数理解析研究所講究録
第 871 巻 1994 年 45-51

Uniquely Parsable Grammars

森田 憲一 Kenichi MORITA[†], 山本 泰則 Yasunori YAMAMOTO[‡], 西原 典孝 Noritaka NISHIHARA[§], 張 治国 Zhiguo ZHANG[§*]

[†] 広島大学工学部
Faculty of Engineering
Hiroshima University
morita@ke.sys.hiroshima-u.ac.jp

[‡] 国立民族学博物館
National Museum of Ethnology
yamty@minpaku.ac.jp

[§] 山形大学工学部
Faculty of Engineering
Yamagata University
nisihara@emtsun.yz.yamagata-u.ac.jp

Abstract: We introduce a new class of grammars called uniquely parsable grammars (UPG). UPG is a kind of phrase structure grammar having a restricted type of rewriting rule set so that parsing can be done without backtracking. After giving basic properties of UPG, we show that, in spite of such restriction to the rules, UPG is universal in its generating ability. That is, the class of languages generated by UPG's is the same as the class of languages accepted by Turing machines. We also define a monotonic UPG (MUPG), a subclass of UPG, and prove that the class of languages generated by MUPG's is the same as the class of languages accepted by deterministic linear-bounded automata.

1 Introduction

Parsing is an important problem in formal language theory, and there were many investigations on efficient parsing methods. When reducing a given word by reverse applications of rewriting rules, there are in general many choices of reductions and thus it needs backtracking to parse it. But, some classes of languages can be parsed without backtracking. Deterministic context-free languages [1] are such ones. For this class (and its various subclasses), many practically useful frameworks, e.g., LR(k) grammars, have been defined and studied to parse them in a deterministic manner. However, besides the subclasses of context-free grammars, there are very few studies on grammar classes having such property.

As for an array grammar, which generates two-dimensional symbol arrays, a uniquely parsable array grammar (UPAG) has been introduced by Yamamoto and Morita [5]. UPAG is a subclass of isometric array grammars (IAG), and has the property of unique parsability. In the case of IAG, it is very difficult to parse array languages for the grammars without this property [3]. So, this framework is useful to find a subclasses of array grammars which have efficient parsing algorithms.

In this paper, we introduce a "uniquely parsable grammar" (UPG), a string grammar version of UPAG. UPG is a grammar whose rewriting rules satisfy the following condition: if a suffix of the righthand side of a rule agrees with a prefix of that of some other rule, then these portions remain unchanged by the reverse application of these rules (precise definition will be given later). By this restriction, UPG languages can be parsed without backtracking.

In what follows, after showing basic properties of UPG, we investigate generating ability of UPG. In spite of the restriction to the rules, UPG is shown to be universal. That is, any language accepted by a Turing machine can be generated by a UPG, and vice versa. We also define a monotonic UPG (MUPG) as a subclass of UPG, and prove that the generating ability of MUPG is exactly characterized by a deterministic linear-bounded automaton.

2 Definitions and Basic Properties of UPG

We introduce here a uniquely parsable grammar, and show that it is parsed without backtracking, especially by a "leftmost reduction".

We first give some definitions. (As for standard notations for strings and languages, see

*Presently, Utsunomiya University

e.g. [2].)

Definition 2.1 A *uniquely parsable grammar* (UPG) is a system

$$G = (N, T, P, S, \$),$$

where N and T are sets of nonterminal and terminal symbols respectively ($N \cap T = \emptyset$), S is a start symbol ($S \in N$), and \$ is a special endmarker ($\$ \notin (N \cup T)$). P is a set of rewriting rules of the following form:

$$\begin{array}{ll} \alpha \to \beta, & \$\alpha \to \$\beta, \\ \alpha\$ \to \beta\$, & \$\alpha\$ \to \$\beta\$, \quad \text{or} \\ \$A\$ \to \$\$, & \end{array}$$

where $\alpha, \beta \in (N \cup T)^+$, $\alpha \neq \beta$, $A \in N$, and α contains at least one nonterminal (a rule of the form $\$A\$ \to \$\$$ is called an *ε-rule*). Furthermore, P satisfies the following condition.

UPG-condition:

1. The righthand side of each rule in P is neither S, $\$S$, $S\$$, nor $\$S\$$.
2. For any two rules $r_1 = \alpha_1 \to \beta_1$ and $r_2 = \alpha_2 \to \beta_2$ in P (r_1 and r_2 may be the same) the following holds.
   (a) If $\beta_1 = \beta_1'\delta$ and $\beta_2 = \delta\beta_2'$ for some $\delta, \beta_1', \beta_2' \in (N \cup T \cup \{\$\})^+$, then $\alpha_1 = \alpha_1'\delta$ and $\alpha_2 = \delta\alpha_2'$ for some $\alpha_1', \alpha_2' \in (N \cup T \cup \{\$\})^*$.
   (b) If $\beta_1 = \gamma\beta_2\gamma'$ for some $\gamma, \gamma' \in (N \cup T \cup \{\$\})^*$, then $r_1 = r_2$ (therefore $\gamma = \gamma' = \varepsilon$ (empty word)). □

UPG-condition 2(a) requires that if some suffix of the righthand side of r_1 agrees with some prefix of that of r_2, then the lefthand sides of r_1 and r_2 also contain them as a suffix and a prefix, respectively. For example, the following pair of rules

$$A \to bA, \quad AC \to Ad$$

satisfies UPG-condition, while

$$A \to bA, \quad EC \to Ad$$

does not. The condition 2(b) says that there are no pair of distinct rules r_1 and r_2 such that the righthand side of r_2 is a substring of that of r_1. Note that there is at most one ε-rule because of 2(b).

Definition 2.2 Let $G = (N, T, P, S, \$)$ be a UPG, and η be a word in $(N \cup T \cup \{\$\})^+$. A rule $\alpha \to \beta$ in P is said to be *applicable* to η if $\eta = \gamma\alpha\delta$ for some $\gamma, \delta \in (N \cup T \cup \{\$\})^*$. Applying $\alpha \to \beta$ to η we obtain $\zeta = \gamma\beta\delta$, and say ζ is *directly derived* from η in G. This is written as $\eta \underset{G}{\Rightarrow} \zeta$. The reflexive and transitive closure of the relation $\underset{G}{\Rightarrow}$ defines the relation of *derivation* and is denoted by $\overset{*}{\underset{G}{\Rightarrow}}$. For any $\eta, \zeta \in (N \cup T \cup \{\$\})^+$ if there are $\xi_1, \xi_2, \cdots, \xi_{n-1}$ such that $\eta \underset{G}{\Rightarrow} \xi_1 \underset{G}{\Rightarrow} \xi_2 \underset{G}{\Rightarrow} \cdots \underset{G}{\Rightarrow} \xi_{n-1} \underset{G}{\Rightarrow} \zeta$, we write it as $\eta \overset{n}{\underset{G}{\Rightarrow}} \zeta$. We use $\Rightarrow, \overset{*}{\Rightarrow}, \overset{n}{\Rightarrow}$ instead of $\underset{G}{\Rightarrow}, \overset{*}{\underset{G}{\Rightarrow}}, \overset{n}{\underset{G}{\Rightarrow}}$ if it is clear which grammar G is used. □

Definition 2.3 Let $G = (N, T, P, S, \$)$ be a UPG, and α be a word in $(N \cup T)^+$. If $\$S\$ \overset{*}{\underset{G}{\Rightarrow}} \$\alpha\$$, we call α a *sentential form* in G. The *language* $L(G)$ generated by G is the set of all sentential forms over T, i.e., $L(G) = \{w \in T^* \mid \$S\$ \overset{*}{\underset{G}{\Rightarrow}} \$w\$\}$. The class of languages generated by the grammar class UPG is denoted by $\mathcal{L}$[UPG] (similar notations are used for other classes of grammars or machines). □

Example 2.1 (1) The grammar

$$G_{\text{paren}} = (\{S\}, \{\,(,)\,\}, P_{\text{paren}}, S, \$)$$

is a UPG that generates all well-formed parentheses strings over $\{\,(,)\,\}$, where P_{paren} consists of the following rules.

$$\begin{array}{lllllll} S & \to & SS & \quad & S & \to & (\,) \\ S & \to & (S) & & & & \end{array}$$

(2) The grammar

$$G_{\text{arith}} = (\{E, T, F\}, \{a, +, *, (,)\}, P_{\text{arith}}, E, \$)$$

is a UPG that generates all arithmetic expressions over $\{a, +, *, (,)\}$, where P_{arith} consists of the following rules.

$$\begin{array}{lllllll} E\$ & \to & E+T\$ & \quad & T & \to & T*F \\ E+ & \to & E+T+ & & \$T & \to & \$F \\ E) & \to & E+T) & & (T & \to & (F \\ \$E\$ & \to & \$T\$ & & +T & \to & +F \\ \$E+ & \to & \$T+ & & F & \to & (E) \\ (E+ & \to & (T+ & & F & \to & a \\ (E) & \to & (T) & & & & \end{array}$$

□

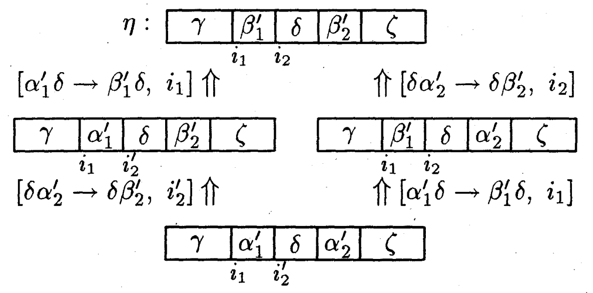
η : γ β'_1 δ β'_2 ζ
i_1 i_2
$[\alpha'_1\delta \to \beta'_1\delta,\ i_1] \Uparrow$
$\Uparrow [\delta\alpha'_2 \to \delta\beta'_2,\ i_2]$
γ α'_1 δ β'_2 ζ
γ β'_1 δ α'_2 ζ
$[\delta\alpha'_2 \to \delta\beta'_2,\ i'_2] \Uparrow$
$\Uparrow [\alpha'_1\delta \to \beta'_1\delta,\ i_1]$
γ α'_1 δ α'_2 ζ
i_1 i'_2

Figure 1: Reductions in UPG. (Any direct reduction does not affect reverse applicability of other rules.)

Definition 2.4 Let $G = (N, T, P, S, \$)$ be a UPG, and $\eta = x_1 x_2 \cdots x_m$ be a word over $(N \cup T \cup \{\$\})$ of length m. If there is a rule $\alpha \to \beta \in P$ such that $\beta = x_i x_{i+1} \cdots x_{i+|\beta|-1}$ for some i, then $\alpha \to \beta$ is said to be *reversely applicable* to η at position i. Any such pair $[\alpha \to \beta,\ i]$ is called a *reversely applicable item* to η. If ξ is obtained by doing so (i.e., $\xi = x_1 \cdots x_{i-1}\ \alpha\ x_{i+|\beta|} \cdots x_m$) we say η is *directly reduced* to ξ, and write it as $\eta \Leftarrow \xi$ or $\eta \overset{[\alpha\to\beta,\,i]}{\Longleftarrow} \xi$. Apparently, $\eta \Leftarrow \xi$ iff $\xi \Rightarrow \eta$. The reflexive and transitive closure of direct reduction $\Leftarrow$ defines the relation of *reduction* and is denoted by $\overset{*}{\Leftarrow}$. The relation $\overset{n}{\Leftarrow}$ is also defined similarly as $\overset{n}{\Rightarrow}$. □

Parsing is considered to be a reducing process of a given word by reverse applications of rules in P. Intuitively speaking, the name "UPG" comes from the following fact. Assume there are two rules $\alpha_1 \to \beta'_1\delta$ and $\alpha_2 \to \delta\beta'_2$ in P, and consider a word $\eta = \gamma\beta'_1\delta\beta'_2\zeta$. Both these rules can be reversely applied to η. If these rules do not satisfy UPG-condition 2(a), the reduction by $\alpha_1 \to \beta'_1\delta$ may prevent the further reduction by $\alpha_2 \to \delta\beta'_2$. However, in UPG, since $\alpha_1 = \alpha'_1\delta$ and $\alpha_2 = \delta\alpha'_2$ hold, $\alpha_2 \to \delta\beta'_2$ can be reversely applied even after the former reduction. Therefore, any reduction does not affect the reverse applicability of other rules (Figure 1). By this property, if $\eta \overset{n}{\Leftarrow} \$S\$$, then *any* reduction starting from η eventually reaches $\$S\$$ in n steps. Therefore, parsing can be done without backtracking. This is stated in the following theorem.

Theorem 2.1 (Unique Parsability Theorem) Let $G = (N, T, P, S, \$)$ be a UPG, and let η be a word in $(N \cup T \cup \{\$\})^+$. If $\eta \overset{n}{\Leftarrow} \$S\$$, then for any reversely applicable item $[\alpha \to \beta,\ i]$ to η and a word ξ such that $\eta \overset{[\alpha\to\beta,\,i]}{\Longleftarrow} \xi$ the following holds.

$$\eta \overset{[\alpha\to\beta,\,i]}{\Longleftarrow} \xi \overset{n-1}{\Longleftarrow} \$S\$$$

In order to prove this theorem, we define a mapping $\varphi : (N \cup T \cup \{\$\})^* \times \mathbf{Z}_+ \to P \times \mathbf{Z}_+ \cup \{-\}$ ($\mathbf{Z}_+$ denotes the set of all positive integers).

$$\varphi(\eta, i) = \begin{cases} [\alpha \to \beta,\ i] & \text{if } \alpha \to \beta \in P, \text{ and} \\ & \eta = \gamma\beta\delta \text{ for some} \\ & \gamma, \delta \in (N \cup T \cup \{\$\})^* \\ & \text{such that } |\gamma| = i - 1. \\ - & \text{elsewhere.} \end{cases}$$

It is easily seen that $\varphi(\eta, i)$ has unique value (if otherwise, G violates UPG-condition 2(b)).

Example 2.2 The value of φ for the grammar G_{arith} in Example 2.1 and the word $\$(T + F) * a\$$ is as follows.

$$\begin{aligned} \varphi(\$(T+F)*a\$,\ 2) &= [\ (E+ \to (T+,\ 2\ ] \\ \varphi(\$(T+F)*a\$,\ 4) &= [\ +T \to +F,\ 4\ ] \\ \varphi(\$(T+F)*a\$,\ 8) &= [\ F \to a,\ 8\ ] \\ \varphi(\$(T+F)*a\$,\ i) &= - \quad (\text{for } i \neq 2, 4, 8) \end{aligned}$$

□

Lemma 2.1 Let $G = (N, T, P, S, \$)$ be a UPG, and η be a word in $(N \cup T \cup \{\$\})^+$. For any i_1, i_2 $(1 \le i_1 < i_2 \le |\eta|)$, if $\varphi(\eta, i_1) = [\alpha_1 \to \beta_1, i_1]$, $\varphi(\eta, i_2) = [\alpha_2 \to \beta_2, i_2]$ and $\eta \overset{\varphi(\eta,i_1)}{\Longleftarrow} \xi_1$, $\eta \overset{\varphi(\eta,i_2)}{\Longleftarrow} \xi_2$, then there exists a unique word σ such that the followings hold.

$$\begin{aligned} &\varphi(\xi_1, i'_2) = [\alpha_2 \to \beta_2, i'_2] \quad (i'_2 = i_2 + |\alpha_1| - |\beta_1|) \\ &\varphi(\xi_2, i_1) = [\alpha_1 \to \beta_1, i_1] \\ &\eta \overset{\varphi(\eta,i_1)}{\Longleftarrow} \xi_1 \overset{\varphi(\xi_1,i'_2)}{\Longleftarrow} \sigma \\ &\eta \overset{\varphi(\eta,i_2)}{\Longleftarrow} \xi_2 \overset{\varphi(\xi_2,i_1)}{\Longleftarrow} \sigma \end{aligned}$$

Proof. First, consider the case $i_2 - i_1 \ge |\beta_1|$. Then η is written as

$$\eta = \gamma\beta_1\delta\beta_2\zeta,$$

where $|\gamma| = i_1 - 1$, $|\gamma\beta_1\delta| = i_2 - 1$. Therefore,

$$\begin{aligned} \xi_1 &= \gamma\alpha_1\delta\beta_2\zeta, \\ \xi_2 &= \gamma\beta_1\delta\alpha_2\zeta. \end{aligned}$$

Now consider ξ_1. The rule $\alpha_2 \to \beta_2$, that is reversely applicable to η at position i_2, is also applicable to ξ_1 at position $i_2+|\alpha_1|-|\beta_1|$ ($= i_2'$). Thus $\varphi(\xi_1, i_2') = [\alpha_2 \to \beta_2, i_2']$. Therefore,

$$\xi_1 \overset{\varphi(\xi_1,i_2')}{\Longleftarrow} \gamma\alpha_1\delta\alpha_2\zeta.$$

Consider ξ_2. The rule $\alpha_1 \to \beta_1$, that is reversely applicable to η at position i_1, is also applicable to ξ_2 at the same position i_1. So, $\varphi(\xi_2, i_1) = [\alpha_1 \to \beta_1, i_1]$, and

$$\xi_2 \overset{\varphi(\xi_2,i_1)}{\Longleftarrow} \gamma\alpha_1\delta\alpha_2\zeta.$$

Thus the lemma holds for $\sigma = \gamma\alpha_1\delta\alpha_2\zeta$ (uniqueness of σ is clear).

Next, consider the case $i_2 - i_1 < |\beta_1|$. Then, from UPG-condition 2(a), η, $\alpha_1 \to \beta_1$, and $\alpha_2 \to \beta_2$ must be written as

$$\begin{aligned} \eta &= \gamma\beta_1'\delta\beta_2'\zeta, \\ \alpha_1 \to \beta_1 &= \alpha_1'\delta \to \beta_1'\delta, \\ \alpha_2 \to \beta_2 &= \delta\alpha_2' \to \delta\beta_2', \end{aligned}$$

where $|\gamma| = i_1 - 1, |\gamma\beta_1'| = i_2 - 1$. Thus,

$$\begin{aligned} \xi_1 &= \gamma\alpha_1'\delta\beta_2'\zeta, \\ \xi_2 &= \gamma\beta_1'\delta\alpha_2'\zeta. \end{aligned}$$

Again in this case, $\alpha_2 \to \beta_2$ can be reversely applied to ξ_1 at $i_2' = i_2+|\alpha_1|-|\beta_1|$, and $\alpha_1 \to \beta_1$ can be reversely applied to ξ_2 at i_1. Thus

$$\begin{aligned} &\varphi(\xi_1, i_2') = [\alpha_2 \to \beta_2, i_2'], \quad (i_2' = i_2 + |\alpha_1| - |\beta_1|) \\ &\varphi(\xi_2, i_1) = [\alpha_1 \to \beta_1, i_1], \\ &\xi_1 \overset{\varphi(\xi_1,i_2')}{\Longleftarrow} \gamma\alpha_1'\delta\alpha_2'\zeta, \\ &\xi_2 \overset{\varphi(\xi_2,i_1)}{\Longleftarrow} \gamma\alpha_1'\delta\alpha_2'\zeta. \end{aligned}$$

Therefore the lemma holds for $\sigma = \gamma\alpha_1'\delta\alpha_2'\zeta$. □

Proof of Theorem 2.1. We prove this by an induction on the number n of steps of reduction.

The case $n = 1$: If $\eta \overset{1}{\Leftarrow} \$S\$$ there is only one reversely applicable item to η, because G satisfies UPG-condition 1 and 2. Thus the theorem holds for $n = 1$.

The case $n > 1$: Assume the theorem holds for n. If $\eta \overset{n+1}{\Leftarrow} \$S\$$, there exists some $\varphi(\eta, i_0) = [\alpha_0 \to \beta_0,\ i_0]$ such that $\eta \overset{\varphi(\eta,i_0)}{\Longleftarrow} \xi_0 \overset{n}{\Leftarrow} \$S\$$. Let $\varphi(\eta, i) = [\alpha \to \beta,\ i]$ be any reversely applicable item to η, and let ξ be a word such that $\eta \overset{\varphi(\eta,i)}{\Longleftarrow} \xi$. We assume $i \neq i_0$, because if $i = i_0$ it is done. We consider only the case $i_0 < i$ (the case $i_0 > i$ is analogous). From Lemma 2.1, the following holds for some σ.

$$\begin{aligned} &\eta \overset{\varphi(\eta,i_0)}{\Longleftarrow} \xi_0 \overset{\varphi(\xi_0,i')}{\Longleftarrow} \sigma \quad (i' = i + |\alpha_0| - |\beta_0|)) \\ &\eta \overset{\varphi(\eta,i)}{\Longleftarrow} \xi \overset{\varphi(\xi,i_0)}{\Longleftarrow} \sigma \end{aligned}$$

On the other hand, from the induction hypothesis,

$$\xi_0 \overset{\varphi(\xi_0,i')}{\Longleftarrow} \sigma \overset{n-1}{\Longleftarrow} \$S\$$$

holds. Therefore,

$$\eta \overset{\varphi(\eta,i)}{\Longleftarrow} \xi \overset{\varphi(\xi,i_0)}{\Longleftarrow} \sigma \overset{n-1}{\Longleftarrow} \$S\$.$$

Thus

$$\eta \overset{\varphi(\eta,i)}{\Longleftarrow} \xi \overset{n}{\Longleftarrow} \$S\$,$$

and the theorem is proved. □

Definition 2.5 Let $G = (N, T, P, S, \$)$ be a UPG. A direct reduction $\eta \Leftarrow \zeta$ in G is called a *direct leftmost reduction* iff $\eta \overset{\varphi(\eta,i_0)}{\Longleftarrow} \zeta$ for $i_0 = \min\{\ i \mid \varphi(\eta, i) \neq -\}$. This is also written as $\eta \underset{\text{lmr}}{\Leftarrow} \zeta$. A reduction

$$\eta_0 \Leftarrow \eta_1 \Leftarrow \cdots \Leftarrow \eta_n$$

is called a *leftmost reduction* iff

$$\eta_0 \underset{\text{lmr}}{\Leftarrow} \eta_1 \underset{\text{lmr}}{\Leftarrow} \cdots \underset{\text{lmr}}{\Leftarrow} \eta_n.$$

If there is a leftmost reduction (of length n) from η_0 to η_n, we write it as $\eta_0 \underset{\text{lmr}}{\overset{*}{\Leftarrow}} \eta_n$ (or $\eta_0 \underset{\text{lmr}}{\overset{n}{\Leftarrow}} \eta_n$). □

Corollary 2.1 Let $G = (N, T, P, S, \$)$ be a UPG, and let η be a word in $(N \cup T \cup \{\$\})^+$. If $\eta \overset{n}{\Leftarrow} \$S\$$, then $\eta \underset{\text{lmr}}{\overset{n}{\Leftarrow}} \$S\$$.

Proof. This is proved by an induction on n.

The case $n = 1$: It is obvious, since if $\eta \overset{1}{\Leftarrow} \$S\$$ there is only one reversely applicable item to η.

The case $n > 1$: Assume the corollary holds for n. Let η be a word such that $\eta \overset{n+1}{\Leftarrow} \$S\$$, and $i_0 = \min\{\ i \mid \varphi(\eta, i) \neq -\}$. Then, from Theorem 2.1,

$$\eta \overset{\varphi(\eta,i_0)}{\Longleftarrow} \xi \overset{n}{\Leftarrow} \$S\$.$$

By the induction hypothesis, $\xi \underset{\text{lmr}}{\overset{n}{\Leftarrow}} \$S\$$ holds. Therefore $\eta \underset{\text{lmr}}{\overset{n+1}{\Leftarrow}} \$S\$$. □

3 Universality of UPG

In this section we show that UPG is universal in its generating power, i.e., $\mathcal{L}$[UPG] is equal to the class $\mathcal{L}$[DTM] of languages accepted by Turing machines (or the class of Type-0 languages). Although UPG is a restricted type of phrase structure grammar, any computing process of a Turing machine can be simulated by a reduction process of it.

Definition 3.1 A *deterministic Turing machine* (DTM) is a system defined by

$$M = (Q, \Sigma, \Gamma, \delta, a_0, q_0, q_f),$$

where Q is a set of states, Σ is a set of input symbols, Γ is a set of tape symbols ($\Gamma \supseteq \Sigma$), $\delta : Q \times \Gamma \to Q \times \Gamma \times \{L, R\} \cup \{-\}$ is a move function ("$-$" means that the machine halts), $a_0 \in \Gamma$ is a blank symbol, $q_0 \in Q$ is an initial state, and $q_f \in Q$ is a final state (halting state). Assume that the tape is one-way infinite to the right, and M never writes the blank symbol a_0.

An *instantaneous description* (ID) of a DTM M is a word $\alpha q \beta$, where $\alpha, \beta \in (\Gamma - \{a_0\})^*$ and $q \in Q$. It represents a computational configuration of M, where non-blank portion of the tape is $\alpha\beta$ and M is reading the left-most symbol of β (if $\beta = \varepsilon$ (empty word) then a_0) in state q. Starting from an initial ID $q_0 w$ ($w \in \Sigma^*$), if M halts with a final ID $q_f \alpha$ for some $\alpha \in (\Gamma - \{a_0\})^+$, then w is said to be *accepted* by M. The set of all accepted words is called the language accepted by M and denoted by $L(M)$. □

Lemma 3.1 $\mathcal{L}[\text{UPG}] \supseteq \mathcal{L}[\text{DTM}]$.

Proof. Let $M = (Q, \Sigma, \Gamma, \delta, a_0, q_0, q_f)$ be an arbitrary DTM, where $\Gamma = \{a_0, a_1, \cdots, a_n\}$, $\Sigma = \{a_1, a_2, \cdots, a_m\}$ ($m \leq n$). We now construct a UPG $G_M = (N, T, P, S, \$)$ that simulates M. It is defined as follows.

$$\begin{aligned} N &= \{S\} \cup \{A_1, \cdots, A_n\} \cup \{B_1, \cdots, B_n\} \\ &\quad \cup \{C_1, \cdots, C_m\} \cup (Q \times \{a_0, \cdots, a_n\}) \\ T &= \Sigma \end{aligned}$$

The set P of rules is as follows:

(1) For each $a_i \in \Gamma - \{a_0\}$ include the following rules in P.

$$S \to SB_i, \quad S \to (q_f, a_i)$$

(2) For each $q_h, q_j \in Q$, $a_i \in \Gamma$, and $a_k, a_l \in \Gamma - \{a_0\}$, if $\delta(q_h, a_i) = (q_j, a_k, L)$ then include the following rule in P.

$$(q_j, a_l)B_k \to A_l(q_h, a_i)$$

(3) For each $q_h, q_j \in Q$, $a_i \in \Gamma$, and $a_k, a_l \in \Gamma - \{a_0\}$, if $\delta(q_h, a_i) = (q_j, a_k, R)$ then include the following rules in P.

$$\begin{aligned} A_k(q_j, a_l) &\to (q_h, a_i)B_l, \\ A_k(q_j, a_0)\$ &\to (q_h, a_i)\$ \end{aligned}$$

(4) For each $a_i, a_j \in \Sigma$, include the following rules in P.

$$\$(q_0, a_i) \to \$C_i, \quad C_iB_j \to a_iC_j, \quad C_i\$ \to a_i\$$$

(5) Include the following rule in P.

$$\$(q_0, a_0)\$ \to \$\$$$

It is easy to verify that G_M satisfies UPG-condition, because M is deterministic.

An ID of M

$$a_{i_1} \cdots a_{i_{h-1}} \, q \, a_{i_h} a_{i_{h+1}} \cdots a_{i_k}$$

is represented by the following word in G_M.

$$\$A_{i_1} \cdots A_{i_{h-1}}(q, a_{i_h})B_{i_{h+1}} \cdots B_{i_k}\$$$

Such a word is generated and updated by the rules of G_M. A computing process of M is simulated by a reduction process in G_M.

We first show that if a terminal word $w \in T^*$ is generated by G_M, then M accepts w. Since $w \in L(G_M)$, there is a reduction $\$w\$ \overset{*}{\Leftarrow} \$S\$$. Consider this reduction process. First, $\$w\$$ is reduced by the rules in (4) (or the rule in (5) if $w = \varepsilon$). Note that the other rules cannot be used until a symbol (q_0, a_i) appears. When $\$w\$$ is reduced to a word containing (q_0, a_i), it represents the initial ID of M with the input w. After that, the rules in (2) and (3) are used to reduce it. It is easy to see that each step of M's movement is simulated by reducing the word using the rules in (2) and (3). Finally, a symbol of the form (q_f, a_i) must appear at the left end (except \$) of the word, which represents a final ID of M. Otherwise, rules in (1) cannot be used for reduction, and the word $\$w\$$ is not reduced to $\$S\$$. Thus, there is a sequence of ID's that leads M to a final state, and w is accepted by M.

Conversely, suppose a word $w = a_{i_1} \cdots a_{i_j} \in \Sigma^*$ is accepted by M, i.e., there is a sequence of ID's starting from the initial ID

$$q_0 a_{i_1} a_{i_2} \cdots a_{i_j}$$

to a final ID

$$q_f a_{i'_1} a_{i'_2} \cdots a_{i'_k}.$$

Then, as we shall see, there is a reduction from $\$w\$$ to $\$S\$$. First, reducing $\$w\$$ by the rules in (4) (or (5) if $w = \varepsilon$), a word representing the initial ID is obtained. Then, by the rules in (2) and (3), the movement of M is simulated. Since M accepts w, $\$w\$$ must be reduced to a word representing the final ID of M. After that, by the rules in (1) it is reduced to $\$S\$$. Thus, $\$w\$ \overset{*}{\Leftarrow} \$S\$$. Therefore $w \in L(G_M)$.

By above, we can see that $L(G_M) = L(M)$. □

Since DTM is universal and thus can easily simulate a derivation process of a UPG, $\mathcal{L}[\text{UPG}] \subseteq \mathcal{L}[\text{DTM}]$ holds. Therefore, the following theorem is obtained.

Theorem 3.1 $\mathcal{L}[\text{UPG}] = \mathcal{L}[\text{DTM}]$.

4 Monotonic UPG

We now introduce a subclass of UPG called a monotonic UPG (MUPG), and show that the class of languages generated by MUPG's is the same as the class accepted by deterministic linear-bounded automata (DLBA).

Definition 4.1 Let $G = (N, T, P, S, \$)$ be a UPG. G is called a *monotonic UPG* (MUPG), iff each rule $\alpha \to \beta$, except an ε-rule, satisfies $|\alpha| \le |\beta|$. □

MUPG is considered to be a restricted type of context-sensitive grammar that satisfies UPG-condition.

Definition 4.2 A *deterministic linear-bounded automaton* (DLBA) is a system defined by

$$M = (Q, \Sigma, \Gamma, \delta, ¢, \$, q_0, q_f),$$

where $Q, \Sigma, \Gamma, \delta, q_0$, and q_f are the same as in DTM. Symbols $¢$ and $\$$ are left and right end-markers of an input word ($¢, \$ \in \Gamma - \Sigma$), and M cannot go beyond them. Therefore M satisfies the following condition: For every $q_h, q_j \in Q$, $a_i, a_k \in \Gamma$ such that $\delta(q_h, a_i) = (q_j, a_k, d)$,

1. If $a_i \notin \{¢, \$\}$ then $a_k \notin \{¢, \$\}$.
2. If $a_i = ¢$ then $a_k = ¢$ and $d = R$.
3. If $a_i = \$$ then $a_k = \$$ and $d = L$.

An ID for DLBA is defined similarly as in DTM. Starting from an initial ID $¢q_0w\$$ ($w \in \Sigma^*$), if M halts with a final ID $¢q_f\alpha\$$ for some $\alpha \in (\Gamma - \{¢, \$\})^*$, then w is said to be *accepted* by M. The set of all accepted words is the language accepted by M and denoted by $L(M)$. □

Lemma 4.1 $\mathcal{L}[\text{MUPG}] \supseteq \mathcal{L}[\text{DLBA}]$.

Proof. Let $M = (Q, \Sigma, \Gamma, \delta, ¢, \$, q_0, q_f)$ be an arbitrarily given DLBA, where $\Gamma = \{¢, \$, a_1, \cdots, a_n\}$, $\Sigma = \{a_1, a_2, \cdots, a_m\}$ ($m \le n$). An MUPG $G_M = (N, T, P, S, \$)$ such that $L(G_M) = L(M)$ is defined as follows.

$$\begin{aligned} N &= \{S\} \cup \{A_1, \cdots, A_n\} \cup \{B_1, \cdots, B_n\} \\ &\quad \cup \{C_1, \cdots, C_m\} \cup (Q \times \{a_1, \cdots, a_n\}) \\ T &= \Sigma \end{aligned}$$

Let $\Gamma' = \Gamma - \{¢, \$\}$. The set P of rules is as follows:

(1) For each $a_i \in \Gamma'$ include the following rules in P.

$$S \to SB_i, \quad S \to (q_f, a_i)$$

(2) For each $q_h, q_j \in Q$, and $a_i, a_k, a_l \in \Gamma'$, if $\delta(q_h, a_i) = (q_j, a_k, L)$ then include the following rule in P.

$$(q_j, a_l)B_k \to A_l(q_h, a_i)$$

(3) For each $q_h, q_j, q_l \in Q$, and $a_i, a_k \in \Gamma'$, if $\delta(q_h, a_i) = (q_j, a_k, L)$ and $\delta(q_j, ¢) = (q_l, ¢, R)$ then include the following rule in P.

$$\$(q_l, a_k) \to \$(q_h, a_i)$$

(4) For each $q_h, q_j \in Q$, and $a_i, a_k, a_l \in \Gamma'$, if $\delta(q_h, a_i) = (q_j, a_k, R)$ then include the following rule in P.

$$A_k(q_j, a_l) \to (q_h, a_i)B_l$$

(5) For each $q_h, q_j, q_l \in Q$, and $a_i, a_k \in \Gamma'$, if $\delta(q_h, a_i) = (q_j, a_k, R)$ and $\delta(q_j, \$) = (q_l, \$, L)$ then include the following rule in P.

$$(q_l, a_k)\$ \to (q_h, a_i)\$$$

(6) For each $a_i, a_j \in \Sigma$, include the following rules in P.

$$\$(q_0, a_i) \to \$C_i, \quad C_iB_j \to a_iC_j, \quad C_i\$ \to a_i\$$$

(7) If $\varepsilon \in L(M)$, then include the following rule in P. (It is easily decided whether $\varepsilon \in L(M)$, since M is a DLBA.)

$$\$S\$ \to \$\$$$

It is easy to verify that G_M satisfies UPG-condition, since M is deterministic. Further, since there is no length-decreasing rule except (7), it also satisfies the condition of MUPG.

An ID of M

$$¢\, a_{i_1} \cdots a_{i_{h-1}}\, q\, a_{i_h} a_{i_{h+1}} \cdots a_{i_k} \$$$

is represented by the following word in G_M.

$$\$A_{i_1} \cdots A_{i_{h-1}} (q, a_{i_h}) B_{i_{h+1}} \cdots B_{i_k} \$$$

G_M simulates M in the almost same way as in Lemma 3.1, except that at the left or the right end of the word, two steps of M's movement are simulated in one reduction step in G_M by the rules in (3) and (5).

By the same argument as in Lemma 3.1, we can see $L(G_M) = L(M)$. □

Lemma 4.2 $\mathcal{L}$[DLBA] $\supseteq \mathcal{L}$[MUPG].

Proof. Let $G = (N, T, P, S, \$)$ be a given MUPG. A DLBA M_G such that $L(M_G) = L(G)$ is constructed in the following manner. Given an input $w \in T^*$, M performs leftmost reduction of $\$w\$$ on the tape (left end-marker ¢ on the tape is identified with \$ in G). In the case of $w = \varepsilon$, the reduction process can be simulated in the finite-state control of M_G, since G is an MUPG. So, in what follows, we consider the case $w \neq \varepsilon$. Scanning the tape from left to right, M_G searches for a reversely applicable rule. If such a rule $\alpha \to \beta$ is found, M_G does the reverse rewriting to the word. This rewriting is always possible on the tape, because $\alpha \to \beta$ satisfies $|\alpha| \leq |\beta|$ (note that ε-rule cannot be used since $w \neq \varepsilon$). If M_G gets the word $\$S\$$ on the tape, M_G halts in the accepting state. If there is no reversely applicable rule, M_G halts in a non-accepting state. Otherwise, M_G repeats the above process of reduction.

From Corollary 2.1 if $w \in L(G)$ then $\$w\$ \overset{*}{\underset{\text{lmr}}{\Leftarrow}} \$S\$$, thus M_G accepts w. Therefore, $w \in L(G)$ implies $w \in L(M_G)$. Conversely, if $w \in L(M_G)$, then, from the above construction of M_G, $\$w\$ \overset{*}{\underset{\text{lmr}}{\Leftarrow}} \$S\$$ holds. Thus $w \in L(G)$. □

From Lemmas 4.1 and 4.2, the equivalence of MUPG and DLBA is established.

Theorem 4.1 $\mathcal{L}$[MUPG] $= \mathcal{L}$[DLBA].

5 Concluding Remarks

We introduced UPG and its subclass MUPG, and studied their language generating ability. Roughly speaking, UPG and MUPG are regarded to be "uniquely parsable" counterparts of Type-0 grammar and context-sensitive grammar. The result $\mathcal{L}$[UPG] $= \mathcal{L}$[DTM] means that the addition of UPG-condition does not affect generating ability of Type-0 grammar at all. On the other hand, $\mathcal{L}$[MUPG] $= \mathcal{L}$[DLBA] shows that the difference between context-sensitive grammar and MUPG is just the difference between nondeterministic and deterministic LBA's.

It is left for the future study to investigate other useful subclasses of UPG's.

References

[1] Ginsburg, S. and Greibach, S.: Deterministic context free languages, *Inf. Control*, **9**, pp.620–648 (1966).

[2] Hopcroft, J.E. and Ullman, J.D.: *Introduction to Automata Theory, Languages, and Computation*, Addison-Wesley, Reading, Massachusetts, p.418 (1979).

[3] Morita, K., Yamamoto, Y. and Sugata, K.: The complexity of some decision problems about two-dimensional array grammars, *Inf. Sci.*, **30**, pp.241–262 (1983).

[4] Yamamoto, Y. and Morita, K.: Hierarchy of one-dimensional uniquely parsable isometric array grammars, *LA Summer Symposium* (1990).

[5] Yamamoto, Y. and Morita, K.: Two-dimensional uniquely parsable isometric array grammars, *Int. J. Pattern Recogn. Artif. Intell.*, **6**, pp.301–313 (1992).