i a_{i+1}\ldots a_j$ of α if $1 \leqslant i \leqslant j \leqslant |\alpha|$, and define $\alpha(i,j)=\varepsilon$ otherwise.

Definition 4.2. The *head-wrapping operations* are the binary functions w_1, w_2, w_3 and $w_4: T\!\uparrow \times T\!\uparrow \rightarrow T\!\uparrow$ defined as follows:

$$w_1[(\alpha,i),(\beta,j)]=(\alpha(1,i)\beta\alpha(i+1,|\alpha|),i),$$

$$w_2[(\alpha,i),(\beta,j)]=(\alpha(1,i)\beta\alpha(i+1,|\alpha|),i+j),$$

$$w_3[(\alpha,i),(\beta,j)]=(\alpha(1,i-1)\beta\alpha(i,|\alpha|),|\beta|+i),$$

$$w_4[(\alpha,i),(\beta,j)]=(\alpha(1,i-1)\beta\alpha(i,|\alpha|),\max(i-1,0)+j).$$

Exceptions:

(1) For each k $(1 \leqslant k \leqslant 4)$, $w_k[(\varepsilon,0),(\varepsilon,0)]$ is defined to be $(\varepsilon,0)$.

(2) For $k=1,3$ and $(\beta,j) \neq (\varepsilon,0)$, $w_k[(\varepsilon,0),(\beta,j)]$ is undefined.

(3) For $k=2,4$ and $(\alpha,i) \neq (\varepsilon,0)$, $w_k[(\alpha,i),(\varepsilon,0)]$ is undefined.

w_1 and w_2 are called *left-wrapping operations* and w_3 and w_4 are called *right-wrapping operations.*

Definition 4.3. Let T be a finite set of terminal symbols and $G=(N,O,F,P,S)$ a gcfg with $N \cap T=\emptyset$. Then G is called a *head grammar* (*hg*) if

(1) $O=T\!\uparrow$, and

(2) each function in F, other than a constant function, is either a headed concatenation operation or a head-wrapping operation.

For an hg G, the language

$$\{\alpha \mid (\alpha,i) \in L(G)\}$$

is called the *underlying language generated by G.* A language L is called a *head language* (*hl*) if L is the underlying language generated by some hg. Let HL denote the class of hl's.

Example 4.4. Let $G=(N,O,F,P,S)$ be an hg where,

(1) $T=\{a,b\}$,

(2) $N=\{S,A,B,D,E\}$,

(3) $F=\{w_3, c_a, c_b\}$, where $c_x[(\alpha, i)]=(\alpha x, i)$ for $x=a, b$, and

(4) $P=\{S\rightarrow(aa,2)\,|\,(bb,2)\,|\,c_a(D)\,|\,c_b(E),\quad D\rightarrow w_3(S,A),\quad E\rightarrow w_3(S,B),\quad A\rightarrow(a,1),$ $B\rightarrow(b,1)\}$.

Then, the underlying language generated by G is

$$\{\alpha^2\,|\,\alpha\in\{a,b\}^+\}.$$

Let G be an hg and assume that (β, i) is a subphrase of $(\alpha, j)\in L(G)$. Then both the strings $\beta(1, i-1)$ and $\beta(i+1, |\beta|)$ are always substrings of α. Although the head $\beta(i, i)$ of (β, i) always appears between $\beta(1, i-1)$ and $\beta(i+1, |\beta|)$ in α, it is not always the case that $\beta(i, i)$ is adjacent to $\beta(1, i-1)$ and/or $\beta(i+1, |\beta|)$ in α. For example, let $G=(N, O, F, P, S)$ be an hg where

$$P=\{A\rightarrow(a_1a_2a_3, 2),\ B\rightarrow(b,1),\ C\rightarrow w_1(A,B),\ S\rightarrow w_3(C,B)\}.$$

Then $(a_1a_2a_3, 2)$ is a subphrase of $(a_1ba_2ba_3, 3)$ and the head a_2 of $(a_1a_2a_3, 2)$ is adjacent to neither a_1 nor a_3 in $a_1ba_2ba_3$. Hence, for a headed string $(\beta, i), (\beta(1, i-1), \beta(i,i), \beta(i+1, |\beta|))$ might be a more appropriate notation than (β, i).

Definition 4.5. An hg $G=(N, O, F, P, S)$ is called a *normal form head grammar* (*nhg*) if G satisfies the following conditions (1) through (3):

(1) The right-hand side of each terminating rule in P is either $(\varepsilon, 0)$ or $(a, 1)$ $(a\in T)$.

(2) Headed concatenation operations in F are limited to the following binary functions c_1 and c_2:

$$c_1[(\alpha, i), (\beta, j)]=(\alpha\beta, i), \text{ and}$$

$$c_2[(\alpha, i), (\beta, j)]=(\alpha\beta, |\alpha|+j).$$

(Exceptions: as in Case 2 and Case 3 of Definition 4.1)

(3) If $A\rightarrow(\varepsilon, 0)\in P$, then A does not appear in the right-hand side of any rule in P.

Lemma 4.6 (Roach [14]). *For a given hg G, an nhg G′ can be constructed such that the underlying languages generated by G and G′ are the same.*

An nhg which uses only left-wrapping operations w_1 and w_2 as wrapping operations is called a *left-wrapping head grammar* (*lhg*). Similarly, an nhg which uses only right-wrapping operations w_3 and w_4 as wrapping operations is called a *right-wrapping head grammar* (*rhg*). The class of the underlying languages generated by lhg's and rhg's are denoted by LHL and RHL, respectively.

It was shown that for a given hg G, we can effectively construct a 2-mcfg G' of $D(G')\leqslant 6$ such that $L(G)$ is the same as the underlying language generated by G and G' satisfies condition (f3), (N1), (N3) and (N4) [6]. Therefore, by Theorem 3.10 the next corollary holds.

Corollary 4.7. *The time complexity of the membership problem for hl's is* $O(n^6)$.

Corollary 4.7 was shown in an earlier paper [22] as a corollary of the fact that the class of languages generated by tag's includes the class of hl's (see Theorem 4.9) and that the time complexity of the membership problem for tal's is $O(n^6)$ [21]. This result is an improvement over the $O(n^7)$-algorithm given by Pollard [12]. Pollard also gives an $O(n^6)$-algorithm of the membership problem for left-wrapping hg's and right-wrapping hg's [12]. In Section 4.1, it is shown that the generative capacities of hg's, lhg's and rhg's are equivalent (Corollary 4.13).

4.2. Equivalence of HL and TAL

Hg's are interesting in that a wide range of "discontinuous constituents" such as *subject-auxiliary inversion* can be defined naturally by using hg's. Pollard defines the headed concatenation operations and the head-wrapping operations as partial functions on headed strings. That is, a function in hg's is undefined when the value of the argument whose head is designated as the head of the value of the function is $(\varepsilon, 0)$ (e.g. if the first argument is $(\varepsilon, 0)$ and the second argument is not $(\varepsilon, 0)$, then w_1 and w_3 are undefined [see Definition 4.2]). This partiality of the functions has led to difficulties in proving certain formal properties of hg's. For example, *CFL* is closed under substitution, and the proof is trivial from the definition. On the other hand, it has not yet been shown that the class *HL* is also closed under substitution, in spite of an affirmative conjecture (see [12, Appendix 2] and [14]).

Vijay-Shanker et al. [22] use pairs of strings (α_1, α_2) (called split strings) instead of headed strings, and consider that a head is not a symbol but a position between two strings α_1 and α_2. They define three operations as total functions on split strings corresponding to the operations of hg's and introduce *modified head grammars* (*mhg's*) which deal with split strings. They showed that the generative capacity of mhg's is equivalent to that of tag's and is not weaker than that of hg's.

In what follows, we show that the generative capacity of mhg's, hg's and tag's are all equivalent, and show as a corollary, that *HL* is a substitution-closed full AFL and the generative capacity of hg's are not weakened even if the head-wrapping operations are restricted to left-wrapping operations or right-wrapping operations. We first define modified head grammars.

Definition 4.8 (Vijay-Shanker et al. [22]). Let T be a set of terminal symbols. A 2-mcfg $G = (N, O, F, P, S_0)$ $(O = T^* \cup (T^*)^2)$ is called a *modified head grammar* (*mhg*) if G satisfies the following conditions (1) through (3):

(1) For each nonterminal symbol A other than S_0, $d(A) = 2$.

(2) There exists a nonterminal symbol S other than S_0 such that $S_0 \to J[S]$ is in P, where

$$J[(x_1, x_2)] = x_1 x_2.$$

S does not appear in the right-hand sides of the rules in P other than $S_0 \to J[S]$.

(3) The nonterminating rules in P other than $S_0 \to J[S]$ has the following form:

$$A \to f[B, D] \text{ with } A, B, D \in N - \{S_0\} \text{ and } f \in \{C_1, C_2, W\}.$$

Functions C_1, C_2 and W are defined as follows:

$$C_1[(x_1, x_2), (y_1, y_2)] = (x_1, x_2 y_1 y_2),$$

$$C_2[(x_1, x_2), (y_1, y_2)] = (x_1 x_2 y_1, y_2),$$

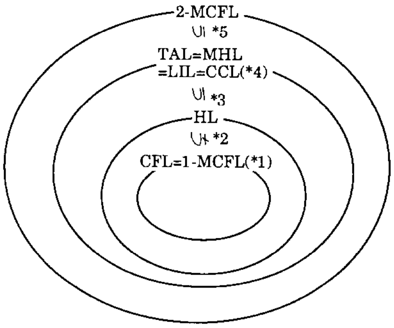
$$W[(x_1, x_2), (y_1, y_2)] = (x_1 y_1, y_2 x_2).$$

A language generated by an mhg is called a *modified head language* (abbreviated as *mhl*) and let MHL denote the class of mhl's.

It has already been shown that $CFL = 1\text{-}MCFL$ [see (2.2)], $CFL \subsetneq HL$ [12] and $HL \subseteq TAL = MHL$ [22], $TAL = LIL$ [20] and $TAL = CCL$ [25], where LIL is the class of languages generated by linear indexed grammars [1] and CCL is the class of languages generated by combinatory categorical grammars [17, 18]. It is obvious that $MHL \subseteq 2\text{-}MCFL$ since mhg's is a subclass of 2-mcfg's. The following theorem summarizes these results (see Figure 2).

Theorem 4.9. $CFL = 1\text{-}MCFL \subsetneq HL \subseteq TAL = MHL = LIL = CCL \subseteq 2\text{-}MCFL$.

In what follows, we show that $MHL \subseteq HL$. That $MHL \subsetneq 2\text{-}MCFL$ is shown in Section 4.3. Let G_0 be an mhg. We will construct an hg weakly equivalent to G_0. By Lemma 2.2, we first construct a 2-mcfg $G = (N, O, F, P, S)$ which is equivalent to G_0 and satisfies condition (f3) and (N1) through (N5) of Lemma 2.2. G has the following property by construction.



*1 See [6], original idea is from [12].
*2 See [12].
*3 See [22].
*4 See [22] for $TAL = MHL$, [20] for $TAL = LIL$ and [25] for $TAL = CCL$.
*5 Trivial.

Fig. 2. Inclusion relations between subclasses of 2-MCFL.

Property DRV. *Each function used in G other than a constant function is obtained from some function f in $\{C_1, C_2, W\}$ by deleting several (possibly zero, but not all) variables in the definition of f* [see (2.1)] *and deleting the resulting components which are the empty strings. (These functions are shown in Table* 1.)

By conditions (N3) and (N4), for any derivation tree t in G, none of ε, (α, ε) and (ε, α) $(\alpha \in T^*)$ is derived from any node other than the root of t. By using this property, we will construct an hg which "simulates derivations" in G without letting the function be undefined (see the discussion at the beginning of this section).

Lemma 4.10. *For a given mhg G_0, an hg G' can be constructed such that the underlying language generated by G' is $L(G_0)$.*

Proof. For a given mhg G_0, let $G=(N, O, F, P, S)$ $(O=T^* \cup (T^*)^2)$ be the 2-mcfg constructed from G_0 by Lemma 2.2. We will construct an hg $G'=(N', T\uparrow, F', P', S)$ satisfying the following condition EQ. We let

$$N'=N \cup \{a \backslash A \mid A \in N,\, a \in T\} \cup \{A_{\text{top}} \mid A \in N,\, d(A)=2\} \cup \{[a] \mid a \in T\} \cup \{EPS\}.$$

Condition EQ. The following conditions (EQ1) through (EQ3) are satisfied.

(EQ1) $\varepsilon \in L(G) \Leftrightarrow (\varepsilon, 0) \in L(G')$

(EQ2) For each $A \in N$ with $d(A)=1$, $a \in T$ and $\alpha \in T^+$ $(|\alpha|>1)$,

$$\alpha \in L_G(A) \text{ and } \alpha(1,1)=a \Leftrightarrow \exists j\ (1 \leqslant j \leqslant |\alpha|-1)\text{: } (\alpha(2,|\alpha|), j) \in L_{G'}(a \backslash A).$$

(EQ3) For each $A \in N$ with $d(A)=2$, $a \in T$ and $\alpha_1, \alpha_2 \in T^+$,

$$(\alpha_1, \alpha_2) \in L_G(A) \text{ and } \alpha(1,1)=a \Leftrightarrow (\alpha_1(2,|\alpha_1|)\alpha_2, |\alpha_1|) \in L_{G'}(a \backslash A).$$

For any tuple $\bar{\alpha}=(\alpha_1, \alpha_2, \ldots, \alpha_n)$ of strings in T^*, $\alpha_1(1,1)$ is said to be the *top symbol* of $\bar{\alpha}$ if $\alpha_1 \neq \varepsilon$. In what follows, we call a tuple of strings merely "a string".

Construction of P'. Consider a rule $A \to f[B, D]$ in P, $\bar{\beta} \in L_G(B)$, $\bar{\delta} \in L_G(D)$ and $\bar{\alpha}=f[\bar{\beta}, \bar{\delta}] \in L_G(A)$. By the definition of the functions in G (see Table 1), the top symbol of $\bar{\alpha}$ is either the top symbol of $\bar{\beta}$ or that of $\bar{\delta}$. When none of the components of the strings derived from B and D is the empty string, it is easily determined, by examining only f, which nonterminal symbol, B or D, derives the string whose top symbol is the top symbol of the string derived from A. Similarly, for a rule $A \to g[B]$ in P, $\bar{\beta} \in L_G(B)$ and $\bar{\alpha}=g[\bar{\beta}] \in L_G(A)$, the top symbol of $\bar{\alpha}$ is the top symbol of $\bar{\beta}$ if none of the components of $\bar{\beta}$ is the empty string. For a rule $R: A \to f[B, D]$ $(f \neq C_1^{(12)})$, assume that the top symbol of a string derived from A is that of B. Then, a rule of the form $a \backslash A \to f'[a \backslash B, D]$ is added to P' for each $a \in T$, where f' is chosen so that condition (EQ2) is satisfied if $d(A)=1$ and (EQ3) is satisfied if $d(A)=2$. If the top symbol of a string derived from A is that of D, a rule of the form $a \backslash A \to f'[B, a \backslash D]$ is added to P' in a similar manner. Assume that $A \to C_1^{(12)}[B, D]$ is in P. Since the second component

Table 1
The functions used in G' constructed from an mhg G by Lemma 2.2.

[1]	$C_1[(x_1,x_2),(y_1,y_2)]$	$=(x_1,x_2y_1y_2)$
[2]	$C_2[(x_1,x_2),(y_1,y_2)]$	$=(x_1x_2y_1,y_2)$
[3]	$W[(x_1,x_2),(y_1,y_2)]$	$=(x_1y_1,y_2x_2)$
[4]	$C_1^{(11)}[x,(y_1,y_2)]$	$=xy_1y_2$
[5]	$C_1^{(12)}[x,(y_1,y_2)]$	$=(x,y_1y_2)$
[6]	$C_1^{(21)}[(x_1,x_2),y]$	$=(x_1,x_2y)$
[7]	$C_2^{(11)}[x,(y_1,y_2)]$	$=(xy_1,y_2)$
[8]	$C_2^{(21)}[(x_1,x_2),y]$	$=(x_1x_2,y)$
[9]	$C_2^{(22)}[(x_1,x_2),y]$	$=x_1x_2y$
[10]	$W^{(11)}[x,(y_1,y_2)]$	$=(y_1,y_2x)$
[11]	$W^{(21)}[(x_1,x_2),y]$	$=(x_1,yx_2)$
[12]	$W^{(22)}[(x_1,x_2),y]$	$=(x_1y,x_2)$
[13]	$C_1^{(11,12)}[(x_1,x_2)]$	$=x_1x_2$
[14]	$C_1^{(11,21)}[x,y]$	$=xy$
[15]	$C_1^{(12,21)}[x,y]$	$=(x,y)$
[16]	$C_1^{(21,22)}[(x_1,x_2)]$	$=(x_1,x_2)$
[17]	$W^{(11,21)}[x,y]$	$=yx$
[18]	$W^{(11,22)}[x,y]$	$=(y,x)$
[19]	The function obtained by deleting three variables ($ID[x]=x$)	

$C_1^{(21)}=C_1^{(22)}, C_2^{(11)}=C_2^{(12)}=W^{(12)}.$
$C_1^{(11,12)}=C_2^{(21,22)}=J.$
$C_1^{(11,21)}=C_1^{(11,22)}=C_2^{(11,22)}=C_2^{(12,22)}=W^{(12,22)}.$
$C_1^{(12,21)}=C_1^{(12,22)}=C_2^{(11,21)}=C_2^{(12,21)}=W^{(12,21)}.$
$C_1^{(21,22)}=C_2^{(11,12)}=W^{(11,12)}=W^{(21,22)}.$
($f^{(i_1 j_1,\dots)}$ denotes the function obtained from f by deleting the j_1th component of the i_1th argument, etc. in the definition of f.)

of the value of $C_1^{(12)}$ (see line [5] in Table 1) begins with the first component y_1 of the second argument, the head of a headed string derived from $a\backslash A$ must be the first symbol of the second argument in order to satisfy (EQ3).

Based on the discussion above, we summarize the construction of P' as ($P'1$) through ($P'7$) below. In what follows, $id[B]$ is an abbreviation of $c_1[B, EPS]$, which is the identity function (a rule with EPS as its left-hand side is constructed in ($P'7$) below). c_1 and c_2 are headed concatenation operations introduced in Definition 4.5.

($P'1$) If $S\to\varepsilon$ is in P, then add $S\to(\varepsilon,0)$ to P'.

($P'2$) Let R be a rule $A\to f[B,D]$ ($f\neq C_1^{(12)}$) in P.

Case (*i*): The top symbol of a string derived from A is that of B.

(i.1) If $d(B)=2$, then add $a\backslash A\to f'[a\backslash B,D]$ to P', where f' is chosen from the functions in hg as shown in Table 2 in order to satisfy (EQ2) and (EQ3).

(i.2) If $d(B)=1$, then add $a\backslash A\to f'[a\backslash B,D]$ to P' as is the Case (i.1). In order that $(\varepsilon,0)$ will not be derived from any node other than the root in any derivation tree in G',

Table 2
Construction $(P'2)$ through $(P'4)$ of Lemma 4.4.

P	P'	
(i.1) *of* $(P'2)$:		
[1] $A\to C_1[B,D]$	$a\backslash A\to c_1[a\backslash B,D]$	$(a\in T)$
[2] $A\to C_2[B,D]$	$a\backslash A\to c_2[a\backslash B,D]$	$(a\in T)$
[3] $A\to W[B,D]$	$a\backslash A\to w_4[a\backslash B,D]$	$(a\in T)$
[6] $A\to C_1^{(21)}[B,D]$	$a\backslash A\to c_1[a\backslash B,D]$	$(a\in T)^{\blacklozenge}$
[8] $A\to C_2^{(21)}[B,D]$	$a\backslash A\to c_2[a\backslash B,D]$	$(a\in T)^{\blacklozenge}$
[9] $A\to C_2^{(22)}[B,D]$	$a\backslash A\to c_1[a\backslash B,D]$	$(a\in T)^{\blacklozenge +}$
[11] $A\to W^{(21)}[B,D]$	$a\backslash A\to w_4[a\backslash B,D]$	$(a\in T)$
[12] $A\to W^{(22)}[B,D]$	$a\backslash A\to w_3[a\backslash B,D]$	$(a\in T)$
(i.2) *of* $(P'2)$:		
[4] $A\to C_1^{(11)}[B,D]$	$a\backslash A\to c_1[a\backslash B,D]$	$(a\in T)^{+}$
	$a\backslash A\to id[D]$	$(a\in L_G(B))$
[7] $A\to C_2^{(11)}[B,D]$	$a\backslash A\to c_2[a\backslash B,D]$	$(a\in T)$
	$a\backslash A\to id[D]$	$(a\in L_G(B))$
[14] $A\to C_1^{(11,21)}[B,D]$	$a\backslash A\to c_1[a\backslash B,D]$	$(a\in T)^{\blacklozenge +}$
	$a\backslash A\to id[D]$	$(a\in L_G(B))$
[15] $A\to C_1^{(12,21)}[B,D]$	$a\backslash A\to c_2[a\backslash B,D]$	$(a\in T)^{\blacklozenge}$
	$a\backslash A\to id[D]$	$(a\in L_G(B))$
(ii.1) *of* $(P'2)$:		
[10] $A\to W^{(11)}[B,D]$	$a\backslash A\to w_4[B,a\backslash D]$	$(a\in T)^{\blacklozenge}$
(ii.2) *of* $(P'2)$:		
[17] $A\to W^{(11,21)}[B,D]$	$a\backslash A\to w_3[B,a\backslash D]$	$(a\in T)^{\blacklozenge +}$
	$a\backslash A\to id[B]$	$(a\in L_G(D))$
[18] $A\to W^{(11,22)}[B,D]$	$a\backslash A\to w_3[B,a\backslash D]$	$(a\in T)^{\blacklozenge}$
	$a\backslash A\to id[B]$	$(a\in L_G(D))$
$(P'3)$:		
[13] $A\to C_1^{(11,12)}[B,D]$	$a\backslash A\to id[a\backslash B]$	$(a\in T)$
[16] $A\to C_1^{(21,22)}[B,D]$	$a\backslash A\to id[a\backslash B]$	$(a\in T)$
[19] $A\to ID[B]$	$a\backslash A\to id[a\backslash B]$	$(a\in T)$
$(P'4)$:		
[5] $A\to C_1^{(12)}[B,D]$	$a\backslash A\to c_2[a\backslash B,D_{\mathrm{top}}]$	$(a\in T)^{\blacklozenge}$
	$a\backslash A\to id[D_{\mathrm{top}}]$	$(a\in L_G(B))$

Line number $[i]$ corresponds to the one in Table 1, where the function appearing in the right-hand side of the rule in P is defined.

the rule $a\backslash B\to(\varepsilon,0)$ is not directly added to P' but "embedded in" nonterminating rule $a\backslash A\to f'[a\backslash B,D]$. That is, for each $a\in T$, add $a\backslash A\to id[D]$ to P' if $a\in L_G(B)$. (It can easily be decided whether $a\in L_G(B)$ or not, as is the case of cfg's.) Note that, since the position of the head of a headed string derived from $a\backslash A$ is arbitrary by condition (EQ2) if $d(A)=1$, $a\backslash A\to c_2[a\backslash B,D]$ may be added to P' instead of $a\backslash A\to c_1[a\backslash B,D]$ in line [4] in Table 2. In the case that for a rule in P there are more than one rule which may be added to P' since the position of the head of a headed string derived from the nonterminal symbol in the left-hand side is arbitrary by condition (EQ2), one of the rules is shown and marked "+" in Table 2.

Case (ii): The top symbol of a string derived from A is that of D.

(ii.1) If $d(D)=2$, then add $a\backslash A \rightarrow f'[B, a\backslash D]$ to P' (see Table 2). Note that, since the head of a headed string derived from B with $d(B)=1$ is always the first symbol, $a\backslash A \rightarrow c_1[a\backslash D, B]$ may be added to P' instead of $a\backslash A \rightarrow w_4[B, a\backslash D]$ in line [10] of Table 2. In the case that for a rule in P there are more than one rule which may be added to P' since the head of a headed string derived from a nonterminal symbol in the right-hand side is always the first symbol, one of the rules is shown and marked "◆" in Table 2.

(ii.2) If $d(D)=1$, then add rules to P' in the same way as in (i.2).

($P'3$) If a rule $A \rightarrow g[B]$ is in P, then add rules to P' in the same way as in (i.1) of ($P'2$).

($P'4$) If a rule $A \rightarrow c_1^{(12)}[B, D]$ is in P, then add $a\backslash A \rightarrow c_2[a\backslash B, D_{\text{top}}]$ for each $a \in T$ and $a\backslash A \rightarrow id[D_{\text{top}}]$ for each $a \in L_G(B)$ to P'.

($P'5$) For each $A \in N$ with $d(A)=1$,

$$A \rightarrow c_1[[a], a\backslash A] \quad \text{for each } a \in T, \text{ and}$$

$$A \rightarrow (a, 1) \quad \text{for each } a \in L_G(A)$$

are added to P'.

($P'6$) For each $A \in N$ with $d(A)=2$,

$$A \rightarrow c_2[[a], a\backslash A] \quad \text{for each } a \in T, \text{ and}$$

$$A_{\text{top}} \rightarrow c_1[[a], a\backslash A] \quad \text{for each } a \in T$$

are added to P'.

($P'7$) The following rules are also added to P' so that $(\varepsilon, 0)$ can be derived from EPS and $(a, 1)$ can be derived from $[a]$ for each $a \in T$:

$$EPS \rightarrow (\varepsilon, 0),$$

$$[a] \rightarrow (a, 1) \quad \text{for each } a \in T.$$

It can be shown by induction on the height of derivation trees that condition EQ is satisfied. We can conclude that $L(G)$ and the underlying language generated by G' are the same by using the fact that EQ is true. Details of the proof are described in Appendix.

By Theorem 4.9 and Lemma 4.10, the following theorem is obtained.

Theorem 4.11. $HL = MHL = TAL = LIL = CCL$.

By Theorem 4.11 and the fact that TAL is a substitution-closed full AFL, the next corollary is obtained.

Corollary 4.12. *HL is a substitution-closed full* AFL.

In the proof of Lemma 4.10, G' is an nhg and uses only w_3 and w_4 as head-wrapping operations. Consequently, for a given mhg G, a right-wrapping hg G' can be constructed such that the underlying language generated by G' is the same as $L(G)$. Similarly, for a given mhg G, a left-wrapping hg G'' can be constructed such that the underlying language generated by G'' is the same as $L(G)$ (in this case, for each $A \in N$ and $a \in T$, a nonterminal symbol A/a is introduced from which every headed string obtained from some string $\bar{\alpha}$ derived from A in G by deleting the last symbol of the last component of $\bar{\alpha}$ is derived.) Therefore, the following corollary holds.

Corollary 4.13. $LHL = RHL = HL$.

4.3. Proper inclusion of HL in 2-MCFL

In this section, we show that HL is properly included in $2\text{-}MCFL$. Consider the following language.

$$\mathrm{RESP} = \{a_1^m a_2^m b_1^n b_2^n c_1^m c_2^m d_1^n d_2^n \mid m, n \geq 0\}.$$

RESP is a 2-mcfl [24, p. 110]. Weir conjectures that RESP is not an mhl, but a proof has not yet been given. Vijay-Shanker [20, Theorem 4.7] proved a pumping lemma for tal's (mhl's). As pointed out by [24, p. 110], however, the lemma is not strong enough to show that RESP is not an mhl. In what follows, another pumping lemma (Lemma 4.14) for mhl's is given, and it is shown by using the lemma that RESP is not an mhl which concludes by Theorem 4.11 that $HL = TAL = MHL \subsetneq 2\text{-}MCFL$.

For a string $\alpha \in T^*$ and a symbol $a \in T$, let $\nu_a(\alpha)$ denote the number of the occurrences of a in α.

Lemma. 4.14 (pumping lemma for mhl's). *Let L be an mhl. Assume that, for a given $n \geq 0$, there exists α in L such that $\nu_a(\alpha) \geq n$ for each $a \in T$. Then, there exists a constant $M \geq 0$, depending only on L, such that for any $n \geq 0$ there exists z in L satisfying the following conditions* (1) *and* (2):

(1) *For each $a \in T$, $\nu_a(z) \geq n$, and*

(2) *z may be written as $z = u_1 x_1 w_1 s_1 u_2 x_2 w_2 s_2 u_3$ such that*

  (a) $|x_1 s_1 x_2 s_2| \geq 1$,

  (b) $|u_2| \leq M$, *and*

  (c) *for all $i \geq 0$, $u_1 x_1^i w_1 s_1^i u_2 x_2^i w_2 s_2^i u_3$ is in L.*

(Note that this lemma is similar to the pumping lemma for 2-mcfl's derived by letting $m = 2$ in Lemma 3.2 in that z may be written as $z = u_1 x_1 w_1 s_1 u_2 x_2 w_2 s_2 u_3$ with

$|x_1 s_1 x_2 s_2| \geqslant 1$ such that x_1, s_1, x_2 and s_2 can be arbitrarily pumped. Lemma 4.14 is stronger than Lemma 3.2 ($m=2$) in that z is shown to be divided into $u_1, x_1, w_1, s_1, u_2, x_2, w_2, s_2$ and u_3 in such a way that the length of the substring u_2 intervening between s_1 and x_2 is not greater than a constant M.)

Proof. Let L be an mhl satisfying the assumption of the lemma. Let G_0 be an mhg satisfying $L(G_0)=L$, and $G=(N,O,F,P,S)$ be a 2-mcfg constructed in Lemma 2.2 which is weakly equivalent to G_0. By the construction of G, property DRV mentioned in Section 4.2 holds for G. In what follows, we consider G. Let n be a nonnegative integer. By the assumption, there exists z_1 in L satisfying that (1) $\nu_a(z_1) \geqslant n$ for each $a \in T$ and (2) $|z_1| \geqslant 2^{|N|+1}$. Let t be a derivation tree of z_1. There exists a path p from the root r to a leaf in t such that the number of the nodes on p which has two children is at least $\log_2 |z_1| = |N|+1$ by the assumption $|z_1| \geqslant 2^{|N|+1}$. Therefore, there exist distinct nodes v and v' on p with a same label (say, $A \in N$) which have two children. The proof is similar to that of Lemma 3.2. The difference is that we must construct a path in t in such a way that $|u_2|$ is not greater than some constant depending only on L.

In what follows, we evaluate the length of u_2. If $d(A)=1$, then z can be divided with $|u_2|=0$. Assume that $d(A)=2$ and let $v_1 v_2 \ldots v_m$ be the path from r to v' ($v_1 = r$ and $v_m = v'$). By property DRV, for each h ($1 \leqslant h \leqslant m$), g_{v_{h+1}, v_h} has the following form:

$$g_{v_{h+1},v_h}[y] \quad : \quad (\gamma_{10}^{(h)} y \gamma_{11}^{(h)}, \gamma_2^{(h)}) \tag{4.1}$$

$$: \quad (\gamma_1^{(h)}, \gamma_{20}^{(h)} y \gamma_{21}^{(h)}) \tag{4.2}$$

$$: \quad \gamma_{10}^{(h)} y \gamma_{11}^{(h)} \tag{4.3}$$

$$g_{v_{h+1},v_h}[(y_1,y_2)] \quad : \quad (y_1 y_2 \gamma_1^{(h)}, \gamma_2^{(h)}) \tag{4.4}$$

$$: \quad (\gamma_1^{(h)}, \gamma_2^{(h)} y_1 y_2) \tag{4.5}$$

$$: \quad (\gamma_{10}^{(h)} y_1 \gamma_{11}^{(h)}, \gamma_{20}^{(h)} y_2 \gamma_{21}^{(h)}) \tag{4.6}$$

$$: \quad \gamma_{10}^{(h)} y_1 y_2 \gamma_{11}^{(h)} \tag{4.7}$$

$$(\gamma_{10}^{(h)}, \gamma_{11}^{(h)}, \ldots \in T^*)$$

Therefore, the length of u_2 is the sum of $|\gamma_{11}^{(h)} \gamma_{20}^{(h)}|$ for $\gamma_{11}^{(h)}$ and $\gamma_{20}^{(h)}$ in (4.6) for each h ($1 \leqslant h \leqslant m$). On the other hand, $|\gamma_{11}^{(h)} \gamma_{20}^{(h)}|$ is positive for $\gamma_{11}^{(h)}$ and $\gamma_{20}^{(h)}$ in (4.6) only if the function appearing in the right-hand side of the applied rule at v_h is W, $W^{(21)}$ or $W^{(22)}$ and v_{h+1} is the first (left) child of v_h (see Table 1). Let such v_h's be $v_{i_1}, v_{i_2}, \ldots, v_{i_d}$ in the order from r to v', and let $l(v)$ denote the sum of the lengths of the components of the strings derived from the second (right) child of v; then

$$|u_2| = \sum_{j=1}^{d} l(v_{i_j}).$$

In order to make $|u_2|$ not greater than some constant depending only on L, we choose a path p from the root r to a leaf in such a way that if the function appearing in the right-hand side of the applied rule at v is W, $W^{(21)}$ or $W^{(22)}$, we let the next node be the second child of v (if possible) in the following way. Let k denote $|N|$.

Let p be a path from the root r to a leaf in t such that the number of the nodes on p which have two children is at least $k+1$ and p satisfies the following conditions (such a path always exists in t):

> Let v be a node on p which has two children, and v_1 and v_2 be the first and the second children of v, respectively. Let j denote the number of the nodes which are in the sequence of nodes from r to v and have two children. If there exists a path from v_2 to a leaf such that the number of the nodes on the path which have two children is $k+1-j$ or more, then the next node to v on p is v_2, and v_1 otherwise.

By the definition of p mentioned above, $l(v_{i_j}) \leqslant 2^{k-j}$. If we choose a pair v, v' of nodes having identical labels which have two children in such a way that v' is nearest to the root r among such pairs, then $d \leqslant k-1$ holds. Therefore,

$$|u_2| = \sum_{j=1}^{d} l(v_{i_j}) \leqslant \sum_{j=1}^{k-1} 2^{k-j} = 2^k - 2.$$

By the definition of z_1 and $z, v_a(z) \geqslant v_a(z_1) \geqslant n$ for each n. Let M be 2^k-2. This completes the proof. □

Both Theorem 4.7 of [20] and Lemma 4.14 of our paper state that z may be written as $z = u_1x_1w_1s_1u_2x_2w_2s_2u_3$ with $|x_1s_1x_2s_2| \geqslant 1$ such that x_1, s_1, x_2 and s_2 can be arbitrarily pumped. The difference between the two lemmas is as follows. Lemma 4.14 states that z may be written as $z = u_1x_1w_1s_1u_2x_2w_2s_2u_3$ in such a way that only the substring u_2 of length not greater than a constant M can intervene between s_1 and x_2. On the other hand, Theorem 4.7 of [20] states that z may be written as $z = u_1x_1w_1s_1u_2x_2w_2s_2u_3$ in such a way that the sum of the lengths of x_1, w_1, s_1, x_2, w_2 and s_2 is not greater than a constant N.

Lemma 4.15. RESP$\notin MHL$.

Proof. Suppose that RESP$\in MHL$ and let M be the constant in Lemma 4.14. Let

$$z = a_1^q a_2^q b_1^r b_2^r c_1^q c_2^q d_1^r d_2^r \quad (q, r > M/2),$$

and divide z as

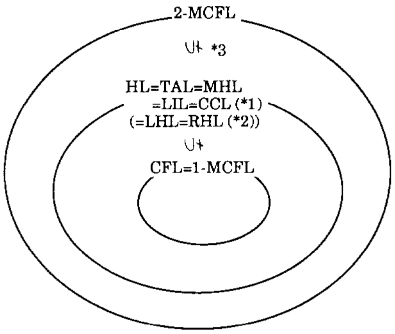
$$z = u_1x_1w_1s_1u_2x_2w_2s_2u_3.$$

The condition "$|x_1s_1x_2s_2| \geqslant 1$ and $u_1x_1^iw_1s_1^iu_2x_2^iw_2s_2^iu_3 \in$RESP for all $i \geqslant 0$" holds only if

(a) $x_1 = a_1^j, s_1 = a_2^j, x_2 = c_1^j, s_2 = c_2^j (1 \leqslant j \leqslant q)$, or

(b) $x_1 = b_1^k, s_1 = b_2^k, x_2 = d_1^k, s_2 = d_2^k (1 \leqslant k \leqslant r)$.

However, neither of (a) and (b) satisfies $|u_2| \leqslant M$. □



*1 Theorem 4.5
*2 Corollary 4.7
*3 Lemma 4.9

Fig. 3. Results of this paper on inclusion relations between subclasses of 2-MCFL.

By Theorems 4.9 and 4.11 and Lemma 4.15, the following inclusion relations hold (see Fig. 3).

Corollary 4.16. $CFL=1\text{-}MCFL\subsetneq HL=MHL=TAL=LIL=CCL\subsetneq 2\text{-}MCFL$.

5. Conclusion

In this paper, is has been shown that the generative capacity of hg's is equivalent to that of tag's and weaker than that of 2-mcfg's, and that the class of head languages is a substitution-closed full AFL. Recently, the authors have developed an Earley-type parsing algorithm for mcfl's. Details of the algorithm is described in [10].

Appendix. Proof of Lemma 4.10

We will show that 2-mcfg $G=(N,O,F,P,S)$ and hg $G'=(N',T\uparrow,F',P',S)$ constructed in the proof of Lemma 4.10 satisfy condition EQ and that $L(G)$ is the same as the underlying language generated by G'. For a derivation tree t, the maximum length of paths in t from the root of t to the leaves is called the height of t.

A.1. Inclusion of $L(G)$ in the underlying language generated by G'

Lemma A.1. *G and G' constructed in Lemma* 4.10 *satisfy the following conditions* (*the "only if" part of* (EQ2) *and* (EQ3), *respectively*):

(1) *For each $A\in N$ with $d(A)=1$, $a\in T$ and $\alpha\in T^+(|\alpha|>1)$,*

$$\alpha\in L_G(A) \text{ and } \alpha(1,1)=a \;\Rightarrow\; \exists j(1\leqslant j\leqslant|\alpha|-1):(\alpha(2,|\alpha|),j)\in L_{G'}(a\backslash A).$$

(2) *For each* $A \in N$ *with* $d(A)=2, a \in T$ *and* $\alpha_1, \alpha_2 \in T^+$,

$$(\alpha_1, \alpha_2) \in L_G(A) \text{ and } \alpha(1,1)=a \;\Rightarrow\; (\alpha_1(2,|\alpha_1|)\alpha_2, |\alpha_1|) \in L_{G'}(a\backslash A).$$

Proof. The lemma is proved by induction on the height of a derivation tree in G rooted at a node labeled with A.

{*Basis* (*height* 2)}

Case (*i*): Assume that α ($|\alpha| > 1$) is in $L_G(A)$ and let t be a derivation tree of α rooted at a node labeled with A. The height of t is 2 only if the applied rule at the root r is either $A \to C_1^{(11,21)}[B, D]$ or $A \to W^{(11,21)}[B, D]$ and the applied rule at the first (second) child of r is $B \to b$ ($D \to d$) for some $b \in T$ ($d \in T$). If the applied rule at r is $A \to C_1^{(11,21)}[B, D]$, then $\alpha = bd$ and $\alpha(1,1)=b$. $b\backslash A \to id[D]$ is in P' by construction (i.2) of ($P'2$) (see line [14] in Table 2) since b is in $L_G(B)$. On the other hand, $D \to (d, 1)$ is in P' by construction ($P'5$) since d is in $L_G(D)$. Therefore,

$$(\alpha(2,|\alpha|), 1) = (d, 1) \text{ is in } L_{G'}(b\backslash A).$$

The proof is similar to the one in the case that the applied rule at r is $A \to W^{(11,12)}[B, D]$.

Case (*ii*): Assume that (α_1, α_2) is in $L_G(A)$. The height of a derivation tree t of (α_1, α_2) is 2 only if the applied rule at the root is either $A \to C_1^{(12,21)}[B, D]$ or $A \to W^{(11,22)}[B, D]$. The proof is analogous to Case (i).

{*Inductive step* (*height greater than* 2)}

Let k be an integer greater than 2 and suppose that the lemma is true for derivation trees of height $k-1$ or less. Let t be a derivation tree in G of height k, and r, v_1 and v_2 be the root of t, the first child of r and the second child of r (if it exists), respectively. Let R be the rule applied at r.

Case (*i*): R is a rule mentioned in (i.1) of ($P'2$) in the construction of P' in Lemma 4.10. For example, let R be $A \to W[B, D]$. By the definition of W, $d(A)=d(B)=d(D)=2$. Let $(\alpha_1, \alpha_2) \in L_G(A)$, $(\beta_1, \beta_2) \in L_G(B)$ and $(\delta_1, \delta_2) \in L_G(D)$ be strings derived from r, v_1 and v_2, respectively; then $(\alpha_1, \alpha_2) = (\beta_1\delta_1, \delta_2\beta_2)$. $\alpha_1, \alpha_2, \beta_1, \beta_2, \delta_1, \delta_2 \in T^+$ since G satisfies conditions (N3) and (N4) of Lemma 2.2. Let $\beta_1(1,1)=a$ and $\delta_1(1,1)=d$. The height of the subtrees of t rooted at v_1 and v_2 is not greater than $k-1$. By the inductive hypothesis,

$$(\beta_1(2,|\beta_1|)\beta_2, |\beta_1|) \in L_{G'}(a\backslash B), \text{ and}$$

$$(\delta_1(2,|\delta_1|)\delta_2, |\delta_1|) \in L_{G'}(d\backslash D).$$

Since $D \to c_2[[d], d\backslash D]$ is in P' by construction ($P'6$),

$$(\delta_1\delta_2, |\delta_1|+1) \in L_{G'}(D).$$

$A \to W[B, D] \in P$ implies $a \backslash A \to w_4[a \backslash B, D] \in P'$ by (i.1) of $(P'2)$ (see line [3] in Table 2). Hence,

$$(\alpha_1(2, |\alpha_1|)\alpha_2, |\alpha_1|) = (\beta_1(2, |\beta_1|)\delta_1\delta_2\beta_2, |\beta_1| + |\delta_1|) \in L_{G'}(a \backslash A).$$

The proofs are analogous in the other cases.

Case (*ii*): R is a rule mentioned in (i.2) of $(P'2)$. For example, let R be $A \to C_1^{(11)}[B, D]$. By the definition of $C_1^{(11)}$, $d(A) = d(B) = 1$ and $d(D) = 2$. Let $\alpha \in L_G(A)$, $\beta \in L_G(B)$ and $(\delta_1, \delta_2) \in L_G(D)$ be strings derived from r, v_1 and v_2, respectively; then $\alpha = \beta\delta_1\delta_2$. By conditions (N3) and (N4), α, β, δ_1, $\delta_2 \in T^+$. Let $\beta_1(1, 1) = a$ and $\delta_1(1, 1) = d$. The height of the subtree of t rooted at v_2 is not greater than $k-1$. By the inductive hypothesis,

$$(\delta_1(2, |\delta_1|)\delta_2, |\delta_1|) \in L_{G'}(d \backslash D).$$

Since $D \to c_2[[d], d \backslash D]$ is in P' by construction $(P'6)$,

$$(\delta_1\delta_2, |\delta_1| + 1) \in L_{G'}(D).$$

There are two subcases:

(1) Assume that $|\beta| = 1$. Then $\beta = a$. Since $A \to C_1^{(11)}[B, D] \in P$ and $a \in L_G(B)$, $a \backslash A \to id[D] \in P'$, i.e. $a \backslash A \to c_1[D, EPS] \in P'$ by (i.2) of $(P'2)$ (see line [4] in Table 2) and $EPS \to (\varepsilon, 0) \in P'$. Therefore,

$$(\alpha(2, |\alpha|), |\delta_1| + 1) = (\delta_1\delta_2, |\delta_1| + 1) \in L_{G'}(a \backslash A).$$

(2) Assume that $|\beta| > 1$. The height of the subtree rooted at v_1 is not greater than $k-1$. By the inductive hypothesis,

$$\exists j (1 \leqslant j \leqslant |\beta| - 1) : (\beta(2, |\beta|), j) \in L_{G'}(a \backslash B), \text{ where } \beta(2, |\beta|) \in T^+ \text{ since } |\beta| > 1.$$

$A \to C_1^{(11)}[B, D] \in P$ implies $a \backslash A \to c_1[a \backslash B, D] \in P'$ by (i.2) of $(P'2)$ (see line [4] in Table 2). Hence,

$$(\alpha(2, |\alpha|), j) = (\beta(2, |\beta|)\delta_1\delta_2, j) \in L_{G'}(a \backslash A).$$

In both subcases (1) and (2),

$$\exists j (1 \leqslant j \leqslant |\alpha| - 1) : (\alpha(2, |\alpha|), j) \in L_{G'}(a \backslash A).$$

The proofs are analogous in the other cases.

Case (*iii*): R is a rule mentioned in (ii) of $(P'2)$ or $(P'3)$. The proof is analogous to Case (i) or (ii).

Case (*iv*): R is a rule mentioned in $(P'4)$, i.e. $A \to C_1^{(12)}[B, D]$. By the definition of $C_1^{(12)}$, $d(A) = 2$, $d(B) = 1$, $d(D) = 2$. Let $(\alpha_1, \alpha_2) \in L_G(A)$, $\beta \in L_G(B)$ and $(\delta_1, \delta_2) \in L_G(D)$ be strings derived from r, v_1 and v_2, respectively; then $(\alpha_1, \alpha_2) = (\beta, \delta_1\delta_2)$. By condition (N3) and (N4), α_1, α_2, β, δ_1, $\delta_2 \in T^+$. Let $\beta(1, 1) = a$ and $\delta_1(1, 1) = d$. The height of the subtree of t rooted at v_2 is not greater than $k-1$. By the inductive hypothesis,

$$(\delta_1(2, |\delta_1|)\delta_2, |\delta_1|) \in L_{G'}(d \backslash D).$$

Since $D_{\text{top}} \to c_1[[d], d\backslash D]$ is in P' by construction ($P'6$),

$$(\delta_1\delta_2, 1) \in L_{G'}(D_{\text{top}}).$$

There are two subcases:

(1) Assume that $|\beta| = 1$. Then $\beta = \alpha_1 = a$. Since $A \to C_1^{(12)}[B, D] \in P$ and $a \in L_G(B)$, $a\backslash A \to id[D_{\text{top}}] \in P'$ by ($P'4$) (see line [5] in Table 2). $|\alpha_1| = 1$ implies $\alpha_1(2, |\alpha_1|) = \varepsilon$. Therefore,

$$(\alpha_1(2, |\alpha_1|)\alpha_2, |\alpha_1|) = (\alpha_2, 1) = (\delta_1\delta_2, 1) \in L_{G'}(a\backslash A).$$

(2) Assume that $|\beta| > 1$. The height of the subtree rooted at v_1 is not greater than $k-1$. By the inductive hypothesis,

$$\exists j (1 \leqslant j \leqslant |\beta| - 1): (\beta(2, |\beta|), j) \in L_{G'}(a\backslash B), \text{ where } \beta(2, |\beta|) \in T^+ \text{ since } |\beta| > 1.$$

$A \to C_1^{(12)}[B, D] \in P$ implies $a\backslash A \to c_2[a\backslash B, D_{\text{top}}] \in P'$ by ($P'4$) (see line [5] in Table 2). Hence,

$$(\alpha_1(2, |\alpha_1|)\alpha_2, |\alpha_1|) = (\beta(2, |\beta|)\delta_1\delta_2, |\beta|) \in L_{G'}(a\backslash A).$$

In both subcases (1) and (2),

$$(\alpha_1(2, |\alpha_1|)\alpha_2, |\alpha_1|) \in L_{G'}(a\backslash A).$$

This completes the proof. □

Let α be in $L_G(S)$. If $|\alpha| > 1$, there exists j $(1 \leqslant j \leqslant |\alpha| - 1)$ such that $(\alpha(2, |\alpha|), j) \in L_{G'}(a\backslash S)$ $(\alpha(1, 1) = a)$ by Lemma A.1 above. Therefore $(\alpha, 1) \in L_{G'}(S)$ since $S \to c_1[[a], a\backslash S] \in P'$ by construction ($P'5$). If $|\alpha| = 1$, then $S \to (\alpha, 1) \in P'$ by ($P'5$). If $\varepsilon \in L(G)$, then $S \to \varepsilon \in P$ by condition (N3). It follows that $S \to (\varepsilon, 0) \in P'$ by ($P'1$), which implies $(\varepsilon, 0) \in L_{G'}(S)$. (That is, the "only if" part of (EQ1) holds.) Therefore, for each α in $L(G)$, there exists $i \geqslant 0$ such that (α, i) is in $L_{G'}(S)$. Hence, $L(G)$ is included in the underlying language generated by G'.

A.2. Inclusion of the underlying language generated by G' in $L(G)$

Lemma A.2. *For each $A \in N, a \in T, \alpha \in T^+$ and $i \geqslant 1$, the conditions* (1) *and* (2) *(the "if" part of* (EQ2) *and* (EQ3), *respectively) hold*:

(1) *$(\alpha, i) \in L_{G'}(a\backslash A)$ and $d(A) = 1$ in $G \Rightarrow a\alpha \in L_G(A)$.*

(2) *$(\alpha, i) \in L_{G'}(a\backslash A)$ and $d(A) = 2$ in $G \Rightarrow (a\alpha(1, i-1), \alpha(i, |\alpha|)) \in L_G(A)$.*

Proof. The lemma is proved by induction on the height of a derivation tree in G' rooted at the node labeled with $a\backslash A$ $(a \in T, A \in N)$.

{Basis (height 2)}.

Let (α, i) be in $L_{G'}(a\backslash A)$ $(\alpha \in T^+)$ and let t be a derivation tree of (α, i) rooted at a node labeled with $a\backslash A$. The height of t is 2 only if the applied rule at the root r of t is

$a\backslash A \to id[B]$ (which is the abbreviation of $a\backslash A \to c_1[B, EPS]$) for some $B \in N$ and the applied rule of the first child of r is $B \to (\alpha, 1)$. Therefore, $|\alpha| = 1$, $i = 1$, $\alpha \in L_G(B)$ and $d(B) = 1$ [by construction $(P'5)$].

If $d(A) = 1$, then $a \in L_G(D)$ and "$A \to C_1^{(11,21)}[D, B] \in P$ or $A \to W^{(11,21)}[B, D] \in P$" by construction $(P'2)$, which implies $a\alpha \in L_G(A)$.

If $d(A) = 2$, then $a \in L_G(D)$ and "$A \to C_1^{(12,21)}[D, B] \in P$ or $A \to W^{(11,22)}[B, D] \in P$" by construction $(P'2)$, which implies $(a, \alpha) \in L_G(A)$.

{*Inductive step* (*height greater than* 2)}.

Let k be an integer greater than 2 and suppose that the lemma is true for derivation trees of height $k-1$ or less. Let (α, i) be in $L_{G'}(a\backslash A)$ $(\alpha \in T^+)$ and let t be a derivation tree of (α, i) of height k rooted at a node labeled with $a\backslash A$. Let r, v_1 and v_2 be the root of t, the first child of r and the second child of r (if it exists), respectively, and let Y and Z be the labels of v_1 and v_2, respectively (if the function appearing in the right-hand side of the applied rule R at r is id, then only Y is considered). Let $(\beta, i_1) \in L_{G'}(Y)$ and $(\delta, i_2) \in L_{G'}(Z)$ (if v_2 exists) be the headed strings derived from v_1 and v_2, respectively. $(\varepsilon, 0)$ is not derived in G' from any nonterminal symbol other than S and EPS in G', and if $(\varepsilon, 0)$ is derived from S in G', then S does not appear in the right-hand side of any rule. Furthermore, EPS does not appear in the right-hand side of any rule other than rules of the form $E \to id[F]$ (i.e. $E \to c_1[F, EPS]$) with $E, F \in N'$. Hence, β and δ are in T^+.

Case (*i*): R is constructed in $(P'2)$ or $(P'3)$, and the function appearing in its right-hand side is not id. For example, let R be $a\backslash A \to w_4[a\backslash B, D]$ with $d(A) = d(B) = d(D) = 2$. Then, $Y = a\backslash B$, $Z = D$ and $(\alpha, i) = (\beta(1, i_1 - 1)\delta\beta(i_1, |\beta|), i_1 + i_2 - 1)$. By construction $(P'6)$, for each $d \in T, D \to c_2[[d], d\backslash D] \in P'$, and there exists no other rule in G' with D as its left-hand side. Hence, let $d = \delta(1, 1)$ and we obtain

$$(\delta(2, |\delta|), i_2 - 1) \in L_{G'}(d\backslash D) \quad \text{with} \quad \delta(2, |\delta|) \in T^+.$$

The height of the subtrees rooted at v_1 and the second child of v_2 is not greater than $k-1$. By the inductive hypothesis,

$$(\alpha\beta(1, i_1 - 1), \beta(i_1, |\beta|)) \in L_G(B), \text{ and}$$

$$(\delta(1, i_2 - 1), \delta(i_2, |\delta|)) \in L_G(D).$$

Since $R \in P'$ and $d(A) = d(B) = d(D) = 2$, $A \to W[B, D] \in P$ by (i.1) of $(P'2)$ (see line [3] in Table 2). Hence,

$$(a\alpha(1, i-1), \alpha(i, |\alpha|)) = (a\beta(1, i_1 - 1)\delta(1, i_2 - 1), \delta(i_2, |\delta|)\,\beta(i_1, |\beta|)) \in L_G(A).$$

The proofs are analogous in the other cases.

Case (*ii*): R is constructed in $(P'4)$, and the function appearing in its right-hand side is not id. Then R is a form of

$$a\backslash A \to c_2[a\backslash B, D_{\text{top}}] \quad \text{with} \quad d(A) = 2,\ d(B) = 1,\ d(D) = 2.$$

$Y=a\backslash B$, $Z=D_{\text{top}}$ and $(\alpha,i)=(\beta\delta,|\beta|+i_2)$. By $(P'6)$, for each $d\in T, D_{\text{top}}\to c_1[[d],d\backslash D]\in P'$, and there exists no other rule in G' with D_{top} as its left-hand side. Hence, $i_2=1$, $i=|\beta|+1$ and there exists $j\geqslant 1$ such that

$$(\delta(2,|\delta|),j)\in L_{G'}(d\backslash D),\text{ where }\delta(2,|\delta|)\in T^+\text{ and }d=\delta(1,1).$$

The height of the subtrees rooted at v_1 and the second child of v_2 is not greater than $k-1$. By the inductive hypothesis,

$$a\beta\in L_G(B),\text{ and}$$

$$(\delta(1,j),\delta(j+1,|\delta|))\in L_G(D).$$

Since $R\in P', A\to C_1^{(12)}[B,D]\in P$ by $(P'4)$ (see line [5] in Table 2). Hence,

$$(a\alpha(1,i-1),\alpha(i,|\alpha|))=(a\beta,\delta)\in L_G(A).$$

Case (*iii*): R is constructed in $(P'2)$, $(P'3)$ or $(P'4)$, and the function appearing in its right-hand side is *id*. For example, let R be $a\backslash A\to id[B]$ with $d(A)=d(B)=1$. Then, $Y=B$ and $(\alpha,i)=(\beta,i_1)$. If $|\beta|=1$, then it is one of the basis cases of the induction. Assume that $|\beta|>1$ and $\beta(1,1)=b$. Since $d(B)=1$, the only rules with B as their left-hand sides are of the form $B\to c_1[[b],b\backslash B]$ $(b\in T)$ and $B\to(b,1)$ $(b\in L_G(B))$ constructed in $(P'5)$. Since $|\beta|>1$, the applied rule at v_1 is $B\to c_1[[b],b\backslash B]$, which implies that there exists $j>0$ such that

$$(\beta(2,|\beta|),j)\in L_{G'}(b\backslash B),\text{ where }\beta(2,|\beta|)\in T^+.$$

The height of the subtree rooted at the second child of v_1 is not greater than $k-1$. By the inductive hypothesis,

$$\beta=L_G(B).$$

"$A\to C_1^{(11,21)}[D,B]\in P$ or $A\to W^{(11,21)}[B,D]\in P$" and $a\in L_G(D)$ by the construction (see Table 2). Hence,

$$a\alpha=a\beta\in L_G(A).$$

The proofs are analogous in the other cases. □

Here we make the following observations:

(a) $S\to(\varepsilon,0)\in P'$ only if $S\to\varepsilon\in P$. Therefore, $(\varepsilon,0)\in L(G')$ implies $\varepsilon\in L_G(S)$. (That is, "if" part of (EQ1) holds.)

(b) The only rules in P' with S as their left-hand sides other than $S\to(\varepsilon,0)$ are (i) $S\to(a,1)$ for each $a\in L_G(S)$ and (ii) $S\to c_1[[a],a\backslash S]$ for each $a\in T$. By (a) and (b), for each $a\in T^*$ and $i\geqslant 0$, $(\alpha,i)\in L_{G'}(S)$ implies $\alpha\in L_G(S)$ [by Lemma A.2 in the case (ii) of (b)].

Hence, the underlying language generated by G' is included in $L(G)$.

By Sections A.1 and A.2, the underlying language generated by G' is the same as $L(G)$, which implies that Lemma 4.10 holds.

Acknowledgment

The authors thank Associate Professor Takao Gunji of Faculty of Language and Culture, Osaka University, who informed us of Pollard's works.

References

[1] G. Gazdar, Applicability of indexed grammars to natural languages, Tech. Report, CSLI-85-34, Center for Study of Language and Information, 1985.

[2] J.E. Hopcroft and J.D. Ullman, *Introduction to Automata Theory, Languages and Computation* (Addison-Wesley, Reading, MA, 1979).

[3] A.K. Joshi, Tree adjoining grammars: How much context-sensitivity is required to provide reasonable structural descriptions, in: D. Dowty, L. Karttunen and A. Zwicky, eds., *Natural Language Parsing: Psychological, Computational and Theoretical Perspectives* (Cambridge University Press, 1985) 206–250.

[4] A.K. Joshi, Private communication, 1988.

[5] A.K. Joshi, L. Levy and M. Takahashi, Tree adjunct grammars, *J. Comput. System Sci.* **10**(1) (1975) 136–163.

[6] T. Kasami, H. Seki and M. Fujii, Generalized context-free grammars, multiple context-free grammars and head grammars, Tech. Report, Osaka University, 1987; also in: *Preprint of WG on Natural Language of IPSJ* **87-NL**-63-1 (1987).

[7] T. Kasami, H. Seki and M. Fujii, Generalized context-free grammars and multiple context-free grammars, *Trans. of IEICE* **J71-D**(5) (1988) (in Japanese) 758–765; English translation: *Systems Comput. Japan* **20**(7) (1989) 43–52.

[8] T. Kasami, H. Seki and M. Fujii, On the membership problem for head grammars and multiple context-free grammars, *Trans. of IEICE* **J71-D**(6) (1988) (in Japanese) 935–941; English translation will appear in: *Systems and Computers in Japan* (Scripta Technica).

[9] T. Kasami, N. Tokura and K. Taniguchi, *Formal Language Theory* (Korona-sha, Tokyo, 1988) (in Japanese) 276–277.

[10] T. Matsumura, H. Seki, M. Fujii ad T. Kasami, Some results on multiple context-free grammars, in: *Paper of Technical Group*, **COMP88**-78, *IEICE* (1989) (in Japanese).

[11] T. Matsumura, H. Seki, M. Fujii and T. Kasami, On the generative capacity of head grammars, tree adjoining grammars and multiple context-free grammars, Tech. Report, Osaka University, 1989.

[12] C.J. Pollard, Generalized phrase structure grammars, head grammars, and natural language, Ph.D. dissertation, Stanford University, 1984.

[13] G.K. Pullum and G. Gazdar, Natural languages and context-free languages, *Linguistics and Philosophy* **4** (1982) 471–504.

[14] K. Roach, Formal properties of head grammars, in: A. Manaster-Ramer, ed., *Mathematics of Language* (John Benjamins, Amsterdam, 1987).

[15] A. Salomma, *Formal Languages* (Academic Press, New York, 1973).

[16] S. Skyum, Parallel context-free languages, *Inform. and Control* **26** (1974) 280–285.

[17] M.J. Steedman, Combinators and grammars, in: R. Oehrle, E. Bach and D. Wheeler, eds., *Categorial Grammars and Natural Language Structures* (Foris, Dordrecht, 1986).

[18] M.J. Steedman, Combinatory grammars and parasitic gaps, in: *Natural Language and Linguistic Theory* (1987).

[19] K. Torii and M. Arisawa, Phrase structure languages by a concurrent derivation, *Trans. of IECE Japan* **54-C** (2) (1971) (in Japanese) 124–131.

[20] K. Vijay-Shanker, A study of tree adjoining grammars, Ph.D. dissertation, University of Pennsylvania, 1987.

[21] K. Vijay-Shanker and A.K. Joshi, Some computational properties of tree adjoining grammars, in: *Proc. 23rd meeting of Assoc. Comput. Ling.* (1985) 82–93.

[22] K. Vijay-Shanker, D.J. Weir and A.K. Joshi, Tree adjoining and head wrapping, in: *Proc. 11th Internat. Conf. on Comput. Ling.* (1986) 202–207.

[23] K. Vijay-Shanker, D.J. Weir and A.K. Joshi, Characterizing structural descriptions produced by various grammatical formalisms, in: *Proc. 25th meeting of Assoc. Comput. Ling.* (1987) 104–111.

[24] D.J. Weir, Characterizing mildly context-sensitive grammar formalisms, Ph.D. dissertation, University of Pennsylvania, 1988.

[25] D.J. Weir and A.K. Joshi, Combinatory categorial grammars: Generative power and relationship to linear context-free rewriting systems, in: *Proc. 26th meeting of Assoc. Comput. Ling.* (1988) 278–285.