The Universal Recognition Problems for Some Subclasses of Lexical-Functional Grammars

Sachiko Ando[†], Ryuichi Nakanishi[†], Hiroyuki Seki[†‡] 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 : ando@ics.es.osaka-u.ac.jp

Abstract The computational complexity of the universal recognition problems for dc-lfg's and for rl-lfg's are investigated. For a class $\mathcal{G}$ of grammars, the universal recognition problem for $\mathcal{G}$ is the one to decide whether G can generate w for a given grammar $G \in \mathcal{G}$ and a string w. We show that the universal recognition problems for dc-lfg's and for rl-lfg's are 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. The time complexity of the universal recognition problems for some subclasses of finite state translation systems are also investigated.

key words lexical-functional grammars, parallel multiple context-free grammars, finite state translation systems, universal recognition problem, computational complexity

1 Introduction

Since 1980's, a number of phrase-structure grammars without transformation have been proposed to define the syntax of natural languages. They are also called unification-based grammars since string derivation is defined in terms of unification operation. There is another stream of grammatical formalism called mildly context-sensitive grammars, such as tree-adjoining grammar, head grammar, *parallel multiple context-free grammar* (*pmcfg*) and *multiple context-free grammar* (*mcfg*)[11, 15]. They have common properties: the generative capacity is between context-free grammar (cfg) and context-sensitive grammar (csg); they are recognizable in polynomial time, phrase structure is simply defined (locality of derivation), and so on.

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* [14] 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*)[14] is a tree transducer with its input domain being the set of derivation trees of a cfg.

Lexical-functional grammar (*lfg*)[9] 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 cfg G_0 called the *underlying cfg* of G and a finite set P_{fs} of equations called *functional schemata* which specify constraints between the f-structures of nodes in a derivation tree. Functional schemata are attached to symbols in productions of G_0. It was shown in [12] 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 [16], subclasses of lfg's, called *nondeterministically copying lfg's* (*nc-lfg's*) and *deterministically copying lfg's* (*dc-lfg's*) were proposed, and the class of dc-lfg's was shown to be recognizable in polynomial time. In this paper, a subclass of dc-lfg's called *right-linear lfg's* (*rl-lfg's*) is proposed. Figure 1 summarizes known results on the generative capacities of pmcfg's, fsts' and these subclasses of lfg's. The

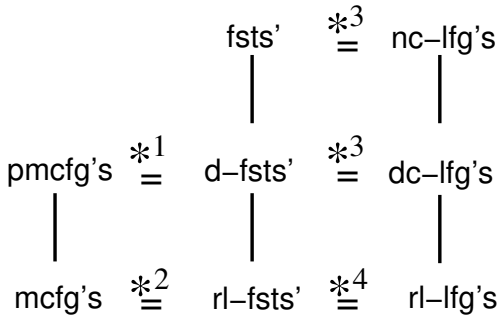

Figure 1: Generative Capacities

equivalence relation $*1$ is shown in [8], the relation $*2$ is a corollary of $*1$, the relations $*3$ are shown in [16], and the relation $*4$ is a corollary of $*3$.

In this paper, the computational complexity of the universal recognition problems for dc-lfg's and for 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 problems. 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 [2] and [6]). For example, although the class of languages generated by generalized phrase structure grammars (gpsg's)[5] 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[2]. 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. And Table 1 summarizes known results on the problems for other classes of grammars, where rlfg's[13] denotes a subclass of lexical functional grammars and ig's denotes the class of indexed

Table 1: Known results

grammars	complexity of universal recognition
cfg's	$\mathcal{P}$-complete[4]
gpsg's[5]	EXP-POLY time-hard[2]
rlfg's[13]	$\mathcal{NP}$-hard[2]
ig's	EXP-POLY time-complete[17]
csg's	PSPACE-complete[10]

grammars.

In **3**, we show that the universal recognition problems for dc-lfg's and for rl-lfg's are EXP-POLY time-complete. It is also shown in **4** that, for a fixed m, the universal recognition problems for dc-lfg's with dimension at most m ($m \geq 1$) and for rl-lfg's with dimension at most m ($m \geq 2$) are both $\mathcal{NP}$-complete. Moreover, it is shown in **5** that, for fixed e ($e \geq 3$), the problems for dc-lfg's with degree e and for rl-lfg's with degree e are both $\mathcal{P}$-complete. The time complexity of the universal recognition problems for deterministic-fsts', for right-linear fsts' and for those with some restriction are also investigated. The results of this papar are summarized in Table 2.

2 Definitions

2.1 Finite State Translation Systems

For a ranked alphabet Σ and a set X of variables, let $\mathcal{T}_\Sigma(X)$ denote the set of Σ-terms (or simply, terms). The rank of $\sigma \in \Sigma$ is denoted by $\rho(\sigma)$. A term in $\mathcal{T}_\Sigma(X)$ is also called a *tree*.

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. 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 whose root is labeled with (a production of which left-hand side is) the start symbol S. We sometimes use a usual graph representation of a tree and use terminology such as node, subtree, child, and so on.

Table 2: The Time 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 [7]/ Theorem 1 in [6]	Corollary 3.5	Theorem 3.4
dimension at most m	$\mathcal{NP}$-complete for $m \geq 1$ / $m \geq 2$		
	Theorem 2 in [7]/ Theorem 1 in [7]	Corollary 4.4	Theorem 4.3
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 [7]/ Theorem 3 in [7]	Corollary 5.4	Theorem 5.3
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

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*)[3, 14] 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 *rules* 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* [3].

A *configuration* of a yT-transducer is an element in $(\Delta \cup Q[\mathcal{T}_\Sigma(\phi)])^*$. *Derivation* of M is defined as follows. Let $t = \alpha_1 q[\sigma(t_1, \ldots, t_n)]\alpha_2$ 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 t' be obtained from v by substituting $t_1, \ldots, t_n$ for $x_1, \ldots, x_n$, respectively, then we define $t \underset{M}{\Rightarrow} \alpha_1 t' \alpha_2$. Let $\overset{*}{\underset{M}{\Rightarrow}}$ be the reflexive and transitive closure of $\Rightarrow$.

A *tree-to-string finite state translation system* (*yT-fsts* or *fsts*)[14] is a pair (M, G) of a yT-transducer $M = (Q, \Sigma, \Delta, q_0, R)$ and a cfg $G = (N, T, P, S)$ which satisfies $\Sigma = T \cup P$ where $\rho(a) = 0$ for $a \in T$ and $\rho(p) = n$ for $p : A \to B_1 \cdots B_n \in P$.

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 the output alphabet and q_0 is the initial state of M.

An fsts (M, G) is called *deterministic-fsts* (or *d-fsts* for short) [3] if the transducer M is deterministic.

Next we define the *degree* of a deterministic yT-transducer $M = (Q, \Sigma, \Delta, q_0, R)$. 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 a 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 the yT-transducer is deterministic, $n_{q,\sigma}$ can be defined uniquely. Define the *degree* of $\sigma \in \Sigma$ as $|Q| + \sum_{q \in Q} n_{q,\sigma}$. If the maximum degree

among the symbols in Σ of M is e, then M is called a *yT-transducer with degree e*. An *fsts with degree e* is an fsts of which yT-transducer is with degree e.

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 $\sigma : A \to X_1 X_2 \cdots X_n$ 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 occurences of $q[x_i]$ in $\beta_1, \beta_2, \cdots, \beta_s$ is at most one.

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) [11, 15].

(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 defined to be $\max\{d(A) \mid A \in N\}$.

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

(G3) Let $(T^*)^d$ denote the set of all the d-tuples of strings over T. Each $f \in F$ 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 $\bar{x}_i = (x_{i1}, x_{i2}, \ldots, x_{id_i(f)})$ 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;

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

where $\alpha_{h,k} \in T^*$ $(0 \leq k \leq n_h)$ and $z_{h,k}$ $(1 \leq k \leq n_h)$ is equal to x_{ij} for some $1 \leq i \leq a(f)$ and $1 \leq j \leq d_i(f)$.

(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 $f[\bar{\alpha}_1, \bar{\alpha}_2, \ldots, \bar{\alpha}_{a(f)}] \in L_G(A)$.

Define $L(G) \triangleq L_G(S)$.

The *degree* of a function f is defined as $r(f) + \Sigma_{i=1}^{a(f)} d_i(f)$, that is, the total number of the components of the arguments and function value of f. If the maximum degree among the functions in F of G is e, then G is called a pmcfg with degree e.

2.3 Subclasses of Lexical-Functional Grammars

Here, we define a subclass of lfg's, called *nondeterministic copying lfg's*[16] (*nc-lfg's*), of which generative capacity is strictly weaker than that of context-sensitive grammars. 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 $\uparrow atr = atm$ $(atr \in N_{atr}, atm \in A_{atm})$ 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:

$$\begin{array}{rcl} p : A & \rightarrow & B_1 \quad B_2 \quad \cdots \quad B_q, \\ E_V & & E_{S1} \; E_{S2} \; \cdots \; E_{Sq} \end{array} \tag{2.2}$$

where $A \in N$, $B_1, B_2, \cdots, B_q \in N \cup T$. E_V is a finite set of V schemata and E_{Sj} $(1 \leq j \leq q)$ is a singleton of an S schema. $\cup_{1 \leq j \leq q} E_{Sj} \cup E_V$ is called the set of functional schemata of p. $A \rightarrow B_1 B_2 \cdots B_q$ in (2.2) 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_i's $(1 \leq i \leq k)$ are distinct attributes, and each of val_i $(1 \leq i \leq k)$ 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 $F.atr_i = val_i$.

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

Let $G_0 = (N, T, P_0, S)$ be the underlying cfg of an nc-lfg $G = (N, T, P, S, N_{atr}, A_{atm})$ and let t be a derivation tree in G_0. Every internal node v in t has an f-structure, which is called the f-structure of v and written as F_v. If an underlying production $p_0 : A \rightarrow B_1 \cdots B_q \in P_0$ is applied at v, then v is labeled with either p_0 itself or p $(\in P)$ of which p_0 is the underlying production, if necessary. Let v_i be the ith child of v $(1 \leq i \leq q)$. 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 $i(1 \leq i \leq q)$th 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. Because the meaning

of a V schema is independent of the position where it is annotated, V schemata will be attached to the left-hand side in what follows.

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 $L(G) = \{w \in T^* \mid S \underset{G}{\overset{*}{\Rightarrow}} w\}$.

The empty string is denoted by ε.

Example 2.1: Let $G_{EX1} = (N, T, P, S, N_{atr}, A_{atm})$ 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;

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

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

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

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

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

The language generated by G_{EX1} is $L(G_{EX1}) = \{a^n b^n c^n d^n \mid n \geq 0\}$. Figure 2 shows a derivation tree of $S \underset{G_{EX1}}{\overset{*}{\Rightarrow}} aabbccdd$ in G_{EX1}. □

For a set X of functional schemata, X is *consistent* iff neither the following (1) nor (2) holds.

(1) $\{\uparrow atr = atm_1, \uparrow atr = atm_2\} \subseteq X$

for some $atr \in N_{atr}$ and some $atm_1, atm_2 \in A_{atm}$ such that $atm_1 \neq atm_2$.

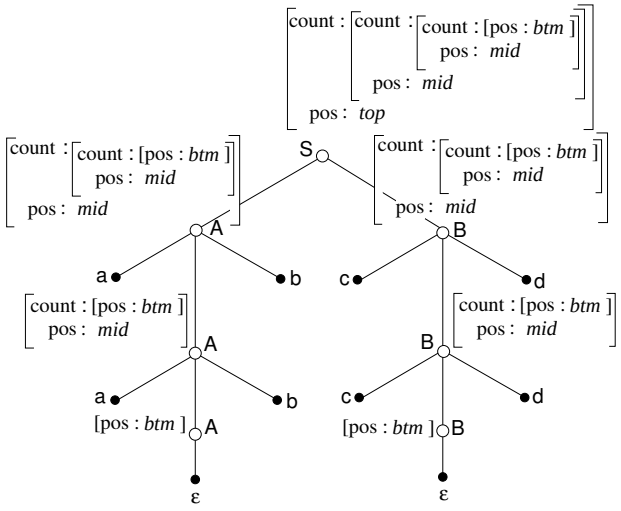

Figure 2: A derivation tree of *aabbccdd*

(2) $\{\uparrow atr = atm, \uparrow atr = \downarrow\} \subseteq X$

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

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. If productions are not consistent, then they are called *inconsistent*.

An nc-lfg G is called a *deterministically copying lfg* (*dc-lfg*), if any distinct productions $A \rightarrow \alpha_1$ and $A \rightarrow \alpha_2$ whose left-hand sides are the same are inconsistent.

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

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 $p_1, p_2, \cdots, p_J$ is at most one.

Example 2.2: Consider G_{EX1} in Example 2.1. Maximal consistent subsets of productions are $sp_1 = \{p_{11}\}$, $sp_2 = \{p_{12}, p_{13}\}$ and $sp_3 = \{p_{14}, p_{15}\}$. For sp_2, each nonterminal appears at most once in the right-hand sides. This condition holds also for sp_1 and sp_3. Hence G_{EX1} is a dc-lfg and is an rl-lfg. □

For a finite set $\mathcal{S}$, $|\mathcal{S}|$ denotes the number of elements in $\mathcal{S}$. 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 G is called an nc-lfg (resp. dc-lfg, rl-lfg) *with degree* e.

3 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 $\triangleq$

$\{L \mid$ 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 problem for pmcfg's is 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''.

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

For a given dc-lfg $G = (N, T, P, S, N_{atr}, A_{atm})$, an equivalent d-fsts (M, G') where $M = (Q, \Sigma, \Delta, q_1, R)$ and $G' = (N', T', P', S')$ can be constructed by [**TRANS 7.1**] in [16]. The sizes of each components of M and G' can be represented in terms of those of G as follows.

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 $\max_{s \in \mathcal{S}} ||s||$. Let SP denote the set of all maximal consistent subsets of P.

By [**TRANS7.1**] in [16], Q has 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.

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

$$\left.\begin{aligned}
|N'| &= |SP| + 2 = O(2^{|P|}) \\
|T'| &= 1 \\
||P'|| &= O(|P'|) \times O(||max(P')||) \\
&= O(|SP| + |P|) \times O(|N_{atr}|) \\
&= O(2^{|P|} \times |N_{atr}|) \\
|Q| &= \max_{sp \in SP} |sp| = m, \text{ the dimension of } G \\
|\Sigma| &= O(|P'|) = O(|SP| + |P|) = O(2^{|P|}) \\
|\Delta| &= |T| \\
||R|| &= O(|R|) \times O(||max(R)||) \\
&= O(\max_{sp \in SP} |sp| \times |SP|) \times O(||max(P)||) \\
&= O(m \times 2^{|P|} \times ||max(P)||)
\end{aligned}\right\} \tag{3.3}$$

Example 3.1: For dc-lfg $G = G_{EX1}$ in Example 2.1, the following d-fsts (M, G'), where $M = (Q, \Sigma, \Delta, q_1, R)$ and $G' = (N', T', P', S')$, is constructed.

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

$$P' = \left\{\begin{array}{l} p'_{start} : S' \rightarrow S_{sp1} \\ p'_{spi} : S_{spi} \rightarrow ZZ \text{ for } i = 1, 2, 3, \\ p'_{guess(spi)} : Z \rightarrow S_{spi} \text{ for } i = 1, 2, 3, \\ p'_{term} : Z \rightarrow d \end{array}\right\} Q = \{q_1, q_2\},\ \Delta = T,$$

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

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

□

[d-fsts ⇒ 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 Section 3.1 in [8]. Obviously, for a string $w \in \Delta^*$, $w \in yL(M, G')$ iff $w \in L(G'')$.

Note that in Section 3.1 in [8], the set N'' of nonterminals in G'' is defined as the union of the set P' of productions in G' and the set T' of terminals in G', and $||G''||$ may be exponential to $||G'||$. However, this construction can be easily modified in such a way that N'' has one-to-one correspondence with $N' \cup T' \cup \{S''\}$ (where S'' is a newly introduced start symbol). Here, we use this modified construction.

The sizes of each components of G'' are:

$$\left. \begin{array}{rcl} |N''| &=& |N'| + |T'| + 1 \\ |T''| &=& |\Delta| + 1 \\ ||P''|| &=& O(|P''|) \times O(||max(P'')||) \\ &=& O((|P'| + |T'|) \times ||max(P')||) \\ ||F|| &=& O(|F|) \times O(||max(F)||) \\ &=& O(|P'| + |T'|) \times O(|Q| \times ||max(R)||) \end{array} \right\} \tag{3.4}$$

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

$$m'' = |Q|. \tag{3.5}$$

Example 3.2: For a d-fsts (M, G') constructed in Example 3.1, the following pmcfg $G'' = (N'', T'', F, P'', S'')$ is constructed.

$$N'' = \{S''\} \cup N' \cup \{D\},$$

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

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

$$F = \left\{ \begin{array}{ll} f_0[(\bar{x}_1)] = \bar{x}_1, & \\ f_1[(\bar{x}_1, \bar{x}_2)] = (x_{11}x_{12}, \$), & \\ f_2[(\bar{x}_1, \bar{x}_2)] = (ax_{11}b, cx_{12}d), & \\ f_3[(\bar{x}_1, \bar{x}_2)] = (\varepsilon, \varepsilon), & f_4[(\bar{x}_1)] = (\$, \$), \\ f_5[] = (\$, \$), & f_S[(\bar{x}_1)] = x_{11} \end{array} \right\},$$

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

It is shown in [7] that the universal recognition problem for pmcfg's is solvable in EXP-POLY time in the following way. Let $G'' = (N'', T'', F, P'', S'')$ and w be a given pmcfg and a string, respectively. Define the table

$$\mathrm{NULL}(A'', (k_1, k_2, \cdots, k_r))$$
$$(A'' \in N'', 1 \leq k_1 < k_2 < \cdots < k_r \leq d(A''))$$

where $\mathrm{NULL}(A'', (k_1, k_2, \cdots, k_r)) = 1$ if there is some $(w_1, \cdots, w_{d(A'')}) \in L_{G''}(A'')$ such that $w_{k_1} = w_{k_2} = \cdots w_{k_r} = \varepsilon$, and $\mathrm{NULL}(A'', (k_1, k_2, \cdots, k_r)) = 0$ otherwise. It is decidable whether $w \in L(G'')$ or not in EXP-POLY time by the following (1) and (2).

(1) Construct the table NULL.

(2) Generate w by simulating a derivation from S'' nondeterministically by Algorithm 1 in [6]. During the execution of this (2), if the symbol \$, which is newly introduced in [**d-fsts** $\Rightarrow$ **pmcfg**], appears on the working tape, then halt since w contains no \$.

When a given dc-lfg $G = (N, T, P, S, N_{atr}, A_{atm})$ is translated into pmcfg G'' by [**dc-lfg** $\Rightarrow$ **d-fsts**] and [**d-fsts** $\Rightarrow$ **pmcfg**], $||G''||$ is exponential to $||G||$. However, the following two lemmas show that the above step (1) and (2) can be executed in EXP-POLY time of $||G||$.

Lemma 3.1: For a given dc-lfg $G = (N, T, P, S, N_{atr},$
$A_{atm})$, the table NULL can be constructed in EXP-POLY time of $||G||$.

Proof. 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. Let m'' be the dimension of G''. It was shown in Lemma 2 in [6] that the table NULL can be constructed in

$$O(|N''| \times 2^{m''} \times |P''_N| \times (2^{m''})^{||max(P''_N)||} \times 2^{m''}) \text{ time}, \tag{3.6}$$

where P''_N is the set of nonterminating productions in P''.

Let m be the dimension of G. By (3.3) through (3.6), the table NULL can be constructed in

$$\begin{aligned} &O(2^{|P|} \times 2^m \times 2^{|P|} \times (2^m)^{|N_{atr}|} \times 2^m) \\ &\qquad = O(2^{||G||^2+4||G||}) \text{ time}. \end{aligned} \tag{3.7}$$

□

Lemma 3.2: 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 equivalent to G as in the proof of Lemma 3.1. Also let m'' be the dimension of G''. It was shown in [6] that the total space needed to execute step (2) is

$$\begin{aligned} &O(log(|N''|) + log(m'') \\ &\qquad + log(m''|w| + ||max(F)||)) \\ &\qquad \times O(m''|w| + ||max(F)||). \end{aligned} \tag{3.8}$$

Let $n = ||G|| + |w|$ and let m be the dimension of G. By (3.3), (3.4), (3.5) and (3.8), the total space needed to execute step(2) is

$$\begin{aligned} & O(log(2^{|P|}) + log(m) \\ & \quad + log(m \times |w| + m \times ||max(P)||)) \\ & \quad \times O(m \times |w| + m \times ||max(P)||) = O(n^3). \end{aligned} \tag{3.9}$$

Furthermore, since $|P''| = O(2^{|P|})$, in step 2(a) of Algorithm 1 in [6], it takes

$$O(log(2^{|P|})) = O(|P|) = O(n) \text{ time}, \tag{3.10}$$

to choose a production nondeterministically.

By Lemma 3 in [6], Theorem 6 in [7], Lemma 3.1, (3.9) and (3.10), 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 ⇒ rl-fsts]

If a given mcfg $G = (N, T, F, P, S)$ satisfies nonerasing condition (Condition f3 in [11]), 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 Section 3.2 in [8]. However, even when G does not satisfy nonerasing condition, the above translation method constructs an equivalent rl-fsts from G if G has no production p which contains a nonterminal C such that $L_G(C) = \emptyset$ in its right-hand side. (Assume that we will use a construction method which is modified in a similar way to the one stated in **[d-fsts ⇒ 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 each components of M and G' are:

$$\left.\begin{aligned} |N'| &= O(|N| \times |P|) \\ |T'| &= 1 \\ ||P'|| &= O(||P||) \\ |Q| &= m, \text{ the dimension of } G \\ |\Sigma| &= 2|P| + 1 \\ |\Delta| &= |T| \\ ||R|| &= O(|R|) \times O(||max(R)||) \\ &= O(|P| \times m) \times O(||max(F)||) \end{aligned}\right\} \tag{3.11}$$

Therefore, $||(M, G')||$ is some polynomial of $||G||$.

Example 3.3: For a given mcfg $G = (N, T, F, P, S)$, where

$$N = \{S, A, B\}\ (d(S) = 1, d(A) = d(B) = 2),$$
$$T = \{a, b, c, d\},$$
$$P = \left\{ \begin{array}{ll} p_0 : S \to f_0[A, B], & p_1 : A \to f_1[A], \\ p_2 : B \to f_2[B], & p_3 : A \to f_3, \\ p_4 : B \to f_3 & \end{array} \right\} \text{ and}$$
$$F = \left\{ \begin{array}{l} f_0[(x_{11}, x_{12}), (x_{21}, x_{22})] \to x_{11}x_{21}x_{12}x_{22}, \\ f_1[(x_{11}, x_{12})] = (ax_{11}, cx_{12}), \\ f_2[(x_{11}, x_{12})] = (bx_{11}, dx_{12}), \\ f_3[] = (\varepsilon, \varepsilon) \end{array} \right\},$$

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

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

□

[rl-fsts ⇒ rl-lfg]

For an rl-fsts (M, G') where $M = (Q, \Sigma, \Delta, q_1, R)$ and $G' = (N', T', P', S')$, an equivalent rl-lfg $G'' = (N'', T'', P'', S'', N_{atr}, A_{atm})$ can be constructed by **[TRANS 7.2]** in [16].

The sizes of each components of G'' are:

$$\left. \begin{array}{rcl} |N''| &=& (|N'| + |T'|) \times |Q| \\ |T''| &=& |\Delta| \\ ||P''|| &=& O(||R||) \\ |N_{atr}| &=& O(||max(P')||) \\ |A_{atm}| &=& |P'| \end{array} \right\} \tag{3.12}$$

Therefore, $||G''||$ is some polynomial of $||(M, G')||$.

Example 3.4: For rl-fsts (M, G') constructed in Example 3.3, the following rl-lfg $G'' = (N'', T'', P'', S'', N_{atr},$
$A_{atm})$ is constructed.

$$N'' = \{X^{[1]}, X^{[2]} \mid X \in N' \cup T'\},\ T'' = \{a, b, c, d\},$$

$$S'' = S^{[1]},\ N_{atr} = \{atr_1, atr_2, prod\},$$

$$A_{atm} = \{p_0, \cdots, p_4, \bar{p}_0, \cdots, \bar{p}_4\},$$

$$P'' =$$

$$\left\{\begin{array}{llll}
S^{[1]}_{p_0} \rightarrow & A^{[1]} & B^{[1]} & A^{[2]} \quad B^{[2]} \\
\{\uparrow prod = p_0\}\{\uparrow atr_1 = \downarrow\} & \{\uparrow atr_2 = \downarrow\} & \{\uparrow atr_1 = \downarrow\} & \{\uparrow atr_2 = \downarrow\}, \\
A^{[1]}_{p_1} \rightarrow a & A^{[1]} & A^{[2]}_{p_1} \rightarrow c & A^{[2]} \\
\{\uparrow prod = p_1\} & \{\uparrow atr_1 = \downarrow\}, & \{\uparrow prod = p_1\} & \{\uparrow atr_1 = \downarrow\}, \\
B^{[1]}_{p_2} \rightarrow b & B^{[1]} & B^{[2]}_{p_2} \rightarrow d & B^{[2]} \\
\{\uparrow prod = p_2\} & \{\uparrow atr_1 = \downarrow\}, & \{\uparrow prod = p_2\} & \{\uparrow atr_1 = \downarrow\}, \\
A^{[1]}_{p_3} \rightarrow \varepsilon, & & A^{[2]}_{p_3} \rightarrow \varepsilon, & \\
\{\uparrow prod = p_3\} & & \{\uparrow prod = p_3\} & \\
B^{[1]}_{p_4} \rightarrow \varepsilon, & & B^{[2]}_{p_4} \rightarrow \varepsilon & \\
\{\uparrow prod = p_4\} & & \{\uparrow prod = p_4\} & \\
S^{[1]} \rightarrow S^{[1]}_{p_0} & & S^{[2]} \rightarrow S^{[2]}_{p_0} & \\
\{\uparrow prod = \bar{p}_0\} & \{\uparrow atr_1 = \downarrow\}, & \{\uparrow prod = \bar{p}_0\} & \{\uparrow atr_1 = \downarrow\}, \\
\multicolumn{4}{c}{\vdots} \\
B^{[1]} \rightarrow B^{[1]}_{p_4} & & B^{[2]} \rightarrow B^{[2]}_{p_4} & \\
\{\uparrow prod = \bar{p}_4\} & \{\uparrow atr_1 = \downarrow\}, & \{\uparrow prod = \bar{p}_4\} & \{\uparrow atr_1 = \downarrow\}
\end{array}\right\}.$$

□

Lemma 3.3: 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 (3.11) and (3.12) in time polynomial of $||G||$. It is known that the universal recognition problem

for mcfg's is EXP-POLY time-hard[6]. Therefore, the universal recognition problem for rl-lfg's is EXP-POLY time-hard. □

Theorem 3.4: (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 3.3. (1) holds from this fact and Lemma 3.2. (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 3.2. This fact together with Lemma 3.3 implies (2). □

Corollary 3.5: The universal recognition problem for d-fsts' and the problem for rl-fsts' are both EXP-POLY time complete. □

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

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

Lemma 4.1: For any fixed $m \geq 1$ and a given dc-lfg $G = (N, T, P, S, N_{atr}, A_{atm})$ with dimension m, an equivalent pmcfg G'' with dimension m can be constructed from G in time polynomial of $||G||$.

Proof. Let G'' be the pmcfg constructed from G by [**dc-lfg** $\Rightarrow$ **d-fsts**] and [**d-fsts** $\Rightarrow$ **pmcfg**] in this order. By (3.3) and (3.5), the dimension of G'' is m. By the definition of the dimension of a dc-lfg, $|SP| = O(|P|^m)$, which is a polynomial of $||G||$ when m is a constant. This fact together with (3.3) and (3.4) implies that $||G''||$ is a polynomial of $||G||$. It can be easily shown that the time consumed by [**dc-lfg** $\Rightarrow$ **d-fsts**] and [**d-fsts** $\Rightarrow$ **pmcfg**] is a polynomial of $||G||$. □

Lemma 4.2: For any fixed m and a given mcfg G with dimension m, an equivalent rl-lfg G'' with dimension m can be constructed from G in time polynomial of $||G||$.

Proof. Let (M, G') be the rl-fsts constructed from G by [**mcfg** $\Rightarrow$ **rl-fsts**], 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**]. Observe that

$$|Q| = m \tag{4.13}$$

by (3.11). On the other hand, in [**rl-fsts** $\Rightarrow$ **rl-lfg**] ([**TRANS 7.2**] in [16]), 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 rl-lfg G''. Hence,

$$\max_{sp \in SP} |sp| = |Q|, \tag{4.14}$$

where SP is the set of consistent subset of productions of G''. By (4.13) and (4.14), G'' is an rl-lfg with dimension m, since the dimension of G'' is defined as $\max_{sp \in SP} |sp|$. By (3.11) and (3.12), it is clear that $||G''||$ is some polynomial of $||G||$. By this fact, it is easy to show that G'' can be constructed in polynomial time of $||G||$. □

Theorem 4.3: **(1)** For any fixed m ($m \geq 1$), the universal recognition problem for dc-lfg's with dimension at most m is $\mathcal{NP}$-complete.

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

Proof. (1) holds from Lemmas 4.1, 4.2, Theorem 2 in [7] and the fact that the class of rl-lfg's is a subclass of dc-lfg's. (2) can be proved in the same way, except for using Theorem 1 in [7] instead of Theorm 2 in [7]. □

Corollary 4.4: For any fixed m ($m \geq 1$), the universal recognition problem for d-fsts' with dimension at most m is $\mathcal{NP}$-complete. For any fixed m ($m \geq 2$), the universal recognition problem for rl-fsts' with dimension at most m is $\mathcal{NP}$-complete. □

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

Lastly, it is shown that the universal recognition problem for dc-lfg's and that for rl-lfg's with bounded degree are $\mathcal{P}$-complete.

Lemma 5.1: For any fixed e ($e \geq 3$) and a given dc-lfg G with degree e, an equivalent pmcfg G'' with degree e 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**]. And let G'' be the pmcfg constructed from (M, G') by [**d-fsts** $\Rightarrow$ **pmcfg**]. 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 step2 (ii) of [**TRANS 7.1**][16]. Thus, the degree of (M, G') is equal to the degree of G. On the other hand, 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 e. By the definition of the dimension of a dc-lfg, $|SP| = O(|P|^e)$, which is a polynomial of $||G||$ when e is a constant. This fact together with (3.3) and (3.4) implies that $||G''||$ is a polynomial of $||G||$. It can be easily shown that the time consumed by [**dc-lfg** $\Rightarrow$ **d-fsts**] and [**d-fsts** $\Rightarrow$ **pmcfg**] is a polynomial of $||G||$. □

Lemma 5.2: For any e ($e \geq 3$) and a given mcfg G with degree e, an equivalent rl-lfg G'' with degree e can be constructed from G in time polynomial of $||G||$.

Proof. Let (M, G') be the rl-fsts constructed from G by [**mcfg** $\Rightarrow$ **rl-fsts**], 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**]. 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 f is equal to the degree of p by [**mcfg** $\Rightarrow$ **rl-fsts**]. Thus, the degree of (M, G') is equal to the degree of G. On the other hand, in [**rl-fsts** $\Rightarrow$ **rl-lfg**] ([**TRANS 7.2**] in [16]), for each production p' in G', 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 definition. It follows that the degree of G'' is equal to the degree of (M, G'), which implies that G'' is the rl-lfg with degree e. By (3.11) and (3.12), it is obvious that $||G''||$ is some polynomial of $||G||$. By this fact, it can be easily shown that G'' can be constructed in time polynomial of $||G||$.

□

Theorem 5.3: (1) For any fixed e ($e \geq 3$), the universal recognition problem for dc-lfg's with degree e is $\mathcal{P}$-complete.

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

Proof. (1) holds from Lemmas 5.1, 5.2, Theorem 4 in [7] and the fact that the class of rl-lfg's is a subclass of dc-lfg's. (2) can be proved in the same way, except for using Theorem 3 in [7] instead of Theorem 4 in [7]. □

Corollary 5.4: For any fixed e ($e \geq 3$), the universal recognition problem for d-fsts' with degree e, and the problem for rl-fsts' with degree e are both $\mathcal{P}$-complete. □

6 Conclusion

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 grammars with one of the "common" parameters bounded are the same. This fact tells that these three grammars are resemble in structure.

References

[1] Ando, S., Nakanishi, R., Seki, H. and Kasami, T. 1992: "Subclasses of Lexical-Functional Grammars Which Are Recognizable in Polynomial Time", IEICE Technical Report, **COMP92-44**.

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

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

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

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

[6] Kaji, Y., Nakanishi, R., Seki, H. and Kasami, T. 1992: "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.

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

[8] Kaji, Y., Nakanishi, R., Seki, H. and Kasami, T. 1992: "Parallel Multiple Context-Free Grammars and Finite State Translation Systems", IEICE Technical Report, **COMP92-34**.

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

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

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

[12] 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.

[13] 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).

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

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

[16] 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. Compt. Ling.:130–139.

[17] 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).