

TWO-WAY PC GRAMMAR SYSTEMS BASED ON REGULAR GRAMMARS

Ing. Petr Kaláb, Doctoral Degree Programme (2)
Dept. of Information Systems, FIT, BUT
E-mail: kalabp@fit.vutbr.cz

Supervised by Dr. Alexander Meduna

ABSTRACT

Besides derivation and communication steps, a two-way PC grammar system can make a reduction step during which it reduces the right-hand side of a context-free production to its left hand-side. This paper proves that every non-unary recursively enumerable language is defined by a centralized two-way grammar system, Γ , with three components in a very economical way. Indeed, Γ 's master has only three nonterminals and one communication production; furthermore, it produces all sentential forms with no more than two occurrences of nonterminals. In addition, during every computation, Γ makes a single communication step. Some variants of two-way PC grammar system are discussed in the conclusion of this paper.

1 INTRODUCTION

The formal language theory has intensively investigated various variants of PC grammar systems, which consist of several components, represented by grammars. This paper introduces some variant of *two-way PC grammar systems*, which make three kinds of computational steps — derivation, reduction, and communication. More precisely, a two-way PC grammar system, Γ , makes a derivation step as usual; that is, it rewrites the left-hand side of a production with its right-hand side. In a reduction step Γ rewrites the right-hand side with the left hand-side. Finally, Γ makes a communication step in a usual PC-grammar-system way; in addition, however, after making this step, it changes the computational way from derivations to reductions and vice versa.

This paper discusses the centralized form of two-way PC grammar systems working in a non-returning mode. That is, since they are centralized, only their first components, called the masters, can cause these systems to make a communication step. A non-returning mode means that after communication step, the components of grammar system continue to process the current string rather than return to their axioms. The present paper concentrates its discussion on their descriptive complexity because this complexity represents an intensively studied area of today's formal language theory.

This paper proves that the centralized (2-CF, RLIN, LLIN) two-way 3-PC grammar

systems characterize the family of non-unary recursively enumerable languages in a very economical way. Indeed, every non-unary recursively enumerable language is defined by a centralized (2-CF, RLIN, LLIN) two-way 3-PC grammar system with one context-free component, one right-linear component and one left-linear component so that during every computation Γ makes a single communication step. In addition, Γ 's three-nonterminal master has only one production with a communication symbol and any of its sentential forms contains no more than two occurrences of nonterminals.

2 DEFINITION

Let k and n be two positive integers. A *two-way k -context-free PC component* is a quadruple, $G = (N, T, P, S)$, where N and T are two disjoint alphabets. Symbols in N and T are referred to as *nonterminal* and *terminals*, respectively, and $S \in N$ is the *start symbol* of G . Set $M = N \setminus \{S\}$. P is a finite set of *productions* such that each $r \in P$ has one of these forms

$S \rightarrow x$, where $x \in (T \cup M)^*$ and x contains no more than k occurrences of symbols from M ,

$A \rightarrow x$, where $A \in M$, $x \in T^*MT^* \cup T^*$ and x contains no more than k occurrences of symbols from M .

Let $u, v \in (N \cup T)^*$. For every $A \rightarrow x \in P$, write $uAv \xrightarrow{d} uxv$ and $uxv \xrightarrow{r} uAv$; d and r stand for a direct *derivation* and a direct *reduction*, respectively. To express that G makes $uAv \xrightarrow{d} uxv$ according to $A \rightarrow x$, write $uAv \xrightarrow{d} uxv [A \rightarrow x]$; $uxv \xrightarrow{r} uAv [A \rightarrow x]$ have an analogical meaning in terms of \xrightarrow{r} . A *two-way k -context-free n -PC grammar system* is an $n + 1$ tuple

$$\Gamma = (Q, G_1, \dots, G_n),$$

where $Q = \{q_i : i = 1, \dots, n\}$, whose members are called *query symbols*, and for all $i = 1, \dots, n$, $G_i = (Q \cup N_i, T, P_i, S_i)$ is a two-way k -context-free PC component such that $Q \cap (N_i \cup T) = \emptyset$ (notice that each G_i has the same terminal alphabet, T); let $q\text{-}P_i \subseteq P_i$ denote the set of all productions in P_i containing a query symbol. A configuration is an n -tuple of the form (x_1, \dots, x_n) , where $x_i \in (Q \cup N_i \cup T)^*$, $1 \leq i \leq n$. The start configuration, σ , is defined as $\sigma = (S_1, \dots, S_n)$. Let Θ denote the set of all configurations of Γ . For every $x \in \Theta$ and $i = 1, \dots, n$, $i\text{-}x$ denote its i th component—that is, if $x = (x_1, \dots, x_i, \dots, x_n)$, then $i\text{-}x = x_i$. For every $x \in \Theta$, define the mapping ${}_x\theta$ over $\{i\text{-}x : 1 \leq i \leq n\}$ as ${}_x\theta(i\text{-}x) = z_1z_2\dots z_{|i\text{-}x|}$ where for all $1 \leq h \leq |i\text{-}x|$, if for some $q_j \in Q$, $i = 1, \dots, n$, $\mathbf{sym}(i\text{-}x, h) = q_j$ and $\mathbf{alph}(j\text{-}x) \cap Q = \emptyset$, then $z_h = j\text{-}x$; otherwise (that is, $\mathbf{sym}(i\text{-}x, h) \notin Q$ or $\mathbf{alph}(j\text{-}x) \cap Q \neq \emptyset$), $z_h = \mathbf{sym}(i\text{-}x, h)$.

Let $y, x \in \Theta$. Write

- $y \xrightarrow{d} x$ in G if $i\text{-}y \xrightarrow{d} i\text{-}x$ in G_i or $i\text{-}y = i\text{-}x$ with $i\text{-}y, i\text{-}x \in T^*$, for all $i = 1, \dots, n$
- $y \xrightarrow{r} x$ in G if $i\text{-}y \xrightarrow{r} i\text{-}x$ in G_i or $i\text{-}y = i\text{-}x$ with $i\text{-}y, i\text{-}x \in \{S_i\} \cup T^*$, for all $i = 1, \dots, n$
- $y \xrightarrow{q} x$ in G if $i\text{-}x = \theta(i\text{-}y)$ in G for all $i = 1, \dots, n$

Informally, Γ works in three computational modes — \xrightarrow{d} , \xrightarrow{r} , \xrightarrow{q} , which symbolically represent a direct *derivation*, *reduction*, and *communication*, respectively. Let $l \geq 1$, $\alpha_j \in \Theta$, $1 \leq i \leq l$, and $\alpha_0 \Vdash \alpha_1 \Vdash \alpha_2 \dots \alpha_{l-1} \Vdash \alpha_l$, where $l_m \in \{d, r, q\}$, $1 \leq m \leq l$; write $\alpha_0 \xRightarrow{*} \alpha_l$ if $l_1 = d$ and each $l_p \in \{d, r, q\}$, $2 \leq p \leq l - 1$, satisfies:

- if $l_p = q$ then $l_{p+1}, l_{p-1} \in \{d, r\}$ and $l_{p+1} \neq l_{p-1}$
- if $l_p \in \{d, r\}$ then $l_{p+1} \in \{q, l_p\}$

Informally, after making a communication step, Γ changes the computational mode from d to r and vice versa; after making a derivation or reduction step, it does not. Consider $\alpha_0 \Rightarrow^* \alpha_l$ that consists of l direct computational steps, $\alpha_0 \Rightarrow \alpha_1 \Rightarrow \alpha_2 \dots \alpha_{l-1} \Rightarrow \alpha_l$, satisfying the above properties. Set $\kappa(\alpha_0 \Rightarrow^* \alpha_l) = \{\alpha_0, \alpha_1, \dots, \alpha_l\}$; that is, $\kappa(\alpha_0 \Rightarrow^* \alpha_l)$ denotes the set of all configurations occurring in $\alpha_0 \Rightarrow^* \alpha_l$. Furthermore, for each $l = 1, \dots, n$, set $\kappa(i-\alpha_0 \Rightarrow^* i-\alpha_l) = \{i-\beta : \beta \in \kappa(\alpha_0 \Rightarrow^* \alpha_l)\}$. Finally, for each $h = 1, \dots, n$, h -computation($i-\alpha_0 \Rightarrow^* i-\alpha_l$) denotes $h-\alpha_0 \Rightarrow h-\alpha_1 \Rightarrow h-\alpha_2 \dots h-\alpha_{l-1} \Rightarrow h-\alpha_l$. The language of Γ , $L(\Gamma)$, is defined as

$$L(\Gamma) = \{z \in T^* : \sigma \Rightarrow^* \alpha \text{ in } \Gamma \text{ with } z = \mathbf{del}(1-\alpha, S_1), \text{ for some } \alpha \in \Theta\}$$

Informally, $L(\Gamma)$ contains $z \in T^*$ if and only if there exists $\alpha \in \Theta$ such that $\sigma \Rightarrow^* \alpha$ in Γ and the deletion of each S_1 in $1-\alpha$ results in z . A computation $\sigma \Rightarrow^* \alpha$ in Γ with $\mathbf{del}(1-\alpha, S_1) \in L(\Gamma)$ is said to be *successful*. By a *two-way n -PC grammar system*, we refer to any two-way k -context-free n -PC grammar system, where $k \geq 1$.

A *two-way right-linear PC component* is a quadruple, $G = (N, T, P, S)$, where N and T are two disjoint alphabets. Symbols in N and T are referred to as *nonterminal* and *terminals*, respectively, and $S \in N$ is the *start symbol* of G . P is a finite set of *productions* such that each $r \in P$ has one of these forms

$$A \rightarrow xB, \text{ where } x \in T^* \text{ and } A, B \in N \text{ or } A \rightarrow x, \text{ where } A, B \in N \text{ and } x \in T^*.$$

Direct derivation and direct reduction are defined in a *two-way k -context-free PC component*.

A *two-way left-linear PC component* is a quadruple, $G = (N, T, P, S)$, where N and T are two disjoint alphabets. Symbols in N and T are referred to as *nonterminal* and *terminals*, respectively, and $S \in N$ is the *start symbol* of G . P is a finite set of *productions* such that each $r \in P$ has one of these forms

$$A \rightarrow Bx, \text{ where } x \in T^* \text{ and } A, B \in N \text{ or } A \rightarrow x, \text{ where } A, B \in N \text{ and } x \in T^*.$$

Direct derivation and direct reduction are defined in a *two-way k -context-free PC component*.

A (2-CF, RLIN, LLIN) *two-way 3-PC grammar system* has 3 PC component: A *two-way 2-context-free PC component*, a *two-way right-linear PC component* and a *two-way left-linear PC component*.

For a two-way k -context-free PC grammar system, $\Gamma = (G_1, \dots, G_n)$, we next introduce some special notions.

Finite index. Let $\sigma \Rightarrow^* x$ be any successful computation in Γ , where $x \in \Theta$, and let $i \in \{1, \dots, n\}$. By i -index($\sigma \Rightarrow^* x$), we denote the maximum number in $\mathbf{length}\{\mathbf{keep}(\kappa(i-\sigma \Rightarrow^* i-x), N_i)\}$. If for every successful computation $\sigma \Rightarrow^* \xi$ in Γ , where $\xi \in \Theta$, there exists $k \geq 1$ such that i -index($\sigma \Rightarrow^* \xi$) $\leq k$, G_i is of a finite index. If G_i is of a finite index, $\mathbf{index}(G_i)$ denotes the minimum number h satisfying i -index($\sigma \Rightarrow^* \xi$) $\leq h$, for every successful computation $\sigma \Rightarrow^* \varpi$ in Γ , where $\varpi \in \Theta$. By $\mathbf{index}(G_i) = \infty$, we express that G_i is not of a finite index. If G_j is of a finite index for all $j = 1, \dots, n$, Γ is of a finite index and $\mathbf{index}(\Gamma)$

denotes the minimum number g satisfying $index(G_l) \leq g$, for all $l = 1, \dots, n$. By $index(\Gamma) = \infty$, we express that Γ is not of a finite index.

q-Degree. For $\sigma \Rightarrow^* x$ in Γ , where $x \in \Theta$, $q\text{-degree}(\sigma \Rightarrow^* x)$ denotes the number of communication steps ($q \Rightarrow$) in $\sigma \Rightarrow^* x$. If for every computation $\sigma \Rightarrow^* \xi$ in Γ , where $\xi \in \Theta$, there exists $k \geq 1$ such that $q\text{-degree}(\sigma \Rightarrow^* \xi) \leq k$, Γ is of a finite $q\text{-degree}$. If Γ is of a finite $q\text{-degree}$, $q\text{-degree}(\Gamma)$ denotes the minimum number h satisfying $q\text{-degree}(\sigma \Rightarrow^* \xi) \leq h$, for every computation $\sigma \Rightarrow^* \xi$ in Γ ; by $q\text{-degree}(\Gamma) = \infty$, we express that Γ is not of a finite $q\text{-degree}$.

Centralized Version. Γ is *centralized* if no query symbol occurs in any production of P_i in $G_i = (N_i, T_i, P_i, S_i)$, for all $i = 2, \dots, n$. In other words, only P_1 can contain some query symbols, so G_1 , called the *master* of Γ , is the only component that can cause Γ to perform a communication step.

3 MAIN RESULT

This section proves that every non-unary recursively enumerable language is defined by a centralized (2-CF, RLIN, LLIN) two-way 3-PC grammar system, $\Gamma = (\{Q_2, Q_3\}, G_1, G_2, G_3)$, such that $index(G_1) = 2$, $index(G_2) = 1$, $index(G_3) = 1$, and $q\text{-degree}(\Gamma) = 1$. As a result, $index(\Gamma) = 2$. In addition, its three-nonterminal master, G_1 , has only one production containing a query symbols.

Lemma 1. For every recursively enumerable language, L , there exists a left-extended queue grammar, Q , satisfying $L(Q) = L$.

Lemma 2 Let Q' be a left-extended queue grammar. Then, there exists a left-extended queue grammar, $Q = (V, T, W, F, s, R)$, such that $L(Q') = L(Q)$, $W = X \cup Y \cup \{1\}$, where $X, Y, \{1\}$ are pairwise disjoint, and every $(a, b, x, c) \in R$ satisfies either $a \in V - T, b \in X, x \in (V - T)^*, c \in X \cup \{1\}$ or $a \in V - T, b \in Y \cup \{1\}, x \in T^*, c \in Y$.

Lemma 4 Let Q be a left-extended queue grammar such that $card(alph(L(Q))) \geq 2$. Then, there exists a centralized (2-CF, RLIN, LLIN) two-way 3-PC grammar system, $\Gamma = (\{Q_2, Q_3\}, G_1, G_2, G_3)$, such that $L(\Gamma) = L(Q)$, $index(G_1) = 2$, $index(G_2) = 1$, $index(G_3) = 1$, $index(\Gamma) = 2$, $q\text{-degree}(\Gamma) = 1$. In addition, Γ 's master, $G_1 = (Q \cup N_1, T, P_1, S_1)$, satisfies $card(N_1) = 3$ and $q\text{-}P_1 = \{A \rightarrow YQ_2Q_3Y\}$.

Proof. Let $Q = (V, T, W, F, s, R)$ be a left-extended queue grammar such that $card(alph(L(Q))) \geq 2$. Assume that $\{0, 1\} \subseteq alph(L(Q)) \cap T$. Furthermore, without any loss of generality, assume that Q satisfies the properties described in Lemma 2 and Corollary 3. Observe that there exist a positive integer, n , and an injection, ι , from VW to $(\{0, 1\}^n - 1^n)$ so that ι remains an injection when its domain is extended to $(VW)^*$ in the standard way (after this extension, ι thus represents an injection from $(VW)^*$ to $(\{0, 1\}^n - 1^n)^*$), a proof of this observation is simple and left to the reader. Based on ι , define the substitution, ν , from V to $(\{0, 1\}^n - 1^n)$ as $\nu(a) = \{\iota(aq) : q \in W\}$ for every $a \in V$. Extend the domain of ν to V^* . Furthermore, define the substitution, μ , from W to $(\{0, 1\}^n - 1^n)$ as $\mu(q) = \{\mathbf{reversal}(\iota(aq)) : a \in V\}$ for every $q \in W$. Extend the domain of μ to W^* . Set $o = 1^n$.

Construction. Introduce the centralized (2-CF, RLIN, LLIN) two-way 3-PC grammar system, $\Gamma = (\{Q_2, Q_3\}, G_1, G_2, G_3)$, where $G_1 = (Q \cup N_1, T, P_1, S_1)$, $G_2 = (N_2, T, P_2, S_2)$, $G_3 =$

(N_3, T, P_3, S_3) , $N_1 = \{S_1, A, Y\}$, and $P_1 = \{S_1 \rightarrow oAo, S_1 \rightarrow oYo, A \rightarrow YQ_2Q_3Y\} \cup \{A \rightarrow \text{reversal}(x)Ax \mid x \in \iota(VW)\} \cup \{Y \rightarrow xYx \mid x \in \iota(VW)\}$. P_2 and P_3 are constructed as follows

1. if $s = a_0q_0$, where $a_0 \in V - T$ and $q_0 \in W - F$,
then add $S_2 \rightarrow u\langle q_0, 1 \rangle$ to P_2 and $S_3 \rightarrow \langle q_0, 1 \rangle t$ to P_3 , for all $u \in v(a_0)$ and $t \in \mu(q_0)$,
2. if $(a, q, y, p) \in R$, where $a \in V - T$, $p, q \in W - F$, and $y \in (V - T)^*$,
then add $\langle q, 1 \rangle \rightarrow u\langle p, 1 \rangle$ to P_2 and $\langle q, 1 \rangle \rightarrow \langle p, 1 \rangle t$ to P_3 , for all $u \in v(y)$ and $t \in \mu(p)$,
3. for every $q \in W - F$, add $\langle q, 1 \rangle \rightarrow o\langle q, 2 \rangle$ to P_2 and $\langle q, 1 \rangle \rightarrow \langle q, 2 \rangle$ to P_3 ,
4. if $(a, q, y, p) \in R$, where $a \in V - T$, $p, q \in W - F$, $y \in T^*$,
then add $\langle q, 2 \rangle \rightarrow y\langle p, 2 \rangle$ to P_2 , $\langle q, 2 \rangle \rightarrow \langle p, 2 \rangle t$ to P_3 , for all $t \in \mu(p)$,
5. if $(a, q, y, p) \in R$, where $a \in V - T$, $q \in W - F$, $y \in T^*$, and $p \in F$,
then add $\langle q, 2 \rangle \rightarrow y$ to P_2 , $\langle q, 2 \rangle \rightarrow o$ to P_3 ,

and N_2 and N_3 contains all symbols occurring in P_2 and P_3 , respectively, that are not in T .

Theorem 5 Let L be a recursively enumerable language such that $\text{card}(\text{alph}(L)) \geq 2$. Then, there exists a centralized (2-CF, RLIN, LLIN) two-way 3-PC grammar system, $\Gamma = (\{Q_2, Q_3\}, G_1, G_2, G_3)$, such that $L(\Gamma) = L$, $\text{index}(G_1) = 2$, $\text{index}(G_2) = 1$, $\text{index}(G_3) = 1$, $\text{index}(\Gamma) = 2$, $q\text{-degree}(\Gamma) = 1$, and Γ 's master, $G_1 = (Q \cup N_1, T, P_1, S_1)$, satisfies $\text{card}(N_1) = 3$, $q\text{-}P_1 = \{A \rightarrow YQ_2Q_3Y\}$.

REFERENCES

- [1] Meduna, A.: Two-Way Metalinear PC Grammar Systems and Their Descriptive Complexity, Acta Cybernetica vol. 2003, No. 4, US, 2003, ISBN 0324-721X
- [2] Vaszil, G.: On simulating Non-returning PC grammar systems with returning systems, Theoretical Computer Science (209) 1-2, 319-329, 1998
- [3] Santean, L.: Parallel Communicating Systems, EATCS Bulletin, 160-171, 1990
- [4] Paun, Gh., Santean, L.: Further Remars about parallel communicating grammar systems, International Journal of Computer Mathematics 34, 187-203, 1990
- [5] Paun, Gh., Salomaa, A., Vicolov, S.: On the generative capacity of parallel communicating grammar systems. International Journal of Computer Mathematics 45, 45-59, 1992
- [6] Csuhaaj-Varju, E., Dassow, J., Kelemen, J., Paun, Gh.: Eco-Grammar Systems: A grammatical framework for life-like interactions, Artificial Life 3, 27-38, 1996
- [7] Csuhaaj-Varju, E.: Cooperating grammar systems. Power and Parameters, LNCS 812, Springer, Berlin, 67-84, 1994
- [8] Csuhaaj-Varju, E., Kelemen, J.: On the Power of Cooperation: a regular Representation of R.E. Languages, Theor. Computer Sci. 81, 305-310, 1991