ON THE NONTERMINAL COMPLEXITY OF LEFT RANDOM CONTEXT E0L GRAMMARS

Petr Zemek
Doctoral Degree Programme (1), FIT BUT
E-mail: xzemek02@stud.fit.vutbr.cz

Supervised by: Alexander Meduna
E-mail: meduna@fit.vutbr.cz

Abstract: The present paper studies the nonterminal complexity of left random context E0L grammars. More specifically, it proves that every recursively enumerable language can be generated by a left random context E0L grammar with nine nonterminals. In the conclusion, some open problems related to the achieved result are stated.

Keywords: Formal languages, left random context E0L grammars, nonterminal complexity

1 INTRODUCTION

A left random context E0L grammar, introduced in [7][1], is a variant of an E0L grammar regulated by context conditions (see Chapter 8 of [2]). Basically, it is an E0L grammar where every rewriting rule is equipped with two finite sets of nonterminals. One set contains permitting symbols while the other has forbidding symbols. A rule like this can rewrite a symbol provided that each of its permitting symbols occurs to the left of the rewritten symbol in the current sentential form while each of its forbidding symbols is absent therein.

As demonstrated by several recent studies, such as [1, 3, 5, 6, 8], the study of descriptional complexity of formal models represents a modern trend in formal language theory. One of the studied parameter is the number of nonterminals needed to sustain the power of a grammar. We follow this trend by showing that every recursively enumerable language can be generated by a left random context E0L grammar with nine nonterminals.

To demonstrate that the number of nonterminals of a computationally complete grammar can be bounded, several techniques have been used [1]. In [5] and [8], the authors simulate a phrase-structure grammar in the Geffert normal form (see [4]), where the number of nonterminals is bounded by a very small constant. In [3], the number of nonterminals of graph controlled, programmed, and matrix grammars to generate any recursively enumerable language is reduced by a simulation of a register machine, which is a Turing machine with integers stored in counters rather than with words written on tapes. A similar variant of a Turing machine, called two-counter machine, is used in [1] to prove that every recursively enumerable language can be generated by a scattered context grammar with only two nonterminals. We use the technique based on the Geffert normal form.

The paper is organized as follows. First, Section 2 gives all the necessary terminology. Then, Section 3 proves that every recursively enumerable language can be generated by a left random context E0L grammar with nine nonterminals. In the conclusion of this paper, Section 4 states two open problems related to the achieved result.

[1]Let us note that [7] represents a not yet published work containing ongoing research.

2 PRELIMINARIES

We assume that the reader is familiar with formal language theory (see [9]). For a set, Q, $\mathrm{card}(Q)$ denotes the cardinality of Q, and 2^Q denotes the power set of Q. For an alphabet (finite non-empty set), V, V^* represents the free monoid generated by V under the operation of concatenation. The unit of V^* is denoted by ε. Define $V^+ = V^* - \{\varepsilon\}$. For a string, $x \in V^*$, $|x|$ denotes the length of x, and $\mathrm{alph}(x)$ denotes the set of symbols occurring in x.

A *phrase-structure grammar* is a quadruple, $G = (N, T, P, S)$, where N is an alphabet of *nonterminals*, T is an alphabet of *terminals*, $N \cap T = \emptyset$, $S \in N$ is the *start symbol*, and $P \subseteq (N \cup T)^* N (N \cup T)^* \times (N \cup T)^*$ is a finite relation, called the set of *rules*. Each $(x, y) \in P$ is written as $x \rightarrow y$. Set $V = N \cup T$. The relation of a *direct derivation*, symbolically denoted by $\Rightarrow$, is defined as follows: if $u, v \in V^*$, and $x \rightarrow y \in P$, then $uxv \Rightarrow uyv$ in G. Let $\Rightarrow^n$ and $\Rightarrow^*$ denote the nth power of $\Rightarrow$, for some $n \geq 0$, and the reflexive-transitive closure of $\Rightarrow$, respectively. The *language of* G is denoted by $L(G)$ and defined as $L(G) = \{w \in T^* \mid S \Rightarrow^* w\}$.

Let G be a phrase-structure grammar. G is in the *Geffert normal form* (see [4]) if it is of the form $G = (\{S, A, B, C\}, T, P \cup \{ABC \rightarrow \varepsilon\}, S)$, where P contains only rules of the form (i) $S \rightarrow uSa$, (ii) $S \rightarrow uSv$, (iii) $S \rightarrow uv$, where $u \in \{A, AB\}^*$, $v \in \{BC, C\}^*$, and $a \in T$.

Lemma 1 (see [4])**.** *Let K be a recursively enumerable language. Then, there is a phrase-structure grammar in the Geffert normal form, G, such that $L(G) = K$. In addition, every successful derivation in G is of the form $S \Rightarrow^* w_1 w_2 w_3$ by rules from P, where $w_1 \in \{A, AB\}^*$, $w_2 \in \{BC, C\}^*$, $w_3 \in T^*$, and $w_1 w_2 w_3 \Rightarrow^* w_3$ is derived by $ABC \rightarrow \varepsilon$.*

A *left random context E0L grammar* (see [7]) is a quadruple, $G = (V, T, P, w)$, where V is the *total alphabet*, $T \subseteq V$ is an alphabet of *terminals*, $N = V - T$ is an alphabet of *nonterminals*, $w \in V^+$ is the *start string*, and $P \subseteq V \times V^* \times 2^N \times 2^N$ is finite. By analogy with phrase-structure grammars, elements of P are called *rules* and instead of $(X, y, U, W) \in P$, we write $\lfloor X \rightarrow y, U, W \rfloor$. The relation of a *direct derivation*, symbolically denoted by $\Rightarrow$, is defined as follows: if $u = X_1 X_2 \cdots X_k$, $v = y_1 y_2 \cdots y_k$, $\lfloor X_i \rightarrow y_i, U_i, W_i \rfloor \in P$, $U_i \subseteq \mathrm{alph}(X_1 X_2 \cdots X_{i-1})$, and $\mathrm{alph}(X_1 X_2 \cdots X_{i-1}) \cap W_i = \emptyset$, for all i, $1 \leq i \leq k$, for some $k \geq 1$, then $u \Rightarrow v$ in G. For $\lfloor X \rightarrow y, U, W \rfloor \in P$, U and W are called the *left permitting context* and the *left forbidding context*, respectively. Let $\Rightarrow^n$ and $\Rightarrow^*$ denote the nth power of $\Rightarrow$, for some $n \geq 0$, and the reflexive-transitive closure of $\Rightarrow$, respectively. The *language of* G is denoted by $L(G)$ and defined as $L(G) = \{x \in T^* \mid w \Rightarrow^* x\}$.

3 MAIN RESULT

In this section, we prove that every recursively enumerable language can be generated by a left random context E0L grammar with nine nonterminals.

Theorem 1. *Let K be a recursively enumerable language. Then, there is a left random context E0L grammar, $H = (V, T, P, w)$, such that $L(H) = K$ and* $\mathrm{card}(N) = 9$.

Proof. Let K be a recursively enumerable language. Then, by Lemma 1, there is a phrase-structure grammar in the Geffert normal form, $G = (\{S, A, B, C\}, T, P \cup \{ABC \rightarrow \varepsilon\}, S)$, such that $L(G) = K$. We next construct a left random context E0L grammar, H, such that $L(H) = L(G)$. Set $N = \{S, A, B, C\}$, $V = \{S, A, B, C\} \cup T$, and $N' = N \cup \{\bar{A}, \hat{A}, \bar{B}, \hat{B}, \#\}$. Define $H = (V', T, P', S\#)$, where initially $V' = N' \cup T$ and $P' = \{\lfloor a \rightarrow a, \emptyset, \emptyset \rfloor \mid a \in T\} \cup \{\lfloor X \rightarrow X, \emptyset, \{\#\} \rfloor \mid X \in \{A, B, C\}\}$. To finish the construction, apply the following six steps:

(1) add $\lfloor \# \rightarrow \#, \emptyset, \{\bar{A}, \bar{B}, \hat{A}, \hat{B}\} \rfloor$, $\lfloor \# \rightarrow \#, \{\hat{A}, \hat{B}, C\}, \{S\} \rfloor$, and $\lfloor \# \rightarrow \varepsilon, \emptyset, N' - \{\#\} \rfloor$ to P';

(2) for each $S \to uSa \in P$, where $u \in \{A, AB\}^*$ and $a \in T$,
add $\lfloor S \to uS\#a,\ \emptyset,\ \{\bar{A}, \bar{B}, \hat{A}, \hat{B}, \#\}\rfloor$ to P';

(3) for each $S \to uSv \in P$, where $u \in \{A, AB\}^*$ and $v \in \{BC, C\}^*$,
add $\lfloor S \to uSv, \emptyset, \{\bar{A}, \bar{B}, \hat{A}, \hat{B}, \#\}\rfloor$ to P';

(4) for each $S \to uv \in P$, where $u \in \{A, AB\}^*$ and $v \in \{BC, C\}^*$,
add $\lfloor S \to uv,\ \emptyset,\ \{\bar{A}, \bar{B}, \hat{A}, \hat{B}, \#\}\rfloor$ to P';

(5) add $\lfloor A \to \bar{A},\ \emptyset,\ \{S, \bar{A}, \bar{B}, \hat{A}, \hat{B}, \#\}\rfloor$, $\lfloor B \to \bar{B},\ \emptyset,\ \{S, \bar{A}, \bar{B}, \hat{A}, \hat{B}, \#\}\rfloor$, $\lfloor A \to \hat{A},\ \emptyset,\ \{S, \bar{A}, \bar{B}, \hat{A}, \hat{B}, \#\}\rfloor$, and $\lfloor B \to \hat{B},\ \emptyset,\ \{S, \bar{A}, \bar{B}, \hat{A}, \hat{B}, \#\}\rfloor$ to P';

(6) add $\lfloor \bar{A} \to A,\ \emptyset,\ \{S, A, B, C, \hat{A}, \hat{B}, \#\}\rfloor$, $\lfloor \bar{B} \to B,\ \emptyset,\ \{S, A, B, C, \hat{A}, \hat{B}, \#\}\rfloor$, $\lfloor \hat{A} \to \varepsilon,\ \emptyset,\ \{S, A, B, C, \hat{A}, \hat{B}, \#\}\rfloor$, $\lfloor \hat{B} \to \varepsilon,\ \{\hat{A}\},\ \{S, A, B, C, \hat{B}, \#\}\rfloor$, $\lfloor C \to \varepsilon,\ \{\hat{A}, \hat{B}\},\ \{S, A, B, C, \#\}\rfloor$ to P'.

H simulates derivations of the form specified in Lemma 1. Rules in P are simulated by rules from (2) through (4). $ABC \to \varepsilon$ is simulated in two steps. First, rules introduced in (5) are used to prepare the application of rules from (6). Then, the latter rules perform the actual erasure of ABC. For example, $AABCBC\#a\# \Rightarrow \bar{A}\hat{A}\hat{B}CBC\#a\# \Rightarrow ABC\#a\#$ in H.

The role of # is twofold. First, it ensures that every sentential form of H is of the form w_1w_2, where $w_1 \in (N' - \{\#\})^*$ and $w_2 \in (T \cup \{\#\})^*$. Since left permitting and left forbidding contexts cannot contain terminals, a mixture of symbols from T and N in H could lead to false sentences. Second, if any of $\bar{A}$, $\bar{B}$, $\hat{A}$, or $\hat{B}$ are present, $ABC \to \varepsilon$ has to be simulated. Therefore, it prevents derivations of the form $Aa \Rightarrow \hat{A}a \Rightarrow a$ in H (notice that the start string of H is $S\#$). Furthermore, we exploit the fact that in every derivation step of H, all symbols have to be rewritten. Consequently, if rules from (5) are used improperly, the derivation is blocked, and so no partial erasures are possible.

Observe that every sentential form of G and H contains at most one occurrence of S. In a derivation step of H, only a single rule from $P \cup \{ABC \to \varepsilon\}$ can be simulated at once. $ABC \to \varepsilon$ can be simulated only if S is not present. #'s can be eliminated in a single step by an application of rules from (1); however, only if there are no nonterminals present in the current sentential form. Based on these observations and on Lemma 1, we see that every successful derivation in H is of the form $S\# \Rightarrow^* w_1w_2\#a_1\#a_2\cdots\#a_n\# \Rightarrow^* \#a_1\#a_2\cdots\#a_n\# \Rightarrow a_1a_2\cdots a_n$, where $w_1 \in \{A, AB\}^*$, $w_2 \in \{BC, C\}^*$, $a_i \in T$, for all i, $1 \le i \le n$, for some $n \ge 0$.

Due to space requirements, we omit some details in what follows. The reader can easily fill them in. To establish $L(H) = L(G)$, we prove two claims. The first claim shows how derivations of G are simulated by H. It is then used to prove $L(G) \subseteq L(H)$. Define the homomorphism φ from V^* to V'^* as $\varphi(X) = X$, for all $X \in N$, and $\varphi(a) = \#a$, for all $a \in T$.

Claim 1. *If $S \Rightarrow^n x \Rightarrow^* z$ in G, for some $n \ge 0$, where $x \in V^*$, $z \in T^*$, then $S\# \Rightarrow^* \varphi(x)\#$ in H.*

Proof. This claim is established by induction on n, where $n \ge 0$. *Basis*: For $n = 0$, the claim clearly holds. *Induction Hypothesis*: Suppose that the claim holds for all derivations of length l or less, where $l \le n$, for some $n \ge 0$. *Induction Step*: Consider any derivation of the form $S \Rightarrow^{n+1} w \Rightarrow^* z$ in G, where $w \in V^*$, $z \in T^*$. Since $n + 1 \ge 1$, this derivation can be expressed as $S \Rightarrow^n x \Rightarrow w \Rightarrow^* z$, for some $x \in V^*$. Without any loss of generality, we may assume that $x = x_1x_2x_3x_4$, where $x_1 \in \{A, AB\}^*$, $x_2 \in \{S, \varepsilon\}$, $x_3 \in \{BC, C\}^*$, and $x_4 \in T^*$ (see Lemma 1 and [4]). Next, we consider all possible forms of $x \Rightarrow w$ in G, covered by the following four cases—(i) through (iv).

(i) (Application of $S \to uSa \in P$.) Let $x = x_1Sx_3x_4$, $w = x_1uSax_3x_4$, and $S \to uSa \in P$, where $u \in \{A, AB\}^*$ and $a \in T$. Then, by the induction hypothesis, $S\# \Rightarrow^* \varphi(x_1Sx_3x_4)\#$ in H. By (2), $\lfloor S \to uS\#a,\ \emptyset,\ \{\bar{A}, \bar{B}, \hat{A}, \hat{B}, \#\}\rfloor \in P'$. Since $\varphi(x_1Sx_3x_4)\# = x_1Sx_3\varphi(x_4)\#$ and $\mathrm{alph}(x_1Sx_3) \cap \{\bar{A},$

$\bar{B}$, $\hat{A}$, $\hat{B}$, $\#\} = \emptyset$, $x_1Sx_3\varphi(x_4)\# \Rightarrow x_1uS\#ax_3\varphi(x_4)\#$ in H. As $\varphi(x_1uSax_3x_4)\# = x_1uS\#ax_3\varphi(x_4)\#$, the induction step is completed for (i).

(ii) (Application of $S \to uSv \in P$.) Let $x = x_1Sx_3x_4$, $w = x_1uSvx_3x_4$, and $S \to uSv \in P$, where $u \in \{A, AB\}^*$ and $v \in \{BC, C\}^*$. Proceed by analogy with (i) by using a rule from (3).

(iii) (Application of $S \to uv \in P$.) Let $x = x_1Sx_3x_4$, $w = x_1uvx_3x_4$, and $S \to uv \in P$, where $u \in \{A, AB\}^*$ and $v \in \{BC, C\}^*$. Proceed by analogy with (i) by using a rule from (4).

(iv) (Application of $ABC \to \varepsilon$.) Let $x = x'_1ABCx'_3x_4$, $w = x'_1x'_3x_4$, where $x_1x_2 = x'_1ABCx'_2$, so $x \Rightarrow w$ in G by $ABC \to \varepsilon$. Then, by the induction hypothesis, $S\# \Rightarrow^* \varphi(x'_1ABCx'_3x_4)\#$ in H. Let $x'_1 = X_1X_2\cdots X_k$, where $k = |x'_1|$. Since $\varphi(x'_1ABCx'_3x_4)\# = x'_1ABCx'_3\varphi(x_4)\#$ and $\mathrm{alph}(x'_1ABCx'_3) \cap \{\bar{A}, \bar{B}, \hat{A}, \hat{B}, \#\} = \emptyset$, $x'_1ABCx'_3\varphi(x_4)\# \Rightarrow \bar{x}'_1\hat{A}\hat{B}Cx'_3\varphi(x_4)\#$ in H by rules from (5), where $\bar{x}'_1 = \bar{X}_1\bar{X}_2\cdots\bar{X}_k$. Since $\mathrm{alph}(\bar{x}'_1) \cap \{S, A, B, C, \hat{A}, \hat{B}, \#\} = \emptyset$, $\hat{A} \in \mathrm{alph}(\bar{x}'_1\hat{A})$, $\hat{A}, \hat{B} \in \mathrm{alph}(\bar{x}'_1\hat{A}\hat{B})$, and $\hat{A}, \hat{B}, C \in \mathrm{alph}(\bar{x}'_1\hat{A}\hat{B}Cx_2)$, $\bar{x}'_1\hat{A}\hat{B}Cx'_3\varphi(x_4)\# \Rightarrow x'_1x'_3\varphi(x_4)\#$ in H by rules from (6). As $\varphi(x'_1x'_3x_4)\# = x'_1x'_3\varphi(x_4)\#$, the induction step is completed for (iv).

Observe that these four cases cover all possible forms of $x \Rightarrow w$ in G, so the claim holds. $\square$

The second claim demonstrates how G simulates derivations of H. It is then used to prove $L(H) \subseteq L(G)$. Define the homomorphism τ from V'^* to V^* as $\tau(X) = X$, for all $X \in N$, $\tau(\bar{A}) = \tau(\hat{A}) = A$, $\tau(\bar{B}) = \tau(\hat{B}) = B$, $\tau(a) = a$, for all $a \in T$, and $\tau(\#) = \varepsilon$.

Claim 2. *If $S\# \Rightarrow^n x \Rightarrow^* z$ in H, for some $n \geq 0$, where $x \in V'^*$, $z \in T^*$, then $S \Rightarrow^* \tau(x)$ in G.*

Proof. This claim is established by induction on n, where $n \geq 0$. *Basis*: For $n = 0$, the claim clearly holds. *Induction Hypothesis*: Suppose that the claim holds for all derivations of length l or less, where $l \leq n$, for some $n \geq 0$. *Induction Step*: Consider any derivation of the form $S\# \Rightarrow^{n+1} w \Rightarrow^* z$ in H, where $w \in V'^*$, $z \in T^*$. Since $n+1 \geq 1$, this derivation can be expressed as $S\# \Rightarrow^n x \Rightarrow w \Rightarrow^* z$, for some $x \in V'^*$. By the induction hypothesis, $S \Rightarrow^* \tau(x)$ in G. Next, we consider all possible forms of $x \Rightarrow w$ in H, covered by the following five cases—(i) through (v).

(i) Let $x = x_1Sx_2$ and $w = x'_1uS\#ax'_2$, where $x_1, x'_1, x_2, x'_2 \in V'^*$, such that $x_1Sx_2 \Rightarrow x'_1uS\#ax'_2$ in H by $\lfloor S \to uS\#a, \emptyset, \{\bar{A}, \bar{B}, \hat{A}, \hat{B}, \#\}\rfloor$—introduced in (2) from $S \to uSa \in P$, where $u \in \{A, AB\}^*$, $a \in T$—and by the rules introduced in the initialization part, in (1), and in (5). Since $\tau(x_1Sx_2) = \tau(x_1)S\tau(x_2)$, $\tau(x_1)S\tau(x_2) \Rightarrow \tau(x_1)uSa\tau(x_2)$ in G. As $\tau(x_1)uSa\tau(x_2) = \tau(x'_1uS\#ax'_2)$, the induction step is completed for (i).

(ii) Let $x = x_1Sx_2$ and $w = x'_1uSvx'_2$, where $x_1, x'_1, x_2, x'_2 \in V'^*$, such that $x_1Sx_2 \Rightarrow x'_1uSvx'_2$ in H by $\lfloor S \to uSv, \emptyset, \{\bar{A}, \bar{B}, \hat{A}, \hat{B}, \#\}\rfloor$—introduced in (3) from $S \to uSv \in P$, where $u \in \{A, AB\}^*$, $v \in \{BC, C\}^*$—and by the rules introduced in the initialization part and in (1) and (5). Proceed by analogy with (i).

(iii) Let $x = x_1Sx_2$ and $w = x'_1uvx'_2$, where $x_1, x'_1, x_2, x'_2 \in V'^*$, such that $x_1Sx_2 \Rightarrow x'_1uvx'_2$ in H by $\lfloor S \to uv, \emptyset, \{\bar{A}, \bar{B}, \hat{A}, \hat{B}, \#\}\rfloor$—introduced in (4) from $S \to uv \in P$, where $u \in \{A, AB\}^*$, $v \in \{BC, C\}^*$—and by the rules introduced in the initialization part and in (1) and (5). Proceed by analogy with (i).

(iv) Let $x = x_1\hat{A}x_2\hat{B}x_3Cx_4$ and $w = x'_1x'_2x'_3x'_4$, where x_1, x'_1, x_2, x'_2, x_3, x'_3, x_4, $x'_4 \in V'^*$, such that $x_1\hat{A}x_2\hat{B}x_3Cx_4 \Rightarrow x'_1x'_2x'_3x'_4$ in H by rules introduced in (6), in the initialization part, and in (1) and (5). Observe that $x_2 = x'_2 = x_3 = x'_3 = \varepsilon$, $\mathrm{alph}(x_1x_3) \cap \{\hat{A}, \hat{B}\} = \emptyset$, and the only occurrence of C that is erased is the one right next to $\bar{B}$; otherwise, this derivation in H is not possible.

Therefore, $x = x_1\hat{A}\hat{B}Cx_4$ and $w = x'_1x'_4$. Since $\tau(x_1\hat{A}\hat{B}Cx_4) = \tau(x_1)ABC\tau(x_4)$, $\tau(x_1)ABC\tau(x_4) \Rightarrow \tau(x_1)\tau(x_4)$ by $ABC \to \varepsilon$ in G. As $\tau(x_1)\tau(x_4) = \tau(x'_1x'_4)$, the induction step is completed for (iv).

(v) Let $x \Rightarrow w$ in H only by rules from the initialization part, from (1), from (5), and by the first two rules from (6). As $\tau(x) = \tau(w)$, the induction step is completed for (v).

Observe that these five cases cover all possible forms of $x \Rightarrow w$ in H, so the claim holds. □

We now prove that $L(H) = L(G)$. Consider a special case of Claim 1 when $x \in T^*$. Then, $S\# \Rightarrow^* \varphi(x)\#$ in H. By (1), $\lfloor \# \to \varepsilon, \emptyset, N' - \{\#\} \rfloor \in P'$. Since $\mathrm{alph}(\varphi(x)\#) \cap (N' - \{\#\}) = \emptyset$, $\varphi(x)\# \Rightarrow x$ in H. Hence, $L(G) \subseteq L(H)$. Consider a special case of Claim 2 when $x \in T^*$. Then, $S \Rightarrow^* x$ in G. Hence, $L(H) \subseteq L(G)$. As $\mathrm{card}(N') = 9$, the theorem holds. □

4 CONCLUSION

The present paper demonstrated that every recursively enumerable language can be generated by a left random context E0L grammar with nine nonterminals. A question to be investigated is whether this bound is, in fact, optimal. Furthermore, in [7], it is proved that left random context E0L grammars without erasing rules—that is, without any rules of the form $X \to \varepsilon$—generate precisely the family of context-sensitive languages. Can we also bound the number of nonterminals in terms of this non-erasing version? We suggest these two open problems for further research.

ACKNOWLEDGEMENT

This work was supported by the research plan MSM0021630528.

REFERENCES

[1] E. Csuhaj-Varjú and G. Vaszil. Scattered context grammars generate any recursively enumerable language with two nonterminals. *Information Processing Letters*, 110:902–907, 2010.

[2] J. Dassow and G. Păun. *Regulated Rewriting in Formal Language Theory*. Springer, 1989.

[3] H. Fernau, R. Freund, M. Oswald, and K. Reinhardt. Refining the nonterminal complexity of graph-controlled, programmed, and matrix grammars. *Journal of Automata, Languages and Combinatorics*, 12(1–2):117–138, 2007.

[4] V. Geffert. Normal forms for phrase-structure grammars. *Theoretical Informatics and Applications*, 25(5):473–496, 1991.

[5] T. Masopust. On the descriptional complexity of scattered context grammars. *Theoretical Computer Science*, 410(1):108–112, 2009.

[6] T. Masopust and A. Meduna. On descriptional complexity of partially parallel grammars. *Fundamenta Informaticae*, 87(3):407–415, 2008.

[7] A. Meduna and P. Zemek. Left random context ET0L grammars. manuscript.

[8] F. Okubo. A note on the descriptional complexity of semi-conditional grammars. *Information Processing Letters*, 110(1):36–40, 2009.

[9] G. Rozenberg and A. Salomaa, editors. *Handbook of Formal Languages, Volumes 1 through 3*. Springer, 1997.