Generative Capacity and Recognition of Lexical-Functional Grammars

Hiroyuki Seki[†‡], Ryuichi Nakanishi[†], Sachiko Ando[†] and Tadao Kasami[‡†]

[†] Faculty of Engineering Science
Osaka University
Toyonaka, Osaka 560, Japan

[‡] Nara Institute of Science and Technology
Ikoma, Nara 630-01, Japan

E-mail : seki@is.aist-nara.ac.jp

Abstract Lexical-functional grammar (lfg) is a well-known non-transformational phrase-structure grammar for natural languages. In this paper, subclasses of lexical-functional grammars are proposed, and generative capacity and universal-recognition complexity of these subclasses are investigated and compared with those of finite state translation systems (fsts') and parallel multiple context-free grammars.

The fixed-language recognition problem for a language L is the problem of deciding whether w belongs to L for a given string w. On the other hand, for a class $\mathcal{G}$ of grammars, the universal recognition problem for $\mathcal{G}$ is the problem of deciding whether G can generate w for a given grammar $G \in \mathcal{G}$ and a string w. Note that the size of an input is $||G|| + |w|$ where $||G||$ and $|w|$ denote the length of the description of G and that of w, respectively, while the input size is only $|w|$ in the case of fixed-language recognition problem.

In this paper, three subclasses of lfg's called nondeterministically copying lfg's (nc-lfg's), deterministically copying lfg's (dc-lfg's) and right-

linear lfg's (rl-lfg's), which constitute a subclass of dc-lfg's, are proposed and the generative capacity of these subclasses is shown to be equal to that of fsts', deterministic fsts' (d-fsts') and right-linear fsts' (rl-fsts'), respectively. It is known that the languages generated by d-fsts' are (fixed-language) recognizable in polynomial time. Hence, the languages generated by dc-lfg's are also recognizable in polynomial time.

Next, computational complexity of the universal recognition problems for dc-lfg's and for rl-lfg's is investigated. It is shown that the universal recognition problems for dc-lfg's and for rl-lfg's are both EXP-POLY time-complete. It is also shown that the problems for dc-lfg's and rl-lfg's with bounded dimension are both $\mathcal{NP}$-complete. Moreover, it is shown that the problems for dc-lfg's and rl-lfg's with bounded degree are both $\mathcal{P}$-complete. Complexity of the problems for d-fsts' and for rl-fsts' is also investigated.

1. Introduction

Since the early 80's, a number of non-transformational phrase structure grammars have been proposed and extensively studied. Among them are lexical-functional grammars[14], functional unification grammars[18], generalized phrase structure grammars[8] and head-driven phrase structure grammars[23]. These grammars are often called unification-based grammars since rule application is formally defined in terms of unification operations in these grammars.

Lexical-functional grammar (*lfg*)[14] is a well-known unification-based grammar. Various syntactical phenomena, such as control, of natural languages can be predicted by lfg and also there are many natural language processing systems based on lfg. In a derivation tree of an lfg, associated with each node v is a finite set F of attribute-value pairs. F is called the *f-structure* of v. An lfg G consists of a context-free grammar (cfg) G_0 (called the *underlying cfg* of G) and functional schemata P_{fs} which specify productions to be used at derivations in G. A functional schema specifies the relation between the f-structure of a node and the f-structures of its children. Functional schemata are attached to symbols in productions of G_0. It has been shown in [19] that the class of languages generated by lfg's is equal to that of recursively enumerable languages even though the underlying cfg's are restricted to regular grammars. In [8, 14, 20], subclasses of lfg's were proposed in order to guarantee the recursiveness (and/or the efficient recognition) of languages generated by lfg's. However, these classes were shown to generate an $\mathcal{NP}$-complete language[19].

There has been another class of grammatical formalisms, called mildly context-sensitive grammars[10, 33]. Tree adjoining grammars (tag's)[9, 32], head grammars (hg's)[22], linear context-free rewriting systems (lcfrs') [31, 33] and parallel multiple context-free grammars[16, 26] fall into this category. Mildly context-sensitive grammars have common properties: They are natural extensions of cfg's; phrase structures are well-defined; their generative capacities are strictly stronger than that of cfg's, and strictly weaker than that of csg's; the languages generated by them are recognizable in polynomial time.

Parallel multiple context-free grammars (*pmcfg's*) and *multiple context-free grammars* (*mcfg's*), which is a subclass of pmcfg's, were introduced in [16, 26]. Mcfg's is the same formalism as the string based lcfrs' except that an lcfrs is required to satisfy non-erasing condition[31] in its definition while a pmcfg is not. However, it is shown in [16] that non-erasing condition does not weaken the generative capacity of mcfg's. For a nonterminal symbol B of a pmcfg, a positive integer, $d(B)$, is associated with B; B derives $d(B)$-tuples of strings. A production rule of a pmcfg is of the form $A \rightarrow f[A_1, A_2, \cdots, A_n]$ ($A, A_1, A_2, \cdots, A_n$ are nonterminals) where the function f is defined as a total function from $(T^*)^{d(A_1)} \times \cdots \times (T^*)^{d(A_n)}$ to $(T^*)^{d(A)}$ in such a way that each component of the value of f is defined as a concatenation of some components of argument tuples and terminal symbols. The class of languages generated by pmcfg's is properly included in that of context-sensitive languages[16]. The languages generated by pmcfg's were shown to be recognizable in polynomial time[17, 26]. For a pmcfg G, if every function f is defined

in such a way that each component of each argument tuple appears at most once in the function value of f, then G is called a multiple context-free grammar[16]. The class of languages generated by pmcfg's properly includes the class of languages generated by mcfg's (see [16]).

On the other hand, there has been much effort to propose well-defined computational models of transformational grammars. Some of them are defined as extensions of devices which operate on strings, such as generalized sequential machines (gsm's), to devices which operate on trees.

A *tree transducer*[24] takes a tree as an input, starts in the initial state with its head scanning the root node of an input. According to the current state and the label of the scanned node, it transforms an input tree into an output tree in a top-down way. A *finite state translation system* (*fsts*)[24, 29] is a tree transducer with its input domain being the set of derivation trees of a cfg. A subclass of fsts' called *deterministic fsts'* (*d-fsts'*) was introduced in [4]. We propose a subclass of d-fsts' called *right-linear fsts'* (*rl-fsts'*).

In this paper, three subclasses of lfg's called *nc-lfg's*, *dc-lfg's* and *rl-lfg's* are proposed, the languages generated by the last two will be shown to be recognizable in polynomial time. Moreover, this paper clarifies the relations among the generative capacities of fsts' and the above mentioned subclasses of lfg's.

1.1 Subclasses of Lfg's

In *nondeterministically copying lfg's* (*nc-lfg's*), a functional schema either specifies the value of a specific attribute, say atr, immediately ($\uparrow atr = val$) or specifies that the value of a specific attribute of a node v is equal to the whole f-structure of a child node of v ($\uparrow atr = \downarrow$).

A subset $\{p_1, p_2, \cdots, p_k\}$ of productions of an nc-lfg G is called consistent if there exists an f-structure that locally satisfies every functional schema of p_i $(1 \leq i \leq k)$. If a subset of productions are not consistent, then it is called inconsistent. An nc-lfg is called a *deterministically copying lfg* (*dc-lfg*) if for each pair of rules $p_1 : A \rightarrow \alpha_1$ and $p_2 : A \rightarrow \alpha_2$ whose left-hand side is the same, $\{p_1, p_2\}$ is always inconsistent. Intuitively, for derivation trees t_1 and t_2 in a dc-lfg G, if the f-structure and nonterminal of the root of t_1 are the same as those of t_2, then t_1 is always identical with t_2.

If, for each nonterminal A and each attribute atr, A with a functional schema $\uparrow atr = \downarrow$ appears at most once in the right-hand sides of any consistent subset of productions of G, then G is called a *right-linear lfg* (*rl-lfg*). The definition of rl-lfg ensures that for any derivation tree t in an rl-lfg G, the number of nodes whose f-structures must be the same with each other is upperbounded by some constant independent of t.

1.2 Generative Capacity

In Section **4** of this paper, it is shown that the generative capacity of nc-lfg's is equal to that of fsts'. As corollaries, it is shown that the generative capacity of dc-lfg's (resp. rl-lfg's) is equal to that of d-fsts'

(resp. rl-fsts').

Let $(y)\mathcal{L}_{\mathcal{G}}$ denote the (yield) languages generated by a class $\mathcal{G}$ of grammars. Known results and new results on the generative capacities of fsts', lfg's and mildly context-sensitive grammars are summarized in Figure 1. The equalities labeled with $*5$ are new results which are proved in Section **4**.

Modified head grammars (mhg's)[30] is a proper subclass of mcfg's. For the equalities labeled with $*1$, $\mathcal{L}_{hg} \subseteq \mathcal{L}_{mhg} = \mathcal{L}_{tag}$ is shown in [30] and $\mathcal{L}_{mhg} \subseteq \mathcal{L}_{hg}$ is shown in [26]. (The equality labeled with $*1'$ in Fig. 1 is a corollary of the equalities labeled with $*1$.) In [1] and [27], subclasses of lfg's called pushdown lfg's (pd-lfg's) and finite-copying lfg's (fc-lfg's) are defined, and $\mathcal{L}_{pd\text{-}lfg} = \mathcal{L}_{mhg}$ and $\mathcal{L}_{fc\text{-}lfg} = \mathcal{L}_{mcfg}$ are shown ($*4$ in Fig. 1).

A number of equivalence relations between the classes of yield languages generated by fsts' and other computational models have been established[4, 6, 13, 27, 34]. Among others, it was shown in [34] that the class of yield languages generated by finite-copying fsts' (fc-fsts')[4] equals to the class of languages generated by mcfg's ($*2$ in Fig. 1).

There is a close relation between fsts' and pmcfg's in their string generation mechanisms. In fact, it was shown in [13] that the class of languages generated by pmcfg's is equal to the class of yield languages generated by d-fsts' ($*3$ in Fig. 1). As an easy corollary of this fact, it can be shown that the generative capacity of mcfg's is equal to that of rl-fsts' ($*3'$ in Fig. 1). Hence, it is concluded that fc-fsts' and rl-fsts' are equal to each other in their string generating capacity ($*2'$ in Fig. 1).

Also, the equalities labeled with $*5'$ are corollaries of $*3'$, $*4$ and $*5$.

Note that all the inclusion relations in Figure 1 are proper; indeed,

$$\{a_1^n a_2^n a_3^n a_4^n \mid n \geq 0\} \in \mathcal{L}_{tag} - \mathcal{L}_{cfg} \quad ([9]),$$

$$\{a_1^n a_2^n \cdots a_{2m-1}^n a_{2m}^n \mid n \geq 0\} \in \mathcal{L}_{mcfg} - \mathcal{L}_{mhg} \text{ for } m \geq 3 \quad ([16, 30, 32]),$$

$$\{a^{2^n} \mid n \geq 0\} \in \mathcal{L}_{d\text{-}fsts} - \mathcal{L}_{fc\text{-}fsts} \quad (\text{Corollary 3.2.8 in } [4]),$$

$$\in \mathcal{L}_{pmcfg} - \mathcal{L}_{mcfg} \quad ([16]), \text{ and}$$

$$\{w \in \{a, b\}^* | \text{the number of occurrences of } a \text{ in } w \text{ is } 2^n \text{ for some } n \geq 0\}$$

$$\in \mathcal{L}_{fsts} - \mathcal{L}_{d\text{-}fsts} \quad (\text{Corollary 3.2.16 in } [4]).$$

It was shown in [5] that the languages generated by d-fsts' are recognizable in polynomial time, and in [13], the languages generated by d-fsts' are shown to be recognizable in $O(n^{e+1})$ time where n is the length of an input string and e is a constant called the degree of a given d-fsts. That is, all languages in the region enclosed with the bold line are recognizable in polynomial time. On the other hand, it was shown in [25] that 3SAT, which is known to be $\mathcal{NP}$-complete can be generated by a (nondeterministic) fsts. Furthermore, it is shown in [13] that a (nondeterministic) fsts can generate an $\mathcal{NP}$-complete language even if the input domain is restricted to a set of monadic trees and the configurations are of state bound two[4].

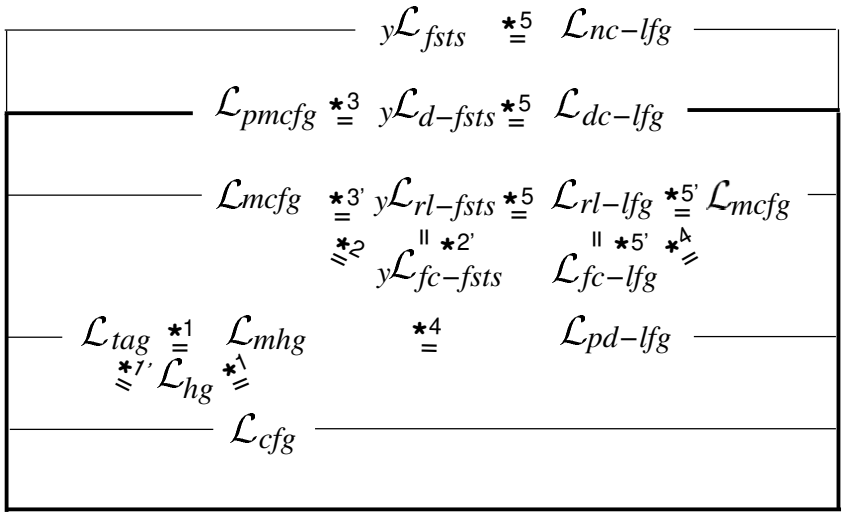

Figure 1: Inclusion relations between classes of languages.

1.3 Universal Recognition

In the later half of this paper, the computational complexity of the universal recognition problems for dc-lfg's and rl-lfg's are investigated. For a class $\mathcal{G}$ of grammars, the *universal recognition problem* for $\mathcal{G}$ is formally defined as follows: take a description of a grammar $G \in \mathcal{G}$ and a string w as an input, decide whether G can generate w. Note that the size of an input is $||G|| + |w|$ where $||G||$ and $|w|$ denote the length of the description of G and that of w, respectively, which is different from the case of fixed-language recognition problem. A *fixed-language recognition problem* for a language L is defined as follows: take a string w as an input, decide whether w belongs to L. The size of an input is $|w|$ only.

Note that the universal recognition problem for $\mathcal{G}_1$ may be harder than that for $\mathcal{G}_2$ even if the generative capacity of $\mathcal{G}_1$ is not stronger than that of $\mathcal{G}_2$ (also see [3]). For example, although the class of languages generated by generalized phrase structure grammars (gpsg's)[8] is included in the class of context-free languages (cfl's), and the universal recognition problem for context-free grammars (cfg's) is known to be solvable in deterministic polynomial time, the problem for gpsg's is known to be EXP-POLY time-hard[3]. Table 1 summarizes known results on complexity of universal/fixed-language recognition problems for some classes of grammars, where rlfg's[21] denotes a subclass of lexical functional grammars and ig's denotes the class of indexed grammars. The column for pmcfg's/mcfg's of Table 2 shows known results on the universal recognition problems for pmcfg's, mcfg's and those with some restrictions.

In Section **5**, complexity of the universal recognition problems for dc-lfg's/rl-lfg's, d-fsts'/rl-fsts' and their subclasses are investigated. The results are shown in Table 2. Hardness properties are shown by reducing the universal recognition problem for pmcfg's (resp. mcfg's) to the problem for d-fsts' (resp. rl-fsts'), which in turn is reduced to the problem for dc-lfg's (resp. rl-lfg's). Containment properties in EXP-POLY time, $\mathcal{NP}$ and $\mathcal{P}$ are proved by the reductions of the opposite direction. Informally, dimension means the maximum number of 'component' productions (a component production in a pmcfg, a translation rule in a d-fsts, and a production in a dc-lfg) that can be applied consistently to a local phrase structure (a nonterminal symbol in a pmcfg, a production rule of a cfg defining the input domain in a d-fsts, and an f-structure in a dc-lfg). On the other hand, degree is the maximum of the total size of such consistent component productions. Table 2 says that if dimension (or degree) is bounded for the three classes of grammars, then the complexity of the universal recognition problems for the classes are reduced in the same way. These results suggest that the three classes of grammars have resemblance in their structures.

Table 1: Complexity of universal/fixed-language recognition

grammars	universal recognition	fixed-language recognition
cfg's	$\mathcal{P}$-complete[7]	$O(n^3)$-time
gpsg's	EXP-POLY time-hard[3]	$O(n^3)$-time
pmcfg's/ mcfg's	see Table 2	$O(n^{e+1})$-time / $O(n^e)$-time[17] (e: the degree of a grammar)
rlfg's	$\mathcal{NP}$-hard[3]	nondeterministic polynomial time[20]
ig's	EXP-POLY time-complete[28]	nondeterministic $O(n)$-space
csg's	PSPACE-complete[15]	PSPACE-complete[15]

Table 2: Complexity of the universal recognition problem for some grammars

subclass	grammars		
	pmcfg's/ mcfg's	d-fsts'/ rl-fsts'	dc-lfg's/ rl-lfg's
no restriction	EXP-POLY time complete		
	Theorem 6 in [12]/ Theorem 1 in [11]	Corollary 5.4	Theorem 5.3
dimension at most m	$\mathcal{NP}$-complete for $m \geq 1$ / $m \geq 2$		
	Theorem 2 in [12]/ Theorem 1 in [12]	Corollary 5.10	Theorem 5.9
definition of dimension	the maximum number of components of a tuple of strings	the number of states	the maximum number of productions in a consistent subset of productions
degree at most e	$\mathcal{P}$-complete for $e \geq 3$		
	Theorem 4 in [12]/ Theorem 3 in [12]	Corollary 5.12	Theorem 5.11
definition of degree	the maximum size of a function	the maximum size of a set of rules for a single input symbol	the maximum size of a consistent subset of productions

2. Preliminaries

2.1 Finite State Translation Systems

A set Σ of symbols is a *ranked alphabet* if, for each $\sigma \in \Sigma$, a unique non-negative integer $\rho(\sigma)$ is associated. $\rho(\sigma)$ is called the *rank of* σ. For a set X, we define the set $\mathcal{T}_\Sigma(X)$ of terms generated by Σ and X as the smallest set such that:

- $\mathcal{T}_\Sigma(X)$ includes X.
- If $\rho(\sigma) = 0$ for $\sigma \in \Sigma$, then $\sigma \in \mathcal{T}_\Sigma(X)$.
- If $\rho(\sigma) = n$ (≥ 1) for $\sigma \in \Sigma$ and $t_1, \ldots, t_n \in \mathcal{T}_\Sigma(X)$, then $t = \sigma(t_1, \ldots, t_n) \in \mathcal{T}_\Sigma(X)$. σ is called the *root symbol*, or shortly, the *root* of t.

Hereafter, a term in $\mathcal{T}_\Sigma(X)$ is also called a *tree*. We sometimes use a usual graph representation of tree and use terminology such as *node*, *subtree*, *child*, and so on. If a node of a tree t has no children, then it is called a *leaf* of t. If a node of a tree is not a leaf, then it is called an *internal node*.

Let $G = (N, T, P, S)$ be a context-free grammar (cfg) where N, T, P and S are a set of *nonterminal symbols*, a set of *terminal symbols*, a set of *productions* and the *start symbol*, respectively. A *derivation tree in cfg* G is a term defined as follows.

(T1) Every $a \in T$ is a derivation tree in G.

(T2) Assume that there are a production $p : A \to X_1 \cdots X_n$ $(A \in N, X_1, \ldots, X_n \in N \cup T)$ in P and n derivation trees $t_1, \ldots t_n$ whose roots are labeled with $\sigma_1, \ldots, \sigma_n$, respectively, and

- if $X_i \in N$, then $\sigma_i = p_i$ is a production $p_i : X_i \to \cdots$, whose left-hand side is X_i, and
- if $X_i \in T$, then $\sigma_i = t_i = X_i$.

Then $p(t_1, \ldots, t_n)$ is a derivation tree in G. Note that for a derivation tree $p(t_1, \cdots, t_n)$ where the root symbol p is a production $A \to \cdots$, A is not necessarily the start symbol.

(T3) There are no other derivation trees.

Let $\mathcal{R}(G)$ be the set of derivation trees in G, and $\mathcal{R}_S(G) \subseteq \mathcal{R}(G)$ be the set of derivation trees of which root symbol is a production $p : S \to \cdots$ having the start symbol S as its left-hand side.

A *tree transducer*[24] defines a mapping from trees to trees. Since we are mainly interested in the string language generated by a tree transducer, a "tree-to-string" version of transducer (see [4]) is used in this paper. For sets Q and X, let

$$Q[X] \triangleq \{q[x] \mid q \in Q, x \in X\}.$$

A *tree-to-string transducer* (*yT-transducer* or simply *transducer*) is defined to be a 5-tuple $M = (Q, \Sigma, \Delta, q_0, R)$ where (1) Q is a finite set of *states*, (2) Σ is an *input ranked alphabet*, (3) Δ is an *output alphabet*, (4) $q_0 \in Q$ is the *initial state*, and (5) R is a finite set of *translation rules* (or *rules* for short) of the form

$$q[\sigma(x_1, \ldots, x_n)] \to v$$

where $q \in Q, \sigma \in \Sigma, n = \rho(\sigma)$ and $v \in (\Delta \cup Q[\{x_1, \ldots, x_n\}])^*$. If any different rules in R have different left-hand sides, then M is called

deterministic[4].

A *configuration* of a yT-transducer is an element in $(\Delta \cup Q[\mathcal{T}_\Sigma(\phi)])^*$ (ϕ denotes the empty set.). *Translation* by M is defined as follows. Let $\alpha = \alpha_1 q[\sigma(t_1, \ldots, t_n)]\alpha_2$ be a configuration where $\alpha_1, \alpha_2 \in (\Delta \cup Q[\mathcal{T}_\Sigma(\phi)])^*$, $q \in Q$, $\sigma \in \Sigma$, $\rho(\sigma) = n$ and $t_1, \ldots, t_n \in \mathcal{T}_\Sigma(\phi)$. Assume that there is a rule $q[\sigma(x_1, \ldots, x_n)] \to v$ in R. Let v' be obtained from v by substituting $t_1, \ldots, t_n$ for $x_1, \ldots, x_n$, respectively, then we define $\alpha \underset{M}{\Rightarrow} \alpha_1 v' \alpha_2$. Let $\overset{*}{\underset{M}{\Rightarrow}}$ be the reflexive and transitive closure of $\Rightarrow$. If $\alpha \overset{*}{\underset{M}{\Rightarrow}} \beta$, then we say α *is translated into* β.

A *tree-to-string finite state translation system* (*yT-fsts* or *fsts*)[24, 29] is a pair (M, G) of a yT-transducer $M = (Q, \Sigma, \Delta, q_0, R)$ and an ε-free cfg $G = (N, T, P, S)$ which satisfies $\Sigma = T \cup P$ where $\rho(a) = 1$ for $a \in T$ and $\rho(p) = n$ for $p : A \to B_1 \cdots B_n \in P$. Note that $\mathcal{R}_S(G) \subseteq \mathcal{T}_\Sigma(\phi)$ holds.

We define $yL(M, G)$, called the *yield language generated by yT-fsts* (M, G), as

$$yL(M, G) \triangleq \{w \in \Delta^* \mid \exists t \in \mathcal{R}_S(G), q_0[t] \overset{*}{\underset{M}{\Rightarrow}} w\}$$

where Δ is an output alphabet and q_0 is the initial state of M.

We note that an fsts (M, G) can be considered to be a model of a transformational grammar: A deep-structure of a sentence is represented by a derivation tree of G, and M can be considered to transform the deep-structure into a sentence (or a surface-structure).

An fsts is called a *deterministic-fsts* (or *d-fsts* for short)[4] if the transducer M is deterministic. Let (M, G) be a d-fsts. If (M, G) satisfies the following condition, then it is called a *right-linear fsts* (or *rl-fsts* for

short).

- Let σ be a production $A \to X_1 X_2 \cdots X_n$ of G or a terminal symbol of G, and let

$$r_j \ : \ q_j[\sigma(x_1, \cdots, x_n)] \to \beta_j \quad (1 \leq j \leq s)$$

be the rules of M such that σ appears in their left-hand sides. For each pair of a state q and a variable x_i $(1 \leq i \leq n)$, the total number of occurrences of $q[x_i]$ in $\beta_1, \beta_2, \cdots, \beta_s$ is at most one.

For a finite set $\mathcal{S}$, $|\mathcal{S}|$ denotes the number of elements in $\mathcal{S}$. Let (M, G) where $M = (Q, \Sigma, \Delta, q_0, R)$ is a d-fsts. We call $|Q|$ the *dimension* of (M, G). If the dimension of (M, G) is m, then (M, G) is called a *d-fsts with dimension m*. The *degree* of (M, G) is defined as follows. For $\sigma \in \Sigma$, let m_σ be the number of rules whose left-hand side is $q[\sigma(x_1, \cdots, x_n)]$ $(q \in Q)$. For $q \in Q$ and $\sigma \in \Sigma$, let $n_{q,\sigma}$ denote the number of occurrences of variables in the right-hand side of the rule whose left-hand side is $q[\sigma(x_1, \ldots, x_n)]$. If no rule is defined for $q[\sigma(x_1, \ldots, x_n)]$, then $n_{q,\sigma} = 0$. Since M is deterministic, $n_{q,\sigma}$ can be defined uniquely. Define the *degree* of $\sigma \in \Sigma$ as $m_\sigma + \sum_{q \in Q} n_{q,\sigma}$. If the maximum degree among the symbols in Σ is e, then the degree of (M, G) is e and (M, G) is called a *d-fsts with degree e*.

Engelfriet introduced a subclass of fsts' called finite-copying fsts' as follows[4]: Let (M, G) $(G = (N, T, P, S))$ be an fsts with output alphabet Δ and initial state q_0, $t \in \mathcal{R}_S(G)$ and t' be a subtree of t. Assume that there is a translation $\delta : q_0[t] \underset{M}{\overset{*}{\Rightarrow}} w \in \Delta^*$. Now, delete all the translation steps which operates on t' from this translation δ. This leads to the

following new translation which keeps t' untouched;

$$\delta' : q_0[t] \underset{M}{\overset{*}{\Rightarrow}} w_1 q_{i_1}[t'] w_2 \cdots w_n q_{i_n}[t'] w_{n+1}$$

where $w_i \in \Delta^*$ for $1 \leq i \leq n+1$. The *state sequence of* t' *in translation* δ is defined to be $\langle q_{i_1}, \ldots, q_{i_n} \rangle$. Translation δ has *copying-bound* k if, for every subtree of t, the length of its state sequence is at most k. An fsts (M, G) $(G = (N, T, P, S))$ is a *finite-copying*, if there is a constant k and for each $w \in yL(M, G)$, there is a derivation tree $t \in \mathcal{R}_S(G)$ and a translation $q_0[t] \underset{M}{\overset{*}{\Rightarrow}} w$ with copying-bound k. It is known that the determinism does not weaken the generative capacity of finite-copying fsts'[4].

If the input alphabet for fsts' is restricted to be monadic, we obtain another classification of fsts'. For a historical reason, such subclasses of fsts' are defined as subclasses of *0L system*. Let (M, G) be an fsts where $M = (Q, \Sigma, \Delta, q_0, R)$ and $G = (N, T, P, S)$. If Σ is monadic, i.e., $\rho(\sigma) \leq 1$ for each $\rho \in \Sigma$, then (M, G) is called an *ET0L system*. Note that, by the definition of fsts, if (M, G) is monadic, then every production in P is of the form either $A \to a\,(A \in N, a \in T)$ or $A \to B\,(A, B \in N)$. If (M, G) is a d-fsts (rl-fsts, fc-fsts, resp.) and Σ is monadic, then (M, G) is called an *EDT0L System* ($ET0L_{RL}$ *system*, $ET0L_{FIN}$ *system*, resp.).

It is known that the class of yield languages generated by fsts', d-fsts' and fc-fsts' properly includes the class of yield languages generated by ET0L systems, EDT0L systems and ET0L_{FIN} systems, respectively (see Corollaries 5.3 and 5.6 in [4]).

2.2 Parallel Multiple Context-Free Grammars

A *parallel multiple context-free grammar* (*pmcfg*) is defined to be a 5-tuple $G = (N, T, F, P, S)$ which satisfies the following conditions (G1) through (G5)[16, 26].

(G1) N is a finite set of *nonterminal symbols*. A positive integer $d(A)$ is given for each nonterminal symbol $A \in N$. The *dimension* of G is $\max\{d(A) \mid A \in N\}$.

(G2) T is a finite set of *terminal symbols* which is disjoint with N.

(G3) F is a finite set of *functions* satisfying the following conditions. For a positive integer d, let $(T^*)^d$ denote the set of all the d-tuples of strings over T. For each $f \in F$ with arity $a(f)$, positive integers $r(f)$ and $d_i(f)$ $(1 \leq i \leq a(f))$ are given, and f is a total 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

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

denote the ith argument of f for $1 \leq i \leq a(f)$.

(f1) For $1 \leq h \leq r(f)$, the hth component of f, denoted by $f^{[h]}$, is defined as;

$$f^{[h]}[\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_{a(f)}] = \alpha_{h,0} x_{\mu(h,0)\eta(h,0)} \alpha_{h,1} \cdots \alpha_{h,n_h-1} x_{\mu(h,n_h-1)\eta(h,n_h-1)} \alpha_{h,n_h} \tag{2.1}$$

where $\alpha_{h,k} \in T^*$ for $0 \leq k \leq n_h$, $1 \leq \mu(h,j) \leq a(f)$ and $1 \leq \eta(h,j) \leq d_{\mu(h,j)}(f)$ for $0 \leq j \leq n_h - 1$.

(G4) P is a finite set of *productions* of the form $A \to f[A_1, A_2, \ldots, A_{a(f)}]$ where $A, A_1, A_2, \ldots, A_{a(f)} \in N$, $f \in F$, $r(f) = d(A)$ and $d_i(f) = d(A_i)$ $(1 \leq i \leq a(f))$. If $a(f) = 0$, i.e., $f \in (T^*)^{r(f)}$, the production is called a *terminating production*, otherwise it is called a *nonterminating production*.

(G5) $S \in N$ is the *start symbol*, and $d(S) = 1$.

If all the functions of a pmcfg G satisfy the following Right Linearity condition, then G is called a *multiple context-free grammar* (*mcfg*).

[**Right Linearity**] For each x_{ij}, the total number of occurrences of x_{ij} in the right-hand sides of (2.1) from $h = 1$ through $r(f)$ is at most one.

The language generated by a pmcfg $G = (N, T, F, P, S)$ is defined as follows. For $A \in N$, let us define $L_G(A)$ as the smallest set satisfying the following two conditions:

(L1) If a terminating production $A \to \bar{\alpha}$ is in P, then $\bar{\alpha} \in L_G(A)$.

(L2) If $A \to f[A_1, A_2, \ldots, A_{a(f)}] \in P$ and $\bar{\alpha}_i \in L_G(A_i)$ $(1 \leq i \leq a(f))$, then $f[\bar{\alpha}_1, \bar{\alpha}_2, \ldots, \bar{\alpha}_{a(f)}] \in L_G(A)$.

The language generated by G, written $L(G)$, is $L_G(S)$.

For a positive integer m, if the dimension of pmcfg G is m, then G is called an pmcfg *with dimension* m. We define an mcfg *with dimension* m in the same way.

The *degree* of a function f is defined as $r(f) + \Sigma_{h=1}^{r(f)} n_h$ where n_h is the total number of occurrences of variables x_{ij} $(1 \leq i \leq a(f), 1 \leq j \leq$

$d_i(f)$) in the right-hand side of (2.1). If the maximum degree among the functions in F of G is e, then the degree of G is e and G is called a pmcfg with degree e. An mcfg with degree e is defined in the same way.

Example 2.1: [16] Let $G_{EX1} = (N, T, F, P, S), N = \{S\}, T = \{a\}, F = \{f_a, f\}, P = \{r_1 : S \rightarrow f_a,\ r_0 : S \rightarrow f[S]\}$, where $f_a = a, f[(x)] = xx$. G_{EX1} is a pmcfg but is not an mcfg since the function f does not satisfy Right Linearity. The language generated by G_{EX1} is $\{a^{2^n} \mid n \geq 0\}$, which cannot be generated by any mcfg (see Lemma 6 in[16]). □

The empty string is denoted by ε.

Example 2.2: Let $G_{EX2} = (N, T, F, P, S)$ be a pmcfg, where $N = \{S, A\}$, $T = \{a, b\}$, $F = \{g[(x_1, x_2)] = x_1x_2,\ f_a[(x_1, x_2)] = (x_1a, x_2a),\ f_b[(x_1, x_2)] = (x_1b, x_2b),\ f_\varepsilon = (\varepsilon, \varepsilon)\}$, and, P=$\{p_0 : S \rightarrow g[A],\ p_1 : A \rightarrow f_a[A],\ p_2 : A \rightarrow f_b[A],\ p_3 : A \rightarrow f_\varepsilon\}$. Note that G_{EX2} is an mcfg. $L(G_{EX2}) = \{ww \mid w \in \{a, b\}^*\}$. □

Lemma 2.1: [16] For a given pmcfg G (resp. mcfg G), we can construct an pmcfg $G' = (N', T', F', P', S')$ (resp. mcfg G') which satisfies $L(G') = L(G)$ and the following non-erasing condition.

[**Non-erasing Condition**] For each f in F', each variable x_{ij} appears at least once in the right-hand side of (2.1) for some $h(1 \leq h \leq r(f))$. □

Lemma 2.2: [17, 26] Let G be a pmcfg with degree e. For a given string w, it is solvable whether $w \in L(G)$ or not in $O(|w|^{e+1})$ time, where $|w|$ denotes the length of w. If G is an mcfg, then the problem is solvable in $O(|w|^e)$ time. □

3. Nondeterministically Copying Lfg's

3.1 Nc-lfg's

A simple subclass of lfg's, called *r-lfg*'s, is introduced in [21], which is shown to generate all recursively enumerable languages[19]. Here, we define *nondeterministically copying lfg's* (*nc-lfg's*) as a proper subclass of r-lfg's. An nc-lfg is defined to be a 6-tuple $G = (N, T, P, S, N_{atr}, A_{atm})$ where: (1) N is a finite set of *nonterminal symbols*, (2) T is a finite set of *terminal symbols*, and (3) P is a finite set of *annotated productions*. Sometimes, a nonterminal symbol, a terminal symbol and an annotated production are abbreviated as a *nonterminal*, a *terminal* and a *production*, respectively. (4) $S \in N$ is the *start symbol*, (5) N_{atr} is a finite set of *attributes*, and (6) A_{atm} is a finite set of *atoms*.

An equation of the form $\uparrow atr = \downarrow$ $(atr \in N_{atr})$ is called an S (*structure synthesizing*) *schema*, and an equation of the form $\uparrow atr = val$ $(atr \in N_{atr}, val \in A_{atm})$ is called a V (*immediate value*) *schema*. A *functional schema* is either an S schema or a V schema.

Each production $p \in P$ has the following form:

$$\begin{array}{ccccccc} p : A & \rightarrow & X_1 & X_2 & \cdots & X_h, \\ E_V & & E_{S1} & E_{S2} & \cdots & E_{Sh} \end{array} \tag{3.2}$$

where $A \in N$, $X_1, X_2, \cdots, X_h \in N \cup T$. E_V is a finite set of V schemata and E_{Sj} $(1 \leq j \leq h)$ is a singleton of an S schema. $E_V \cup \cup_{1 \leq j \leq h} E_{Sj}$ is called the set of functional schemata of p. $A \rightarrow X_1 X_2 \cdots X_h$ in (3.2) is called the *underlying production of* p. Let P_0 be the set of all the underlying productions of P. Cfg $G_0 = (N, T, P_0, S)$ is called the *underlying cfg of* G.

An *f-structure* of G is recursively defined as a set $F = \{\langle atr_1, val_1\rangle, \langle atr_2, val_2\rangle, \cdots, \langle atr_k, val_k\rangle\}$ where $atr_1, atr_2, \cdots$, and atr_k are distinct attributes, and each of $val_1, val_2, \cdots$, and val_k is an atom or an f-structure. We say that $val_i (1 \leq i \leq k)$ is the *value of* atr_i in F and write $F.atr_i = val_i$. If $\langle atr, val\rangle \notin F$ for any val, then $F.atr$ is undefined.

For a cfg $G' = (N', T', P', S')$, derivation relations in G', denoted by $A \underset{G'}{\Rightarrow} \alpha$ and $A \overset{*}{\underset{G'}{\Rightarrow}} \alpha$ $(A \in N', \alpha \in (N' \cup T')^*)$ are defined in the usual way.

Suppose $G_0 = (N, T, P_0, S)$ is the underlying cfg of an nc-lfg $G = (N, T, P, S, N_{atr}, A_{atm})$. Let t be a derivation tree in G_0. Every node v labeled with a nonterminal in t has an f-structure, which is called the f-structure of v and written as F_v. If an underlying production $p_0 : A \rightarrow X_1 \cdots X_h \in P_0$ is applied at v, then we also say that the annotated production $p \in P$ is applied at v where p_0 is the underlying production of p. Let v_i be the ith child of v $(1 \leq i \leq h)$. We define the values of both sides of a functional schema attached to the symbol in p (on v) as follows:

- the value of $\uparrow atr (atr \in N_{atr})$ is $F_v.atr$,
- the value of $\downarrow$ in an S schema is F_{v_i} if the S schema is attached to the ith $(1 \leq i \leq h)$ symbol in the right-hand side of p, and
- the value of atom atm in a V schema is atm itself.

We say that v *satisfies* functional schemata if for each functional schema $l_{fs} = r_{fs}$ of p, the values of l_{fs} and r_{fs} on v are defined and equals with each other. In this case, it is also said that F_v *locally satisfies* the functional

schemata of p. (NOTE: Because the meaning of a V schema is independent of the position where it is annotated, V schemata are attached to the left-hand side in this paper.)

For a nonterminal $A \in N$ and a sentential form $\alpha \in (N \cup T)^*$, let t be a derivation tree of a derivation $A \underset{G_0}{\overset{*}{\Rightarrow}} \alpha$. If all internal nodes in t satisfy functional schemata, then α is said to *be derived from* A and written as $A \underset{G}{\overset{*}{\Rightarrow}} \alpha$. In this case, the tree t is called a *derivation tree of* $A \underset{G}{\overset{*}{\Rightarrow}} \alpha$. We also call t a derivation tree (of α) in G simply. The *language generated by* an nc-lfg G, denoted by $L(G)$, is defined as $L(G) = \{w \in T^* \mid S \underset{G}{\overset{*}{\Rightarrow}} w\}$.

Example 3.1: Let $G_{EX3} = (N, T, P, S, N_{atr}, A_{atm})$ be an nc-lfg where $N = \{S, A, B\}$, $T = \{a, b, c, d\}$, $N_{atr} = \{count, pos\}$, $A_{atm} = \{top, mid, btm\}$, and productions in P are;

$$p_{11}: \underset{\{\uparrow pos = top\}}{S} \rightarrow \underset{\{\uparrow count = \downarrow\}}{A} \quad \underset{\{\uparrow count = \downarrow\}}{B},$$

$$p_{12}: \underset{\{\uparrow pos = mid\}}{A} \rightarrow a \quad \underset{\{\uparrow count = \downarrow\}}{A} \quad b,$$

$$p_{13}: \underset{\{\uparrow pos = mid\}}{B} \rightarrow c \quad \underset{\{\uparrow count = \downarrow\}}{B} \quad d,$$

$$p_{14}: \underset{\{\uparrow pos = btm\}}{A} \rightarrow \varepsilon,$$

$$p_{15}: \underset{\{\uparrow pos = btm\}}{B} \rightarrow \varepsilon.$$

The language generated by G_{EX3} is $L(G_{EX3}) = \{a^n b^n c^n d^n \mid n \geq 0\}$. Figure 2 shows a derivation tree of $S \underset{G_{EX3}}{\overset{*}{\Rightarrow}} aabbccdd$ in G_{EX3}. □

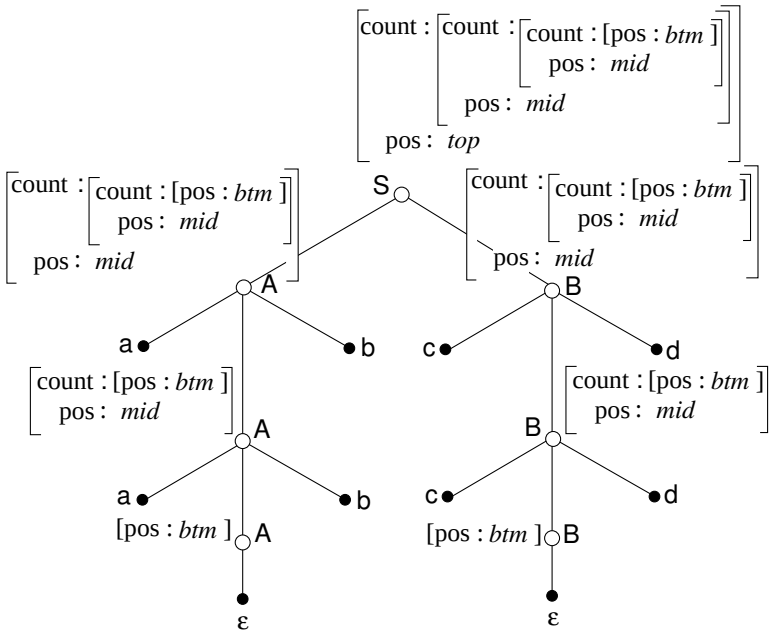

Figure 2: A derivation tree of *aabbccdd*

Example 3.2: Let $G_{EX4} = (N, T, P, S, N_{atr}, A_{atm})$ be an nc-lfg where $N = \{S\}$, $T = \{a\}$, $N_{atr} = \{log\}$, $A_{atm} = \{one\}$, and productions in P are;

$$p_{21} : \underset{}{S} \rightarrow \underset{\{\uparrow log = \downarrow\}}{S} \underset{\{\uparrow log = \downarrow\}}{S},$$

$$p_{22} : \underset{\{\uparrow log = one\}}{S} \rightarrow a.$$

The language generated by G_{EX4} is $L(G_{EX4}) = \{a^{2^n} \mid n \geq 0\}$. □

For a set $\mathcal{F}$ of functional schemata, $\mathcal{F}$ is *consistent* iff neither the following (1) nor (2) holds.

(1) $\{\uparrow atr = val_1, \uparrow atr = val_2\} \subseteq \mathcal{F}$

for some $atr \in N_{atr}$ and some $val_1, val_2 \in A_{atm}$ such that $val_1 \neq val_2$.

(2) $\{\uparrow atr = val, \uparrow atr = \downarrow\} \subseteq \mathcal{F}$

for some $atr \in N_{atr}$ and some $val \in A_{atm}$.

Productions $p_1, \cdots, p_n$ are called *consistent* if $\cup_{1 \leq i \leq n} E^{(i)}$ is consistent where $E^{(i)}$ is the set of functional schemata of p_i. In this case, the set $\{p_1, \cdots, p_n\}$ of productions is also called consistent. If productions are not consistent, then they are called *inconsistent*.

3.2 Dc-lfg's and Rl-lfg's

If an nc-lfg G satisfies the following condition, then G is called a *deterministically copying lfg* (*dc-lfg*).

- If $p_1 : A \rightarrow \alpha_1$ and $p_2 : A \rightarrow \alpha_2$ are distinct productions of G whose left-hand side is the same symbol A, then p_1 and p_2 are inconsistent.

Let G be a dc-lfg. If every consistent subset $\{p_1, p_2, \cdots, p_n\}$ of productions satisfies the following condition, then G is called a *right-linear lfg* (*rl-lfg*).

- For each pair of a nonterminal A and an attribute atr of G, the total number of occurrences of $\overset{A}{\{\uparrow atr = \downarrow\}}$ in the right-hand sides of $p_1, p_2, \cdots, p_n$ is at most one.

Example 3.3: Consider G_{EX3} in Example 3.1. Maximal consistent subsets of productions are $sp1 = \{p_{11}\}$, $sp2 = \{p_{12}, p_{13}\}$ and $sp3 = \{p_{14}, p_{15}\}$. For $sp2$, each nonterminal appears at most once in the right-hand sides. This condition holds also for $sp1$ and $sp3$. Hence G_{EX3} is a dc-lfg and is an rl-lfg. Similarly, G_{EX4} in Example 3.2 is a dc-lfg. $\overset{S}{\{\uparrow log = e\}}$ appears twice in p_{21}. Hence G_{EX4} is not an rl-lfg. □

Let $G = (N, T, P, S, N_{atr}, A_{atm})$ be an nc-lfg (resp. dc-lfg, rl-lfg). For a consistent subset sp of P, $|sp|$ is called the *dimension of sp*. The *dimension of* G is defined as the maximum dimension m among all consistent subsets of P, and in this case G is called an nc-lfg (resp. dc-lfg, rl-lfg) *with dimension* m. Define the *degree* of a consistent subset sp of P as $|sp| + \sum_{p_i \in sp} n_i$ where n_i denotes the number of occurrences of nonterminals which appear in the right-hand side of a production p_i. If the maximum degree among all consistent subsets of P is e, then the degree of G is e and G is called an nc-lfg (resp. dc-lfg, rl-lfg) *with degree* e.

3.3 Other Subclasses

In [1, 27], another subclass of lfg's called fc-lfg's is defined as follows. Suppose $G = (N, T, P, S, N_{atr}, A_{atm})$ is an nc-lfg. Let $\{\{e_1, e_2, \cdots, e_n\}\}$ denote the multiset which consists of elements $e_1, e_2, \cdots, e_n$ that are not necessarily distinct. An *SPN* (*SubPhrase Nonterminal*) *multiset* in G is recursively defined as the following (1) through (3):

(1) $\{\{S\}\}$ is an SPN multiset.

(2) Suppose that $\{\{A_1, A_2, \cdots, A_h\}\}$ $(A_1, A_2, \cdots, A_h \in N)$ is an SPN multiset. Let $A_1 \rightarrow \alpha_1$, $\cdots$, $A_h \rightarrow \alpha_h$ be consistent productions. For each $atr \in N_{atr}$, let MS_{atr} be the multiset consisting of all the nonterminals which appear in $\alpha_1, \cdots, \alpha_h$ and have an S schema $\uparrow atr = \downarrow$. If MS_{atr} is not empty, then MS_{atr} is also an SPN multiset.

(3) There is no other SPN multiset.

An nc-lfg such that the number of SPN multisets in G is finite is called a *finite-copying lfg* (*fc-lfg*).

Example 3.4: Consider G_{EX3} in Example 3.1 again. $\{\{S\}\}$ is an SPN multiset by the above (1). Since p_{12} is the only production whose left-hand side is S, and A and B have the same S schema, we know that $\{\{A, B\}\}$ is an SPN multiset by the above (2). For an SPN multiset $\{\{A, B\}\}$, we must consider consistent productions p_{12} and p_{13}, but no new SPN multiset is obtained by the above (2). Similarly, no new SPN multiset is obtained from p_{14} and p_{15}. Hence, SPN multisets in G_{EX3} are $\{\{S\}\}$ and $\{\{A, B\}\}$ in all and therefore G_{EX3} is an fc-lfg.

For G_{EX4} in Example 3.2, there are infinite number of multisets $\{\{S\}\}, \{\{S,S\}\}, \{\{S,S,S,S\}\}, \cdots$. Hence, G_{EX4} is not an fc-lfg. □

In [1], it is shown that mcfg's and fc-lfg's have the same generative capacity (*4 in Fig. 1). By the definitions, every rl-lfg is also an fc-lfg, but an fc-lfg is not necessarily an rl-lfg. However, as shown in Figure 1, rl-lfg's and fc-lfg's have the same generative capacity.

We can introduce natural subclasses of nc-lfg's which correspond to ET0L systems and their subclasses. If an nc-lfg $G = (N, T, P, S, N_{atr}, A_{atm})$ satisfies the following Monadicness condition, then G is called a *monadic nc-lfg*.

[**Monadicness**] The set N_{atr} of attributes consists of only two elements, say, $N_{atr} = \{atr_S, atr_V\}$, and, atr_S appears only in S schemata in P and atr_V appears only in V schemata.

A dc-lfg (resp. rl-lfg, fc-lfg) which satisfies the above condition is called a *monadic dc-lfg*, a *monadic rl-lfg* and a *monadic fc-lfg*, respectively.

Example 3.5: For G_{EX3} in Example 3.1, by letting $atr_S = count$ and $atr_V = pos$, we know that G_{EX3} is a monadic rl-lfg. □

4. Generative Capacity of Nc-lfg's and Their Subclasses

4.1 $\mathcal{L}_{nc\text{-}lfg} \subseteq y\mathcal{L}_{fsts}$

First, we show $\mathcal{L}_{nc\text{-}lfg} \subseteq y\mathcal{L}_{fsts}$. For a given nc-lfg $G = (N, T, P, S, N_{atr} = \{atr_1, \cdots, atr_n\}, A_{atm})$, an equivalent fsts (M, G') is constructed in the following way.

Let τ be a derivation tree in nc-lfg G and the f-structure of the root node of t be $F = \{\langle atr_1, F_1\rangle, \cdots, \langle atr_n, F_n\rangle\}$. F is represented by a derivation tree $t = p'_{sp}(t_1, \cdots, t_n)$ in G', where $t_i (1 \leq i \leq n)$ is a derivation tree in G' which represents F_i recursively. Moreover, sp is a set of productions in P such that F locally satisfies the functional schemata of all productions in sp. M translates t into the yield of τ, i.e., the terminal string obtained by concatenating the labels of leaves of τ, in a top-down way.

[TRANS 1] Let $N = \{A_1, \cdots, A_m\}$, $S = A_1$ and $N_{atr} = \{atr_1, \cdots, atr_n\}$. Define SP as the set of all consistent subsets of P.

Step 1: $G' = (N', \{d\}, P', S')$, where $N' = \{S_{sp} | sp \in SP\} \cup \{S', Z\}$ (S', Z and d are newly introduced symbols) and

$$
\begin{aligned}
P' \;=\; & \{p'_{start(\bar{sp})} : S' \rightarrow S_{\bar{sp}} \\
& \quad | \bar{sp} \in SP \text{ and } \bar{sp} \text{ includes a production} \\
& \qquad\qquad \text{whose left-hand side is } S\} \\
& \cup \{p'_{sp} : S_{sp} \rightarrow \underbrace{Z \cdots Z}_{n} | sp \in SP\} \\
& \cup \{p'_{guess(sp)} : Z \rightarrow S_{sp} | sp \in SP\} \\
& \cup \{p'_{term} : Z \rightarrow d\}.
\end{aligned}
$$

For a derivation tree t in G' and a node v in t to which p'_{sp} is applied for some $sp \in SP$, the subtree rooted by the ith child of v represents the value of attribute atr_i.

Step 2: $M = (Q, \Sigma, T, q_1, R)$ is defined as follows.

Q is $\{q_1, \ldots, q_m\}$. A state q_j $(1 \leq j \leq m)$ corresponds to nonterminal A_j in N. R is the set of rules such that:

(i) For each $\bar{sp} \in SP$ that includes a production whose left-hand side is S,

$$q_1[p'_{start(\bar{sp})}(x)] \rightarrow q_1[x] \tag{4.3}$$

belongs to R.

(ii) For each $sp \in SP$,

$$q_j[p'_{guess(sp)}(x)] \rightarrow q_j[x] \quad (1 \leq j \leq m) \tag{4.4}$$

belongs to R.

(iii) Let t be a derivation tree in G'. When p'_{sp} is the production applied at the root of t and a state of M is $q_{\mu(0)}$, M chooses a production p whose left-hand side is $A_{\mu(0)}$, if exists, in sp. Note that since productions in sp are consistent, there is an f-structure which locally satisfies the functional schemata of all productions in sp. Consider an arbitrary (not necessarily maximal) consistent subset $sp \in SP$ of productions. For each production $p \in sp$ such that

$$\begin{array}{lccccccccc} p: & A_{\mu(0)} & \rightarrow & \alpha_0 & A_{\mu(1)} & \alpha_1 & \ldots & \alpha_{L-1} & A_{\mu(L)} & \alpha_L \\ & E_V & & \{\uparrow atr_{\nu(1)} = \downarrow\} & & \ldots & & \{\uparrow atr_{\nu(L)} = \downarrow\} & & \end{array} \tag{4.5}$$

where $A_{\mu(l)} \in N$ and $\alpha_l \in T^*(0 \leq l \leq L)$, the following rule belongs to R:

$$q_{\mu(0)}[p'_{sp}(x_1, \ldots, x_n)] \rightarrow \alpha_0 q_{\mu(1)}[x_{\nu(1)}]\alpha_1 \ldots \alpha_{L-1} q_{\mu(L)}[x_{\nu(L)}]\alpha_L. \quad (4.6)$$

(iv) No other rule belongs to R. □

Lemma 4.1: Let (M, G') be an fsts constructed from an nc-lfg by **TRANS 1**. For $i_1, \cdots, i_J \in \{1, \ldots, m\}$ (different i_j's are not necessarily distinct) and $w_j \in T^*(1 \leq j \leq J)$, the following **(C1)** and **(C2)** are equivalent.

(C1) There exist derivation trees $\tau_j (1 \leq j \leq J)$ of $A_{i_j} \overset{*}{\underset{G}{\Rightarrow}} w_j$ such that

$$F_{v_1} = F_{v_2} = \ldots = F_{v_J} \quad (4.7)$$

where v_j is the root of τ_j.

(C2) There exists a tree $t = p'_{sp}(\cdots) \in \mathcal{R}(G')$ for some $sp \in SP$ such that $q_{i_j}[t] \overset{*}{\underset{M}{\Rightarrow}} w_j$ $(1 \leq j \leq J)$.

Proof. **(C1)** $\Rightarrow$ **(C2)**: By induction on the maximum length K of derivations $A_{i_j} \overset{*}{\underset{G}{\Rightarrow}} w_j$ $(1 \leq j \leq J)$ in **(C1)**.

Basis: If $K = 1$, then for $1 \leq j \leq J$ there exists a production

$$p_j: \quad \begin{matrix} A_{i_j} \\ E_V \end{matrix} \quad \rightarrow \quad w_j$$

in P and, by (4.7), $p_1, p_2, \cdots, p_J$ are consistent. Let $sp = \{p_1, p_2, \cdots, p_J\}$. By **step 2** of **TRANS 1**, rules

$$q_{i_j}[p'_{sp}(x_1, \cdots, x_n)] \rightarrow w_j \quad (1 \leq j \leq J)$$

are in R. Let $t_d = p'_{term}(d)$, then $t = p'_{sp}(t_d, \cdots, t_d) \in \mathcal{R}(G')$ by **step 1** of **TRANS 1** and, for $1 \leq j \leq J$,

$$q_{i_j}[p'_{sp}(t_d, \cdots, t_d)] \underset{M}{\Rightarrow} w_j.$$

Induction: Suppose that the claim is true for $K \leq \eta$. We examine the case that $K = \eta + 1$. For $1 \leq j \leq J$, let

$$\begin{array}{lllllll} p_j: & A_{i_j} & \rightarrow & \alpha_{j0} & A_{\mu(j,1)} & \alpha_{j1} \cdots \alpha_{jL_j-1} & A_{\mu(j,L_j)}\ \alpha_{jL_j} \\ & E_{V_j} & & & \{\uparrow atr_{\nu(j,1)} = \downarrow\} & & \{\uparrow atr_{\nu(j,L_j)} = \downarrow\} \end{array} \tag{4.8}$$

$$(\alpha_{jl} \in T^*(0 \leq l \leq L_j) \text{ and } A_{i_j}, A_{\mu(j,l)} \in N(1 \leq l \leq L_j))$$

be the production applied at the root v_j of τ_j in condition **(C1)**. By (4.7), $p_1, p_2, \cdots, p_J$ are consistent. Let $\tau_{jl}(1 \leq j \leq J, 1 \leq l \leq L_j)$ be the subtree of τ_j of which the root v_{jl} is a child of v_j and labeled with $A_{\mu(j,l)}$. Also suppose that τ_{jl} is a derivation tree of

$$A_{\mu(j,l)} \underset{G}{\overset{*}{\Rightarrow}} w_{jl} \tag{4.9}$$

in G, and

$$\begin{array}{rcl} A_{i_j} & \underset{G}{\Rightarrow} & \alpha_{j0} A_{\mu(j,1)} \alpha_{j1} \cdots \alpha_{jL_j-1} A_{\mu(j,L_j)} \alpha_{jL_j} \\ & \underset{G}{\overset{*}{\Rightarrow}} & \alpha_{j0} w_{j1} \alpha_{j1} \cdots \alpha_{jL_j-1} w_{jL_j} \alpha_{jL_j} = w_j. \end{array}$$

Since S schema $\{\uparrow atr_{\nu(j,l)} = \downarrow\}$ is attached to nonterminal $A_{\mu(j,l)}$ and the root of τ_{jl} is v_{jl}, it follows from (4.7) that $\nu(j_1, l_1) = \nu(j_2, l_2)$ implies $F_{v_{j_1 l_1}} = F_{v_{j_2 l_2}}$. Hence, by the inductive hypothesis and (4.9), for $1 \leq u \leq n$, there exists a tree $t_u = p'_{sp_u}(\cdots) \in \mathcal{R}(G')$ such that

$$q_{\mu(j,l)}[t_{\nu(j,l)}] \underset{M}{\overset{*}{\Rightarrow}} w_{jl} \quad (1 \leq j \leq J, 1 \leq l \leq L_j). \tag{4.10}$$

Let $t'_u = p'_{guess(sp_u)}(t_u)$ and $t = p'_{sp}(t'_1, \cdots, t'_n)$ where $sp = \{p_1, \cdots, p_J\}$. By the construction of P' in **step 1** of **TRANS 1**, $t \in \mathcal{R}(G')$ holds. Moreover,

$$\begin{aligned} q_{i_j}[t] &\underset{M}{\Rightarrow} \alpha_{j0} q_{\mu(j,1)}[t'_{\nu(j,1)}]\alpha_{j1} \cdots \alpha_{jL_{j-1}} q_{\mu(j,1)}[t'_{\nu(j,L_j)}]\alpha_{jL_j} \\ &\qquad (\because (4.8) \text{ and } \textbf{step2} \text{ (iii)}) \\ &\underset{M}{\overset{*}{\Rightarrow}} \alpha_{j0} q_{\mu(j,1)}[t_{\nu(j,1)}]\alpha_{j1} \cdots \alpha_{jL_{j-1}} q_{\mu(j,1)}[t_{\nu(j,L_j)}]\alpha_{jL_j} \\ &\qquad (\because (4.4)) \\ &\underset{M}{\overset{*}{\Rightarrow}} \alpha_{j0} w_{j1} \alpha_{j1} \cdots \alpha_{jL_{j-1}} w_{jL_j} \alpha_{jL_j} = w_j. \\ &\qquad (\because (4.10)) \end{aligned}$$

This completes the proof of the inductive step.

The converse (**(C2)** $\Rightarrow$ **(C1)**) can be proved in a similar way by induction on the number of internal nodes of a derivation tree $t \in \mathcal{R}(G')$ in Condition **(C2)**. □

Lemma 4.2: $\mathcal{L}_{nc\text{-}lfg} \subseteq y\mathcal{L}_{fsts}$

Proof. Let (M, G') be the fsts constructed from an nc-lfg G by **TRANS 1**. By Lemma 4.1, $S(= A_1) \underset{G}{\overset{*}{\Rightarrow}} w$ if and only if there exists a tree $t \in \mathcal{R}_S(G')$ such that $q_1[t] \underset{M}{\overset{*}{\Rightarrow}} w$. □

Corollary 4.3: $\mathcal{L}_{dc\text{-}lfg} \subseteq y\mathcal{L}_{d\text{-}fsts}$

Proof. In **TRANS 1**, if G is a dc-lfg, then no $sp \in SP$ contains distinct productions whose left-hand sides are the same and hence the constructed transducer M becomes deterministic by the construction (**step 2** (iii)).

□

Corollary 4.4: $\mathcal{L}_{rl\text{-}lfg} \subseteq y\mathcal{L}_{rl\text{-}fsts}$

Proof. Assume that G is an rl-lfg in **TRANS 1**. Clearly, $p'_{start}(\bar{sp})$ and $p'_{guess}(sp)$ meet the definition of rl-fsts by **step 2** (i) and (ii). For a consistent set sp of productions in P considered in **step 2** (iii), $\overset{A_j}{\{\uparrow atr_i = \downarrow\}}$ occurs at most once in the right-hand sides of sp since G is an rl-lfg. Hence, each $q_j[x_i]$ occurs at most once in the right-hand sides of rules in R constructed for sp in **step 2** (iii), implying that (M, G') is an rl-fsts. The corollary follows from Lemma 4.2. □

For a finite set $\mathcal{S}$, $||s||$ $(s \in \mathcal{S})$ denotes the description length of s (under appropriate encoding). $||\mathcal{S}||$ denotes $\sum_{s\in\mathcal{S}} ||s||$, and $||max(\mathcal{S})||$ denotes $\max_{s\in\mathcal{S}} ||s||$.

By **TRANS 1**, Q has a one-to-one correspondence with N. However, the construction method can be easily modified in such a way that $|Q|$ is equal to $\max_{sp\in SP} |sp|$, i.e., the dimension of G. For a given dc-lfg $G = (N, T, P, S, N_{atr}, A_{atm})$, let (M, G') where $M = (Q, \Sigma, \Delta, q_1, R)$ and $G' = (N', T', P', S')$ be an equivalent d-fsts constructed from G by the modified version of **TRANS 1**. The sizes of the components of M and G' can be represented in terms of those of G as follows. (Let SP denote the set of all maximal consistent subsets of P.)

$$\left.\begin{array}{l} |N'|=|SP|+2=O(2^{|P|}) \\ |T'|=1 \\ ||P'||=O(|P'|)\times O(||max(P')||)=O(|SP|+|P|)\times O(|N_{atr}|) \\ \quad =O(2^{|P|}\times|N_{atr}|) \\ |Q|=\max_{sp\in SP}|sp|=m, \text{ the dimension of } G \\ |\Sigma|=O(|P'|)=O(|SP|+|P|)=O(2^{|P|}) \\ |\Delta|=|T| \\ ||R||=O(|R|)\times O(||max(R)||) \\ \quad =O(\max_{sp\in SP}|sp|\times|SP|)\times O(||max(P)||) \\ \quad =O(m\times 2^{|P|}\times||max(P)||) \end{array}\right\} \quad (4.11)$$

Example 4.1: For dc-lfg $G = G_{EX3}$ in Example 3.1 (also see Example 3.3), the following d-fsts (M, G'), where $M = (Q, \Sigma, \Delta, q_1, R)$ and $G' = (N', T', P', S')$, is constructed.

$$N' = \{S_{sp1}, S_{sp2}, S_{sp3}, S', Z\},\ T' = \{d\},$$

$$P' = \left\{ \begin{array}{l} p'_{start} : S' \rightarrow S_{sp1} \\ p'_{spi} : S_{spi} \rightarrow ZZ \text{ for } i = 1, 2, 3, \\ p'_{guess(spi)} : Z \rightarrow S_{spi} \text{ for } i = 1, 2, 3, \\ p'_{term} : Z \rightarrow d \end{array} \right\},$$

$$Q = \{q_1, q_2\},\ \Delta = T,$$

$$\Sigma = \{p'_{start}, p'_{sp1}, p'_{sp2}, p'_{sp3}, p'_{guess(sp1)}, p'_{guess(sp2)}, p'_{guess(sp3)}, p'_{term}\},\ \text{and}$$

$$R = \left\{ \begin{array}{l} q_1[p'_{start}(x)] \rightarrow q_1[x], \\ q_j[p'_{guess(spi)}(x)] \rightarrow q_j[x] \text{ for } i = 1,2,3 \text{ and } j = 1,2, \\ q_1[p'_{sp1}(x_1,x_2)] \rightarrow q_1[x_1]q_2[x_1], \\ q_1[p'_{sp2}(x_1,x_2)] \rightarrow aq_1[x_1]b, \\ q_2[p'_{sp2}(x_1,x_2)] \rightarrow cq_2[x_1]d, \\ q_1[p'_{sp3}(x_1,x_2)] \rightarrow \varepsilon, \\ q_2[p'_{sp3}(x_1,x_2)] \rightarrow \varepsilon \end{array} \right\}.$$

□

Lemma 4.5: Let G be a dc-lfg with dimension m ($m \geq 1$) and with degree e ($e \geq 3$). Then, the d-fsts (M, G') constructed from G by **TRANS 1** is with dimension m and with degree $\max(e, 2m)$.

Proof. By (4.11), the dimension of (M, G') is m. For each consistent subset sp of productions in G, the set of rules of the form $q[p'_{sp}(\cdots)] \rightarrow \cdots$ is constructed and the degree of p'_{sp} is equal to the degree of sp by **step 2** (iii) of **TRANS 1**. For $p'_{start(\bar{sp})}$ and $p'_{guess(sp)}$ in **step 2** (i) and (ii), the degrees are 2 and $2m$, respectively. Thus, the degree of (M, G') is $\max(e, 2m)$. □

4.2 $y\mathcal{L}_{fsts} \subseteq \mathcal{L}_{nc\text{-}lfg}$

Next, $y\mathcal{L}_{fsts} \subseteq \mathcal{L}_{nc\text{-}lfg}$ is shown. For a given fsts (M, G), the following algorithm constructs an nc-lfg G' such that $L(G') = yL(M, G)$.

[TRANS 2] Suppose that a given fsts is (M, G), where $G = (N, T, P, S)$, $M = (Q, \Sigma, \Delta, q_1, R)$ and $Q = \{q_1, q_2, \cdots, q_m\}$. Let n be the maximum length of the right-hand side of a production in P. An nc-lfg $G' = (N', \Delta, P', S', N_{atr}, A_{atm})$ is constructed as follows.

Step 1:
$$\begin{aligned} N' &= \{C^{[j]} \mid C \in N, 1 \leq j \leq m\} \\ &\quad \cup \{a^{[j]} \mid a \in T, 1 \leq j \leq m\}, \\ S' &= S^{[1]}, \\ N_{atr} &= \{atr_i \mid 1 \leq i \leq n\} \cup \{prod\}, \text{ and} \\ A_{atm} &= \{p \mid p \text{ is (the label of) a production in } P\} \cup \{nil\}. \end{aligned}$$

Each pair of a symbol (either nonterminal or terminal) X of G and a state q_j of M is represented by a single nonterminal $X^{[j]}$ in G'.

A derivation tree $t = p(t_1, \cdots, t_h)$ in G is represented by an f-structure $\{\langle prod, p\rangle, \langle atr_1, F_1\rangle, \cdots, \langle atr_h, F_h\rangle\}$ of G' where F_i $(1 \leq i \leq h)$ is an f-structure which represents the subtree t_i recursively.

Step 2: A move when M at state q_j reads a symbol p which is the label of a production $p : C \rightarrow \cdots$, can be simulated by a production in G' whose left-hand side is $C^{[j]}_{\{\uparrow prod = p\}}$.

The set P' of productions of G' is formally defined as follows.

(i) Let $p : C \rightarrow X_1 \cdots X_h$ be a production in P where $C \in N$, $X_i \in N \cup T$ $(1 \leq i \leq h)$, and let:

$$r_j \ : \ q_j[p(x_1, \cdots, x_h)] \rightarrow \alpha_{j0} q_{\mu(j,1)}[x_{\nu(j,1)}] \alpha_{j1} \cdots q_{\mu(j,L_j)}[x_{\nu(j,L_j)}] \alpha_{jL_j}$$

be a rule in R where $\alpha_{jk} \in \Delta^*$ $(0 \leq k \leq L_j)$, $q_{\mu(j,l)} \in Q$, and $x_{\nu(j,l)} \in \{x_1, \cdots, x_h\}$ $(1 \leq l \leq L_j)$.

Then, the following production belongs to P':

$$C^{[j]}_{\{\uparrow prod = p\}} \rightarrow \alpha_{j0} \, X^{[\mu(j,1)]}_{\nu(j,1)\ \{\uparrow atr_{\nu(j,1)} = \downarrow\}} \, \alpha_{j1} \cdots \, X^{[\mu(j,L_j)]}_{\nu(j,L_j)\ \{\uparrow atr_{\nu(j,L_j)} = \downarrow\}} \, \alpha_{jL_j}.$$

(ii) Let $q_j[a] \to \beta_j$ be a rule in R where $a \in T$ and $\beta_j \in \Delta^*$. Then the production $a^{[j]} \to \beta_j$ belongs to P'.

(iii) No other production belongs to P'. □

Lemma 4.6: Let G' be an nc-lfg constructed from an fsts (M, G) by **TRANS 2**.

(i) Let $a \in T$ and $w \in \Delta^*$. Then, the following **(C1.1)** and **(C1.2)** are equivalent.

(C1.1) $q_j[a] \underset{M}{\Rightarrow} w$.

(C1.2) There exists a derivation tree τ of $a^{[j]} \underset{G'}{\Rightarrow} w$ such that $F_v = \emptyset$ where v is the root of τ.

(ii) Let $J > 0$, $w_j \in \Delta^*$ $(1 \leq j \leq J)$ and

$$p : C \to X_1 \cdots X_h \tag{4.12}$$

be a production in P. Then, the following **(C2.1)** and **(C2.2)** are equivalent.

(C2.1) There exists a tree $p(t_1, \cdots, t_h) \in \mathcal{R}(G)$ such that

$$q_{i_j}[p(t_1, \cdots, t_h)] \underset{M}{\overset{*}{\Rightarrow}} w_j (1 \leq j \leq J).$$

(C2.2) There exist derivation trees τ_j $(1 \leq j \leq J)$ of $C^{[i_j]} \underset{G'}{\overset{*}{\Rightarrow}} w_j$ in G' $(1 \leq i_j \leq m)$ such that

$$F_{v_1} = \cdots = F_{v_J} \tag{4.13}$$

where v_j $(1 \leq j \leq J)$ is the root of τ_j.

Proof of (i). It can be easily shown by **step 2** (ii) of **TRANS 2**.

Proof of (ii). **(C2.1)** $\Rightarrow$ **(C2.2)** : Suppose that **(C2.1)** holds. That is, there exist a tree $p(t_1, \cdots, t_h) \in \mathcal{R}(G)$ and rules

$$q_{i_j}[p(x_1, \cdots, x_h)] \rightarrow \alpha_{j0} q_{\mu(j,1)}[x_{\nu(j,1)}]\alpha_{j1} \cdots q_{\mu(j,L_j)}[x_{\nu(j,L_j)}]\alpha_{jL_j} \quad (4.14)$$

for $1 \leq j \leq J$, such that

$$\begin{aligned} q_{i_j}[p(t_1, \cdots, t_h)] &\underset{M}{\Rightarrow} \alpha_{j0} q_{\mu(j,1)}[t_{\nu(j,1)}]\alpha_{j1} \cdots q_{\mu(j,L_j)}[t_{\nu(j,L_j)}]\alpha_{jL_j} \\ &\overset{*}{\underset{M}{\Rightarrow}} \alpha_{j0} w_{j1}\alpha_{j1} \cdots w_{jL_j}\alpha_{jL_j} = w_j. \end{aligned} \quad (4.15)$$

We show **(C2.2)** by induction on the number $\eta(t) > 1$ of symbols which appear in t. By (4.12), (4.14) and **step 2** (i) of **TRANS 2**, for $1 \leq j \leq J$ there exists a production

$$\begin{array}{ccccccc} C^{[i_j]} & \rightarrow & \alpha_{j0} & X_{\nu(j,1)}^{[\mu(j,1)]} & \alpha_{j1} \cdots & X_{\nu(j,L_j)}^{[\mu(j,L_j)]} & \alpha_{jL_j} \\ \{\uparrow prod = p\} & & & \{\uparrow atr_{\nu(j,1)} = \downarrow\} & & \{\uparrow atr_{\nu(j,L_j)} = \downarrow\} & \end{array} \quad (4.16)$$

in P'. Consider the subterm t_i $(1 \leq i \leq h)$ of $t = p(t_1, \cdots, t_h)$. There are two cases.

(1) Assume that $\eta(t_i) = 1$, i.e., $t_i = a$ for some $a \in T$. If $t_{\nu(j,l)} = t_i (= a)$ in (4.15), then by (i) of this lemma, there exists a derivation tree τ_{jl} of

$$a^{[\mu(j,l)]} \underset{G'}{\Rightarrow} w_{jl} \quad (4.17)$$

such that

$$F_{v_{jl}} = \emptyset \quad (4.18)$$

where v_{jl} is the root of τ_{jl}.

(2) Assume that $\eta(t_i) > 1$ and $t_i = t_{\nu(j_1,l_1)} = \cdots = t_{\nu(j_s,l_s)}$ in (4.15). We can write $t_i = p'(t'_1, \cdots, t'_k)$ for some production

$$p' \ : \ D \rightarrow Y_1 \cdots Y_k$$

in P. By inductive hypothesis, there exist derivation trees $\tau_{j_u l_u} (1 \leq u \leq s)$ of

$$D^{[\mu(j_u, l_u)]} \underset{G'}{\overset{*}{\Rightarrow}} w_{j_u l_u} \tag{4.19}$$

in G' such that

$$F_{v_{j_1 l_1}} = F_{v_{j_2 l_2}} = \cdots = F_{v_{j_s l_s}} (= F_i) \tag{4.20}$$

where $v_{j_u l_u}$ is the root of $\tau_{j_u l_u}$. For $1 \leq j \leq J$, let τ_j be a tree such that

- the production (4.16) is applied at the root v_j of τ_j,
- the f-structure of v_j is

$$F_{v_j} = \{\langle prod, p\rangle, \langle atr_1, F_1\rangle, \cdots, \langle atr_h, F_h\rangle\},$$

  and,

- the subtree of which the root is labeled with $X^{[\mu(j,l)]}_{\nu(j,l)}$ in (4.16) is τ_{jl}.

Then, by (4.16) through (4.20), τ_j is a derivation tree of

$$C^{[i_j]} \underset{G'}{\overset{*}{\Rightarrow}} \alpha_{j0} w_{j1} \alpha_{j1} \cdots w_{jL_j} \alpha_{jL_j} = w_j$$

in G'. This completes the induction.

The converse (**(C2.2)** $\Rightarrow$ **(C2.1)**) can be proved by induction on the total number $s\,(\geq 1)$ of internal nodes of derivation trees in $\tau_1, \cdots, \tau_J$. □

Lemma 4.7: $y\mathcal{L}_{fsts} \subseteq \mathcal{L}_{nc\text{-}lfg}$.

Proof. Let $G' = (N', \Delta, P', S^{[1]}, N_{atr}, A_{atm})$ be an nc-lfg constructed from an fsts (M, G) by **TRANS 2**, where $G = (N, T, P, S)$ and $M = (Q, \Sigma, \Delta, q_1, R)$. By Lemma 4.6 (ii), for a production $p : S \rightarrow X_1 \cdots X_h$ in P and $w \in \Delta^*$, $q_1[p(t_1, \cdots, t_h)] \underset{M}{\overset{*}{\Rightarrow}} w$ for some $p(t_1, \cdots, t_h) \in \mathcal{R}(G)$ iff there exists a derivation tree of $S^{[1]} \underset{G'}{\overset{*}{\Rightarrow}} w$ in G'. □

Corollary 4.8: $y\mathcal{L}_{d\text{-}fsts} \subseteq \mathcal{L}_{dc\text{-}lfg}$.

Proof. Assume that an fsts (M, G) in **TRANS 2** is a d-fsts. For each pair of a rule $p : C \to \cdots$ in P and a state q_j, there is at most one rule of the form $q_j[p(\cdots)] \to \cdots$. Hence, by **step 2** (i), there exists at most one production whose left-hand sides is $\overset{C^{[j]}}{\underset{\{\uparrow prod = p\}}{\cdot}}$. Similarly, for each pair of a terminal $a \in T$ and a state q_j, there are no distinct rules of the form $q_j[a] \to \cdots$. It follows from **step 2** (ii) that there exists at most one production whose left-hand side is $a^{[j]}$. Hence, the nc-lfg G' constructed by **TRANS 2** is a dc-lfg. This, together with Lemma 4.7, implies the corollary. □

Corollary 4.9: $y\mathcal{L}_{rl\text{-}fsts} \subseteq \mathcal{L}_{rl\text{-}lfg}$.

Proof. In **step 2** (i) of **TRANS 2**, the left-hand side of the productions of G' constructed from a production p of G has the V schema $\uparrow prod = p$. Hence, the productions constructed from a same production $p : C \to \cdots$ are consistent, and any two productions constructed from distinct productions $p_1 : C_1 \to \cdots$ and $p_2 : C_2 \to \cdots$ are always inconsistent. Let p be a production of G. If (M, G) is an rl-fsts, then for each pair of a variable x_i and a state q_j, $q_j[x_i]$ appears at most once in the right-hand sides of rules $r_1, r_2, \cdots$ for p in **step 2** (i). Hence, $\overset{X_i^{[j]}}{\underset{\{\uparrow atr_i = \downarrow\}}{\cdot}}$ appears at most once in the right-hand sides of productions constructed in **step 2** (i). It follows from the definition of rl-lfg that the constructed nc-lfg G' is an rl-lfg. Therefore, Lemma 4.7 implies the corollary. □

For an rl-fsts (M, G) where $M = (Q, \Sigma, \Delta, q_1, R)$ and $G = (N, T, P,$

S), let $G' = (N', T', P', S', N_{atr}, A_{atm})$ be an equivalent rl-lfg constructed by **TRANS 2**.

The sizes of the components of G' are:

$$\left.\begin{array}{rcl} |N'| &=& (|N| + |T|) \times |Q| \\ |T'| &=& |\Delta| \\ ||P'|| &=& O(||R||) \\ |N_{atr}| &=& O(||max(P)||) \\ |A_{atm}| &=& |P| \end{array}\right\} \tag{4.21}$$

Example 4.2: For rl-fsts (M, G') in Example 5.2, the following rl-lfg $G'' = (N'', T'', P'', S'', N_{atr}, A_{atm})$ is constructed.

$$N'' = \{X^{[1]}, X^{[2]} \mid X \in N' \cup T'\},\ T'' = \{a, b, c, d\},$$
$$S'' = S^{[1]},\ N_{atr} = \{atr_1, atr_2, prod\},$$
$$A_{atm} = \{p_0, \cdots, p_4, \bar{p_0}, \cdots, \bar{p_4}\},$$

$$P'' = \left\{\begin{array}{ll} S^{[1]}_{p_0} \rightarrow A^{[1]} B^{[1]} & A^{[2]} B^{[2]} \\ \{\uparrow prod = p_0\}\{\uparrow atr_1 = \downarrow\}\{\uparrow atr_2 = \downarrow\} & \{\uparrow atr_1 = \downarrow\}\{\uparrow atr_2 = \downarrow\}, \\ A^{[1]}_{p_1} \rightarrow a\ A^{[1]} & A^{[2]}_{p_1} \rightarrow c\ A^{[2]} \\ \{\uparrow prod = p_1\}\ \{\uparrow atr_1 = \downarrow\}, & \{\uparrow prod = p_1\}\ \{\uparrow atr_1 = \downarrow\}, \\ B^{[1]}_{p_2} \rightarrow b\ B^{[1]} & B^{[2]}_{p_2} \rightarrow d\ B^{[2]} \\ \{\uparrow prod = p_2\}\ \{\uparrow atr_1 = \downarrow\}, & \{\uparrow prod = p_2\}\ \{\uparrow atr_1 = \downarrow\}, \\ A^{[1]}_{p_3} \rightarrow \varepsilon, & A^{[2]}_{p_3} \rightarrow \varepsilon, \\ \{\uparrow prod = p_3\} & \{\uparrow prod = p_3\} \\ B^{[1]}_{p_4} \rightarrow \varepsilon, & B^{[2]}_{p_4} \rightarrow \varepsilon \\ \{\uparrow prod = p_4\} & \{\uparrow prod = p_4\} \\ S^{[1]} \rightarrow S^{[1]}_{p_0} & S^{[2]} \rightarrow S^{[2]}_{p_0} \\ \{\uparrow prod = \bar{p_0}\}\ \{\uparrow atr_1 = \downarrow\}, & \{\uparrow prod = \bar{p_0}\}\ \{\uparrow atr_1 = \downarrow\}, \\ \vdots & \\ B^{[1]} \rightarrow B^{[1]}_{p_4} & B^{[2]} \rightarrow B^{[2]}_{p_4} \\ \{\uparrow prod = \bar{p_4}\}\ \{\uparrow atr_1 = \downarrow\}, & \{\uparrow prod = \bar{p_4}\}\ \{\uparrow atr_1 = \downarrow\} \end{array}\right\}.$$

□

Lemma 4.10: Let (M, G) $(M = (Q, \Sigma, \Delta, q_1, R))$ be a d-fsts with dimension m $(m \geq 2)$ and with degree e $(e \geq 3)$. Assume that there exists an input symbol $\sigma_m \in \Sigma$ such that for each $q \in Q$ there exists a rule whose left-hand side is $q[\sigma_m(x_1, \cdots, x_n)]$ in R. Then, the dc-lfg G' constructed from G by **TRANS 2** is with dimension m and degree e.

Proof. Let $M = (Q, \Sigma, \Delta, q_1, R)$ and let $G = (N, T, P, S)$.

(*for dimension*): By **TRANS 2**, for each input tree symbol $\sigma \in \Sigma$, the set of rules of the form $q[\sigma(\cdots)] \rightarrow \cdots$ is translated into a maximal subset of consistent productions of dc-lfg G' because of the V schema $\uparrow prod = p$ in **step 2** (i). Hence,

$$\max_{sp \in SP} |sp| = |Q| \, (= m), \tag{4.22}$$

by the assumption for the symbol σ_m, where SP is the set of consistent subset of productions of G'. Therefore, G' is an dc-lfg with dimension m, since the dimension of G' is defined as $\max_{sp \in SP} |sp|$.

(*for degree*): By **TRANS 2**, for each production $p \in P$, the set of rules of the form $q[p(\cdots)] \rightarrow \cdots$ is translated into a maximal subset of consistent productions in G'. Thus, the degree of sp is equal to the degree of p by the definition. It follows that the degree of G' is equal to the degree of (M, G). □

Theorem 4.11: $\mathcal{L}_{nc\text{-}lfg} = y\mathcal{L}_{fsts}$.

Proof. By Lemmas 4.2 and 4.7. □

Theorem 4.12: $\mathcal{L}_{dc\text{-}lfg} = y\mathcal{L}_{d\text{-}fsts}$.

Proof. By Corollaries 4.3 and 4.8. □

Theorem 4.13: $\mathcal{L}_{rl\text{-}lfg} = y\mathcal{L}_{rl\text{-}fsts}$.

Proof. By Corollaries 4.4 and 4.9. □

Corollary 4.14: The class of languages generated by monadic nc-lfg's (monadic dc-lfg's, rl-lfg's, resp.) is equal to the class of yield languages generated by ET0L systems (EDT0L systems, ET0L_{RL} systems, resp.).

Proof. We only sketch the proof for monadic nc-lfg's = ET0L systems. The remaining two equalities can be proved in the same way. In **TRANS 2**, if a given fsts (M, G) is an ET0L system, then the set of attributes of the constructed nc-lfg G' is $N_{atr} = \{atr_1\} \cup \{prod\}$. By **step 2** of **TRANS 2**, G' satisfies Monadicness condition in the definition of monadic nc-lfg with letting $atr_S = atr_1$ and $atr_V = prod$.

Conversely, assume that $G = (N, T, P, S, N_{atr} = \{atr_S, atr_V\}, A_{atm})$ is a monadic nc-lfg. If we translate G by using **TRANS 1**, the resulting fsts is not an ET0L, since $n = |N_{atr}| = 2$. However, atr_S appears only in S schemata and atr_V appears only in V schemata, we can modify **TRANS 1** so that it translates a monadic nc-lfg into an equivalent ET0L system in the following way. Let $atr_1 = atr_S$ in **TRANS 1**.

- In **step 1**, let

$$\begin{aligned} P' \;=\; & \{p'_{start(\bar{sp})} : S' \to S_{\bar{sp}} \\ & \quad | \bar{sp} \in SP \text{ and } \bar{sp} \text{ includes a production} \\ & \qquad\qquad \text{whose left-hand side is } S\} \\ & \cup \{p'_{sp} : S_{sp} \to Z\} \\ & \cup \{p'_{guess(sp)} : Z \to S_{sp} | sp \in SP\} \\ & \cup \{p'_{term} : Z \to d\}. \end{aligned}$$ □

5. Universal Recognition Problems for Dc-lfg's and Rl-lfg's

5.1 Universal Recognition of Dc-lfg's/Rl-lfg's

In this section, the universal recognition problem for dc-lfg's and that for rl-lfg's are shown to be EXP-POLY time-complete, where

EXP-POLY time $\stackrel{\triangle}{=}$

$\{L \mid$ there exist some $c > 1$ and some polynomial p such that,

for a given w, whether $w \in L$ is solvable in deterministic

$O(c^{p(|w|)})$ time $\}$.

As shown in Table 2, the problems for pmcfg's and mcfg's are known to be EXP-POLY time complete. First, the problem for dc-lfg's is shown to be solvable in EXP-POLY time by translating a given dc-lfg G into an equivalent d-fsts (M, G') and then translating (M, G') into an equivalent pmcfg G''.

For a grammar G (or an fsts (M, G)), $||G||$ (or $||(M, G)||$) denotes the description length of G (or (M, G)), under appropriate encoding.

[dc-lfg $\Rightarrow$ d-fsts]

For a given dc-lfg G, an equivalent d-fsts (M, G') can be constructed by **TRANS 1** in Section **4.1**.

[d-fsts $\Rightarrow$ pmcfg]

For a given d-fsts (M, G'), where $M = (Q, \Sigma, \Delta, q_1, R)$ and $G' = (N', T', P', S')$, a pmcfg $G'' = (N'', T'', F, P'', S'')$ such that $yL(M, G') = L(G'') \cap \Delta^*$ can be constructed by the translation method in Section 3.1 in [13]. Obviously, for a string $w \in \Delta^*$, $w \in yL(M, G')$ iff $w \in L(G'')$.

Note that, in Section 3.1 in [13], for each production p' in P', a nonterminal, say $R_{p'}$, of pmcfg is added to N''. Let

$$p' : A \to Y_1 \cdots Y_k \in P' \tag{5.23}$$

be a production where $A \in N', Y_1, \cdots, Y_k \in N' \cup T'$. For each combination $Z_1, \cdots, Z_k$ of nonterminals in P'' that satisfies

(a) if $Y_i \in T'$ then $Z_i = Y_i$, and,

(b) if $Y_i \in N'$ then $Z_i = R_{p'_i}$ where the left-hand side of p'_i is Y_i,

the following production of pmcfg is constructed:

$$R_{p'} \to f_{p'}[Z_1, \cdots, Z_k],$$

where $f_{p'}$ is constructed from rules $r : q[p'(\cdots)] \to \cdots$ in R ($q \in Q$), such that p' appears in their left-hand sides. The number of nonterminals in the right-hand side of a production in P' is $O(||G'||)$ and, the number of productions whose left-hand side is Y_i is also $O(||G'||)$. Hence, $||G''||$ may be exponential to $||G'||$. However, $f_{p'}$ has enough information that the above production p' in P' and rules r have, even if this construction is modified in such a way that N'' has a one-to-one correspondence with $N' \cup T' \cup \{S''\}$ (S'' is a newly introduced start symbol) and, for a production (5.23), the following production of pmcfg is constructed:

$$A \to f_{p'}[Y_1, \cdots, Y_k].$$

S'' is used for extracting the component that corresponds to the initial state (see Example 5.1). Here, we use this modified construction. The

sizes of the components of G'' are:

$$\left.\begin{array}{rcl} |N''| &=& |N'| + |T'| + 1 \\ |T''| &=& |\Delta| + 1 \\ ||P''|| &=& O(|P''|) \times O(||max(P'')||) \\ &=& O((|P'| + |T'|) \times ||max(P')||) \\ ||F|| &=& O(|F|) \times O(||max(F)||) \\ &=& O(|P'| + |T'|) \times O(|Q| \times ||max(R)||) \end{array}\right\} \quad (5.24)$$

Let m'' be the dimension of G'', then

$$m'' = |Q|. \quad (5.25)$$

Example 5.1: For a d-fsts (M, G') constructed in Example 4.1, the following pmcfg $G'' = (N'', T'', F, P'', S'')$ is constructed.

$N'' = \{S''\} \cup N' \cup \{D\}$, where $d(S'') = d(S') = d(S_{sp1}) = d(D) = 1$ and $d(S_{sp2}) = d(S_{sp3}) = d(Z) = 2$,

$T'' = \Delta \cup \{\$\}$ ($\$$ is a newly introduced symbol),

$$P'' = \left\{\begin{array}{ll} S' \to f_0[S_{sp1}], & S_{sp1} \to f_1[Z, Z], \\ S_{sp2} \to f_2[Z, Z], & S_{sp3} \to f_3[Z, Z], \\ Z \to f_4[S_{sp1}], & Z \to f_5[S_{sp2}], \\ Z \to f_5[S_{sp3}], & Z \to f_6[D], \\ D \to f_7, & S'' \to f_0[S'] \end{array}\right\} and$$

$$F = \left\{\begin{array}{ll} f_0[x_{11}] = x_{11}, & f_1[\bar{x_1}, \bar{x_2}] = x_{11}x_{12}, \\ f_2[\bar{x_1}, \bar{x_2}] = (ax_{11}b, cx_{12}d), & f_3[\bar{x_1}, \bar{x_2}] = (\varepsilon, \varepsilon), \\ f_4[x_{11}] = (x_{11}, \$), & f_5[\bar{x_1}] = \bar{x_1}, \\ f_6[x_{11}] = (\$, \$), & f_7[] = \$ \end{array}\right\},$$

where $\bar{x_i} = (x_{i1}, x_{i2})$ for $i = 1, 2$. □

Lemma 5.1: The universal recognition problem for dc-lfg's is solvable in EXP-POLY time.

Proof. Let $G = (N, T, P, S, N_{atr}, A_{atm})$ and w be a given dc-lfg and a string, respectively, and let $G'' = (N'', T'', F, P'', S'')$ be the pmcfg constructed from G by [**dc-lfg** $\Rightarrow$ **d-fsts**] and [**d-fsts** $\Rightarrow$ **pmcfg**] in this order. Also let m'' be the dimension of G''. A nondeterministic algorithm that solves the universal recognition problem for pmcfg's is presented in [12]. This recognition algorithm is applied to G''. The algorithm consists of two steps. The first step of the algorithm can be executed in

$$O(|N''| \times 2^{m''} \times |P''_N| \times (2^{m''})^{||max(P''_N)||} \times 2^{m''}) \text{ time,} \tag{5.26}$$

where P''_N is the set of nonterminating productions in P'' (see Lemma 2 in [11]). The second step of the algorithm (Algorithm 1 in [11]) can be executed nondeterministically in

$$\begin{aligned} O(log(|N''|) + log(m'') + log(m''|w| + ||max(F)||)) \\ \times O(m''|w| + ||max(F)||) \end{aligned} \tag{5.27}$$

space. Let $n = ||G|| + |w|$ and let m be the dimension of G. By (4.11) and (5.24) through (5.26), the execution of the first step takes

$$O(2^{|P|} \times 2^m \times 2^{|P|} \times (2^m)^{|N_{atr}|} \times 2^m) = O(2^{||G||^2+4||G||}) \text{ time.} \tag{5.28}$$

By (4.11), (5.24), (5.25) and (5.27), the total space needed to execute the second step is

$$\begin{aligned} O(log(2^{|P|}) + log(m) + log(m \times |w| + m \times ||max(P)||)) \\ \times O(m \times |w| + m \times ||max(P)||) = O(n^3). \end{aligned} \tag{5.29}$$

By (5.28) and (5.29), the universal recognition problem for dc-lfg's is solvable in EXP-POLY time. □

Next, we shall show that the universal recognition problem for rl-lfg's is EXP-POLY time-hard by reducing the problem for mcfg's to that for rl-lfg's.

[**mcfg** $\Rightarrow$ **rl-fsts**]

If a given mcfg $G = (N, T, F, P, S)$ satisfies non-erasing condition (see Section **2.2**), then an equivalent rl-fsts (M, G') where $M = (Q, \Sigma, \Delta, q_1, R)$ and $G' = (N', T', P', S')$ can be constructed by the translation method in Section 3.2 in [13]. However, even when G does not satisfy non-erasing condition, the above translation method constructs an equivalent rl-fsts from G if G has no production p which contains a non-terminal C such that $L_G(C) = \emptyset$ in its right-hand side. (Here, we will use a construction method which is modified in a similar way to the one stated in [**d-fsts** $\Rightarrow$ **pmcfg**].) Since whether $L_G(A) = \emptyset$ or not can be checked in a similar way to the detection of useless symbols for a given cfg, there exists a procedure that deletes such productions in time polynomial of $||G||$.

The sizes of the components of M and G' are:

$$\left.\begin{array}{rcl} |N'| & = & O(|N| \times |P|) \\ |T'| & = & 1 \\ ||P'|| & = & O(||P||) \\ |Q| & = & m, \text{ the dimension of } G \\ |\Sigma| & = & 2|P| + 1 \\ |\Delta| & = & |T| \\ ||R|| & = & O(|R|) \times O(||max(R)||) \\ & = & O(|P| \times m) \times O(||max(F)||) \end{array}\right\} \quad (5.30)$$

Example 5.2: For a given mcfg $G = (N, T, F, P, S)$, where

$$N = \{S, A, B\}\ (d(S) = 1, d(A) = d(B) = 2),$$

$$T = \{a, b, c, d\},$$

$$P = \left\{ \begin{array}{ll} p_0 : S \rightarrow f_0[A, B], & p_1 : A \rightarrow f_1[A], \\ p_2 : B \rightarrow f_2[B], & p_3 : A \rightarrow f_3, \\ p_4 : B \rightarrow f_3 & \end{array} \right\} \text{ and}$$

$$F = \left\{ \begin{array}{l} f_0[(x_{11}, x_{12}), (x_{21}, x_{22})] \rightarrow x_{11}x_{21}x_{12}x_{22}, \\ f_1[(x_{11}, x_{12})] = (ax_{11}, cx_{12}), \\ f_2[(x_{11}, x_{12})] = (bx_{11}, dx_{12}), \\ f_3[] = (\varepsilon, \varepsilon) \end{array} \right\},$$

the following rl-fsts (M, G') where $M = (Q, \Sigma, \Delta, q_1, R)$ and $G' = (N', T', P', S')$ is constructed.

$$N' = \{S, S_{p_0}, A, A_{p_1}, A_{p_3}, B, B_{p_2}, B_{p_4}\},\ T' = \{\$\},$$

$$P' = \left\{ \begin{array}{lll} p_0 : S_{p_0} \to AB, & p_1 : A_{p_1} \to A, & p_2 : B_{p_2} \to B, \\ p_3 : A_{p_3} \to \$, & p_4 : B_{p_4} \to \$, & \bar{p_0} : S \to S_{p_0}, \\ \bar{p_1} : A \to A_{p_1}, & \bar{p_2} : B \to B_{p_2}, & \bar{p_3} : A \to A_{p_3}, \\ \bar{p_4} : B \to B_{p_4} & & \end{array} \right\},$$

$Q = \{q_1, q_2\}$, $\Sigma = \{p_0, \cdots, p_4, \bar{p_0}, \cdots, \bar{p_4}, \$\}, \Delta = \{a, b, c, d\}$ and

$$R = \left\{ \begin{array}{ll} q_1[p_0(x_1, x_2)] \to q_1[x_1]q_1[x_2]q_2[x_1]q_2[x_2], & \\ q_1[p_1(x)] \to aq_1[x], & q_2[p_1(x)] \to cq_2[x], \\ q_1[p_2(x)] \to bq_1[x], & q_2[p_2(x)] \to dq_2[x], \\ q_j[p_i(x)] \to \varepsilon \text{ for } i = 3, 4 \text{ and } j = 1, 2, & \\ q_j[\bar{p}_i(x)] \to q_j[x] \text{ for } 0 \leq i \leq 4 \text{ and } j = 1, 2 & \end{array} \right\}.$$

□

[**rl-fsts** $\Rightarrow$ **rl-lfg**]

For an rl-fsts (M, G'), an equivalent rl-lfg G'' can be constructed by **TRANS 2** in Section **4.2**.

Lemma 5.2: The universal recognition problem for rl-lfg's is EXP-POLY time-hard.

Proof. For a given mcfg G, an equivalent rl-lfg G'' can be constructed by (5.30) and (4.21) in time polynomial of $||G||$. It is known that the universal recognition problem for mcfg's is EXP-POLY time-hard[11]. Therefore, the universal recognition problem for rl-lfg's is EXP-POLY time-hard. □

Theorem 5.3: (1) The universal recognition problem for dc-lfg's is EXP-POLY time complete.

(2) The universal recognition problem for rl-lfg's is EXP-POLY time complete.

Proof. (1): Since the class of rl-lfg's is a subclass of dc-lfg's by the definition of rl-lfg, the universal recognition problem for dc-lfg's is EXP-POLY time-hard by Lemma 5.2. Containment in EXP-POLY time is from Lemma 5.1. (2): By the same reason as in the proof of (1), the universal recognition problem for rl-lfg's is solvable in EXP-POLY time by Lemma 5.1. This fact together with Lemma 5.2 implies (2). □

Corollary 5.4: The universal recognition problem for d-fsts' and the problem for rl-fsts' are both EXP-POLY time complete. □

5.2 Universal Recognition of Dc-lfg's/Rl-lfg's with Bounded Dimension/Degree

In this section, we shall show that the universal recognition problems for dc-lfg's and rl-lfg's with bounded dimension are $\mathcal{NP}$-complete and that the problems for dc-lfg's and rl-lfg's with bounded degree are $\mathcal{P}$-complete.

Theorem 5.5: [12] For any fixed m, the universal recognition problem for mcfg's with dimension m ($m \geq 2$) or less and that for pmcfg's with dimension m ($m \geq 1$) or less are $\mathcal{NP}$-complete. □

Theorem 5.6: [12] For any fixed e ($e \geq 3$), the universal recognition problem for mcfg's with degree e or less and that for pmcfg's with degree e or less are $\mathcal{P}$-complete. □

Lemma 5.7: Let G be a dc-lfg with dimension m ($m \geq 1$) and with degree e ($e \geq 3$). Then, the pmcfg G'' constructed from G by the translation in Section **5.1** is with dimension m and with degree $\max(e, 2m)$. Furthermore, G'' can be constructed from G in time polynomial of $||G||$.

Proof. Let (M, G') be the d-fsts constructed from G by [**dc-lfg** $\Rightarrow$ **d-fsts**] in Section **5.1**. And let G'' be the pmcfg constructed from (M, G') by [**d-fsts** $\Rightarrow$ **pmcfg**] in Section **5.1**. By Lemma 4.5, (M, G') is a d-fsts with dimension m and with degree $\max(e, 2m)$.

By (5.25), the dimension of G'' is m. In [**d-fsts** $\Rightarrow$ **pmcfg**], for each p'_{sp}, a function f is constructed from the rules of the form $q[p'_{sp}(\cdots)] \rightarrow \cdots$ and the degree of f is equal to the degree of p'_{sp}. Hence, the degree of G'' is equal to the degree of (M, G'). Consequently, the degree of G'' is $\max(e, 2m)$.

By the definition of the degree of a dc-lfg, $|SP| = O(|P|^m)$, which is a polynomial of $||G||$ when m is a constant. This fact together with (4.11) and (5.24) implies that $||G''||$ is a polynomial of $||G||$. It can be easily shown that the time consumed in [**dc-lfg** $\Rightarrow$ **d-fsts**] and [**d-fsts** $\Rightarrow$ **pmcfg**] is a polynomial of $||G||$. □

Lemma 5.8: Let G be an mcfg with dimension m ($m \geq 2$) and with degree e ($e \geq 3$). Then, the rl-lfg G'' constructed from G by the translation in Section **5.1** is with dimension m and degree $\max(e, 2m)$.

Proof. Let (M, G') be the rl-fsts constructed from $G = (N, T, F, P, S)$ by [**mcfg** $\Rightarrow$ **rl-fsts**] in Section **5.1**, where $M = (Q, \Sigma, \Delta, q_1, R)$ and $G' = (N', T', P', S')$. And let G'' be the rl-lfg constructed from (M, G') by [**rl-fsts** $\Rightarrow$ **rl-lfg**] in Section **5.1**.

By (5.30) and Lemma 4.10, G'' is an rl-lfg with dimension m. In [**mcfg** $\Rightarrow$ **rl-fsts**], for each production $p : A \rightarrow f[\cdots]$ in G, the set of rules of the form $q[p(\cdots)] \rightarrow \cdots$ is constructed and the degree of p is equal to the degree of f. Also, for each $A \in N$ and $p \in P$ whose left-hand side is A, a production $p' : A \rightarrow A_p$ and rules $q[p'(x)] \rightarrow q[x]$ ($q \in Q$) are constructed. Thus, the degree of (M, G') is $\max(e, 2m)$. This fact together with Lemma 4.10 implies that G'' is the rl-lfg with degree $\max(e, 2m)$. □

Theorem 5.9: (1) For any fixed m ($m \geq 1$), the universal recognition problem for dc-lfg's with dimension m or less is $\mathcal{NP}$-complete.

(2) For any fixed m ($m \geq 2$), the universal recognition problem for rl-lfg's with dimension m or less is $\mathcal{NP}$-complete.

Proof. (1): Containment in $\mathcal{NP}$ is from Lemma 5.7 and Theorem 5.5. For $\mathcal{NP}$-hardness, use Lemma 5.8, Theorem 5.5 and the fact that the class of rl-lfg's is a subclass of dc-lfg's. By (5.30) and (4.21), the reduction can be done in polynomial time. We can prove (2) in the same way. □

Corollary 5.10: For any fixed m ($m \geq 1$), the universal recognition problem for d-fsts' with dimension m or less is $\mathcal{NP}$-complete. For any fixed m ($m \geq 2$), the universal recognition problem for rl-fsts' with dimension m or less is $\mathcal{NP}$-complete. □

Theorem 5.11: (1) For any fixed e ($e \geq 3$), the universal recognition problem for dc-lfg's with degree e or less is $\mathcal{P}$-complete.

(2) For any fixed e ($e \geq 3$), the universal recognition problem for rl-lfg's with degree e or less is $\mathcal{P}$-complete.

Proof. (1): By the definition of the degree of a dc-lfg, if the degree is bounded, then the dimension is also bounded. This fact together with Lemma 5.7 implies that a dc-lfg with bounded degree can be translated into an equivalent pmcfg with bounded degree in polynomial time. Hence, the universal recognition problem for dc-lfg's with degree e is in $\mathcal{P}$ by Theorem 5.6. $\mathcal{P}$-hardness is from the fact that cfg's can be considered as a proper subclass of rl-lfg's with degree three or less, and the universal recognition problem for cfg's is $\mathcal{P}$-complete. We can prove (2) in the same way. □

Corollary 5.12: For any fixed e ($e \geq 3$), the universal recognition problem for d-fsts' with degree e or less, and the problem for rl-fsts' with degree e or less are both $\mathcal{P}$-complete. □

6. Conclusion

In this paper, three subclasses of lfg's called nc-lfg's, dc-lfg's and rl-lfg's are proposed. Also, relations between the generative capacity of these subclasses, pmcfg's and fsts' are clarified. Results are shown as equalities labeled with $*5$ in Figure 1. As a corollary, it is shown that the languages generated by dc-lfg's and rl-lfg's can be recognized in polynomial time.

In the last part of this paper, complexity of the universal recognition problems for dc-lfg's/rl-lfg's, dc-fsts'/rl-fsts' and their subclasses are investigated. Results are summarized in Table 2.

Pmcfg's, fsts', and lfg's were proposed from different motivation and developed independently. Nevertheless, as shown in Table 2, we can define "common" parameters, namely, dimension and degree, among these grammars. Furthermore, the complexity of the universal recognition problems for subclasses of these three classes of grammars, with one of the "common" parameters bounded, are the same. This fact tells that these three classes of grammars resemble in structure.

References

[1] Ando, S., Nakanishi, R., Seki, H. and Kasami, T. 1993: "A Polynomial-Time Recognizable Subclasses of Lexical-functional Grammars", Submitted to IEICE Trans. Inf. and Syst. (conditionally accepted on Feb. 1994), also available as ICS Research Report, **93-ICS-6**, Department of Information and Computer Sciences, Osaka University (1993).

[2] Ando, S., Nakanishi, R., Seki, H. and Kasami, T. 1993: "The Universal Recognition Problems for Some Subclasses of Lexical-Functional Grammars", Information Science Technical Report, **NAIST-IS-TR93011**, Graduate School of Information Science, Nara Institute of Science and Technology, also available as IEICE Technical Report, **COMP93-67** (1994).

[3] Barton, G.E., Berwick, R.C. and Ristad, E.S. 1987: "Computational Complexity and Natural Language", MIT Press.

[4] Engelfriet, J., Rosenberg, G. and Slutzki, G. 1980: "Tree Transducers, L Systems, and Two-Way Machines", J. Comput. & Syst. Sci., **20**:150–202.

[5] Engelfriet, J. 1986: " The Complexity of Languages Generated by Attribute Grammars", SIAM J. Comput., **15**, 1:70–86.

[6] Engelfriet, J. and Heyker, L. 1991: "The String Generating Power of Context-Free Hypergraph Grammars", J. Comput. & Syst. Sci., **43**:328–360.

[7] Jones, N.D. and Laaser, W.T. 1976: "Complete Problems for Deterministic Polynomial Time", Theoretical Computer Science, **3**, 1:105–118.

[8] Gazdar, G., Klein, E., Pullum, G. and Sag, I. 1985: "Generalized Phrase Structure Grammar", Basil Blackwell.

[9] Joshi, A.K., Levy, L. and Takahashi, M. 1975: "Tree Adjunct Grammars", J. Comput. & Syst. Sci., **10**:136–163.

[10] Joshi, A.K., Vijay-Shanker, K. and Weir, D.J. 1988: "The Convergence of Mildly Context-Sensitive Grammar Formalisms", Technical Report, University of Pennsylvania.

[11] Kaji, Y., Nakanishi, R., Seki, H. and Kasami, T. 1992a: "The Universal Recognition Problem for Multiple Context-Free Grammars and for Linear Context-Free Rewriting Systems", IEICE Trans. Inf. and Syst., **E75-D**, 1:78–88.

[12] Kaji, Y., Nakanishi, R., Seki, H. and Kasami, T. 1992b: "The Universal Recognition Problem for Parallel Multiple Context-Free Grammars and for Their Subclasses", IEICE Trans. Inf. and Syst., **E75-D**, 4:499–508.

[13] Kaji, Y., Seki, H. and Kasami, T. 1992c: "Parallel Multiple Context-Free Grammars and Finite State Translation Systems", IEICE Technical Report, **COMP92-34**, also in IEICE Trans. Inf. and Syst. (to appear).

[14] Kaplan, R. and Bresnan, J. 1982: "Lexical-Functional Grammar", The Mental Representation of Grammatical Relations, Bresnan, J. (ed.), MIT Press:173–281.

[15] Karp, R.M. 1972: "Reducibility Among Combinatorial Problems", Complexity of Computer Computations, Plenum Press:85–104.

[16] Kasami, T., Seki, H. and Fujii, M. 1988a: "Generalized Context-Free Grammar and Multiple Context-Free Grammar", Trans. IEICE, **J71-D-I**, 5:758–765 (in Japanese).

[17] Kasami, T., Seki, H. and Fujii, M. 1988b: "On the Membership Problem for Head Languages and Multiple Context-Free Languages ", Trans. IEICE, **J71-D-I**, 6:935–941 (in Japanese).

[18] Kay, M. 1983: "Unification Grammar", Technical Report, Xerox Palo Alto Research Center.

[19] Nakanishi, R., Seki, H. and Kasami, T. 1992: "On the Generative Capacity of Lexical-Functional Grammars", IEICE Trans. Inf. and Syst., **E75-D**, 7:509–516.

[20] Nishino, T. 1991: "Mathematical Analysis of Lexical-Functional Grammars —Complexity, Parsability, and Learnability—", Research Report **TDU-IS-24**, Tokyo Denki University, also in Language Research, **27**, 1, Language Research Institute, Seoul National University (1991).

[21] Nishino, T. 1988: "Relating Attribute Grammars and Lexical-Functional Grammars", Research Report **TDU-IS-7**, Tokyo Denki University, also in Information Sciences, **66**:1–22 (1992).

[22] Pollard, C.J. 1984: "Generalized Phrase Structure Grammars, Head Grammars and Natural Language", Ph.D. dissertation, Stanford University.

[23] Pollard, C.J. 1985: "Lectures on HPSG", Unpublished manuscript, Center for the Study of Language and Information, Stanford University.

[24] Rounds, W.C. 1970: "Mappings and Grammars on Trees", Mathematical Systems Theory, **4**, 3:257–287.

[25] Rounds, W.C. 1973: "Complexity of Recognition in Intermediate-Level Languages", IEEE 14th Annual Symp. on SWAT., 145–158.

[26] Seki, H., Matsumura, T., Fujii, M. and Kasami, T. 1991: "On Multiple Context-Free Grammars", Theoretical Computer Science, **88**, 2:191–229.

[27] Seki, H., Nakanishi, R., Kaji, Y., Ando, S. and Kasami, T. 1993: "Parallel Multiple Context-Free Grammars, Finite-State Translation Systems, and Polynomial-Time Recognizable Subclasses of Lexical-Functional Grammars", Proc. of 31st Annual Meeting of Assoc. Comput. Ling., 130–139.

[28] Tanaka, S. and Kasai, T. 1985: "The Emptiness Problem for Indexed Languages is Exponential Time Complete", Trans. IEICE, **J68-D**, 10:1727–1734 (in Japanese).

[29] Thatcher, J.W. 1967: "Characterizing Derivation Trees of Context-Free Grammars through a Generalarization of Finite Automata Theory", J. Comput. & Syst. Sci., **1**:317–322.

[30] Vijay-Shanker, K., Weir, D.J. and Joshi, A.K. 1986: "Tree Adjoining and Head Wrapping", Proc. of 11th Intl. Conf. on Comput. Ling., 202–207.

[31] Vijay-Shanker, K., Weir, D.J. and Joshi, A.K. 1987a: "Characterizing Structural Descriptions Produced by Various Grammatical Formalisms", Proc. of 25th Annual Meeting of Assoc. Comput. Ling., 104–111.

[32] Vijay-Shanker, K. 1987b: "A Study of Tree Adjoining Grammars", Ph.D. thesis, University of Pennsylvania.

[33] Weir, D.J. 1988: "Characterizing Mildly Context-Sensitive Grammar Formalisms", Ph.D. thesis, University of Pennsylvania.

[34] Weir, D.J. 1992: "Linear Context-Free Rewriting Systems and Deterministic Tree-Walking Transducers", Proc. of 30th Annual Meeting of Assoc. Comput. Ling.