TWO–WAY LINEAR PC GRAMMAR SYSTEMS AND THEIR DESCRIPTIONAL COMPLEXITY

Ing. Petr KALÁB, Doctoral Degree Programme (3)
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 five 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.

1 INTRODUCTION

The formal language theory has intensively investigated various variants of PC grammar systems (see [10]), 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 vise versa.

This paper discusses the centralized form of two-way linear 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 descriptional complexity because this complexity represents an intensively studied area of today's formal language theory.

This paper proves that the centralized two-way linear 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 two-way linear PC grammar system with one linear component and four regular 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 n be a positive integer. A *component of two-way PC grammar system* 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 this form

$A \rightarrow x$, where $x \in (T \cup N)^*$ and $A \in N$.

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

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

where $Q = \{q_i : i = 1, \ldots, n\}$, whose members are called *query symbols*, and for all $i = 1, \ldots, n$, $G_i = (Q \cup N_i, T, P_i, S_i)$ is a component of two-way PC grammar system such that $Q \cap (N_i \cup T) = \varnothing$ (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 $(x_1, \ldots, x_n)$, where $x_i \in (Q \cup N_i \cup T)^*$, $1 \leq i \leq n$. The start configuration, σ, is defined as $\sigma = (S_1, \ldots, S_n)$. Let Θ denote the set of all configurations of Γ. For evey $x \in \Theta$ and $i = 1, \ldots, n$, $i\text{-}x$ denote its ith component – that is, if $x = (x_1, \ldots, x_i, \ldots, x_n)$, 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 ${}_x\theta(i\text{-}x) = z_1 z_2 \ldots z_{|i\text{-}x|}$ where for all $1 \leq h \leq |i\text{-}x|$, if for some $q_j \in Q$, $i = 1, \ldots, n$, $\mathbf{sym}(i\text{-}x, h) = q_j$ and $\mathbf{alph}(j\text{-}x) \cap Q = \varnothing$, 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 $\neq \varnothing$), $z_h = \mathbf{sym}(i\text{-}x, h)$.

- $y \Rightarrow_d x$ in G if $i\text{-}y \Rightarrow_d i\text{-}x$ in G_i or $i\text{-}y = i\text{-}x$ with $i\text{-}y, i\text{-}x \in T^*$, for all $i = 1, \ldots, n$;
- $y \Rightarrow_r x$ in G if $i\text{-}y \Rightarrow_r i\text{-}x$ in G_i or $i\text{-}y = i\text{-}x$ with $i\text{-}y, i\text{-}x \in \{S_i\} \cup T^*$, for all $i = 1, \ldots, n$;
- $y \Rightarrow_q x$ in G if $i\text{-}x = \theta(i\text{-}y)$ in G for all $i = 1, \ldots, n$.

Informally, Γ works in three computational modes – $\Rightarrow_d$, $\Rightarrow_r$, $\Rightarrow_q$, which symbolically represent a direct *derivation*, *reduction*, and *communication*, respectively. Let $l \geq 1$, $\alpha_j \in \Theta$, $1 \leq i \leq l$, and $\alpha_0 \Rightarrow_{l1} \alpha_1 \Rightarrow_{l2} \alpha_2 \ldots \alpha_{l-1} \Rightarrow_{ll} \alpha_l$, where $l_m \in \{d, r, q\}$, $1 \leq m \leq l$; write $\alpha_0 \Rightarrow^* \alpha_l$ 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 vise versa; after making a derivation or reduction step, it does not. Consider $\alpha_0 \Rightarrow^* \alpha_l$ that consists of l direct computational steps, $\alpha_0 \Rightarrow_{l1} \alpha_1 \Rightarrow_{l2} \alpha_2 \ldots \alpha_{l-1} \Rightarrow_{ll} \alpha_l$, satisfying the above properties. Set $\kappa(\alpha_0 \Rightarrow^* \alpha_l) = \{\alpha_0, \alpha_1, \ldots, \alpha_l\}$; that is, $\kappa(\alpha_0 \Rightarrow^* \alpha_l)$

denotes the set of all configurations occurring in $\alpha_0 \Rightarrow^* \alpha_l$. Furthermore, for each $l = 1, \ldots, n$, set $\kappa(i\text{–}\alpha_0 \Rightarrow^* i\text{–}\alpha_l) = \{ i\text{–}\beta\text{: } \beta \in \kappa(\alpha_0 \Rightarrow^* \alpha_l)\}$. Finally, for each $h = 1, \ldots, n$, *h-computation*$(i\text{–}\alpha_0 \Rightarrow^* i\text{–}\alpha_l)$ denotes $h\text{-}\alpha_0 \ {}_{l1}\Rightarrow\ h\text{-}\alpha_1 \ {}_{l2}\Rightarrow\ h\text{-}\alpha_2 \ldots h\text{-}\alpha_{l-1} \ {}_{ll}\Rightarrow\ h\text{-}\alpha_l$. The *language of* Γ, $L(\Gamma)$, is defined as

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

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-α results in z. A computation $\sigma \Rightarrow^* \alpha$ in Γ with $\mathbf{del}(1\text{-}\alpha, S_1) \in L(\Gamma)$ is said to be *successful*.

The components of the *linear two-way PC grammar system* are simple linear grammars.

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

Finite index. Let $\sigma \Rightarrow^* x$ be any successful computation in Γ, where $x \in \Theta$, and let $i \in \{1, \ldots, n\}$. By *i-index*$(\sigma \Rightarrow^* x)$, we denote the maximum number in $\mathbf{length}\{\mathbf{keep}(\kappa(i\text{–}\sigma \Rightarrow^* i\text{–}x), N_i)$. 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, *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 *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 $j = 1, \ldots, n$, Γ *is of a finite index* and *index*(Γ) denotes the minimum number g satisfying *index*$(G_l) \leq g$, for all $l = 1, \ldots, n$. 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-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-degree*$(\sigma \Rightarrow^* \xi) \leq k$, Γ *is of a finite q-degree*. If Γ is of a finite q-degree, *q-degree*(Γ) denotes the minimum number h satisfying *q-degree*$(\sigma \Rightarrow^* \xi) \leq h$, for every computation $\sigma \Rightarrow^* \xi$ in Γ; by *q-degree*$(\Gamma) = \infty$, we express that Γ is not of a finite q-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 $i = 2, \ldots, n$. 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 two-way linear 5-PC grammar system, $\Gamma = (\{Q_2, Q_3, Q_4, Q_5\}, G_1, G_2, G_3, G_4, G_5)$ such that *index*$(G_1) = 2$, *index*$(G_2) =$ *index*$(G_3) =$ *index*$(G_4) =$ *index*$(G_5) = 1$, and *q-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. Moreover G_2, G_3, G_4 and G_5 are regular components or grammars.

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

Proof. Recall that every recursively enumerable language is generated by a queue grammar (see [2]). Clearly, for every queue grammar, there exists an equivalent left-extended

queue grammar. Thus, this lemma holds. ■

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 $a \in V - T$, $b \in Y \cup \{1\}$, $x \in T^*$, $c \in Y$.

Proof: See Lemma 1 in [3]. ■

Lemma 3 Let Q be a left-extended queue grammar such that $card(\mathbf{alph}(L(Q))) \geq 2$. Then, there exists a centralized linear two-way 5-PC grammar system, $\Gamma = (\{Q_2, Q_3, Q_4, Q_5\}, G_1, G_2, G_3, G_4, G_5)$, such that $L(\Gamma) = L(Q)$, $index(G_1) = 2$, $index(G_2) = index(G_3) = index(G_4) = index(G_5) = 1$, $index(\Gamma) = 2$. In addition, Γ's master, $G_1 = (\{Q_2, Q_3, Q_4, Q_5\} \cup N_1, T, P_1, S_1)$, satisfies $card(N_1) = 3$ and $q\text{-}P_1 = \{A \rightarrow Q_4Q_2Q_3Q_4Q_5\}$.

Proof. Let $Q = (V, T, W, F, s, R)$ be a left-extended queue grammar such that $card(\mathbf{alph}(L(Q))) \geq 2$. Assume that $\{0, 1\} \subseteq \mathbf{alph}(L(\Gamma)) \cap T$. 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 represent 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 a centralized two-way linear 5-PC grammar system, $\Gamma = (\{Q_2, Q_3, Q_4, Q_5\}, G_1, G_2, G_3, G_4, G_5)$, where $G_1 = (Q \cup N_1, T, P_1, S_1)$, $G_2 = (N_2, T, P_2, S_2)$, $G_3 = (N_3, T, P_3, S_3)$, $G_3 = (N_3, T, P_3, S_3)$, $G_4 = (N_4, T, P_4, S_4)$, $G_5 = (N_5, T, P_5, S_5)$, $N_1 = \{S_1, A_1, Y\}$, $P_1 = \{S_1 \rightarrow oA_1, S_1 \rightarrow oYo, A_1 \rightarrow Q_4Q_2Q_3Q_4Q_5\} \cup \{A_1 \rightarrow \mathbf{reversal}(x)A_1 : x \in \iota(VW)\} \cup \{Y \rightarrow xYx : x \in \iota(VW)\}$, $N_4 = \{S_4, Y\}$, $P_4 = \{S_4 \rightarrow Y, Y \rightarrow Y\}$, $N_5 = \{S_5, A_5\}$ and $P_5 = \{S_5 \rightarrow A_5o, A_5 \rightarrow \varepsilon\} \cup \{A_5 \rightarrow A_5x : x \in \iota(VW)\}$. 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 $u \in \nu(a_0)$ 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 $u \in \nu(y)$ 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 $card(\mathbf{alph}(L))) \geq 2$. Then, there exists a centralized two-way linear 5-PC grammar system, $\Gamma = (\{Q_2, Q_3, Q_4, Q_5\},$

G_1, G_2, G_3, G_4, G_5), such that $L(\Gamma) = L(Q)$, $index(G_1) = 2$, $index(G_2) = index(G_3) = index(G_4) = index(G_5) = 1$, $index(\Gamma) = 2$, $q\text{-}degree(\Gamma) = 1$, and Γ's master, $G_1 = (\{Q_2, Q_3, Q_4, Q_5\} \cup N_1, T, P_1, S_1)$ satisfies $q\text{-}P_1 = \{A \rightarrow Q_4Q_2Q_3Q_4Q_5\}$ and $card(N_1) = 3$.

Proof. This theorem follows from Lemmas 1, 2, and 3. ■

Interesting properties. Observe other properties of investigated two-way linear PC grammar system. Examined two-way linear PC grammar system is consisted of five components. Four are regular grammars and only one component is linear grammar. Furthermore in derivation phase of computation are used only regular productions and in reduction phase of computation are used linear productions.

ACKNOWLEDGMENTS

This work was supported by the GAČR grant 201/04/0441.

REFERENCES

[1] Csuhaj-Varju, E.: Cooperating grammar systems. Power and Parameters, LNCS 812, Springer, Berlin, 67-84, 1994

[2] Kleijn, H. C. M., Rozenberg, G.: On the Generative Power of Regular Pattern Grammars, Acta Informatica 20, 391-411, 1983

[3] Meduna, A.: Simultaneously One-Turn Two-Pushdown Automata, International Journal of Computer Mathematics 80, 679-687, 2003.

[4] Meduna, A.: Two-Way Metalinear PC Grammar Systems and Their Descriptional Complexity, Acta Cybernetica 2003, US, s. 126-137, ISSN 0324-721X

[5] Kaláb, P.: A Two-Way PC Grammar Systems Based on Regular Grammars, In: Proceedings of 10th Conference and Competition STUDENT EEICT 2004, Brno, CZ, FIT VUT, 2004, s. 252-256, ISBN 80-214-2635-7

[6] Vaszil, G.: On simulating Non-returning PC grammar systems with returning systems, Theoretical Computer Science (209) 1-2, 319-329, 1998

[7] Salomaa, A.: Formal Languages, Academic Press, New York, 1973

[8] Paun, Gh. and Santean, L.: Further Remars about parallel communicating grammar systems, International Journal of Computer Mathematics 34, 187-203, 1990

[9] Paun, Gh. and Santean, L.: Parallel communicating grammar systems: the regular case, Ann. Univ. Buc., Ser. Matem.-Inform. 38, 55-63, 1989

[10] Csuhaj-Varju, E., Dassow, J., Kelemen, J., Paun, Gh.: Grammar Systems: A Grammatical Approach to Distribution and Cooperation, Gordon and Breach, London, 1994