Simple Restriction in Context-Free Rewriting

Tomáš Masopust[1,*]

Faculty of Information Technology, Brno University of Technology
Božetěchova 2, Brno 61266, Czech Republic

Abstract

Many rewriting systems with context-free productions and with controlled derivations have been studied. On one hand, these systems preserve the simplicity of applications of context-free productions and, on the other hand, they increase the generative power to cover more aspects of natural and programming languages. However, with λ-productions, many of these systems are computationally complete. It gives rise to a natural question of what are the simplest restrictions of the derivation process of context-free grammars to obtain the universal power. In this paper, we present such a simple restriction introducing so-called restricted context-free rewriting systems. These systems are context-free grammars with a function assigning a nonterminal coupled with $+$ or $-$ to each nonterminal. A production is applicable if it is applicable as a context-free production and if the symbol assigned to the left-hand side of the production is coupled with $+$, then this symbol has to appear in the sentential form, while if coupled with $-$, it must not appear in the sentential form. This restriction is simpler than most of the other restrictions, since the context conditions are assigned to nonterminals, not to productions, and their type is the simplest possible—a nonterminal.

Key words: Formal languages, context-free grammar, rewriting system, derivation restriction, generative power.
2000 MSC: 68Q42, 68Q45

1. Introduction

Over its history, formal language theory has investigated and studied many variants of regulated grammars based on context-free productions in order to increase the generative power of context-free grammars so that they are able to cover more aspects of natural and programming languages. The main idea of the regulation is to omit some of the context-free derivations so that although the production under consideration is applicable to the current sentential form as a context-free production, it is not applicable according to the regulation. Besides the other (mostly equivalent) regulating mechanisms, the following type of regulation characterizes the basic idea of a restriction discussed in this paper: A production is applicable to the current sentential form if it is applicable as a context-free production and, in addition, some symbols have to appear in the sentential form, while some others must not. Representatives of such regulated grammars are, for instance, random context grammars (see [1] for more details). It is well-known that random context grammars characterize the family of recursively enumerable languages if λ-productions are allowed, and a proper subfamily of the family of context-sensitive languages if λ-productions are not allowed. In addition, it is obvious that the latter language family has the property that every recursively enumerable language is a homomorphic image of a language of this family.

Undoubtably, regulating mechanisms are of some interest because they give a characterization of non-context-free languages by applications of only simple context-free productions. However, besides random

*Corresponding author.
Email address: tomas.masopust@mail.muni.cz (Tomáš Masopust)
[1]Tel. +420541141323, Fax. +420541141270

context grammars, there are many other regulated grammars using different types of regulating mechanisms, such as matrix grammars, graph controlled grammars, programmed grammars etc. (see [1, 2]). Many of these grammars characterize the family of recursively enumerable languages if λ-productions are allowed. This observation gives rise to a very natural question of what are the simplest extensions of context-free grammars by a control of derivation to obtain the universal power.

In this paper, we present such a simple mechanism regulating the applications of context-free productions according to the appearance of some symbols in the current sentential form. More specifically, we introduce and study so-called restricted context-free rewriting system, which is a context-free based rewriting system with an additional function assigning a nonterminal symbol coupled with a symbol $+$ or $-$ to each of its nonterminals. A production of such a system is applicable if it is applicable as a context-free production and, in addition, if the symbol assigned to the left-hand side of the production (to the rewritten nonterminal) is coupled with $+$, then this symbol has to appear in the current sentential form, while if it is coupled with $-$, then it is not allowed to appear in the current sentential form. Observe that this restriction is simpler than most of the other restrictions used in the literature, since the context conditions are assigned to nonterminals, not to productions, and their type is the simplest possible—a nonterminal symbol.

As the main result, we present a characterization of recursively enumerable languages in terms of restricted context-free rewriting systems. This characterization results in some new normal forms for random context grammars and their variants discussed in the literature (see [3–7]), as well as for matrix grammars. As it is not hard to see that any restricted context-free rewriting system can be thought of as a (very simple type of) random context grammars, it immediately follows that nonterminals coupled with both $+$ and $-$ are required because it is well-known that random context grammars with all permitting (forbidding, respectively) sets being empty characterize a proper subfamily of the family of (even) recursive languages (see [8] and also [9, 10]).

Finally, in Section 4 and in the conclusion of this paper, we discuss some questions concerning restricted context-free rewriting systems without λ-free productions and summarize open problems.

2. Preliminaries and Basic Definitions

In this paper, we assume that the reader is familiar with formal language theory (see [1, 11, 12]). For an alphabet (finite nonempty set) V, V^* represents the free monoid generated by V. The unit of V^* is denoted by λ. Set $V^+ = V^* - \{\lambda\}$. For $w \in V^*$, $|w|$ denotes the length of w and $alph(w)$ denotes the set of all symbols occurring in w. Let $\mathscr{L}(\mathrm{RE})$ and $\mathscr{L}(\mathrm{CS})$ denote the families of recursively enumerable and context-sensitive languages, respectively.

A *context-free grammar* is a quadruple $G = (N, T, P, S)$, where N is the alphabet of nonterminals, T is the alphabet of terminals such that $N \cap T = \emptyset$, $V = N \cup T$ is the total alphabet, $S \in N$ is the start symbol, and P is a finite set of productions of the form $A \to x$, where $A \in N$ and $x \in V^*$. If $x \in V^+$, then the production $A \to x$ is said to be λ-free. G is λ-free if all its productions are λ-free. For two strings $u, v \in V^*$, we define the relation $uAv \Rightarrow uxv$ provided that $A \to x \in P$. The language generated by G is defined as $L(G) = \{w \in T^* : S \Rightarrow^* w\}$, where $\Rightarrow^*$ is the reflexive and transitive closure of the relation $\Rightarrow$. The family of languages generated by context-free grammars and λ-free context-free grammars are denoted by $\mathscr{L}(\mathrm{CF})$ and $\mathscr{L}(\mathrm{CF} - \lambda)$, respectively. Note that it is well-known that these two language families coincide, i. e., $\mathscr{L}(\mathrm{CF}) = \mathscr{L}(\mathrm{CF} - \lambda)$.

An *unordered scattered context grammar (with appearance checking)* is a quintuple $G = (N, T, P, S, R)$, where N is the alphabet of nonterminals, T is the alphabet of terminals such that $N \cap T = \emptyset$, $V = N \cup T$ is the total alphabet, $S \in N$ is the start symbol, P is a finite set of productions of the form $(A_1, A_2, \ldots, A_n) \to (w_1, w_2, \ldots, w_n)$, $n \geq 1$, where $A_i \in N$ and $w_i \in V^*$, for all $i = 1, \ldots, n$, and R is a finite set of context-free productions. If $w_i \in V^+$, for all $i = 1, \ldots, n$, then the production is said to be λ-free. G is λ-free if all its productions are λ-free. A production $(A_1, A_2, \ldots, A_n) \to (w_1, w_2, \ldots, w_n) \in P$ is applied to a string $x = x_1 A_{i_1} x_2 A_{i_2} x_3 \ldots x_u A_{i_u} x_{u+1}$, where $x_i \in V^*$, for $i = 1, \ldots, u+1$, provided that

1. $(A_{i_1}, A_{i_2}, \ldots, A_{i_u})$ is a permutation of a subsequence of $(A_1, A_2, \ldots, A_n)$, and
2. if $A_j \in \{A_1, A_2, \ldots, A_n\} - \{A_{i_1}, A_{i_2}, \ldots, A_{i_u}\}$, then A_j does not occur in x and $A_j \to w_j \in R$.

This application results in the string $y = x_1 w_{i_1} x_2 w_{i_2} x_3 \dots x_u w_{i_u} x_{u+1}$, written as $x \Rightarrow y$. The language generated by G is defined as $L(G) = \{w \in T^* : S \Rightarrow^* w\}$, where $\Rightarrow^*$ is the reflexive and transitive closure of the relation $\Rightarrow$. The families of languages generated by unordered scattered context grammars and λ-free unordered scattered context grammars are denoted by $\mathscr{L}(\mathrm{uSC}, ac)$ and $\mathscr{L}(\mathrm{uSC} - \lambda, ac)$, respectively.

An unordered scattered context grammar $G = (N, T, P, S, R)$ is said to be *2-limited* if

1. $(A_1, \dots, A_n) \to (w_1, \dots, w_n) \in P$ implies $n \leq 2$ and $|w_i| \leq 2$, for $i = 1, 2$; and
2. $n = 1$ implies $A_1 = S$.

Mayer [13] proved that $\mathscr{L}(\mathrm{uSC}, ac) = \mathscr{L}(\mathrm{RE})$ and that every recursively enumerable language is generated by a 2-limited unordered scattered context grammar. The proof of the latter result follows by the standard construction introduced in [14] and by the corresponding modification of R. Thus, as a result of the construction given in [14], we can without loss of generality assume that $(A, B) \to (x, y) \in P$ implies that $A \neq B$.

To prove the main result, the following lemma is needed.

Lemma 1. *For every unordered scattered context grammar G', there is a 2-limited unordered scattered context grammar $G = (N, T, P, S, R)$ such that S does not occur on the right-hand side of any production and if $(A, B) \to (x, y) \in P$, then $A \neq B$.*

Proof. By [13], there is a 2-limited unordered scattered context grammar $G = (N, T, P, S)$ such that $L(G') = L(G)$ and if $(A, B) \to (x, y) \in P$, then $A \neq B$. If S occurs on the right-hand side of a production, we construct an equivalent 2-limited unordered scattered context grammar $\bar{G}$ as follows. Let S' and S_1 be two new nonterminals not contained in N and set $N' = N \cup \{S', S_1\}$, $P' = P \cup \{(S') \to (S_1 S)\}$, and replace all productions of the form $(S) \to (w)$ in P' with $(S_1, S) \to (S_1, w)$ and $(S_1, S) \to (\lambda, w)$. Then, $\bar{G} = (N', T, P', S', R)$ is as required and it is not hard to see that $L(G) = L(\bar{G})$. □

3. Restricted Context-Free Rewriting Systems

A *restricted context-free rewriting system* is a quintuple $G = (N, T, P, S, f)$, where (N, T, P, S) is a context-free grammar and $f : N \to \{+, -\} \times N$ is a function.

For two strings $u, v \in V^*$, where $V = N \cup T$, and a production $A \to x \in P$, we define the relation $uAv \Rightarrow uxv$ provided that

1. either $f(A) = (+, X)$ and $X \in \mathit{alph}(uAv)$,
2. or $f(A) = (-, X)$ and $X \notin \mathit{alph}(uAv)$,

where $X \in N$ is a nonterminal. The language generated by G is defined as $L(G) = \{w \in T^* : S \Rightarrow^* w\}$, where $\Rightarrow^*$ is the reflexive and transitive closure of the relation $\Rightarrow$.

The families of languages generated by restricted context-free rewriting systems and λ-free restricted context-free rewriting systems are denoted by $\mathscr{L}(\mathrm{rRS}, \mathrm{CF})$ and $\mathscr{L}(\mathrm{rRS}, \mathrm{CF} - \lambda)$, respectively.

4. Examples

In this paper, we prove that restricted context-free rewriting systems with λ-productions are computationally complete. On the other hand, however, the question of what is the generative power of λ-free restricted context-free rewriting systems is open. Therefore, in this section, we present two examples of λ-free restricted context-free rewriting systems demonstrating their ability to generate non-context-free and non-semi-linear languages.

Example 1. Let $G = (\{S, A, B, C, A', B', C'\}, \{a\}, P, S, f)$ be a restricted context-free rewriting system, where $P = \{S \to ABC,\ A \to aA',\ A \to a,\ B \to bB',\ B \to b,\ C \to cC',\ C \to c,\ A' \to A,\ B' \to B,\ C' \to C\}$ and f is defined as follows.

1. $f(S) = (+, S)$,
2. $f(A) = (-, C')$,
3. $f(B) = (-, A)$,
4. $f(C) = (-, B)$,
5. $f(A') = (+, C')$,
6. $f(B') = (+, A)$,
7. $f(C') = (+, B)$.

Then, it is quite obvious that $L(G) = \{a^n b^n c^n : n \geq 1\}$. □

The next example shows that there are unary non-context-free languages generated by λ-free restricted context-free rewriting systems.

Example 2. Let $G = (\{S, A, B, X, X', Y, Z, a'\}, \{a\}, P, S, f)$ be a restricted context-free rewriting system, where P and f are defined as follows.

1. $S \to BX$,
2. $B \to aA^2$,
3. $B \to a'$,
4. $A \to B$,
5. $X \to Y$,
6. $X \to Z$,
7. $Y \to X'$,
8. $X' \to X$,
9. $a' \to a$,
10. $Z \to a$.

1. $f(S) = (+, S)$,
2. $f(B) = (+, X)$,
3. $f(A) = (+, Y)$,
4. $f(X) = (-, B)$,
5. $f(Y) = (-, A)$,
6. $f(X') = (-, a')$,
7. $f(a') = (+, Z)$,
8. $f(Z) = (-, a')$.

Consider a sentential form $a^{2^k-1}B^{2^k}X$, for some $k \geq 0$. Clearly, BX is of this form for $k = 0$. Then, the only successful derivations are the following (in what follows, $\Rightarrow_{(x)}$ means that the derivation step is made by production (x)):

$$\begin{array}{lllll} a^{2^k-1}B^{2^k}X & \Rightarrow^*_{(2)} & a^{2^k-1}a^{2^k}A^{2^{k+1}}X & \Rightarrow_{(5)} & a^{2^{k+1}-1}A^{2^{k+1}}Y \\ & \Rightarrow^*_{(4)} & a^{2^{k+1}-1}B^{2^{k+1}}Y & \Rightarrow_{(7)} & a^{2^{k+1}-1}B^{2^{k+1}}X' \\ & \Rightarrow_{(8)} & a^{2^{k+1}-1}B^{2^{k+1}}X & & \end{array}$$

and

$$a^{2^k-1}B^{2^k}X \Rightarrow^*_{(3)} a^{2^k-1}{a'}^{2^k}X \Rightarrow_{(6)} a^{2^k-1}{a'}^{2^k}Z \Rightarrow^*_{(9)} a^{2^k-1}a^{2^k}Z \Rightarrow_{(10)} a^{2^{k+1}}.$$

Then, by induction, we have that $L(G) = \{a^{2^n} : n \geq 1\}$. □

5. Main Result

In this section, we prove that every recursively enumerable language can be generated by a restricted context-free rewriting system.

Theorem 2. $\mathscr{L}(\mathrm{rRS}, \mathrm{CF}) = \mathscr{L}(\mathrm{RE})$.

Proof. Let $L \in \mathscr{L}(\mathrm{RE})$. Without loss of generality, L is generated by a 2-limited unordered scattered context grammar $G = (N, T, P, S, R)$ satisfying Lemma 1. Let n be the number of productions in P. Then,

$$P = \bigcup_{i=1}^{k-1}((S) \to (w_i)) \cup \bigcup_{i=k}^{n}((A_{i1}, A_{i2}) \to (w_{i1}, w_{i2}))\,,$$

where $1 \leq k \leq n$, and $A_{i1} \neq A_{i2}$. Construct $G' = (N', T, P', S', f)$ as follows.

- For each $p = (A, B) \to (x, y) \in P$, if $A \to x \in R$, add $(B, A) \to (y, x)$ to P.

Set $N' = N \cup \{A' : A \in N\} \cup \{B_p : p = (A, B) \to (x, y) \in P\} \cup \{A_l : A \in N\} \cup \{A_r : A \in N\} \cup \{\#, \$, X, Y\} \cup \{p, p', p'', p''', p^{iv}, p^v : p \in P\}$ (all these sets are pairwise disjoint), and define P' as follows.

1. For each $(S) \to (w) \in P$, add $S \to w$ with $f(S) = (+, S)$ to P'.
2. For each $A \in N$, add
   (a) $A \to A_l$ and $A \to A_r$ with $f(A) = (+, A)$ to P'.
3. For each $p = (A, B) \to (x, y) \in P$, add the following productions to P':
   (a) $A_l \to X\#A'$ with $f(A_l) = (-, \#)$
   (b) $B_r \to B_p\$X$ with $f(B_r) = (-, \$)$
   (c) $X \to \lambda$ with $f(X) = (-, Y)$
   (d) $A' \to p$ with $f(A') = (-, X)$
   (e) $p \to p'$ with $f(p) = (+, B_p)$
   (f) $B_p \to y$ with $f(B_p) = (+, p')$
   (g) $p' \to Yx$ with $f(p') = (-, B_p)$
   (h) $\# \to \lambda$ with $f(\#) = (+, Y)$
   (i) $\$ \to \lambda$ with $f(\$) = (+, Y)$
   (j) $Y \to \lambda$ with $f(Y) = (-, X)$
4. If $B \to y \in R$, add to P' also
   (a) $A' \to p''$
   (b) $p'' \to p'''$ with $f(p'') = (-, B)$
   (c) $p''' \to p^{iv}$ with $f(p''') = (-, B_l)$
   (d) $p^{iv} \to p^v$ with $f(p^{iv}) = (-, B_r)$
   (e) $p^v \to Yx$ with $f(p^v) = (-, \$)$

where p, p', p'', p''', p^{iv}, and p^v are new nonterminals added to N'.

As the main idea of the proof is quite obvious from the construction, we only explain the meaning of some symbols. The precise formal proof follows. Clearly, # and \$ blocks the applications of productions (3a) and (3b), respectively. Symbols X and Y play the crucial role in the construction; X is introduced at the beginning of the simulation of a production and verifies that the previous simulation has been completed, while Y is introduced after the simulation of a production to express that the simulation is complete. Then, # and \$ can be removed and a new simulation can begin. Notice that having both X and Y in the sentential form blocks the derivation. Therefore, Y has to be removed before the new simulation starts.

To prove that $L(G) \subseteq L(G')$, consider a derivation step, $w_1Aw_2Bw_3 \Rightarrow w_1xw_2yw_3$, of G according to $p = (A, B) \to (x, y)$. In G', the derivation is as follows.

$$\begin{array}{llll}
w_1Aw_2Bw_3 & \Rightarrow_{(2a)} w_1A_lw_2Bw_3 & \Rightarrow_{(2a)} & w_1A_lw_2B_rw_3 \\
& \Rightarrow_{(3a)} w_1X\#A'w_2B_rw_3 & \Rightarrow_{(3b)} & w_1X\#A'w_2B_p\$Xw_3 \\
& \Rightarrow_{(3c)} w_1\#A'w_2B_p\$Xw_3 & \Rightarrow_{(3c)} & w_1\#A'w_2B_p\$w_3 \\
& \Rightarrow_{(3d)} w_1\#pw_2B_p\$w_3 & \Rightarrow_{(3e)} & w_1\#p'w_2B_p\$w_3 \\
& \Rightarrow_{(3f)} w_1\#p'w_2y\$w_3 & \Rightarrow_{(3g)} & w_1\#Yxw_2y\$w_3 \\
& \Rightarrow_{(3h)} w_1Yxw_2y\$w_3 & \Rightarrow_{(3i)} & w_1Yxw_2yw_3 \\
& \Rightarrow_{(3j)} w_1xw_2yw_3\,. & &
\end{array}$$

If there is no B in the sentential form and $B \to y \in R$, the derivation is as follows.

$$\begin{array}{llllll}
w_1Aw_2 & \Rightarrow_{(2a)} w_1A_lw_2 & \Rightarrow_{(3a)} w_1X\#A'w_2 & \Rightarrow_{(3c)} & w_1\#A'w_2 \\
& \Rightarrow_{(4a)} w_1\#p''w_2 & \Rightarrow_{(4b)} w_1\#p'''w_2 & \Rightarrow_{(4c)} & w_1\#p^{iv}w_2 \\
& \Rightarrow_{(4d)} w_1\#p^vw_2 & \Rightarrow_{(4e)} w_1\#Yxw_2 & \Rightarrow_{(3h)} & w_1Yxw_2 \\
& \Rightarrow_{(3j)} w_1xw_2\,. & & &
\end{array}$$

The proof of this inclusion then follows by induction.

To prove that $L(G') \subseteq L(G)$, consider a successful derivation in G'. As it depends only on the presence of some symbols in the sentential form, their positions are disregarded. Assume that the current sentential form is $w = w_1A_lw_2B_rw_3$, where $w \in (N \cup T \cup \{C_l : C \in N\} \cup \{C_r : C \in N\})^*$.

As the derivation is successful, productions constructed in (2a) have been applied in a correct form, it means that for some production $(A, B) \to (x, y)$, there are symbols A_l and B_r in the sentential form. As these symbols cannot be replaced while there is # or \$ in the sentential form, respectively, we do not consider the possibly applicable productions constructed in (2a). Thus, now only productions constructed in $(3a)$ and $(3b)$ are applicable.

3a: (These labels are sequences of productions applied so far.) $(3a)$ is applied, i. e.

$$w_1 A_l w_2 B_r w_3 \quad \Rightarrow_{(3a)} \quad w_1 X \# A' w_2 B_r w_3 \,, \tag{1}$$

then only $(3b)$ and $(3c)$ are applicable.

3a3b: $(3b)$ is applied, then only $(3c)$ is applicable (twice), then only $(3d)$ and $(4a)$ (however, $(4a)$ blocks the derivation). Thus, $(3d)$ is applied, and then only $(3e)$, then only $(3f)$, then only $(3g)$, i. e.

$$\begin{aligned}
w_1 X \# A' w_2 B_r w_3 \quad &\Rightarrow_{(3b)} \quad w_1 X \# A' w_2 B_p \$ X w_3 && (2)\\
&\Rightarrow_{(3c)} \quad w_1 \# A' w_2 B_p \$ X w_3 && (3)\\
&\Rightarrow_{(3c)} \quad w_1 \# A' w_2 B_p \$ w_3 && (4)\\
&\Rightarrow_{(3d)} \quad w_1 \# p w_2 B_p \$ w_3 && (5)\\
&\Rightarrow_{(3e)} \quad w_1 \# p' w_2 B_p \$ w_3 && (6)\\
&\Rightarrow_{(3f)} \quad w_1 \# p' w_2 y \$ w_3 && (7)\\
&\Rightarrow_{(3g)} \quad w_1 \# Y x w_2 y \$ w_3 \,, && (8)
\end{aligned}$$

and only $(3h)$, $(3i)$, and $(3j)$ are applicable. However, $(3j)$ blocks the derivation.

3a3b3c3c3d3e3f3g3h: $(3h)$ is applied, i. e.

$$w_1 \# Y x w_2 y \$ w_3 \quad \Rightarrow_{(3h)} \quad w_1 Y x w_2 y \$ w_3 \,, \tag{9}$$

and only $(3a)$, $(3i)$, and $(3j)$ are applicable. However, $(3a)$ introduces X to the sentential form, which blocks the derivation because neither X nor Y can be removed (see productions $(3c)$ and $(3j)$).

3a3b3c3c3d3e3f3g3h3i: $(3i)$ is applied, i. e.

$$w_1 Y x w_2 y \$ w_3 \quad \Rightarrow_{(3i)} \quad w_1 Y x w_2 y w_3 \,, \tag{10}$$

and only $(3a)$, $(3b)$, and $(3j)$ are applicable. However, both $(3a)$ and $(3b)$ introduce X to the sentential form, which blocks the derivation as explained above. Thus, we obtain

$$w_1 Y x w_2 y w_3 \quad \Rightarrow_{(3j)} \quad w_1 x w_2 y w_3 \,. \tag{11}$$

3a3b3c3c3d3e3f3g3h3j: $(3j)$ is applied, then only $(3a)$, then only $(3c)$, i. e.

$$\begin{aligned}
w_1 Y x w_2 y \$ w_3 \quad &\Rightarrow_{(3j)} \quad w_1 x w_2 y \$ w_3 && (12)\\
&\Rightarrow_{(3a)} \quad w_{11} X \# C' w_{12} x w_2 y \$ w_3 && (13)\\
&\Rightarrow_{(3c)} \quad w_{11} \# C' w_{12} x w_2 y \$ w_3 \,, && (14)
\end{aligned}$$

and only $(3d)$ and $(4a)$ are applicable. However, by $(3d)$,

$$w_{11} \# C' w_{12} x w_2 y \$ w_3 \quad \Rightarrow_{(3d)} \quad w_{11} \# q w_{12} x w_2 y \$ w_3 \,, \tag{15}$$

and, by $(4a)$,

$$\begin{aligned}
w_{11} \# C' w_{12} x w_2 y \$ w_3 \quad &\Rightarrow_{(4a)} \quad w_{11} \# q'' w_{12} x w_2 y \$ w_3 && (16)\\
&\Rightarrow_{(4b)} \quad w_{11} \# q''' w_{12} x w_2 y \$ w_3 && (17)\\
&\Rightarrow_{(4c)} \quad w_{11} \# q^{iv} w_{12} x w_2 y \$ w_3 && (18)\\
&\Rightarrow_{(4d)} \quad w_{11} \# q^{v} w_{12} x w_2 y \$ w_3 \,, && (19)
\end{aligned}$$

and the derivation is blocked. (In the last derivation step, we assume that there are no D, D_l, and D_r in the sentential form, for $q = (C, D) \to (u, v)$. If there is one of them, the derivation is blocked earlier.)

3a3b3c3c3d3e3f3g3i: $(3i)$ is applied, i. e.

$$w_1\#Yxw_2y\$w_3 \quad \Rightarrow_{(3i)} \quad w_1\#Yxw_2yw_3\,, \tag{20}$$

then only $(3b)$, $(3h)$, and $(3j)$ are applicable. However, $(3b)$ introduces X to the sentential form, which blocks the derivation because neither X nor Y can be removed.

3a3b3c3c3d3e3f3g3i3h: $(3h)$ is applied, i. e.

$$w_1\#Yxw_2yw_3 \quad \Rightarrow_{(3h)} \quad w_1Yxw_2yw_3\,, \tag{21}$$

and the derivation continues as in (10).

3a3b3c3c3d3e3f3g3i3j: $(3j)$ is applied, then only $(3b)$, and then only $(3c)$ is applicable, i. e.

$$\begin{aligned} w_1\#Yxw_2yw_3 \quad &\Rightarrow_{(3j)} \quad w_1\#xw_2yw_3 && (22)\\ &\Rightarrow_{(3b)} \quad w_1\#xw_2yw_{31}D_q\$Xw_{32} && (23)\\ &\Rightarrow_{(3c)} \quad w_1\#xw_2yw_{31}D_q\$w_{32}\,, && (24) \end{aligned}$$

and the derivation is blocked.

3a3c: $(3c)$ is applied, i. e.

$$w_1X\#A'w_2B_rw_3 \quad \Rightarrow_{(3c)} \quad w_1\#A'w_2B_rw_3\,, \tag{25}$$

and only $(3b)$, $(3d)$, and $(4a)$ are applicable. If $(3b)$ is applied, $w_1\#A'w_2B_rw_3 \Rightarrow_{(3b)} w_1\#A'w_2B_p\$Xw_3$, the derivation continues as in (3).

3a3c3d: $(3d)$ is applied, then only $(3b)$ is applicable, then only $(3c)$ and $(3e)$ are applicable. If $(3c)$ is applied, the derivation continues as in (5). Thus, $(3e)$ is applied. Then, only $(3f)$ and $(3c)$ are applicable. If $(3c)$ is applied, the derivation continues as in (6). Thus, $(3f)$ is applied. Then, only $(3c)$ and $(3g)$ are applicable. If $(3c)$ is applied, the derivation continues as in (7). Thus, $(3g)$ is applied;

$$\begin{aligned} w_1\#A'w_2B_rw_3 \quad &\Rightarrow_{(3d)} \quad w_1\#pw_2B_rw_3 && (26)\\ &\Rightarrow_{(3b)} \quad w_1\#pw_2B_p\$Xw_3 && (27)\\ &\Rightarrow_{(3e)} \quad w_1\#p'w_2B_p\$Xw_3 && (28)\\ &\Rightarrow_{(3f)} \quad w_1\#p'w_2y\$Xw_3 && (29)\\ &\Rightarrow_{(3g)} \quad w_1\#Yxw_2y\$Xw_3\,, && (30) \end{aligned}$$

and the derivation is blocked; neither X not Y can be removed.

3a3c4a: $(4a)$ is applied, $w_1\#A'w_2B_rw_3 \Rightarrow_{(4a)} w_1\#p''w_2B_rw_3$, then it is not hard to see that the derivation will block; of course, only productions constructed in $(3b)$ are applicable from the group of productions constructed in (3), which introduce \$, and then productions from the group of productions constructed in (4) will block the derivation because there is B, B_l, B_r, or \$ in the sentential form.

3b: $(3b)$ is applied, then only $(3a)$ and $(3c)$ are applicable. If $(3a)$ is applied, the derivation continues as in (2). Thus, $(3c)$ is applied. Then, only $(3a)$ followed by $(3c)$ is applicable;

$$\begin{aligned} w_1A_lw_2B_rw_3 \quad &\Rightarrow_{(3b)} \quad w_1A_lw_2B_p\$Xw_3 && (31)\\ &\Rightarrow_{(3c)} \quad w_1A_lw_2B_p\$w_3 && (32)\\ &\Rightarrow_{(3a)} \quad w_1X\#A'w_2B_p\$w_3 && (33)\\ &\Rightarrow_{(3c)} \quad w_1\#A'w_2B_p\$w_3\,, && (34) \end{aligned}$$

and the derivation continues as in (4).

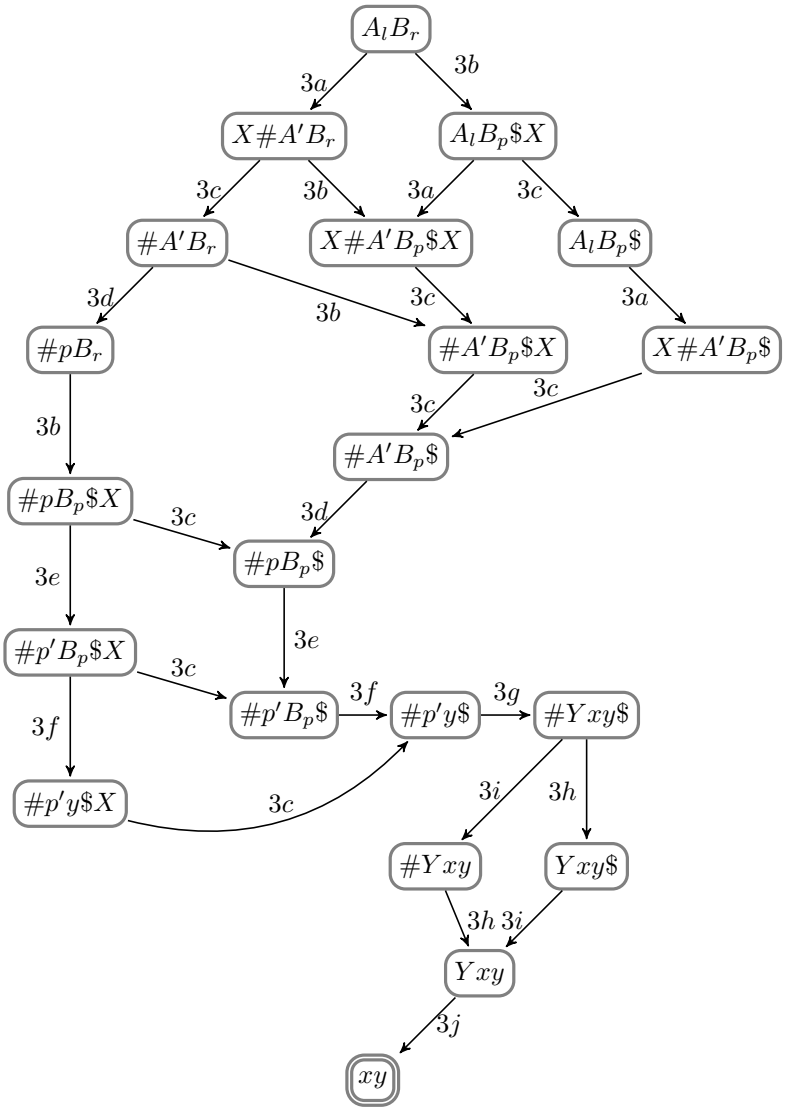

Figure 1: All possible applications of productions in a successful derivation simulating the production $(A, B) \to (x, y)$. Productions that block the derivation are omitted. Nodes contain all symbols of the current sentential form that are not included in $N \cup T \cup \{C_l : C \in N\} \cup \{C_r : C \in N\}$.

This can be depicted graphically as shown in Figure 1. Note that $(A, B) \to (x, y)$ does the same in G.

Next, assume that there are no symbols B, B_r, and B_l in the sentential form and let $w_1 A_l w_2 \in (N \cup T \cup \{C_l : C \in N\} \cup \{C_r : C \in N\})^*$. Then, the successful derivation is of the following form:

$$\begin{array}{llll} w_1 A_l w_2 & \Rightarrow_{(3a)} & w_1 X \# A' w_2 & \Rightarrow_{(3c)} \quad w_1 \# A' w_2 \\ & \Rightarrow_{(4a)} & w_1 \# p'' w_2 & \Rightarrow_{(4b)} \quad w_1 \# p''' w_2 \\ & \Rightarrow_{(4c)} & w_1 \# p^{iv} w_2 & \Rightarrow_{(4d)} \quad w_1 \# p^{v} w_2 \\ & \Rightarrow_{(4e)} & w_1 \# Y x w_2 \,. & \end{array}$$

The derivation then continues as in (20). The proof then follows by induction. □

From the construction in the proof of Theorem 2, we have the following corollary.

Corollary 3. *Every recursively enumerable language is generated by a restricted context-free rewriting system $G = (N, T, P, S, f)$, where $A \to x \in P$ implies $|x| \leq 3$.*

6. Consequences

In this section, we present several consequences of the main result.

Recall that a *random context grammar (with appearance checking)* is a quadruple $G = (N, T, P, S)$, where N, T, and S are as in a context-free grammar, and P is a finite set of productions of the form $(A \to x, Q, R)$, where $A \to x$ is a context-free production and $Q, R \subseteq N$ are permitting and forbidding sets, respectively. For two strings $u, v \in V^*$, where $V = N \cup T$, and a production $(A \to x, Q, R) \in P$, we define the relation $uAv \Rightarrow uxv$ provided that all symbols of Q appear in uAv, and no symbol of R appears in uAv. The language generated by G is defined as $L(G) = \{w \in T^* : S \Rightarrow^* w\}$, where $\Rightarrow^*$ is the reflexive and transitive closure of the relation $\Rightarrow$. The families of languages generated by random context grammars and λ-free random context grammars are denoted as $\mathscr{L}(\mathrm{RC}, \mathrm{CF}, ac)$ and $\mathscr{L}(\mathrm{RC}, \mathrm{CF} - \lambda, ac)$, respectively.

It is well-known that $\mathscr{L}(\mathrm{RC}, \mathrm{CF}, ac) = \mathscr{L}(\mathrm{RE})$ and $\mathscr{L}(\mathrm{RC}, \mathrm{CF} - \lambda, ac) \subset \mathscr{L}(\mathrm{CS})$.

In addition, using the simulation of matrix grammars by random context grammars (see [1, Theorem 1.2.3]), it follows from the results proved in [1] that for every recursively enumerable language L, there exists a random context grammar $G = (N, T, P, S)$ with $L(G) = L$ such that all productions are of the form $(A \to x, Q, R)$ with $A \in N$, $x \in V^*$, $|x| \leq 2$, and R, Q are two disjoint subsets of N.

The following corollary of the main result gives a new normal form for random context grammars. Note that Conditions (2) and (3) improve the previous normal form. On the other hand, however, Condition (1) requires $|x| \leq 3$, and it is an open problem whether we can also have $|x| \leq 2$.

Corollary 4. *For every recursively enumerable language L, there exists a random context grammar $G = (N, T, P, S)$ such that $L = L(G)$ and each production $(A \to x, Q, R) \in P$ satisfies the following three conditions:*

1. $|x| \leq 3$,
2. $|Q \cup R| \leq 1$, and
3. if $(A \to x, Q_1, R_1), (A \to y, Q_2, R_2) \in P$, then $Q_1 = Q_2$ and $R_1 = R_2$.

Proof. For each production $A \to x$ of the restricted context-free rewriting system, we introduce the production $(A \to x, Q, R)$ so that $Q = \{X : (+, X) \in f(A)\}$ and $R = \{X : (-, X) \in f(A)\}$. The statement of the corollary then follows immediately from the definition and Theorem 2. □

In addition, this corollary also demonstrates that many variants of random context grammars, such as semi-conditional grammars (see [7]), simple semi-conditional grammars (see [6]) or conditional context-free rewriting systems (see [5]) are computationally complete if λ-productions are allowed. This normal form holds for them as well.

A *matrix grammar (with appearance checking)* is a quintuple $G = (N, T, M, S, F)$, where N, T, and S are as in a context-free grammar, M is a finite set of finite sequences of the form $[r_1, r_2, \ldots, r_n]$, $n \geq 1$, where r_i is a context-free production, for all $i = 1, 2, \ldots, n$, and F is a finite set of context-free productions. For two strings $u, v \in V^*$, where $V = N \cup T$, and a matrix $[r_1, r_2, \ldots, r_n] \in M$, we define the relation $u \Rightarrow v$ provided that there are sentential forms $x_0, x_1, \ldots, x_n \in V^*$ such that $u = x_0$, $v = x_n$, and either $x_{i-1} \Rightarrow x_i$ by r_i, or r_i is not applicable to x_{i-1}, $r_i \in F$ and $x_i = x_{i-1}$. The language generated by G is defined as $L(G) = \{w \in T^* : S \Rightarrow^* w\}$, where $\Rightarrow^*$ is the reflexive and transitive closure of the relation $\Rightarrow$. The families of languages generated by matrix grammars and λ-free matrix grammars are denoted as $\mathscr{L}(\mathrm{M}, \mathrm{CF}, ac)$ and $\mathscr{L}(\mathrm{M}, \mathrm{CF} - \lambda, ac)$, respectively.

It is also well-known that $\mathscr{L}(\mathrm{M}, \mathrm{CF}, ac) = \mathscr{L}(\mathrm{RE})$ and that $\mathscr{L}(\mathrm{M}, \mathrm{CF} - \lambda, ac) = \mathscr{L}(\mathrm{RC}, \mathrm{CF} - \lambda, ac)$.

The following normal form for matrix grammars is shown in [1, Lemmas 1.2.3 and 1.3.1]. For every recursively enumerable language L, there exists a matrix grammar $G = (N \cup \{Z\}, T, M, S, F)$, for some $Z \notin N \cup T$, such that $L(G) = L$, all matrices are of the form $[A \to x]$, $[A \to x, X \to Y]$ or $[A \to x, X \to \lambda]$ with $A, X, Y \in N$, $x \in V^*$, $|x| \leq 2$, and F consists only of productions of the form $A \to Z$, for $A \in N$.

Using the standard simulation of random context grammars by matrix grammars (see [1, Theorem 1.2.3]), the following normal form for matrix grammars is an immediate consequence of the previous corollary.

Corollary 5. *For every recursively enumerable language L, there exists a matrix grammar $G = (N \cup \{Z\}, T, M, S, F)$, for some $Z \notin N \cup T$, such that $L = L(G)$, Z is replaced only with itself and each matrix is of one of the following two forms:*

1. $[A \to A, B \to x]$, where $A, B \in N$ and $|x| \leq 3$, or
2. $[A \to Z, B \to x]$, where $A, B \in N$, $|x| \leq 3$, and $A \to Z \in F$.

In addition, if $[A \to X, B \to x] \in M$ and $[A \to Y, B \to y] \in M$ are two matrices of G, then $X, Y \in \{A, Z\}$ and $X = Y$.

Again, we do not know whether the corollary also holds in case $|x| \leq 2$.

7. Conclusion

In this paper, we studied restricted context-free rewriting systems with λ-productions. In formal language theory, however, the λ-free case is of a great interest as well. Nevertheless, the generative power of λ-free restricted context-free rewriting systems is an open problem. Note that the proof of Theorem 2 cannot be trivially modified because it uses several arbitrary symbols that have to be introduced and removed many times during the derivation process. On the other hand, it is well-known (see [1, 13]) that $\mathscr{L}(uSC - \lambda, ac) = \mathscr{L}(X, CF - \lambda, ac)$, where $X \in \{RC, M\}$. Obviously and by the fact that context-free languages are closed under homomorphism,

$$\mathscr{L}(\mathrm{CF}) \subset \mathscr{L}(rRS, CF - \lambda) \subseteq \mathscr{L}(uSC, CF - \lambda, ac)\,.$$

Therefore, proving that λ-free restricted context-free rewriting systems are equivalent to λ-free unordered scattered context grammars could introduce analogous normal forms for λ-free variants of random context and matrix grammars. On the other hand, proving that the inclusion is proper would give a better characterization of the family of languages having the property that every recursively enumerable language is a homomorphic image of a language from that family. As far as the author knows, there is no other language family characterized by context-free grammars with a regulating mechanism, properly included in the family generated by λ-free random context (matrix) grammars, having this property.

Some examples demonstrating the generative power of λ-free restricted context-free rewriting systems are presented above, cf. Examples 1 and 2 in Section 4. Can those languages be generated by λ-free restricted context-free rewriting systems with all symbols coupled only with + (only with −)? And can the set of all prime numbers be generated by such a system (with nonterminals coupled with both + and −)? Note also that it is known that the language families generated by (λ-free) restricted context-free rewriting systems with nonterminals coupled only with + (only with −, respectively) are weaker than the general case. Specifically, they are included in the language families generated by random context grammars

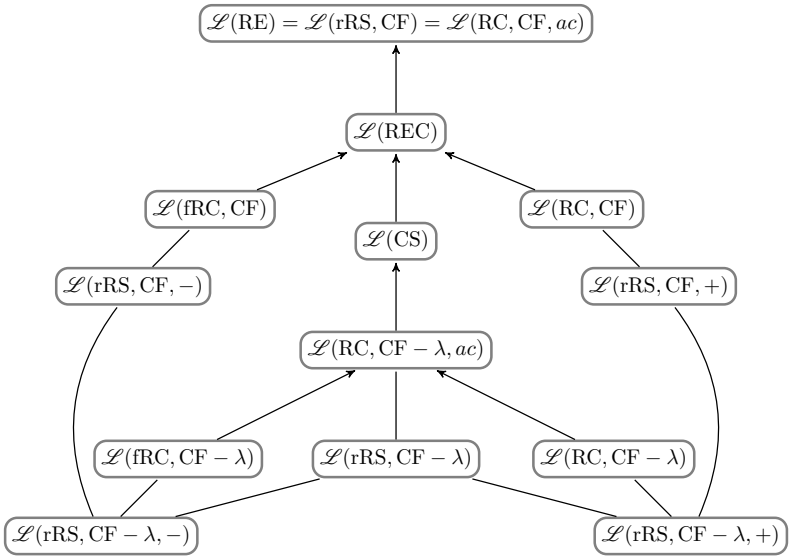

Figure 2: A hierarchy of language families. If two families are connected by a line (an arrow), then the upper family includes (includes properly) the lower family. If two families are not connected, then they are not necessary incomparable. $\mathscr{L}(\mathrm{rRS}, \mathrm{X}, \mathrm{y})$, $X \in \{\mathrm{CF}, \mathrm{CF} - \lambda\}$, $y \in \{+, -\}$, denotes the language family generated by (λ-free) restricted context-free rewriting systems, where all nonterminals are coupled with y. $\mathscr{L}(\mathrm{RC}, \mathrm{X})$ $\big(\mathscr{L}(\mathrm{fRC}, \mathrm{X})\big)$, $X \in \{\mathrm{CF}, \mathrm{CF} - \lambda\}$, denotes the language family generated by (λ-free) random context grammars $\big($by (λ-free) forbidding grammars, respectively$\big)$.

without appearance checking (also called permitting grammars) and forbidding random context grammars, respectively, which are known to be properly included in the family of recursive languages or in the family of random context (matrix) languages if λ-productions are or are not allowed, respectively (see [8–10] and Figure 2 for an overview of the language hierarchy).

Finally, note that it is an interesting mathematically challenging question to ask what is the generative power of restricted context-free rewriting systems with the function being injective.

Acknowledgements

The author gratefully acknowledges very useful suggestions and comments of the anonymous referee improving this paper. This work was supported by the Czech Ministry of Education under the Research Plan No. MSM 0021630528.

References

[1] J. Dassow, G. Păun, Regulated Rewriting in Formal Language Theory, Springer-Verlag, Berlin, 1989.
[2] H. Fernau, R. Freund, M. Oswald, K. Reinhardt, Refining the nonterminal complexity of graph-controlled, programmed, and matrix grammars, Journal of Automata, Languages and Combinatorics 12 (1–2) (2007) 117–138.
[3] T. Masopust, Formal models: Regulation and reduction, Ph.D. thesis, Brno University of Technology, Brno (2007).
[4] T. Masopust, A note on the generative power of some simple variants of context-free grammars regulated by context conditions, in: A. H. Dediu, A. M. Ionescu, C. Martín-Vide (Eds.), LATA 2009 proceedings, Vol. 5457 of Lecture Notes in Computer Science, Springer-Verlag, 2009, pp. 554–565.
[5] T. Masopust, A. Meduna, On context-free rewriting with a simple restriction and its computational completeness, RAIRO – Theoretical Informatics and Applications 43 (2) (2009) 365–378.
[6] A. Meduna, A. Gopalaratnam, On semi-conditional grammars with productions having either forbidding or permitting conditions, Acta Cybernetica 11 (4) (1994) 307–324.
[7] G. Păun, A variant of random context grammars: Semi-conditional grammars, Theoretical Computer Science 41 (1985) 1–17.

[8] H. Bordihn, H. Fernau, Accepting grammars and systems, Tech. Rep. 22/94, Universität Karlsruhe, Fakultät für Informatik (1994).
URL http://citeseer.ist.psu.edu/article/bordihn95accepting.html
[9] S. Ewert, A. P. J. van der Walt, A pumping lemma for random permitting context languages, Theoretical Computer Science 270 (1–2) (2002) 959–967.
[10] A. P. J. van der Walt, S. Ewert, A shrinking lemma for random forbidding context languages, Theoretical Computer Science 237 (1-2) (2000) 149–158.
[11] A. Salomaa, Formal languages, Academic Press, New York, 1973.
[12] J. E. Hopcroft, J. O. Ullman, Introduction to Automata Theory, Languages and Computation, Addison-Wesley, 1979.
[13] O. Mayer, Some restrictive devices for context-free grammars, Information and Control 20 (1972) 69–92.
[14] S. Greibach, J. Hopcroft, Scattered context grammars, Journal of Computer and System Sciences 3 (1969) 233–247.