One-Side Pumping and Two-Side Pumping by Complete CF(¢,$)-grammars

František Mráz[1,*], Martin Plátek[1], Dana Pardubská[2] and Daniel Průša[3]

[1]Charles University, Department of Computer Science, Malostranské nám. 25, 118 00 Praha 1, Czech Republic
[2]Comenius University in Bratislava, Department of Computer Science, Mlynská Dolina, 84248 Bratislava, Slovakia
[3]Czech Technical University, Department of Cybernetics, Karlovo nám. 13, 121 35 Praha 2, Czech Republic

Abstract

We introduce context-free grammar with sentinels, CF(¢,$)-grammar, as a generalization of recently introduced LR(¢,$)-grammar. Original LR(¢,$)-grammars can be used to construct deterministic pumping restarting automata performing correctness and error preserving pumping analysis by reduction on each word over its input alphabet. Pumping analysis by reduction involves step-wise simplification of an input word by removing at most two continuous parts of the current word while preserving the correctness or incorrectness of the word. Each such simplification step corresponds to removing portions of the current word that can be "pumped" according to the pumping lemma for context-free languages, and thus, it does not use any nonterminals.

One-side pumping grammars are CF(¢,$)-grammars that allow removing just one continuous part in each step of pumping reduction. A complete CF(¢,$)-grammar generates both a language and its complement with sentinels. We show that complete one-side pumping CF(¢,$)-grammars characterize the class of regular languages, while LR(¢,$)-grammars that allow two-side pumping reductions characterize the class of deterministic context-free languages. LR(¢,$)-grammars that do not allow any one-side pumping reduction generate non-regular languages only.

Keywords

restarting automata, LR(0) grammars, complete grammars, deterministic context-free languages

1. Introduction

This paper builds upon and improves the findings from papers [1, 2, 3] that introduced and investigated certain restrictions for deterministic monotone restarting pumping automata (det-mon-RP-automata). The motivation for introducing general restarting automata was to model analysis by reduction.

Analysis by reduction is a method from linguistics. Here it serves for checking the correctness of an input word by step-wise rewriting some part of the current tape with a shorter one until we obtain a simple word for which we can decide its correctness easily. In general, analysis by reduction is nondeterministic, and in one step, we can rewrite a sub-string of a length limited by a constant with a shorter string. An input word is accepted if there is a sequence of reductions such that the final simple word is from the target language. Then, intermediate words obtained during the analysis are also accepted. Each reduction must be *error preserving*, i.e., no word outside the target language can be rewritten into a word from the language.

Our interest in studying pumping restarting automata was awakened by papers [4, 5] investigating the complexity of parsing deterministic context-free languages. This paper's linguistic and non-linguistic motivations can already be found in [1, 2, 6]. We aim to develop formal tools supporting the characterization and localization of syntactical errors in artificial and natural languages. This paper should contribute to a complete taxonomy of different types of syntactical errors encountered when parsing deterministic context-free languages.

In this paper, we study some types of deterministic analysis by reduction. We are mainly interested in a strongly constrained version of analysis by reduction called *pumping analysis by reduction*. Pumping analysis by reduction is a reduction analysis with the following additional restriction. In each step of pumping analysis by reduction, the current word is not completely rewritten. Instead, at most two continuous segments of the current word are deleted.

When a restarting automaton works on a word, the word is always delimited by sentinels – ¢ on the left and $ on the right end of its tape. Therefore, we consider here context-free grammars with sentinels, shortly CF(¢,$)-grammars, that generate only words of the form $\{¢\} \cdot w \cdot \{\$\}$, where w is a word over an alphabet Σ containing neither ¢ nor $.

Here, we use so-called complete CF(¢,$)-grammars

ITAT'23: Information Technologies – Applications and Theory, September 22–26, 2023, Tatranské Matliare, Slovakia
*Corresponding author.
frantisek.mraz@mff.cuni.cz (F. Mráz); martin.platek@mff.cuni.cz (M. Plátek); pardubska@dcs.fmph.uniba.sk (D. Pardubská); prusapa1@fel.cvut.cz (D. Průša)
0000-0001-3869-3340 (F. Mráz); 0000-0003-3147-6442 (M. Plátek); 0000-0001-9383-8117 (D. Pardubská); 0000-0003-4866-5709 (D. Průša)

that generate all words from $\{¢\} \cdot \Sigma^* \cdot \{\$\}$. Informally, a complete CF(¢,$)-grammar G_C has two initial nonterminals, S_A and S_R. The set of words derived from the nonterminal S_A is a language of the form $\{¢\} \cdot L \cdot \{\$\}$, for some language $L \subseteq \Sigma^*$. The language L is called the inner language of G_C. The set of words derived from the nonterminal S_R is complementary to $\{¢\} \cdot L \cdot \{\$\}$ with respect to $\{¢\} \cdot \Sigma^* \cdot \{\$\}$, that is $\{¢\} \cdot (\Sigma^* \setminus L) \cdot \{\$\}$.

Pumping lemma for context-free languages [7] implies that, for each complete CF(¢,$)-grammar G, there exists a constant p such that each word w generated by G derived from S_A (or S_R, respectively) of length greater than p can be written as a concatenation of some words x_1, x_2, x_3, x_4, and x_5, where x_2x_4 is nonempty, and all words $x_1x_2^ix_3x_4^ix_5$, for all integers $i \geq 0$, can be derived from S_A (or S_R, respectively). Hence, during reducing pumping analysis, $x_1x_2x_3x_4x_5$ can be reduced into $x_1x_3x_5$. If x_2 or x_4 is empty, we say the pumping reduction is a one-side reduction.

It is well-known that each deterministic context-free language can be generated by an LR(0) grammar ([7]). In [3], it was shown that for each deterministic context-free language L, there exists a complete CF(¢,$)-grammar, which is an LR(0) grammar with the inner language L. Moreover, the language L is accepted by a deterministic restarting RP-automaton M that performs pumping analysis by reduction on all input words (both from L and its complement). The last phase of the computation of M on an input word w produces a terminal word w' that is not longer than the above constant p. If $¢w'\$$ is derived from S_A according to G_C, then w' (and thus also w) is accepted by M. Otherwise if $¢w'\$$ is derived from S_R according to G_C, then w' (and thus also w) is rejected by M.

CF(¢,$)-grammars that allow only left-side pumping reductions (x_4 is empty in all possible pumping reductions) or only right-side pumping reductions (x_2 is empty in all pumping reductions) generate only regular languages. On the other hand, we show that two-side pumping CF(¢,$)-grammars that are LR(0) and do not allow one-side pumping at all generate non-regular languages only.

The paper has the following structure. Section 2 introduces the main notions of CF(¢,$)-grammars, LR(0) grammars, and pumping notions. In Section 3, we show that the class of inner languages of one-side CF(¢,$)-grammars coincides with the class of regular languages. Conversely, Section 4 shows that the inner language of a complete CF(¢,$)-grammar, which is LR(0) and does not allow any one-side pumping reduction, is not regular. Section 5 presents further results dealing with one- and two-side pumping grammars. Section 6 concludes the paper and sketches directions for further research.

2. Basic Notions and Results

In what follows, we will work with the class of context-free languages and some of its subclasses, like deterministic context-free languages (see [8]). Let Σ be a finite nonempty alphabet. A language $L \subseteq \Sigma^*$ is context-free if it is generated by a context-free grammar $G = (N, \Sigma, S, R)$, where N is a finite set of nonterminals, Σ is a finite set of terminals, $N \cap \Sigma = \emptyset$, $S \in N$ is the initial symbol, and R is a finite set of rules of the form $X \to \alpha$, for $X \in N$ and $\alpha \in (N \cup \Sigma)^*$.

We say that α directly derives β (denoted as $\alpha \Rightarrow \beta$) by G if $\alpha = \nu A \xi$, $\beta = \nu\gamma\xi$ for some $\alpha, \beta, \gamma, \nu, \xi \in (N \cup \Sigma)^*$, $A \in N$ and $A \to \gamma \in R$. The reflexive and transitive closure of the relation $\Rightarrow$ is denoted as $\Rightarrow^*$. If, additionally, ξ is a terminal word, we say that the derivation step $\alpha = \nu A \xi \Rightarrow \nu\gamma\xi = \beta$ is a rightmost derivation step and denote it as $\alpha \Rightarrow^{\mathrm{r}} \beta$. Obviously, $\Rightarrow^{\mathrm{r}*}$ denotes the reflexive and transitive closure of the relation $\Rightarrow^{\mathrm{r}}$.

For each context-free grammar $G = (N, \Sigma, S, R)$, there exists a context-free grammar $G' = (N', \Sigma, S', R')$ that generates the same language as G such that for each nonterminal X from N', there is at least one word $w \in \Sigma^*$ for which it holds $X \Rightarrow^* w$ and there exist words $\alpha, \beta \in (N' \cup \Sigma)^*$ for which it holds $S' \Rightarrow^* \alpha X \beta$. We say that grammar G' is *reduced* [8].

2.1. CF(¢,$)-grammars

In [3], we introduced LR(¢,$)-grammars and complete LR(¢,$)-grammars to provide a formal tool for identifying syntax errors within the analysis-by-reduction process performed by RP-automata. Additionally, we sought to characterize and differentiate deterministic context-free languages (DCFL) from regular languages based on the decidable syntactic properties of the formal model. The concept of pumping reduction emerged as a crucial element in achieving these goals. In this section, we present the definitions and results from [3] concerning LR(¢,$)-grammars but now in a new, more general setting of CF(¢,$)-grammars.

Definition 1. *Let N and Σ be two disjoint alphabets, $¢, \$ \notin (N \cup \Sigma)$ and $G = (N, \Sigma \cup \{¢, \$\}, S, R)$ be a context-free grammar generating a language of the form $\{¢\} \cdot L \cdot \{\$\}$, where $L \subseteq \Sigma^*$, and S does not occur in the right-hand side of any rule from R. We say that G is a CF(¢,$)-grammar. The language L is the* internal language of G, *and it is denoted as $L_{\mathrm{in}}(G)$. W.l.o.g., we suppose that a CF(¢,$)-grammar does not contain rewriting rules of the form $A \to \lambda$ for any nonterminal $A \in N$, where λ denotes the empty word.*

Closure properties of the class of context-free languages imply that for a CF(¢,$)-grammar G, both lan-

guages $L(G)$ and $L_{\text{in}}(G)$ are context-free. The added right sentinel \$ facilitates recognition of languages. E.g., if L is a deterministic context-free language, then it can be generated by an LR(1)-grammar (see [7]). But $L \cdot \{\$\}$ and $\{¢\} \cdot L \cdot \{\$\}$ are both generated by simpler LR(0) grammars. The left sentinel ¢ is included in CF(¢,\$)-grammars for compatibility with RP-automata.

2.2. Pumping Notions of CF(¢,\$)-grammars

This section studies the pumping properties of CF(¢,\$)-grammars. We start with several definitions and notations.

Let $G = (N, \Sigma \cup \{¢, \$\}, S, R)$ be a CF(¢,\$)-grammar, x, u_1, v, u_2, y be words over Σ, $|u_1| + |u_2| > 0$, $|v| > 0$, and $A \in N$ be a nonterminal. If

$$S \Rightarrow^* ¢xAy\$ \Rightarrow^* ¢xu_1Au_2y\$ \Rightarrow^* ¢xu_1vu_2y\$ \quad (1)$$

we say that (x, u_1, A, v, u_2, y) is a *pumping infix* by G, $xu_1vu_2y \rightsquigarrow_{P(G)} xvy$ is a *pumping reduction* by G and the word $¢xu_1vu_2y\$$ is a *pumped word* according to G.

Note that we omitted the sentinels in the pumping infix and pumping reduction, and $xu_1vu_2y \in L_{\text{in}}(G)$.

The relation $\rightsquigarrow^*_{P(G)}$ is the reflexive and transitive closure of the relation $\rightsquigarrow_{P(G)}$.

On the other hand, if (x, u_1, A, v, u_2, y) is a pumping infix by G, then all words of the form $¢xu_1^ivu_2^iy\$$, for all integers $i \geq 0$, belong to $L(G)$.

We call *pumped* each derivation tree corresponding to a derivation of the form (1). Likewise, we say that the word $¢xu_1vu_2y\$$ is pumped, and the derivation (1) is pumped.

Let $G = (N, \Sigma \cup \{¢, \$\}, S, R)$ be a CF(¢,\$)-grammar, t be the number of nonterminals of G, and k be the maximal length of the right-hand side of the rules from R. Let T be a derivation tree according to G. If T has more than k^t leaves, a path exists from a leaf to the root of T such that it contains at least $t + 1$ nodes labeled by nonterminals. As G has only t nonterminals, at least two nodes on the path are labeled with the same nonterminal A. In that case, there is a derivation of the form (1), and T is a pumped derivation tree. We say that $K_G = k^t$ is the *grammar number of* G.

Note that for any word from $L(G)$ of length greater than K_G, some pumping infix by G must correspond. On the other hand, each word generated by G that is not pumped is of length at most K_G.

Lemma 1. *Let $G = (N, \Sigma \cup \{¢, \$\}, S, R)$ be a CF(¢,\$)-grammar. If G generates w, then there exists a sequence of words $w_1, \ldots, w_n$ from $L(G)$, for some integer $n \geq 1$, such that $w = w_1$, there are pumping reductions $w_i \rightsquigarrow_{P(G)} w_{i+1}$, for all $i = 1, \ldots, n-1$, and there is no $w_{n+1} \in \Sigma^*$ such that $w_n \rightsquigarrow_{P(G)} w_{n+1}$.*

Proof: (Sketch) Let $w \in L(G)$ and T be a derivation tree with the symbols of w in its leaves.

In T, if there is no path from its root to a leaf on which two nodes are labeled with the same nonterminal, then the lemma statement holds for $n = 1$, as there is no pumping reduction possible.

Suppose there is a path from the root of T to a leaf such that two nodes on the path are labeled with the same nonterminal A. In that case, we can build a pumping infix (x, u_1, A, v, u_2, y) (for some $x, u_1, v, u_2, y \in \Sigma^*$ and $A \in N$) by G such that $w = xu_1vu_2y$, then $w = xu_1vvu_2y \rightsquigarrow_{P(G)} xvy$ and the derivation tree T can be modified into the derivation tree T_1 for the word xvy by replacing the subtree corresponding to the derivation of $A \Rightarrow^* u_1vu_2$ with the subtree for the derivation $A \Rightarrow^* v$. Further, we can again try to find a path with repeating nonterminal in tree T_1 and construct another pumping reduction. In this way, we can continue until we obtain a word with a derivation tree, in which there is no path with repeating nonterminal. In this way, we obtain the desired sequence of pumping reductions. □

Definition 2. *Let (x, u_1, A, v, u_2, y) be a pumping infix by a CF(¢,\$)-grammar G. We say that the pumping infix is a* core pumping infix *if there is a derivation tree T by G that corresponds to the derivation*

$$S \Rightarrow^* ¢xAy\$ \Rightarrow^* ¢xu_1Au_2y\$ \Rightarrow^* ¢xu_1vu_2y\$ \quad (2)$$

such that the path between the root r_1 of the subtree corresponding to the derivation of u_1Au_2 from A in (2) to the root r_2 of the subtree corresponding to the derivation of v (but without r_2) does not contain two distinct nodes labeled with the same nonterminal.

We write $xu_1vu_2y \rightsquigarrow_{P(G,\text{core})} xvy$, and say that the reduction $xu_1vu_2y \rightsquigarrow_{P(G,\text{core})} xvy$ is a core pumping reduction by G. The transitive and reflexive closure of $\rightsquigarrow_{P(G,\text{core})}$ is denoted in the standard way as $\rightsquigarrow^*_{P(G,\text{core})}$.

Note that in the above derivation (2), the length of the words x, u_1, v, u_2, y is not limited. A general pumping reduction $w \rightsquigarrow_{P(G)} w'$ corresponds to removing a segment between any nodes r_1 and r_2 labeled with the same nonterminal A occuring on a path from the root of a derivation tree for w. The pumping reduction is core if there is no other node labeled with A between r_1 and r_2, and all nodes between r_1 and r_2 are labeled with distinct nonterminals. The statement of Lemma 1 can be easily extended to pumping analysis using core reductions only.

Corollary 1. *Let $G = (N, \Sigma \cup \{¢, \$\}, S, R)$ be a CF(¢,\$)-grammar. If G generates w, then there exists a sequence of words $w_1, \ldots, w_n$ from $L(G)$, for some integer $n \geq 1$, such that $w = w_1$, there are core pumping reductions $w_i \rightsquigarrow_{P(G,\text{core})} w_{i+1}$, for all $i = 1, \ldots, n-1$, and there is no $w_{n+1} \in \Sigma^*$ such that $w_n \rightsquigarrow_{P(G)} w_{n+1}$.*

Definition 3. *Let G be a CF(¢,$)-grammar. We say that a pumping infix (x, u_1, A, v, u_2, y) by G is:*

- *a* one-side pumping infix *if $u_1 = \lambda$, or $u_2 = \lambda$,*
- *a* two-side pumping infix *if $u_1 \neq \lambda$, and $u_2 \neq \lambda$,*
- *a* left-side pumping infix *if $u_2 = \lambda$, and*
- *a* right-side pumping infix *if $u_1 = \lambda$.*

Correspondingly, we say that a pumping reduction is a one-side (two-side, left-side, or right-side) pumping reduction if the corresponding pumping infix is a one-side (two-side, left-side, or right-side) pumping infix.

One-side/Two-side pumping grammars. Let G be a CF(¢,$)-grammar. We say that G is a *left-side pumping CF(¢,$)-grammar* if all its core pumping infixes are left-side pumping infixes. Similarly, we say that G is a *right-side pumping CF(¢,$)-grammar* if all its core pumping infixes are right-side pumping infixes. We say that G is a *one-side pumping CF(¢,$)-grammar* if it is left-side or right-side pumping CF(¢,$)-grammar. If G is neither left-side nor right-side pumping grammar, we say that G is a *two-side pumping CF(¢,$)-grammar.*

Example 1. *Let $G = (\{S, S', A\}, \{a, b, c, ¢, \$\}, S, R)$ be a CF(¢,$)-grammar, where R is the set of rules:*

$$S \to ¢S'\$, \qquad S' \to c \mid A, \quad A \to aS' \mid S'b.$$

The grammar G is a two-side pumping grammar. All its core pumping infixes are of one of the following forms: $(x, a, S', v', \lambda, y)$, $(x, \lambda, S', v', b, y)$, (x, a, A, v, λ, y) or (x, λ, A, v, b, y), for any words $x \in \{a\}^*$, $y \in \{b\}^*$, $v, v' \in \{a\}^* \cdot c \cdot \{b\}^*$, $|v| \geq 2$. All the core pumping infixes are one-side, but the grammar is two-side pumping as it has both left-side and right-side pumping infixes.

There also exist two-side pumping infixes by G, like $\pi_1 = (\lambda, a, S', c, b, \lambda)$ or $\pi_2 = (aa, a, A, aac, b, bbb)$. The former is not core pumping infix since there is a path between two nodes labeled with S' (from which $aS'b$ and c are derived) in the corresponding derivation tree containing another node labeled with S'. Similarly, the derivation tree corresponding to the pumping infix π_2 contains a sub-path between nodes labeled with A (from which aAb and aac are derived) that contains another node labeled with A.

Example 2. *Let $G' = (\{S, A, B\}, \{a, b, ¢, \$\}, S, R)$ be a CF(¢,$)-grammar with the following set of rules:*

$$\begin{array}{l} S \to ¢A\$ \mid ¢B\$, \\ A \to aAb \mid ab, \\ B \to aB \mid bB \mid a \mid b. \end{array}$$

Evidently, $L(G') = \{¢\} \cdot \{a, b\}^+ \cdot \{\$\}$.

Consider the pumping infix $(aa, a, A, aabb, b, bb)$. This pumping infix is a core and two-side pumping infix; therefore, grammar G' is a two-side pumping grammar.

However, every word from $L(G')$ can be reduced to $¢a\$$ or $¢b\$$ using left-side core pumping reductions, where the nonterminal A is not used.

Interestingly, after omitting all rules that include A from the grammar, we obtain a one-side (more precisely, left-side) pumping grammar generating the same language as the original grammar.

2.3. Complete CF(¢,$)-grammars

In this subsection, we generalize complete LR(¢,$)-grammars from [3] to complete CF(¢,$)-grammars.

A complete CF(¢,$)-grammar is a grammar that enables analysis of a language and its complement. If a complete CF(¢,$)-grammar is used in an analytic mode, it returns a derivation tree for each input word of the form $¢w\$$, where $w \in \Sigma^*$. The nonterminal under their root distinguishes the accepting and rejecting analytic trees. That means that the accepted words have accepting trees only, and the rejected words have rejecting trees only.

Definition 4. *Let $G_C = (N, \Sigma \cup \{¢, \$\}, S, R)$ be a CF(¢,$)-grammar. Then G_C is called a* complete CF(¢,$)-grammar *if*

1. *$S \to S_A \mid S_R$, where $S_A, S_R \in N$, are the only rules in R containing the initial nonterminal S. No other rule of G_C contains S_A or S_R in its right-hand side.*
2. *The languages $L(G_A)$ and $L(G_R)$ generated by the grammars $G_A = (N, \Sigma \cup \{¢, \$\}, S_A, R)$ and $G_R = (N, \Sigma \cup \{¢, \$\}, S_R, R)$, respectively, are disjoint and complementary with respect to $\{¢\} \cdot \Sigma^* \cdot \{\$\}$. That is, $L(G_A) \cap L(G_R) = \emptyset$ and $L(G_C) = L(G_A) \cup L(G_R) = \{¢\} \cdot \Sigma^* \cdot \{\$\}$.*

We will denote the grammar as $G_C = (G_A, G_R)$. Further, we will call G_A and G_R as accepting and rejecting grammar of the complete CF(¢,$)-grammar G_C, respectively.

Obviously, for each word of the form $¢w\$$, where $w \in \Sigma^*$, there is some derivation tree T according to G_C. The node under the root of T is labeled either by S_A or S_R. If it is S_A, the word is generated by the accepting grammar G_A. Otherwise, it is generated by the rejecting grammar G_R.

Moreover, for each word, two or more derivation trees can exist, but all of them are accepting, or all of them are rejecting.

The following lemma will be used below to show that we can decide if a given complete CF(¢,$)-grammar is one-side CF(¢,$)-grammar.

Lemma 2. *Let G be a reduced CF(¢,$)-grammar. If there is a core two-side (left-side or right-side, respectively) pumping infix by G, then there is a two-side (left-side or right-side, respectively) core pumping infix by G for a word of length at most K_G^3.*

Proof: Let $\pi = (x, u_1, A, v, u_2, y)$ be a two-side core pumping infix by G. There is a derivation tree T corresponding to a derivation

$$S \Rightarrow^* ¢xAy\$ \Rightarrow^* ¢xu_1Au_2y\$ \Rightarrow^* ¢xu_1vu_2y\$ \quad (3)$$

where the sub-path P_{r_1,r_2} between the root r_1 of the subtree corresponding to a derivation of u_1Au_2 from A to the root r_2 of the subtree corresponding to the derivation of v (but without r_2) in (3) does not contain two distinct nodes labeled with the same nonterminal.

Suppose there are two distinct nodes on the path P_{r_1} between the root of T and the node r_1 labeled with the same nonterminal. In that case, we can perform the corresponding pumping reduction and still preserve a two-side pumping infix in the reduced derivation tree.

Similarly, let P be a path between an arbitrary leaf (including leaves under r_2) and the closest node on the path P_{r_2} between the root of T and r_2. If P contains two distinct nodes labeled with the same nonterminal, we perform the corresponding pumping reduction. In the obtained derivation tree, we can still find a two-side core pumping reduction, as the pumping reduction does not delete any node from the sub-path P_{r_1,r_2}. Note that G is reduced and, according to Definition 1, G does not contain any rule of the form $X \to \lambda$. Hence, a reduction on a path P from a leaf to a node on the path P_{r_2} (path P contains only one node from P_{r_2}) cannot delete all terminal leaves of the corresponding subtree.

In this way, we obtain a derivation tree with a two-side pumping infix of height at most $3t$, where t is the number of nonterminals of G. Such a tree has at most $k^{3t} = K_G^3$ leaves.

The proofs for left-side and right-side pumping infixes are similar. □

Lemma 2 implies that we can decide whether given CF(¢,$)-grammar is one-side pumping grammar by inspecting all pumping infixes in all derivation trees for words of length at most K_G^3. Below we show how to do such a test more efficiently.

For a context-free grammar $G = (N, \Sigma, S, R)$, we define its *size* as $|G| = \sum_{r \in R} |r|$, where $|r|$ denotes the number of terminals and nonterminals in the rule r, including the left-hand side nonterminal, i.e., if $r = A \to \gamma$, where $A \in N$ and $\gamma \in (\Sigma \cup N)^*$, then $|r| = 1 + |\gamma|$.

Theorem 1. *Let $G = (N, \Sigma \cup \{¢, \$\}, S, R)$ be a reduced CF(¢,$)-grammar. An algorithm that runs in $O(|G|)$ time can decide whether G is one-side pumping.*

Proof: As grammar G is reduced and does not have any rule with the empty right-hand side, each nonterminal can be used in some derivation, and from each nonterminal, only nonempty terminal strings can be derived.

We construct a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ as follows: $\mathcal{V} = N$, and $(A, B) \in \mathcal{E}$ if and only if there is a rule $A \to \gamma$ in R, where $B \in N$ is included in γ. Note that $\mathcal{E}$ can contain loops and $|\mathcal{E}| = O(|G|)$.

Let $\Sigma_1 = \Sigma \cup \{¢, \$\}$. Define the function $\ell : \mathcal{E} \to 2^{\{\mathsf{left},\mathsf{right},\mathsf{both}\}}$ such that for each $e = (A, B) \in \mathcal{E}$,

- $\mathsf{left} \in \ell(e)$ if there is a rule $A \to \alpha B$ in R, where $A, B \in N$ and $\alpha \in (N \cup \Sigma_1)^+$,
- $\mathsf{right} \in \ell(e)$ if there is a rule $A \to B\beta$ in R, where $A, B \in N$ and $\beta \in (N \cup \Sigma_1)^+$, and
- $\mathsf{both} \in \ell(e)$ if there is a rule $A \to \alpha B \beta$ in R, where $A, B \in N$ and $\alpha, \beta \in (N \cup \Sigma_1)^+$.

Let $W = (e_1, \ldots, e_n)$ be a directed walk in $\mathcal{G}$ such that $e_i = (v_{i-1}, v_i) \in \mathcal{E}$, for $i = 1, \ldots, n$. As G is reduced, there exists a derivation tree T for G containing a path $P = (f_1, \ldots, f_n)$, where $f_i = (u_{i-1}, u_i)$, for some nodes $u_0, \ldots, u_n$ of T such that the nodes on P are labeled with nonterminals $v_0, \ldots, v_n$.

Based on Definition 2 of core pumping infix, we can conclude that G is a left-side pumping grammar if and only if, for each walk $W = (e_1, \ldots, e_m)$ in $\mathcal{G}$ starting and ending in the same node, it holds

$$\bigcup_{e \in W} \ell(e) \subseteq \{\text{left}\}. \quad (4)$$

Similarly, G is a right-side pumping grammar if and only if for each walk $W = (e_1, \ldots, e_m)$ in $\mathcal{G}$ starting and ending in the same node it holds

$$\bigcup_{e \in W} \ell(e) \subseteq \{\text{right}\}. \quad (5)$$

Obviously, all edges from W are in the same strongly connected component of graph $\mathcal{G}$. We can construct all strongly connected components of graph $\mathcal{G}$ using Tarjan's algorithm [9] in time $O(|\mathcal{V}|+|\mathcal{E}|) = O(|G|)$. Then, for each strongly connected component $\mathcal{C}$, we can test whether $\mathcal{C}$ satisfies the condition (4) or (5) in time linear with the size of $\mathcal{C}$. Hence, we can decide whether grammar G is one-side in time $O(|\mathcal{V}| + |\mathcal{E}|) = O(|G|)$. □

2.4. LR(0) Grammars

One of the main results of this paper strongly utilizes the theory of LR(0) grammars [7]. For any LR(0) grammar G, we can construct a deterministic parser that not only accepts each word $w \in L(G)$ but also produces a unique derivation tree for w. Then, with such a derivation tree, we can unambiguously associate the rightmost derivation of the word w according to G.

Let us recall the definition and properties of LR(0) grammars from [7].

Definition 5 ([7]). *Let $G = (N, \Sigma, S, R)$ be a context-free grammar and $\gamma \in (N \cup \Sigma)^*$. A* handle of γ *is an ordered pair (ρ, i), $\rho \in R, i \geq 0$ such that there exists $A \in N, \alpha, \beta \in (N \cup \Sigma)^*$ and $w \in \Sigma^*$ such that*

(a) $S \Rightarrow^{r*} \alpha A w \Rightarrow^{r} \alpha\beta w = \gamma$,
(b) $\rho = A \to \beta$, *and*
(c) $i = |\alpha\beta|$.

While a handle in a string is generally not uniquely defined, this is not the case for LR(0) grammars.

Definition 6. *Let $G = (N, \Sigma, S, R)$ be a reduced context-free grammar such that $S \Rightarrow^{r+} S$ is not possible in G. We say G is an* LR(0) *grammar if, for each $w, w', x \in \Sigma^*$, $\eta, \alpha, \alpha', \beta, \beta' \in (N \cup \Sigma)^*$, and $A, A' \in N$,*

(a) $S \Rightarrow^{r*} \alpha A w \Rightarrow^{r} \alpha\beta w = \eta w$, *and*
(b) $S \Rightarrow^{r*} \alpha' A' x \Rightarrow^{r} \alpha'\beta' x = \eta w'$

implies $(A \to \beta, |\alpha\beta|) = (A' \to \beta', |\alpha'\beta'|)$.

Thus, if G is an LR(0) grammar, then the rightmost derivation of the word w by G and the left-right analysis are unique (deterministic). This paper considers $LR(0)$ grammars to a significant extent as analytical grammars. A language generated by an LR(0) grammar is called LR(0) language.

It is shown in [7] that every LR(0) language is deterministic context-free, and for each deterministic context-free language $L \subseteq \Sigma^*$ and symbol $\$ \notin \Sigma$, the language $L \cdot \{\$\}$ is LR(0). The construction of an "LR-style parser" is also given there. The parser is a deterministic pushdown automaton that reads the input word from left to right and stores the partially processed prefix of the input word in its stack until the right-hand side of the rewriting rule of the grammar is identified, and the right-hand side is then replaced with the corresponding left-hand side. The input word is accepted if it is reduced to the starting nonterminal in its stack.

In literature, several automata models were based on the analysis by reduction. In [3], we introduced so-called RP-automata, the restarting automata [2] that differ only slightly from the original RW-automata introduced in [10] and from reducing automata presented in [11]. RP-automata perform only pumping reductions.

2.5. LR(¢,$)-grammars

Definition 7. *Let $¢, \$ \notin (N \cup \Sigma)$ and $G = (N, \Sigma \cup \{¢, \$\}, S, R)$ be a CF(¢,$)-grammar that is also $LR(0)$ grammar. We say that G is an LR(¢,$)-grammar.*

Classes of languages. In what follows, $\mathcal{L}(A)$, where A is some (sub)class of grammars or automata, denotes the class of languages generated/accepted by grammars/automata from A. Similarly, for (sub)class A of CF(¢,$)-grammars we denote $\mathcal{L}_{\text{in}}(A) = \{L \mid \{¢\} \cdot L \cdot \{\$\} \in \mathcal{L}(A)\}$.

Based on the closure properties of DCFL shown, e.g., in [7], internal languages of LR(¢,$)-grammars can be used to represent all deterministic context-free languages.

Proposition 1 ([3]). $\mathcal{L}_{\text{in}}(LR(¢, \$)) = \text{DCFL}$.

Note. It is not hard to see that the languages from $\mathcal{L}(CF(¢, \$))$ are prefix-free and suffix-free languages at the same time.

The next theorem shows the importance of complete grammar $G_C = (G_A, G_R)$.

Theorem 2 ([3]). *For any LR(¢,$)-grammar G_A, there exists a complete LR(¢,$)-grammar $G_C = (G_A, G_R)$.*

Example 3. *Consider the non-regular deterministic context-free language $L = \{¢a^n b^n \$ \mid n \geq 1\}$ with the internal language $L_{\text{in}} = \{a^n b^n \mid n \geq 1\}$ that is generated by the reduced LR(¢,$)-grammar $G = (\{S, S_1, a, b\}, \{a, b\} \cup \{¢, \$\}, R, S)$, with the set of rules R:*

$$\begin{array}{lcl} S & \to & ¢S_1\$, \\ S_1 & \to & aS_1b \mid ab. \end{array}$$

Consider the sentence $\gamma = ¢aaabbb\$$. For example, the pair $(S_1 \to ab, 5)$ is a handle of γ (cf. Definition 5), as

$$S \Rightarrow^{r*} ¢aaS_1bb\$ \Rightarrow^{r} ¢aaabbb\$$$

and the division of γ into α, β, w is unique:

$$\gamma = \underbrace{¢aa}_{\alpha}\underbrace{ab}_{\beta}\underbrace{bb\$}_{w}.$$

We can see that G is a linear LR(¢, $)-grammar, as
(a) $S \Rightarrow^{r*} \alpha A w \Rightarrow^{r} \alpha\beta w = \eta w$, *and*
(b) $S \Rightarrow^{r*} \alpha' A' x \Rightarrow^{r} \alpha'\beta' x = \eta w'$
obviously implies $(A \to \beta, |\alpha\beta|) = (A' \to \beta', |\alpha'\beta'|)$, because $A = S_1$, $\alpha = a^n$, $w = a^n$, $\beta = aS_1b$, for some $n \geq 0$.

Fig. 1 illustrates the pumping infix (x, a, S_1, ab, b, y) in a derivation tree for $\gamma = ¢xaabby\$ \in L(G)$, where $x = a^i$, $y = b^i$, for any $i \geq 0$.

3. One-Side Pumping Reductions

This section will show that left-side and right-side pumping CF(¢,$)-grammars generate only regular languages. At first, we will study the rightmost derivations according to right-side pumping CF(¢,$)-grammars.

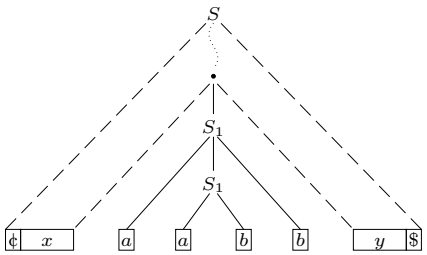

Figure 1: A derivation tree for $\gamma = ¢xaabby\$$ with pumping infix (x, a, S_1, ab, b, y).

Lemma 3. *Let $G = (N, \Sigma \cup \{¢, \$\}, S, R)$ be a right-side pumping CF(¢,$)-grammar. Let $\Sigma_1 = \Sigma \cup \{¢, \$\}$, k be the maximal length of the right-hand side of the rules of G and $t = |N|$. Let, for some $n \geq 1$, $\alpha_i \in (N \cup \Sigma_1)^* \cdot N$, $w_i \in \Sigma_1^*$, for all $i = 1, \ldots, n$,*

$$S \Rightarrow^{\mathrm{r}} \alpha_1 w_1 \Rightarrow^{\mathrm{r}} \cdots \Rightarrow^{\mathrm{r}} \alpha_{n-1} w_{n-1} \Rightarrow^{\mathrm{r}} w_n \quad (6)$$

be a rightmost derivation of a terminal word $w_n \in \Sigma_1^*$ according to G. Then, each α_i is of length at most kt, for all $i = 1, \ldots, n-1$.

Proof: For a contradiction, assume $|\alpha_i| > kt$, that is $\alpha_i = X_1 \cdots X_m$, where $m > kt$, $X_1, \ldots, X_{n-1} \in (N \cup \Sigma_1)$, and $X_m \in N$, for some i between 1 and $n-1$. Consider the partial derivation tree T_i corresponding to the sentential form $X_1 \cdots X_m w_i$. Let $p(X_j)$ denote the parent node of the node labeled with X_j, for $j = 1, \ldots, m$. As (6) is a rightmost derivation, it is possible that several nodes labeled with $X_1, \ldots, X_m$ have a common parent, but at most k nodes have the same parent, as k is the maximal length of a rule in R.

We will show that $p(X_j)$ is on the path from the root of T_i to $p(X_m)$, for all $j = 1, \ldots, m$.

Let us suppose that $p(X_j)$ is not on the path P from the root of T_i to $p(X_m)$. Then, let u, $u \neq p(X_j)$, denote the closest ancestor of $p(X_j)$ and $p(X_m)$ in T_i. The nodes labeled with X_j and X_m are descendants of two different child nodes of u.

Obviously, the nodes $p(X_j)$ and $p(X_m)$ are labeled by nonterminals. As (6) is a rightmost derivation, the node $p(X_j)$ cannot be rewritten before rewriting the nonterminal X_m into a terminal string. Hence, $p(X_j) = u$ and $p(X_j)$ is on the path P.

Thus, the set $Q = \{p(X_1), \ldots, p(X_m)\}$ contains more than $\frac{m}{k} > \frac{kt}{k} = t$ nodes, and all of them are on the path P. There exist two nodes $p(X_{j_1})$ and $p(X_{j_2})$ in Q, $p(X_{j_1}) \neq p(X_{j_2})$, labeled with the same nonterminal A such that between the nodes $p(X_{j_1})$ and $p(X_{j_2})$, there is no other node labeled with A. Then, there is a core pumping infix $\pi = (x, u_1, A, v, u_2, y)$ by G such that u_1 is nonempty, as G is reduced, G does not contain any rule of the form $B \to \lambda$, and u_1 contains at least a terminal string derived from X_{j_1}. Hence, π is not right-side pumping infix – a contradiction with the assumption that G is right-side pumping CF(¢,$)-grammar. □

Lemma 4. *Each language generated by a right-side pumping CF(¢,$)-grammar is regular.*

Proof: Let $G = (N, \Sigma \cup \{¢, \$\}, S, R)$ be a right-side pumping CF(¢,$)-grammar. Let $\Sigma_1 = \Sigma \cup \{¢, \$\}$, k be the maximal length of the right-hand side of the rules of G and $t = |N|$.

We will construct a regular (left-linear) grammar $G' = (N', \Sigma_1, S', R')$ generating the same language as G. The set of nonterminals N' will consist of sequences of nonterminals and terminals of G enclosed in square brackets. We will construct G' inductively.

Let $S' = [S]$, $N' = \bigcup_{i=0}^{\infty} N'_i$ and $R' = \bigcup_{i=0}^{\infty} R'_i$ (we use infinite unions, but we will see below that both N' and R' will be finite)

$$N'_0 = \{[S]\} \cup \{[\alpha] \mid S \to \alpha \in R\},$$
$$R'_0 = \{[X] \to [\alpha] \mid X \to \alpha \in R\}.$$

For $i \geq 0$, let

$$\begin{aligned} N'_{i+1} = N'_i \cup \{[\alpha\beta] \mid\ & \exists X \in N, \alpha, \beta \in (N \cup \Sigma_1)^* : \\ & [\alpha X] \in N'_i \text{ and } X \to \beta \in R\} \\ \cup \{[\alpha] \mid\ & \exists w \in \Sigma_1^* : [\alpha w] \in N'_i\}, \end{aligned}$$

$$\begin{aligned} R'_{i+1} = R'_i \cup \{[\alpha X] \to [\alpha\beta] \mid\ & X \in N, [\alpha X] \in N'_i, \\ & \alpha, \beta \in (N \cup \Sigma_1)^* : \\ & X \to \beta \in R\} \\ \cup \{[\alpha w] \to [\alpha] w \mid\ & \alpha \in (N \cup \Sigma_1)^* \cdot N, \\ & w \in \Sigma_1^+ : [\alpha w] \in N'_i\} \\ \cup \{[w] \to w \quad \mid\ & w \in \Sigma_1^*, [w] \in N'_i\}. \end{aligned}$$

If $[\gamma]$ is a nonterminal from N', then γ is a substring of a sentential form obtained during a rightmost derivation $S \Rightarrow^{\mathrm{r}*} \eta\gamma\xi$ according to G, for some $\eta \in (N \cup \Sigma_1)^*$, $\xi \in \Sigma_1^*$.

Either, $[\gamma] = [\alpha\beta]$ was obtained by rewriting $[\alpha X]$, for some $X \in N$, $[\alpha X] \in N'$, and $X \to \beta \in R$. Then, according to Lemma 3, αX cannot be longer than kt. Thus γ is of length at most $kt + k - 1$.

Or, $[\gamma] = [\alpha]$ was obtained by rewriting $[\alpha w]$, for some $\alpha \in (N \cup \Sigma_1)^* \cdot N$, $w \in \Sigma_1^+$. Then, according to Lemma 3, α cannot be longer than kt.

In both cases, if $[\gamma]$ is a nonterminal from N', for some $\gamma \in (N \cup \Sigma_1)^*$, then the length of γ is limited by a constant. Hence, N' and R' are finite sets.

It is easy to show that, for all $\alpha \in (N \cup \Sigma_1)^*$, $X \in N$, $y, w \in \Sigma_1^*$, it holds:

$$S \Rightarrow^{\mathrm{r}*} \alpha X y \Rightarrow^{\mathrm{r}*} w \text{ iff } [S] \Rightarrow^* [\alpha X] y \Rightarrow^* w.$$

Thus $L(G) = L(G')$, the grammar G' is left linear and generates a regular language, and $L(G)$ is a regular language. □

Let us illustrate the construction from the above proof by an example.

Example 4. *The grammar* $G = (\{S, A, B, C\}, \{a, b, c, ¢, \$\}, S, R)$ *with the set of rules* R*:*

$$S \to ¢A\$, \qquad A \to CaB \mid c,$$
$$B \to Bb \mid b, \quad C \to AcB.$$

is a right-side pumping CF(¢,$)-grammar. We can construct an equivalent left linear grammar G' *with the following set of rules:*

$[S] \to [¢A\$]$,
$[¢A] \to [¢CaB] \mid [¢c]$,
$[¢CaB] \to [¢CaBb] \mid [¢Cab]$,
$[¢Cab] \to [¢C]ab$,
$[¢AcB] \to [¢AcBb] \mid [¢Acb]$,
$[¢Acb] \to [¢A]cb$.
$[¢A\$] \to [¢A]\$$,
$[¢c] \to ¢c$,
$[¢CaBb] \to [¢CaB]b$,
$[¢C] \to [¢AcB]$,
$[¢AcBb] \to [¢AcB]b$,

Corollary 2. *Each language generated by a left-side pumping CF(¢,$)-grammar is regular.*

Proof: Let $G = (N, \Sigma \cup \{¢, \$\}, S, R)$ be a left-side pumping CF(¢,$)-grammar. Then the CF(¢,$)-grammar $G^{(\mathrm{R})} = (N, \Sigma \cup \{¢, \$\}, S, R^{(\mathrm{R})})$ obtained by reversing the right-hand sides of all rules of G, except that the sentinels ¢ and $ must not change their positions in the rules, generates the language $L(G^{(\mathrm{R})}) = \{¢\} \cdot [L_{\mathrm{in}(G)}]^{\mathrm{R}} \cdot \{\$\}$ and $G^{(\mathrm{R})}$ is a right-side pumping CF(¢,$)-grammar. Lemma 4 says the language $L(G^{(\mathrm{R})})$ is regular. The closure of the class of regular languages on quotients, reversal, and concatenation implies that $L(G)$ is regular, too. □

We get the following theorem as a simple consequence of Lemma 4 and Corollary 2.

Theorem 3. *Let* $G_C = (G_A, G_R)$ *be a complete one-side pumping CF(¢,$)-grammar. Then, both* $L(G_A)$ *and* $L_{\mathrm{in}}(G_A)$ *are regular languages.*

4. Two-Side Pumping

Theorem 3 has a straightforward consequence: non-regular languages cannot be generated with one-side pumping grammars.

Corollary 3. *Let* $G = (V, \Sigma \cup \{¢, \$\}, R, S)$ *be a CF(¢,$)-grammar such that* $L_{\mathrm{in}}(G)$ *is a non-regular language. Then* G *is a two-side pumping grammar.*

The opposite is not true – two-side pumping grammars can generate regular languages.

Example 5. *Consider the linear LR(¢,$)-grammar* G_1 *given by the following rules, where* S *is the starting nonterminal:*

$$S \to ¢S_1\$ \mid ¢T_1 \mid T_2\$, \quad T_1 \to aT_1 \mid aS_1\$,$$
$$S_1 \to aS_1b \mid ab, \quad T_2 \to T_2b \mid ¢S_1b.$$

It is easy to see that G_1 *enables two-side pumping reductions.* $L(G_1) = \{¢a^n b^m\$ \mid n > 0, m > 0\}$ *and* $L(G_1)$ *obviously is a regular language.*

The same language can, of course, be generated by the following left-side pumping grammar G_r *with starting nonterminal* S *and the following set of rules:*

$$S \to ¢T_1, \qquad T_1 \to aT_1 \mid aT_2$$
$$T_2 \to bT_2 \mid b\$.$$

Totally two-side pumping CF(¢,$)-grammars. A reduced CF(¢,$)-grammar G is called *totally two-side pumping* if there is at least one two-side core infix and no one-side pumping infix by G.

Example 6. *Let* G_t *be the following CF(¢,$)-grammar with starting nonterminal* S*, further nonterminals* A, B*, terminal alphabet* $\Sigma_1 = \Sigma \cup \{¢, \$\}$*, for* $\Sigma = \{a, b\}$*, and rules:*

$$S \to ¢A\$,$$
$$A \to aB \mid bB \mid a \mid b,$$
$$B \to Aa \mid Ab.$$

All core pumping infixes by G_t *are two-side and of the form* (x, u_1, X, v, u_2, y)*, where* $u_1, u_2 \in \Sigma$*,* $x, v, y \in \Sigma^*$ *such that either* $X = A$*,* $|v|$ *is odd and* $|x| = |y|$*, or* $X = B$*,* $|v|$ *is even and* $|x| = |y| + 1$*.*

Thus, grammar G_t *is a totally two-side pumping CF(¢,$)-grammar that generates the regular language* $L(G_t) = \{¢w\$ \mid w \in \{a, b\}^{2i+1}$ *for some* $i \geq 0\}$*. But* G_t *is not an LR(¢,$)-grammar.*

For a pumping infix (x, u_1, A, v, u_2, y), we say that u_1vu_2 is its *middle part*. Realize that although the length of core pumping infix is potentially unbounded, it is possible to identify core infixes with the length of their middle part limited by a constant. Such pumping infixes will be called bottom pumping infixes.

Definition 8. *Let* G *be a CF(¢,$)-grammar and* $\pi = (x, u_1, A, v, u_2, y)$ *be a pumping infix by* G*. We say that* π *is a* bottom pumping infix *if there is no other pumping infix reducing inside its middle part* u_1vu_2*. That is, if there are terminal words* x_1, u_1', v', u_2', y_1 *and a pumping infix* $\pi' = (xx_1, u_1', B, v', u_2', y_1y)$ *by* G *such that*

$$xu_1vu_2y = xx_1u_1'v'u_2'y_1y$$

then $\pi' = \pi$*.*

Correspondingly, we say that a pumping reduction is a bottom pumping reduction if the corresponding pumping infix is a bottom pumping infix. We write

$$x_1u_1vu_2y_1 \rightsquigarrow_{P(G_A, \mathrm{bottom})} x_1vy_1.$$

Let $\pi = (x, u_1, A, v, u_2, y)$ be a pumping infix, and T be a derivation tree by the grammar G corresponding to π. The tree T has a subtree $T_{u_1vu_2}$ that derives the middle part u_1vu_2. The pumping infix π is a bottom pumping infix if and only if the subtree $T_{u_1vu_2}$ has exactly one path from its root to its leaf on which two different nodes are labeled with the same nonterminal. Hence, the height of $T_{u_1vu_2}$ is at most $t+1$, where t is the number of nonterminals of G. Therefore, the length of the middle part u_1vu_2 is at most $k^{t+1} = t \cdot K_G$, where k is the maximal length of the right-hand side of the rules of G and K_G is the grammar number of G.

Evidently, each bottom pumping infix by G is a core pumping infix by G, and we could prove a stronger version of Corollary 1.

Corollary 4. *Let $G = (N, \Sigma \cup \{¢, \$\}, S, R)$ be a CF(¢,$)-grammar. If G generates w, then there exists a sequence of words $w_1, \ldots, w_n$ from $L(G)$, for some integer $n \geq 1$, such that $w = w_1$, there are bottom pumping reductions $w_i \rightsquigarrow_{P(G,\mathrm{bottom})} w_{i+1}$, for all $i = 1, \ldots, n-1$, and there is no $w_{n+1} \in \Sigma^*$ such that $w_n \rightsquigarrow_{P(G)} w_{n+1}$.*

Now we can show that in contrast to general two-side pumping CF(¢,$)-grammars, totally two-side pumping LR(¢,$)-grammars cannot generate regular languages.

Theorem 4. *Let G be a reduced totally two-side pumping LR(¢,$)-grammar. Then $L(G)$ is not a regular language.*

Proof: Let $G = (N, \Sigma \cup \{¢, \$\}, R, S)$ be a reduced totally two-side pumping LR(¢,$)-grammar. There is at least one two-side core pumping infix by G. For a contradiction, suppose that $L(G)$ is a regular language. We will show that there exists at least one one-side pumping infix by G, which contradicts the assumption that G is totally two-side pumping.

If $L(G)$ is a regular language, there is a deterministic finite automaton M such that $L(M) = L(G)$. Let q denote the number of states of M.

Let $\pi = (x, au_1, A, v, bu_2, y)$, where $a, b \in \Sigma$, $x, u_1, v, u_2, y \in \Sigma^*$, be a two-side bottom core pumping infix by G such that $|x|, |y| \leq K_G^3$. Such bottom pumping infix exists as G is reduced, G has at least one two-side pumping infix π', and, according to Lemma 2, there exists a two-side pumping infix π'' by G for a word of length at most K_G^3. We can suppose that π'' is a bottom pumping infix. If not, we can perform a finite sequence of pumping reductions that shorten all paths with repeated nonterminal except the one corresponding to a two-side pumping infix until we get a bottom core pumping infix π.

As π is a pumping infix by G, all words

$$w_i = ¢x(au_1)^i v(bu_2)^i y\$, \text{ for } i \geq 0,$$

are in $L(G)$. Since G is an LR(0) grammar, exactly one derivation tree exists for each word from $L(G)$.

Since M has q states, within its accepting computation on w_i, where $i > q$, automaton M visits at least two occurrences of a (in front of u_1) in the same state. Thus, there is a positive integer s such that $0 < s < q+1$, and for each non-negative integer i, the word $w_i' = ¢x(au_1)^{s \cdot i}(au_1)^{q+1} v(bu_2)^{q+1} y\$$ is in $L(G)$.

Let i_0 be an integer greater than $tK_G(K_G^3 + 2)$. Let us consider the derivation tree T_α by G for the word $\alpha = w_{si_0+q+1}$ and the derivation tree T_β by G for the word $\beta = w'_{i_0}$. The words α and β have the common prefix $\gamma = ¢x(au_1)^{si_0+q+1} v(bu_2)^{q+1}$.

According to Corollary 4, there is a sequence of the bottom core pumping reductions $\alpha_i \rightsquigarrow_{P(G,\mathrm{bottom})} \alpha_{i+1}$, for $i = 1, \ldots, n-1$ and $\alpha_1 = \alpha$, and there is no $\alpha_{n+1} \in \Sigma^*$ such that $\alpha_n \rightsquigarrow_{P(G)} \alpha_{n+1}$. Suppose it is the leftmost sequence of core bottom pumping reductions.

Similarly, for the word β, there is a sequence of the leftmost bottom core pumping reductions $\beta_j \rightsquigarrow_{P(G,\mathrm{bottom})} \beta_{j+1}$, for $j = 1, \ldots, m-1$ and $\beta_1 = \beta$, and there is no $\beta_{m+1} \in \Sigma^*$ such that $\beta_m \rightsquigarrow_{P(G)} \beta_{m+1}$.

The initial part of the sequence of the leftmost core bottom pumping reductions starting from α_1 until the first pumping reduction that uses a middle part that contains a symbol outside the prefix γ makes the same changes as the initial part of the sequence of the leftmost core bottom pumping reductions starting from β_1 until the first pumping reduction uses a middle part that contains a symbol outside the prefix γ.

In the case of α, after the common prefix of the sequence of bottom core pumping reductions, the subsequent reductions will continue deleting pairs of subwords au_1 and bu_2. In the case of β, the following pumping reductions must delete most of $¢x(au_1)^{si_0}$ and $y\$$.

Let us inspect the derivation tree T_α. Let $\mathcal{T}_\alpha = \{T_{\alpha,1}, \ldots, T_{\alpha,n_\alpha}\}$ be the set of all maximal subtrees of T_α such that all their leaves are in $¢(au_1)^{si_0}$. The set of subtrees $\mathcal{T}_\alpha$ is nonempty, and one of the trees in $\mathcal{T}_\alpha$ contains ¢.

Let $\mathcal{T}_\beta = \{T_{\beta,1}, \ldots, T_{\beta,n_\beta}\}$ be the set of all maximal subtrees of T_β such that all their leaves are in $¢(au_1)^{si_0}$. The set of subtrees $\mathcal{T}_\beta$ is nonempty, and one of the trees in $\mathcal{T}_\beta$ contains ¢.

Additionally, the sets $\mathcal{T}_\alpha$ and $\mathcal{T}_\beta$ are equal. Why? Because they are built during LR(0) analysis of the prefix γ, all reductions are made by the corresponding deterministic LR(0) analyzer when it scans the prefix γ.

For T_α, the series of core bottom pumping reductions can continue with pumping reductions corresponding to pumping infixes of the form π. In T_β, we can find a series of bottom core reductions that delete (most of) $(au_1)^{si_0}$. However, all these reductions in T_β must have the middle part that includes at least one symbol from the suffix $\omega = y\$$. If any of these reductions do not include any symbol from ω in its middle part, they must have already been done in the sequence of reductions performed inside the prefix γ.

Each bottom pumping reduction can shorten the current word by at most $t \cdot K_G$ symbols (the upper limit of the length of the middle part of a bottom pumping infix). Hence, the sequence of reduction is of length at least

$$\frac{si_0 \cdot |au_1| + |y|}{t \cdot K_G} > \frac{tK_G(K_G^3 + 2)}{tK_G} = K_G^3 + 2.$$

As the suffix ω of β is of length at most $K_G^3 + 1$, at least one of these bottom reductions does not delete any symbol from ω, while its middle part must include at least one symbol from ω. Such bottom reduction is a left-side pumping reduction.

We have proved that if a two-side LR(¢,$)-grammar accepts a regular language, then it has at least one core left-side pumping reduction. □

A CF(¢,$)-grammar G is totally two-side pumping if it only has two-side pumping infixes. A slight modification of the procedure from the proof of Theorem 1 gives an algorithm that decides whether G is totally two-side pumping grammar in $O(|G|)$ time.

5. Refinement Results

We use the following notations for our types of context-free grammars. Prefix lin- denotes the linear CF-grammars, similarly 1s- the one-side pumping CF-grammars, lfs- the left-side pumping CF-grammars, rs- the right-side pumping CF-grammars, and ttsp- the totally two-side pumping CF($\text{¢}, \$$)-grammars.

Moreover, we denote the set of accepting grammars of complete CF(¢,$)-grammars as CCFA.

Corollary 5. *It holds the following:*

$$\mathcal{L}_{\text{in}}(\textit{1s-}LR(\text{¢}, \$)) = \mathcal{L}_{\text{in}}(\textit{lfs-}LR(\text{¢}, \$)) = \mathcal{L}_{\text{in}}(\textit{rs-}LR(\text{¢}, \$)) = \text{REG}.$$

$$\mathcal{L}_{\text{in}}(\textit{1s-CCFA}) = \mathcal{L}_{\text{in}}(\textit{lfs-CCFA}) = \mathcal{L}_{\text{in}}(\textit{rs-CCFA}) = \text{REG}.$$

Proof: The corollary is a consequence of Theorem 3 and of the fact that for each regular language L, there exists a left-linear LR(0) grammar and a right-linear LR(0) grammar that both generate L. Recall that regular languages are closed on both left and right quotients. □

The next result follows from the previous proof.

Corollary 6. *It holds the following:*

$$\mathcal{L}_{\text{in}}(\textit{lin-1s-}LR(\text{¢}, \$)) = \mathcal{L}_{\text{in}}(\textit{lin-lfs-}LR(\text{¢}, \$)) = \mathcal{L}_{\text{in}}(\textit{lin-rs-}LR(\text{¢}, \$)) = \text{REG}.$$

$$\mathcal{L}_{\text{in}}(\textit{lin-1s-CCFA}) = \mathcal{L}_{\text{in}}(\textit{lin-lfs-CCFA}) = \mathcal{L}_{\text{in}}(\textit{lin-rs-CCFA}) = \text{REG}.$$

Corollary 7. $\mathcal{L}_{\text{in}}(\textit{lin-ttsp-}LR(\text{¢}, \$)) \subset \mathcal{L}_{\text{in}}(\textit{lin-}LR(\text{¢}, \$)) \subset \mathcal{L}_{\text{in}}(LR(\text{¢}, \$)) = \text{DCFL}.$

Proof: We can see that each language from $\mathcal{L}_{\text{in}}(\text{lin-}LR(\text{¢}, \$))$ is a linear context-free language. On the other hand, the Dyck language is from DCFL, and it is not a linear context-free language [7]. The class $\mathcal{L}_{\text{in}}(\text{lin-ttsp-}LR(\text{¢}, \$))$ does not contain any regular language. On the other hand, the class $\mathcal{L}_{\text{in}}(\text{lin-}LR(\text{¢}, \$))$ contains all regular languages. □

Corollary 8.

$$\mathcal{L}_{\text{in}}(\textit{ttsp-}LR(\text{¢}, \$)) \subset \mathcal{L}_{\text{in}}(LR(\text{¢}, \$)) = \text{DCFL}.$$

Proof: Strictness of the inclusion follows from the fact that the class $\mathcal{L}_{\text{in}}(\text{ttsp-}LR(\text{¢}, \$))$ does not contain any regular language, and the class $\mathcal{L}_{\text{in}}(LR(\text{¢}, \$))$ contains all regular languages. □

The class of context-free languages is not closed on complement. Hence, complete CF(¢,$)-grammars generate only a subset of the class of context-free languages as their inner languages. Nevertheless, they can also generate languages that are not deterministic context-free languages. We give an example of complete CF(¢,$)-grammar $Gab_C = (Gab_A, Gab_R)$ such that $L_{\text{in}}(Gab_A)$ and its complement are non-regular (nondeterministic) context-free languages.

Example 7. *We start with the grammar Gab_A generating the language $\{\text{¢}a^n b^m \$ \mid 0 < n \leq m \leq 2n\}$. It is well known that this language is not a deterministic context-free language.*

$$\begin{aligned} S_A &\to \text{¢}S_1\$, \\ S_1 &\to aS_1b \mid aS_1bb \mid ab \mid abb. \end{aligned}$$

The complement of $L_{in}(Gab_A)$ is the inner language of a grammar Gab_R that generates the language

$$\begin{aligned} &\{\text{¢}a^n b^m \$ \mid 0 \leq m < n\} \cup \\ &\{\text{¢}a^n b^m \$ \mid m > 2n \geq 0\} \cup \\ &(\text{¢}\{a,b\}^* ba \{a,b\}^* \$) \cup \text{¢}\$. \end{aligned}$$

The grammar Gab_R with the starting symbol S_R has the following rules:

$$\begin{array}{lcllcl} S_R & \to & \text{¢}S_2\$, & S_R & \to & \text{¢}S_4\$, \\ S_2 & \to & aS_2 \mid S_3, & S_4 & \to & aS_4bb \mid S_4b \mid b, \\ S_3 & \to & aS_3b \mid a, & & & \end{array}$$

$$\begin{aligned} S_R &\to \text{¢}S_5\$ \mid \text{¢}\$, \\ S_5 &\to ba, \\ S_5 &\to aS_5 \mid bS_5 \mid S_5a \mid S_5b. \end{aligned}$$

Corollary 9.

$$DCFL \subset \mathcal{L}_{in}(CCFA) \subset CFL.$$

Proof: The first proper inclusion follows from the previous example, and the second one follows from the fact that the class of context-free languages is not closed under complement. □

6. Conclusion and Future Work

In this paper, we introduced and studied complete CF(¢,$)-grammars. We have shown that left-side pumping complete CF(¢,$)-grammars and right-side pumping complete CF(¢,$)-grammars characterize regular languages. On the other hand, general pumping LR(¢,$)-grammars characterize DCFL, and totally two-side pumping LR(¢,$)-grammars generate non-regular deterministic context-free languages only. These results imply similar results for one-side pumping and two-side pumping RP(LR(¢,$))-automata from [3].

Next, we will focus on studying the regular and non-regular characteristics of two-side core pumping patterns in RP(LR(¢,$))-automata and CF(¢,$)-grammars. We aim to utilize these characteristics to develop tools for effective localization of syntactic errors in deterministic context-free languages (DCFL). We will demonstrate that restarting automata can serve as error-sensible analyzers for complete CF(¢,$)-grammars.

Finally, we aim to present a construction that transforms a monotone restarting automaton with pumping properties into a complete LR(¢,$)-grammar, while maintaining the same analysis by reduction and recognizing the same languages. A preliminary step towards this goal was already taken in [11].

7. Acknowledgments

The research has been supported by grant 1/0601/20 of the Slovak Scientific Grant Agency VEGA (Dana Pardubská) and by grant 19-21198S of the Czech Science Foundation (Daniel Průša).

References

[1] F. Mráz, D. Pardubská, M. Plátek, J. Šíma, Pumping deterministic monotone restarting automata and DCFL, in: M. Holena, T. Horváth, A. Kelemenová, F. Mráz, D. Pardubská, M. Plátek, P. Sosík (Eds.), Proceedings of the 20th Conference Information Technologies – Applications and Theory (ITAT 2020), volume 2718 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2020, pp. 51–58. URL: http://ceur-ws.org/Vol-2718/paper13.pdf.

[2] M. Plátek, F. Mráz, D. Pardubská, D. Průša, J. Šíma, On separations of LR(0)-grammars by two types of pumping patterns, in: B. Brejová, L. Ciencialová, M. Holena, F. Mráz, D. Pardubská, M. Plátek, T. Vinar (Eds.), Proceedings of the 21st Conference Information Technologies – Applications and Theory (ITAT 2021), volume 2962 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2021, pp. 140–146. URL: http://ceur-ws.org/Vol-2962/paper05.pdf.

[3] M. Plátek, F. Mráz, D. Pardubská, D. Průša, On pumping RP-automata controlled by complete LRG(¢, $)-grammars, in: L. Ciencialová, M. Holena, R. Jajcay, T. Jajcayová, F. Mráz, D. Pardubská, M. Plátek (Eds.), Proceedings of the 22nd Conference Information Technologies – Applications and Theory (ITAT 2022), volume 3226 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2022, pp. 111–121. URL: https://ceur-ws.org/Vol-3226/paper13.pdf.

[4] J. Šíma, M. Plátek, One analog neuron cannot recognize deterministic context-free languages, in: T. Gedeon, K. W. Wong, M. Lee (Eds.), Neural Information Processing – 26th International Conference, ICONIP, Proceedings, Part III, volume 11955 of *Lecture Notes in Computer Science*, Springer, 2019, pp. 77–89. doi:10.1007/978-3-030-36718-3_7.

[5] P. Jančar, J. Šíma, The simplest non-regular deterministic context-free language, in: F. Bonchi, S. J. Puglisi (Eds.), 46th International Symposium on Mathematical Foundations of Computer Science, MFCS 2021, volume 202 of *LIPIcs*, Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021, pp. 63:1–63:18. doi:10.4230/LIPIcs.MFCS.2021.63.

[6] M. Lopatková, M. Plátek, P. Sgall, Towards a formal model for functional generative description: Analysis by reduction and restarting automata, Prague Bull. Math. Linguistics 87 (2007) 7–26. URL: http://ufal.mff.cuni.cz/pbml/87/lopatkova-et-al.pdf.

[7] M. A. Harrison, Introduction to Formal Language Theory, Addison-Wesley, USA, 1978.

[8] J. Hopcroft, J. Ullman, Introduction to Automata Theory, Languages, and Computation, Addison-Wesley, N. Reading, MA, 1980.

[9] R. E. Tarjan, Depth-first search and linear graph algorithms, SIAM J. Comput. 1 (1972) 146–160. doi:10.1137/0201010.

[10] P. Jančar, F. Mráz, M. Plátek, J. Vogel, On monotonic automata with a restart operation, J. Autom. Lang. Comb. 4 (1999) 287–311. doi:10.25596/jalc-1999-287.

[11] M. Procházka, On reducing automata and their normalizations, in: L. Ciencialová, M. Holena, R. Jajcay, T. Jajcayová, F. Mráz, D. Pardubská, M. Plátek (Eds.), Proceedings of the 22nd Conference Information Technologies – Applications and Theory (ITAT 2022), volume 3226 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2022, pp. 130–141. URL: https://ceur-ws.org/Vol-3226/paper15.pdf.