REMARKS ON REAL-TIME DETERMINISTIC CONTEXT-FREE LANGUAGES

Yoshihide Igarashi
Department of Computer Science
Gunma University, Kiryu, 376 Japan

1. Introduction

The context-free languages are most important language family for the study of compiler design techniques and language specifications. In particular, characterizations of deterministic context-free languages by automata are important for parsing algorithms [3][7]. Several subclasses of deterministic context-free languages have been studied in a way that we ask whether placing restrictions on the deterministic pushdown automata affects the family of languages accepted [4][5][6][10]. The real-time deterministic context-free languages are one of such subclasses.

In this paper we establish a pumping lemma for the real-time deterministic context-free languages. The lemma is an interesting character of the subclass and useful to show that a given deterministic context-free language is not real-time.

In the main we employ the definitions and notation given in standard texts such as [3] or [8]. If $w$ is a word (i.e., a string of symbols), $|w|$ denotes its length. $\varepsilon$ denotes the word of zero length. If $x$ is a pair of words, $|x|$ denotes the length of its second component (i.e., if $x = (q, a)$, $|x| = |a|$). If $S$ is a set, $\#(S)$ denotes the number of elements in $S$. A deterministic pushdown automaton (abbreviated DPDA) is a deterministic acceptor with a one-way input tape, a pushdown tape, and a finite state control. It can be specified by a 7-tuple $(Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$,
where

(1) $Q$ is a finite set of states,
(2) $\Sigma$ is a finite set of input symbols (the input alphabet),
(3) $\Gamma$ is a finite set of pushdown symbols (the pushdown alphabet),
(4) $q_0$ is in $Q$ (the initial state),
(5) $Z_0$ is in $\Gamma$ (the start symbol),
(6) $F \subseteq Q$ (the set of final states), and
(7) $\delta$ is a mapping from $Q \times (\Sigma \cup \{\varepsilon\}) \times \Gamma$ to the finite subsets of $Q \times \Gamma^*$ which has the following restrictions: For each $q$ in $Q$ and $Z$ in $\Gamma$
   (a) either $\delta(q, a, Z)$ contains exactly one element for all $a$ in $\Sigma$
   and $\delta(q, \varepsilon, Z) = \emptyset$, or $\delta(q, \varepsilon, Z)$ contains exactly one element and
   $\delta(q, a, Z) = \emptyset$ for each $a$ in $\Sigma$, and (b) if $\delta(q, \pi, Z_0) \neq \emptyset$ for
   $\pi$ in $\Sigma \cup \{\varepsilon\}$, then $\delta(q, \pi, Z_0) = \{(p, Z_0 \gamma)\}$ for some $p$ in $Q$ and $\gamma$
   in $\Gamma^*$.

Certain strings over $\Gamma$ are interpreted as the contents of the pushdown store. We assume that the bottom of the store is on the left and top on the right. A configuration is a pair from $Q \times \Gamma^*$. The initial configuration
$(q_0, Z_0)$ is denoted by $c_s$. A DPDA makes a move $(q, \alpha A)^W (p, \alpha \gamma)$ if and
only if there is some transition $\delta(q, \pi, A) = (p, \gamma)$. In particular, if
$\pi = \varepsilon$, it is called an $\varepsilon$-move. If $\pi$ is in $\Sigma$, then this symbol is considered
to have been read. A computation is a sequence of such moves through suc-
cessive configurations. Suppose $w$ is a string over $\Sigma$. If we obtain con-
figuration $c'$ from configuration $c$ by the successive read of $w$, the computa-
tion is denoted by $c \xrightarrow{w} c'$. A word $w$ is accepted by DPDA $M = (Q, \Sigma, \Gamma, \delta,$
$q_0, Z_0, F)$ if for some configuration $c$ with the first component of $c$ belong-
ing to $F$, $(q_0, Z_0)^W c$. The language accepted by $M$ is denoted by $L(M)$. That
is, $L(M) = \{w \in \Sigma^* \mid c_s = (q_0, Z_0)^W c$, the first component of $c$ belongs
to F). The language accepted by a DPDA is called a deterministic context-free language (abbreviated DCFL).

Let \( c_1 \vdash^w c' \) be a computation. \( c_1 \) is a stacking configuration in the computation if and only if it is not followed by any configuration of height \( \leq |c_1| \) in the computation. Note that, whether or not \( c_1 \) is a stacking configuration depends on what computation is considered. That is, if we say that \( c_1 \) is a stacking configuration in the computation \( c \vdash^w c' \), it means that \( c_1 \) is a stacking configuration for the whole of \( c \vdash^w c' \).

DPDA \( M = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F) \) is said to be quasi-real-time if and only if there exists an integer \( t \geq 0 \) such that for any \( q, q' \) in \( Q \) and \( \gamma, \gamma' \) in \( \Gamma^* \), \( (q, \gamma) \vdash^E \ldots \vdash^E (q', \gamma') \) implies that the number of steps of this computation is not greater than \( t \). In particular, \( M \) is said to be real-time if and only if \( t = 0 \) (i.e., if and only if \( \delta(q, \varepsilon, Z) = \emptyset \) for all \( q \) in \( Q \) and \( Z \) in \( \Gamma \)). A language \( L \) is called (quasi-) real-time if and only if \( L = L(M) \) for some (quasi-) real-time DPDA \( M \). Our (quasi-) real-time DCFL's correspond to \( \Delta_0 \)-(quasi-) real-time languages defined in [4] and [6]. It is known that the class of quasi-real-time DCFL's coincides with the class of real-time DCFL's [4][6].

2. Pumping Lemmas for Real-Time DCFL's

The pumping lemma and Ogden's lemma are useful and fundamental properties of CFL's [1][3][9][11]. Wise has established a necessary and sufficient version of the classic pumping lemma for CFL's [13], and Jaffe has established a necessary and sufficient pumping lemma for regular languages [9]. Stanat has recently shown another characterization of regular languages using a modified pumping lemma [12]. It is also interesting to ask whether we can derive a useful pumping lemma for each of well-known subclasses of
DCFL's, or to ask whether we can establish a necessary and sufficient pumping lemma for such a subclass.

In this section we first show a simple pumping lemma for real-time DCFL's. Then we show a version of the pumping lemma which will be useful to show that a language is not a real-time DCFL.

**Definition 1.** Let $L$ be a language (i.e., a subset of $\Sigma^*$). $x$ in $\Sigma^*$ is equivalent under $L$ to $y$ in $\Sigma^*$ (denoted by $x \equiv_L y$) if and only if for any $w$ in $\Sigma^*$ both $xw$ and $yw$ are in $L$ or both $xw$ and $yw$ are not in $L$.

The relation $\equiv_L$ is an equivalence relation on $\Sigma^*$. $x \not\equiv_L y$ means that $x$ and $y$ are not equivalent under $L$.

**Lemma 1 (Simple pumping lemma for real-time DCFL's).** Let $L$ be a real-time DCFL. Then there are a pair of constants $k_1 > 0$ and $k_2$, depending only on $L$, that satisfy the following property (*):

(*) If $x_1, x_2, \ldots, x_n$ are $n$ strings on $\Sigma$ such that

(*-1) for any $1 \leq i < j \leq n$ $x_i \not\equiv_L x_j$, and

(*-2) for each $i$ ($1 \leq i \leq n$) there is $y_i$ in $\Sigma^*$ satisfying

(*-2-1) $x_i y_i$ is in $L$, and

(*-2-2) $|y_i| \leq (\log_2 n)/k_1 + k_2$,

then for at least one $r$ ($1 \leq r \leq n$) we may write $x_r = x_{r_1} x_{r_2} x_{r_3}$ such that

(*-3) $|x_{r_2}| \geq 1$, and

(*-4) for all $t \geq 0$ $x_{r_1} (x_{r_2})^t x_{r_3} y_r$ is in $L$.

**Proof.** Let $L$ be recognized by a real-time DPDA $M = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$. Without loss of generality we may assume that $\#(\Gamma)$ is not less than 2. For $w$ in $\Sigma^*$ let $\text{CONF}_M(w)$ be the configuration of $M$ when input string $w$ has been read (i.e., $c_s = (q_0, Z_0) \vdash w \text{CONF}_M(w)$). Let $k_1 = \log_2 \#(\Gamma)$ and $k_2 = (\log_2(\#(\Gamma) - 1) - \log_2 \#(Q))/\log_2 \#(\Gamma) - \#(Q)\#(\Gamma) - 1$. Let $x_1, \ldots, x_n$ be
n strings over $\Sigma$ that satisfy (\#-1) and (\#-2) above, and let $h = \max \{|\text{CONF}_M(x_i)| : 1 \leq i \leq n\}$. From (\#-1) all of $\text{CONF}_M(x_1), \text{CONF}_M(x_2), \ldots, \text{CONF}_M(x_n)$ are distinct. Therefore, $\#(Q)(1 + \#(\Gamma) + \ldots + (\#(\Gamma))^{h-1}) \geq n$. Note that the leftmost symbol of the pushdown store is always $Z_0$. Solving this inequality we have

$$h > (\log_2 n + \log_2 (\#(\Gamma) - 1) - \log_2 \#(Q))/\log_2 \#(\Gamma)$$

$$= (\log_2 n)/k_1 + k_2 + \#(Q)/(\#(\Gamma) + 1).$$

Let $r$ be an index such that $h = |\text{CONF}_M(x_r)|$. From this inequality and (\#-2) $|\text{CONF}_M(x_r)| > \#(Q)\#(\Gamma) + 1 + |y_r|$. Therefore, for the whole computation of the input string $x_r y_r$ there are at least $\#(Q)\#(\Gamma) + 1$ stacking configurations among the configurations from $c_s$ to $\text{CONF}_M(x_r)$. Hence, there are at least two configurations in this part such that their pairs of the states and top pushdown tape symbols are identical. Let these configurations be $\text{CONF}_M(x_{r_1})$ and $\text{CONF}_M(x_{r_1}x_{r_2})$. Since $x_{r_1}y_r$ is in $L$, for all $t > 0$ $x_{r_1}^t x_{r_2} x_{r_3} y_r$ is in $L$, where $x_r = x_{r_1} x_{r_2} x_{r_3}$ and $|x_r| > 1$. Q. E. D.

The notation $\text{CONF}_M$ introduced in the above proof will be used in the following. The above lemma is not strong enough to use it as a tool for proving that a given DCFL is not real-time. For example, $L = \{a^i b^j c^k a^l \mid i > 0, j > k > 0\}$ is not a real-time DCFL. However, we cannot lead any contradiction by using Lemma 1 from the assumption that $L$ is a real-time DCFL. We, therefore, are requested to prepare a powerful version of Lemma 1 for this purpose. This situation is analogous to the fact that Ogden's lemma is more powerful than the classic pumping lemma for CFL's. The next lemma is such a version for real-time DCFL's.

**Lemma 2** (Strong pumping lemma for real-time DCFL's). Let $L$ be a real-time DCFL. Then there are constants $k_1, k_2 > 0$ and $k_3$, depending only on $L$, that satisfy the following property (\#):
Let $n$ be an integer such that $n > k_1$, and let $m$ be an integer. If there are $n$ strings $x_1, \ldots, x_n$ on $\Sigma$ such that for each pair of $i$ and $j$ ($1 \leq i \leq n$, $1 \leq j \leq m$) there is a string $y_{ij}$ satisfying

\[(\ast-1)\] for each $i$ ($1 \leq i \leq n$) and for any pair of $j_1$ and $j_2$ ($1 \leq j_1 < j_2 \leq m$) $x_1 y_{ij_1} \not\in L x_1 y_{ij_2}$,

\[(\ast-2)\] for any pair of $i_1$ and $i_2$ ($1 \leq i_1 < i_2 \leq n$) and for any pair of $j_1$ and $j_2$ ($1 \leq j_1 \leq m$, $1 \leq j_2 \leq m$) the concatenation of $x_{i_1}$ and any initial substring of $y_{i_1 j_1}$ and the concatenation of $x_{i_2}$ and any initial substring of $y_{i_2 j_2}$ are not equivalent under $L$ (i.e., if $\overline{y_{i_1 j_1}}$ is an initial substring of $y_{i_1 j_1}$, and if $\overline{y_{i_2 j_2}}$ is an initial substring of $y_{i_2 j_2}$, then $x_{i_1} \overline{y_{i_1 j_1}} \not\in L x_{i_2} \overline{y_{i_2 j_2}}$), and

\[(\ast-3)\] for each pair of $i$ ($1 \leq i \leq n$) and $j$ ($1 \leq j \leq m$) there exists a string $w_{ij}$ such that $x_i y_{ij} w_{ij}$ is in $L$ and $|w_{ij}| \leq (\log_2 m)/k_2 + k_3$,

then there exists at least one pair of $p$ and $q$ ($1 \leq p \leq n$, $1 \leq q \leq m$) such that

\[(\ast-4)\] we may write $x = x_1 x_2 x_3$ where $|x_1 p_2| \geq 1$, and

\[(\ast-5)\] for all $t \geq 0$, $x_1 (x_2 p_3)^t p_2 p_3 p_4 p_5$ is in $L$.

Proof. Let $L$ be accepted by a real-time DPDA $M = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$. Without loss of generality we may assume that $\#(\Gamma)$ is not less than 2.

The proof will proceed as the proof of the previous lemma. Let $k_1 = \#(Q)(1 + \#(\Gamma) + \ldots + (\#(\Gamma))^{\#(Q)} \#(\Gamma))$ and $k_2 = \log_2 \#(\Gamma)$, and let $k_3 = (\log_2 (\#(\Gamma) - 1) - \log_2 \#(Q))/\log_2 \#(\Gamma) - \#(Q) \#(\Gamma) - 1$. If $m \leq k_1$, then $(\log_2 m)/k_2 + k_3 < 0$. In this case, for any pair of $i$ ($1 \leq i \leq n$) and $j$ ($1 \leq j \leq m$) there does not exist $w_{ij}$ satisfying $(\ast-3)$. Therefore, in this case the assertion of the lemma holds. We suppose that $m > k_1$ and that there
exist \( x_i \) (\( 1 \leq i \leq n \)), \( y_{ij} \) (\( 1 \leq i \leq n \), \( 1 \leq j \leq m \)) and \( w_{ij} \) (\( 1 \leq i \leq n \), \( 1 \leq j \leq m \)) satisfying (*-1), (*-2) and (*-3), where \( n > k \).

Consider the following classes of strings in \( \Sigma^* \).

\[
A(1) = \{ x_1 y_{11}, x_1 y_{12}, \ldots, x_1 y_{1m} \}
\]

\[
A(2) = \{ x_2 y_{21}, x_2 y_{22}, \ldots, x_2 y_{2m} \}
\]

\[
\vdots
\]

\[
A(n) = \{ x_n y_{n1}, x_n y_{n2}, \ldots, x_n y_{nm} \}.
\]

From (*-1) for each \( i \) (\( 1 \leq i \leq n \)) all of \( \text{CONF}_M(x_i y_{ij}) \), \ldots, \( \text{CONF}_M(x_i y_{im}) \) should be distinct. Therefore, for each \( i \) (\( 1 \leq i \leq n \)) there exists at least one element in \( A(i) \), say \( x_i y_{ij} \), such that \( \left| \text{CONF}_M(x_i y_{ij}) \right| \geq g \), where \( g \) is the least integer satisfying \( \#(Q)(1 + \#(R) + \ldots + (\#(R)^{g-1}) \geq m \). Let these strings be \( x_1 y_{ij}, \ldots, x_n y_{nj} \). For each \( i \) (\( 1 \leq i \leq n \)) let \( \overline{y}_{ij} \) be an initial substring of \( y_{ij} \) such that \( \left| \text{CONF}_M(x_i \overline{y}_{ij}) \right| = \min \left| \text{CONF}_M(x_i \overline{y}_{ij}) \right| \). From (*-2) all of \( \text{CONF}_M(x_i y_{ij}) \), \ldots, \( \text{CONF}_M(x_n y_{nj}) \) should be distinct. From this fact and \( n > k \) there exists at least one element, say \( x_{p} y_{pj} \) among \( x_1 y_{ij}, \ldots, x_n y_{nj} \) such that \( \left| \text{CONF}_M(x_{p} y_{pj}) \right| \geq \#(Q)(\#(R) + 2 \). That is, for any initial substring \( \overline{y}_{pj} \) of \( y_{pj} \) \( \left| \text{CONF}_M(x_{p} y_{pj}) \right| \geq \#(Q)(\#(R) + 2 \). Hence, for the computation from \( c_s \) to \( \text{CONF}_M(x_{p} y_{pj}) \) there are at least \( \#(Q)(\#(R) + 1 \) stacking configurations in the first \( |x_p| \) steps. Since \( \left| \text{CONF}_M(x_{p} y_{pj}) \right| \geq g \) and \( |w_{pj}| \leq (\log_2 n)/k_2 + k_3 \), the height of the pushdown tape during the last \( |w_{pj}| \) steps of \( c_s = (q_0, Z_0) \rightarrow \ldots \rightarrow \text{CONF}_M(x_{p} y_{pj} w_{pj}) \) is at least \( \#(Q)(\#(R) + 2 \). Hence, for the computation \( c_s \rightarrow \ldots \rightarrow \text{CONF}_M(x_{p} y_{pj} w_{pj}) \) the first \( \#(Q)(\#(R) + 1 \) stacking configurations locate in the first \( |x_p| \) steps of the computation. Thus there are at least two stacking configurations in the first \( |x_p| \) steps of the computation \( c_s \rightarrow \ldots \rightarrow \text{CONF}_M(x_{p} y_{pj} w_{pj}) \) such that their pairs of states and top pushdown tape symbols are identical. Let these configurations be
CONF\textsubscript{M}(x\textsubscript{p1}) and CONF\textsubscript{M}(x\textsubscript{p1}x\textsubscript{p2}), where \(|x\textsubscript{p2}| \geq 1\). Removing or repeating the part of the computation corresponding to \(x\textsubscript{p2}\) does not affect the last state of the whole computation. Since \(x\textsubscript{p1}p\textsubscript{p1}p\textsubscript{p1}p\textsubscript{p2}x\textsubscript{p2}\) is in L, for all \(t \geq 0\) \(x\textsubscript{p1}(x\textsubscript{p2})^tx\textsubscript{p3}y\textsubscript{pq}w\textsubscript{pq}\) is in L, where \(q = j\textsubscript{p}\) and \(x\textsubscript{p} = x\textsubscript{p1}x\textsubscript{p2}x\textsubscript{p3}\). Q. E. D.

For a certain string in a real-time DCFL Lemma 2 specifies a range of the pumping position of the string, whereas Lemma 1 does not. This specification of the pumping position is indispensible to use the lemma as a tool to show that a given language is not a real-time DCFL.

3. Applications

Strong pumping lemma (Lemma 2) guarantees a scheme for proving that a given language is not a real-time DCFL. We show this proving scheme by examples.

**Example 1.** \(L_1 = \{a^ib^ja^i | i, j \geq 1\}\)

Harrison and Havel proved that \(L_1\) is not a \(\Lambda_2\)-real-time language (Theorem 2.4 of [4]). The class of \(\Lambda_2\)-real-time languages is properly included in the class of \(\Lambda_0\)-real-time languages [4] (i.e., real-time DCFL's of this paper). By using Lemma 2 we can easily show that \(L_1\) is not a real-time DCFL.

Assume for the sake of contradiction that \(L_1\) is a real-time DCFL. Let \(k_1, k_2, k_3\) be constants described in Lemma 2. Let \(n > k_1\) and let \(m\) be an integer such that \(n \leq (\log_2 m)/k_2 + k_3\). We choose \(x_1 = a^i, y_{ij} = b^j\) and \(w_{ij} = a^i\) for each \(i (1 \leq i \leq n)\) and each \(j (1 \leq j \leq m)\). Then (\#-1), (\#-2) and (\#-3) are satisfied. Then from (\#-4) and (\#-5) for some pair of \(i\) and \(j\) we may write \(a^i = a^{i_1}a^{i_2}a^{i_3}, where i_2 \geq 1\) and for all \(t \geq 0\) \(a^{i_1}(a^{i_2})^{t}a^{i_3}b^ja^i\) is in \(L_1\). This is a contradiction. We, therefore, conclude that \(L_1\) is not a real-time DCFL.
Lemma 2 is powerful enough for our purpose. In fact, we do not know at present any DCFL that is not real-time but that cannot be proved by Lemma 2 not to be real-time. However, it may be valuable to prepare a version of Lemma 2 that seems to be easier for the reader to use it. In the rest of this section we describe such a version although it is essentially the same as Lemma 2.

Definition 1. Let \( f(n) \) be a function from nonnegative integers to nonnegative integers. A language \( L \) is \( f(n) \)-characteristic if and only if the following property (*) is satisfied:

(*) For arbitrary positive integers \( n \) and \( m \) there exist \( n \) strings \( x_1, \ldots, x_n \) and \( n \times m \) strings \( y_{ij} \) \((1 \leq i \leq n, 1 \leq j \leq m)\) such that

(*-1) for each \( i \) \((1 \leq i \leq n)\) and for any pair of \( j_1 \) and \( j_2 \) \((1 \leq j_1 < j_2 \leq m)\) \( x_i y_{ij} \notin L x_i y_{ij} \),

(*-2) for any pair of \( i_1 \) and \( i_2 \) \((1 \leq i_1 < i_2 \leq n)\), any \( j_1 \) and \( j_2 \) \((1 \leq j_1 \leq m, 1 \leq j_2 \leq m)\), the concatenation of \( x_{i_1} \) and any initial substring of \( y_{i_1 j_1} \) and the concatenation of \( x_{i_2} \) and any initial substring of \( y_{i_2 j_2} \) are not equivalent under \( L \), and

(*-3) for any pair of \( i \) and \( j \) there exists a string \( w_{ij} \) such that

(*-3-1) \( |w_{ij}| = f(n) \),

(*-3-2) \( x_i y_{ij} w_{ij} \) is in \( L \), and

(*-3-3) for any non-null substring \( x''_i \) of \( x_i \), there exists a non-negative integer \( t \) such that \( x_i'(x'')^t - x_i y_{ij} w_{ij} \) is not in \( L \), where \( x_i = x_i' x''_i x_i'' \).

Lemma 3. If there is a function \( f(n) \) such that \( L \) is \( f(n) \)-characteristic, then \( L \) is not a real-time DCFL.

Proof. Let \( L \) be \( f(n) \)-characteristic. Assume for the sake of contradiction that \( L \) is accepted by a real-time DPDA \( M = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F) \).
Let \( n \) and \( m \) be integers such that \( n > k_1 \) and \( f(n) \leq \frac{(\log_2 m)}{k_2 + k_3} \), where \( k_1, k_2 \) and \( k_3 \) are constants given in the proof of Lemma 2. Let \( x_{ij} (1 \leq i \leq n), y_{ij} (1 \leq i \leq n, 1 \leq j \leq m) \) and \( w_{ij} (1 \leq i \leq n, 1 \leq j \leq m) \) be strings satisfying conditions \((*)-1\), \((*)-2\) and \((*)-3\) of Definition 1. These strings satisfy conditions \((*)-1\), \((*)-2\) and \((*)-3\) of Lemma 2. Therefore, \((*)-4\) and \((*)-5\) of Lemma 2 should hold since \( L \) is assumed to be a real-time DCFL. However, \((*)-4\) and \((*)-5\) of Lemma 2 are contrary to \((*)-3-3\) of Definition 1. We, therefore, conclude that our assumption is wrong. That is, \( L \) is not a real-time DCFL.

Q. E. D.

**Example 2.** \( L_2 = \{a^ib^ja^i | i, j \geq 1\} \). This language has been given by Gisburg and Greibach \(^{(2)}\) as an example of a DCFL that is not real-time. By using Lemma 3 we prove that \( L_2 \) is not a real-time DCFL. Let \( f(n) = n \). For \( n > 1 \) and \( m \geq 1 \) we choose \( x_i = a^i (1 \leq i \leq n), y_{ij} = b^j (1 \leq i \leq n, 1 \leq j \leq m) \) and \( w_{ij} = a^i (1 \leq i \leq n, 1 \leq j \leq m) \). Then \((*)-1\), \((*)-2\) and \((*)-3\) in Definition 1 hold. That is, \( L_2 \) is \( n \)-characteristic. From Lemma 3 \( L_3 \) is not a real-time DCFL.

**Example 3.** \( L_3 = \{a^ib^jc^ra^i | i \geq 1, j \geq r \geq 1\} \). Let \( f(n) = n + 1 \). For \( n \geq 1 \) and \( m \geq 1 \) we choose \( x_i = a^i (1 \leq i \leq n), y_{ij} = b^j (1 \leq i \leq n, 1 \leq j \leq m) \) and \( w_{ij} = c^a (1 \leq i \leq n, 1 \leq j \leq m) \). Then \((*)-1\), \((*)-2\) and \((*)-3\) in Definition 1 hold. Therefore, \( L_3 \) is \((n + 1)\)-characteristic, and from Lemma 3 it is not a real-time DCFL.

**Example 4.** \( L_4 = \{a^ib^jc^pa^q | i, j, p \geq 1, i \neq q \text{ and } j \neq p\} \). Let \( f(n) = n! + n + 1 \). For \( n \geq 1 \) and \( m \geq 1 \) we choose \( x_i = a^i (1 \leq i \leq n), y_{ij} = b^{i+1} (1 \leq i \leq n, 1 \leq j \leq m) \) and \( w_{ij} = c^{i+1} (1 \leq i \leq n, 1 \leq j \leq m) \). Then it is obvious that \((*)-1\), \((*)-2\), \((*)-3-1\) and \((*)-3-2\) in Definition 1 hold. For any non-null substring \( a^r \) of \( a^i \) \( r = |a^r| \) is a divisor of \( i! \). Thus we can write \( a^{i-r}(a^r)(i!/r)+1 = a^{i!+1} \). Therefore, for any \( r \) \( (1 \leq r \leq i) \) and
t = \frac{i!}{r}, \ a^{i-r(a^r)_{t+1}} b^{j+1} c^{i+1} d^{i+1} = a^{i+1} b^{j+1} c^{i+1} d^{i+1} \text{ is not in } L_4. \text{ Thus } (*)-3-3 \text{ in Definition 1 hold, too. Therefore, } L_4 \text{ is } (n!+n+1)\text{-characteristic, and from Lemma 3 it is not a real-time DCFL.}

Note that \( L_5 = \{ a^{i} b^{j} c^{r} a^{i} \mid 1 \leq j \leq r, 1 \geq i \} \) is a real-time DCFL. Therefore, for any function \( f(n) \) \( L_5 \) is not \( f(n) \)-characteristic. For example, suppose that for \( n \geq 1 \) and \( m \geq 1 \) we choose \( x_i = a^i \) \( (1 \leq i \leq n) \), and \( y_{i,j} = b^j \) \( (1 \leq i \leq n, 1 \leq j \leq m) \). In this case, when \( m \) is sufficiently large compared with \( f(n) \), say \( m = 2 f(n) \), we cannot choose any \( w_{i,j} \) \( (1 \leq i \leq n, 1 \leq j \leq m) \) that satisfies \((*)-3-1\) and \((*)-3-2\) in Definition 1 simultaneously. Therefore, these choices of \( x_i \) \( (1 \leq i \leq n) \) and \( y_{i,j} \) \( (1 \leq j \leq m) \) are not successful to show that \( L_5 \) would be \( f(n) \)-characteristic.

We do not know at present whether Lemma 2 is a sufficient condition for real-time DCFL's. We invite the reader to consider the following problems worthy of further investigation:

1) Is Lemma 2 a necessary and sufficient condition for real-time DCFL's?

2) Find an elegant characterization of real-time DCFL's that is a necessary and sufficient condition for real-time DCFL's.

3) Find an elegant characterization of each subclass of DCFL's.

References


SIGACT NEWS 10(2) (1978) 48-49.


