String-Partitioning Systems and An Infinite Hierarchy

Zbyněk Křivka *
krivka@fit.vutbr.cz

Rudolf Schönecker *
schonec@fit.vutbr.cz

Abstract: This paper introduces and discusses string-partitioning systems. This formalization consists of partitioning the rewritten string into several parts, which the systems rewrite by rules that specify to which part they are applied. Based on the number of parts, the present paper establishes an infinite hierarchy of language families that coincides with the hierarchy resulting from the programmed grammars of finite index, so these systems actually represent a counterpart to these grammars. In its conclusion, this paper suggests some open problem areas.

Keywords: string-partitioning systems; programmed grammars; finite index; infinite hierarchy.

1 Introduction

As opposed to classical formal models, which are classifiable under their main properties into two groups - generative grammars or accepting automata, string-partitioning systems is a formal model, that has properties from both - it is an accepting device that uses states to control its computation.

However, M works with strings divided into several parts by a special bounder symbol, $\#$. During each computational step, the system rewrittes an occurrence of $\#$ with a string, possibly containing other $\#$s, and, thereby, rearrange the string division. If used in this way, starting from $\#$, M yields a string x containg no $\#$, x is in the language of M.

Based on this simple rewriting mechanism, we demonstrate that these automata give rise to an infinite hierarchy of language families based on the number of parts of the rewritting strings. More precisely, the systems that divide their strings into no more than n parts are less powerful than the systems that make this division up to $n + 1$ parts, for all $n \geq 1$. In addition, we demonstrate that this hierarchy coincides with the hierarchy resulting from the programmed grammars of index n, for all $n \geq 1$ (see analogy with matrix grammars of finite index – page 160 in [1]). In this sense, the string-partitioning systems represent a counterpart to these grammars, which has lacked any automata counterpart of this kind so far; in this sense, the present paper fills this gap.

In its conclusion, this paper suggests some variants of string-partitioning systems to study in the future.

2 Preliminaries

This paper assumes that the reader is familiar with the formal language theory (see [2]). For a set, Q, $card(Q)$ denotes the cardinality of Q. For an alphabet, V, V^* represents the free monoid

* Department of Information Systems, Faculty of Information Technology, Brno University of Technology, Božetěchova 2, 612 66 Brno, Czech Republic

generated by V under the operation of concatenation. The identity of V^* is denoted by ε. Set $V^+ = V^* - \{\varepsilon\}$; algebraically, V^+ is thus the free semigroup generated by V under the operation of concatenation. For $w \in V^*$, $|w|$ denotes the length of w, and for $W \subseteq V$, $occur(w, W)$ denotes the number of occurrences of symbols from W in w and $sym(w, i)$ denotes the i-th symbol of w; for instance, $sym(abcd, 3) = c$.

A *context-free grammar* is a quadruple, $G = (V, T, P, S)$, where V is a total alphabet, $T \subseteq V$ is an alphabet of terminals, $S \in (V - T)$ is the start symbol, and P is a finite set of *rules* of the form $q\colon A \to v$, where $A \in (V - T)$, $v \in V^*$ and q is a label of this rule. If $q\colon A \to v \in P$, $x, y \in V^*$, G makes a derivation step from xAy to xvy according to $q\colon A \to v$, symbolically written as $xAy \Rightarrow xvy\ [q\colon A \to v]$ or, simply, $xAy \Rightarrow xvy$. In the standard manner, we define $\Rightarrow^m$, where $m \geq 0$, $\Rightarrow^+$, and $\Rightarrow^*$. The *language of* G, $L(G)$, is defined as $L(G) = \{w \in T^* \mid S \Rightarrow^* w\}$. A language, L, is *context-free* if and only if $L = L(G)$, where G is a context-free grammar.

For $p \in P$, $rhs(p)$ and $lhs(p)$ denotes right-side and left-side handle of rule p, respectively, $lab(p)$ denotes label of rule p and for set of rules P, $lab(P)$ denotes set of all labels of rules from P.

A *programmed grammar* (see page 28 in [1]) is a quadruple, $G = (V, T, P, S)$, where V is a total alphabet, $T \subseteq V$ is an alphabet of terminals, $S \in (V - T)$ is the start symbol, and P is a finite set of rules of the form $q\colon A \to v, g(q)$, where $q\colon A \to v$ is a context free rule labeled by q and $g(q)$ is a set of rule labels associated with this rule. After an application of a rule of this form in an ordinary context way, in the next step a rule labeled by a label from $g(q)$ has to be applied. Thus G makes a derivation step, symbolically denoted by $\Rightarrow$, by analogy with a context-free grammar. In the standard manner, we define $\Rightarrow^m$, where $m \geq 0$, $\Rightarrow^+$, and $\Rightarrow^*$. The language of G, $L(G)$, is defined as $L(G) = \{w \in T^* \mid S \Rightarrow^* w\}$.

Let G be a programmed grammar, and let T, and S be its terminal alphabet, and axiom, respectively. For a derivation $D\colon S = w_1 \Rightarrow w_2 \Rightarrow \cdots \Rightarrow w_r = w \in T^*$, where $r > 1$, according to G, we set $Ind(D, G) = \max\{occur(w_i, V - T) \mid 1 \leq i \leq r\}$, and, for $w \in T^*$, we define $Ind(w, G) = \min\{Ind(D, G) \mid D$ is a derivation for w in $G\}$. The *index of grammar* (see page 151 in [1]) G is defined as $Ind(G) = \sup\{Ind(w, G) \mid w \in L(G)\}$. For a language L in the family $\mathcal{L}(X)$ of languages generated by grammars of some type X, we define $Ind_X(L) = \inf\{Ind(G) \mid L(G) = L, G$ is of type $X\}$. For a family $\mathcal{L}(X)$, we set $\mathcal{L}_n(X) = \{L \mid L \in \mathcal{L}(X)$ and $Ind_X(L) \leq n\}$, $n \geq 1$ and $\mathcal{L}_{fin}(X) = \bigcup\limits_{n \geq 1} L_n(X)$.

3 Definitions

Let I be a set of positive integers $\{1, 2, \ldots, k\}$. A *string-partitioning system* is a quadruple $H = (Q, \Sigma, s, R)$, where Q is a finite set of states, Σ is an alphabet containing a special symbol, $\#$, called a *bounder*, $s \in Q$ is a start state and $R \subseteq Q \times I \times \{\#\} \times Q \times \Sigma^*$ is a finite relation whose members are called *rules*. A rule $(q, n, \#, p, x) \in R$, where $n \in I$, $q, p \in Q$ and $x \in \Sigma^*$, is written as $r\colon q\,{}_n\# \to px$ hereafter, where r is unique label and can be omitted.

A *configuration* x of H is a string $x \in Q(\Sigma \cup \{\#\})^*$.

H makes a *derivation step* from $pu\#v$ to $quxv$ by using $r\colon p\,{}_n\# \to qx$, where $occur(u, \#) = n - 1$, symbolically written $pu\#v \Rightarrow quxv\ [r]$ in H or simply $pu\#v \Rightarrow quxv$.

Let $\Rightarrow^*$ denote the transitive and reflexive closure of $\Rightarrow$. The *language derived* by H, $L(H)$, is defined as

$$L(H) = \{w \mid s\# \Rightarrow^* qw,\ q \in Q, w \in (\Sigma - \{\#\})^*\}.$$

A string-partitioning system H is of index k, if for every configuration qx, $s\# \Rightarrow^* qx$ holds $occur(x, \#) \leq k$.

Example 1. $H = (\{s, p, q, f\}, \{a, b, c, \#\}, s, R)$, where R contains:

1. $s\ {}_1\# \to p\ \#\#$
2. $p\ {}_1\# \to q\ a\#b$
3. $q\ {}_2\# \to p\ \#c$
4. $p\ {}_1\# \to f\ ab$
5. $f\ {}_1\# \to f\ c$

$L(H) = \{a^n b^n c^n \mid n \geq 1\}$, holds that $Ind(H) = 2$.
Example of a derivation resulting string $aaabbbccc$: $s\# \Rightarrow p\#\#$ [1] $\Rightarrow qa\#b\#$ [2] $\Rightarrow$ $pa\#b\#c$ [3] $\Rightarrow qaa\#bb\#c$ [2] $\Rightarrow paa\#bb\#cc$ [3] $\Rightarrow faaabbb\#cc$ [4] $\Rightarrow faaabbbccc$ [5].

Let $\mathcal{L}_k(SPS)$, and $\mathcal{L}_k(P, CF)$ denote the families of languages derived by string-partitioning systems, and programmed languages of index k, $k \geq 1$, based on context-free grammar, respectively.

4 Results

This section establishes an infinite hierarchy of language families resulting from the string-partitioning systems defined in the previous section.

Lemma 1. *For every $k \geq 1$, $\mathcal{L}_k(P, CF) \subseteq \mathcal{L}_k(SPS)$*
Let $k \geq 1$. For every programmed grammar of index k, G, there is a string-partitioning system of index k, H, such that $L_k(G) = L_k(H)$.

Construction. Let $k \geq 1$ be a positive integer. Let $G = (V, T, P, S)$ is programmed grammar of index k, where $N = V - T$. Introduce the string-partitioning system of index k, $H = (Q, T \cup \{\#\}, s, R)$, where $\# \notin T$, $s = \langle\sigma\rangle$, σ is a new symbol, R and Q are constructed by performing the following steps:

1. For each $p\colon S \to \alpha \in P$, $\alpha \in V^*$, add $\langle\sigma\rangle_1\# \to \langle[p]\rangle\#$ to R, $\langle[p]\rangle$ is new state in Q
2. If $A_1A_2 \dots A_j \dots A_h \in N^*$, $h \in \{1, 2, \dots, k\}$, $p\colon A_j \to x_0B_1x_1B_2x_2 \dots x_{n-1}B_nx_n$, $g(p) \in P$, $j \in \{1, 2, \dots, h\}$ for $n \geq 0$, $x_0, x_t \in T^*$, $B_t \in N$, $1 \leq t \leq n$ and $n + h - 1 \leq k$, then
   (a) if $g(p) = \emptyset$, then $\langle A_1A_2 \dots A_{j-1}[p]A_{j+1} \dots A_h\rangle$, $\langle A_1A_2 \dots B_1 \dots B_n \dots A_h\rangle$ are new states in Q and the rule $\langle A_1A_2 \dots A_{j-1}[p]A_{j+1} \dots A_h\rangle_j\# \to \langle A_1A_2 \dots B_1 \dots B_n \dots A_h\rangle$ $x_0\#x_1 \dots x_{n-1}\#x_n$ is added to R
   (b) for every $q \in g(p)$, $q\colon D_d \to \alpha$, $\alpha \in V^*$ add new states $\langle A_1A_2 \dots A_{j-1}[p]A_{j+1} \dots A_h\rangle$ and $\langle D_1D_2 \dots [q] \dots D_{n+h-1}\rangle$ to Q and add the following rule to R:
   $\langle A_1A_2 \dots A_{j-1}[p]A_{j+1} \dots A_h\rangle_j\# \to \langle D_1D_2 \dots [q] \dots D_{n+h-1}\rangle x_0\#x_1 \dots x_n$,
   where $A_1 \dots A_{j-1}B_1 \dots B_nA_{j+1} \dots A_h = D_1 \dots D_{h+n-1}$, $B_1 \dots B_n = D_j \dots D_{j+n-1}$ for some $d \in \{1, 2, \dots, n + h - 1\}$.

Basic Idea. The information necessary for the simulation is recorded inside of states. Every Q's state label carries string of nonterminals from N^* where one symbol of this string is replaced by P_G's rule label.

Let us have a configuration $x_0A_1x_1 \dots x_{h-1}A_hx_h$ in some programmed grammar $G = (N, T, P, S)$ of index k, where $x_i \in T^*$ for $0 \leq i \leq h \leq k$ and $A_l \in N$ for $1 \leq l \leq h$, and let $p\colon A_j \rightarrow \alpha$ is applicable in the next step to the nonterminal A_j, $1 \leq j \leq h$.

Then, new configuration of equivalent string-partitioning system H is of the form $\langle A_1A_2 \dots A_{j-1}[p]A_{j+1} \dots A_h\rangle x_0\#x_1 \dots x_{n-1}\#x_h$ and encodes simulated nonterminals in G's sentential form and next applicable rule label.

Claim 2 *If $S \Rightarrow^m x_0A_1x_1A_2x_2 \dots x_{n-1}A_hx_h$ in G, then $\langle\sigma\rangle\# \Rightarrow^r \langle A_1A_2 \dots A_h\rangle x_0\#x_1 \dots x_h$ $[q_1q_2 \dots q_r]$ in H, for $m \geq 0, r \geq 1$. For $g(q_r) \neq \emptyset$ exists a rule $q_{r+1}\colon A_j \rightarrow y_0B_1y_1 \dots y_{h-1}B_ny_n$, $n + h - 1 \leq k$, $q_{r+1} \in g(q_r)$ and $A_j = [q_{r+1}]$, $q_1, \dots, q_r, q_{r+1} \in lab(R)$.*

Proof. This claim is established by induction on m.

Basis: Let $m = 0$. For $S \Rightarrow^0 S$ in G there exists $\langle\sigma\rangle\# \Rightarrow^1 \langle[p]\rangle\#$ in H, where $p\colon S \rightarrow \alpha \in P$ and $\langle\sigma\rangle_1\# \Rightarrow \langle[p]\rangle\# \in R$.

Induction Hypothesis: Suppose that Claim 2 holds for all derivations of length m or less for some $m \geq 0$.

Induction Step: Consider $S \Rightarrow^m y\ [p_1p_2 \dots p_m]$, where $y = x_0A_1x_1 \dots x_{n-1}A_hx_h$ and $p_1, \dots, p_m$, $p_{m+1} \in lab(P)$ so that $y \Rightarrow x\ [p_{m+1}]$. If $m = 0$, then $p_{m+1} \in \{p \mid lhs(p) = S, p \in lab(P)\}$ otherwise $p_{m+1} \in g(p_m)$. For $p_{m+1}\colon A_j \rightarrow y_0B_1y_1 \dots y_{n-1}B_ny_n$ is x in the form: $x = x_0A_1x_1 \dots A_{j-1}x_{j-1}y_0B_1y_1 \dots y_{n-1}\ B_ny_nx_jA_{j+1} \dots x_{h-1}A_hx_h$, for $x_0, \dots, x_h \in T^*$ and $y_0, \dots, y_n \in T^*$. Based on the induction hypothesis, there exists the derivation $\langle\sigma\rangle\# \Rightarrow^r$ $\langle A_1A_2 \dots A_{j-1}[p_{m+1}]A_{j+1} \dots A_h\rangle x_0\#x_1 \dots x_{h-1}\#x_h\ [q_1q_2 \dots q_r] \Rightarrow$ $\langle A_1A_2 \dots A_{j-1}B_1 \dots B_nA_{j+1} \dots A_h\rangle x_0\# \dots \#x_{j-1}y_0\# \dots \#y_nx_j\# \dots \#x_h\ [q_{r+1}]$, $r \geq 1$, $q_i \in lab(R)$, $1 \leq i \leq r + 1$. If $g(p_{m+1}) \neq \emptyset$, then exists a rule $p_{m+2} \in g(p_{m+1})$ and a sequence $D_1D_2 \dots D_{n+h-1}$ so that $A_1A_2 \dots A_{j-1}B_1 \dots B_nA_{j+1} \dots A_h = D_1D_2 \dots D_{n+h-1}$, where for at most one $d \in \{1, 2, \dots, n + h - 1\}$ is $D_d = [q_{r+2}]$, $q_{r+2} \in g(q_{r+1})$.

Claim 3 *If $S \Rightarrow^z x$ in G, then $\langle\sigma\rangle\# \Rightarrow^* \langle\rangle x$ in H for some $z \geq 0$, $x \in T^*$.*

Proof. Consider Claim 2 with $h = 0$. At this point, if $S \Rightarrow^z x_0$, then $\langle\sigma\rangle\# \Rightarrow^* \langle\rangle x_0$ and so $x_0 = x$.

Lemma 4. *For every $k \geq 1$, $\mathcal{L}_k(SPS) \subseteq \mathcal{L}_k(P, CF)$*
Let $k \geq 1$. For every string-partitioning system of index k, H, exists equivalent programmed grammar of index k, G, such that $L_k(G) = L_k(H)$.

Construction. Let $k \geq 1$ be a positive integer. Let $H = (Q, T \cup \{\#\}, s, R)$ is string-partitioning system of index k, where $\Sigma = T \cup \{\#\}$. Introduce the programmed grammar of index k, $G = (V, T, P, S)$, where the sets of nonterminals $N = V - T$ and rules P are constructed as follows:

1. $P = \emptyset$,
2. $S = \langle s, 1, 1\rangle$,
3. $N = \{\langle p, i, h\rangle \mid p \in Q, 1 \leq i \leq k, 1 \leq h \leq k, i \leq h\} \cup \{\langle q', i, h\rangle \mid q \in Q, 1 \leq i \leq k, 1 \leq h \leq k, i \leq h\} \cup \{\langle q'', i, h\rangle \mid q \in Q, 1 \leq i \leq k, 1 \leq h, i \leq h \leq k\}$,
4. For every rule $r\colon p\ _i\# \rightarrow qy \in R$, $y = y_0\#y_1 \dots y_{m-1}\#y_m$, $y_0, y_1, y_2 \dots y_m \in T^*$, if $m = 0$, then $h_{max} = k$ else $h_{max} = k - m + 1$, add the following set to P:

(i) $\{\langle p, j, h\rangle \rightarrow \langle q', j, h+m-1\rangle,$
$\{r' \mid$ if $j+1 = i$ then $r'\colon \langle p, i, h\rangle \rightarrow \langle q'', i, h+m-1\rangle$ else $r'\colon \langle p, j+1, h\rangle \rightarrow \langle q', j+1, h+m-1\rangle\ \}$
$\mid 1 \leq j < i, i \leq h \leq h_{max}\}$
$\cup$
(ii) $\{\langle p, i, h\rangle \rightarrow \langle q'', i, h+m-1\rangle,$
$\{r' \mid$ if $i = h$, then $r'\colon \langle q'', i, h+m-1\rangle \rightarrow y_0\langle q', i, h+m-1\rangle y_1\langle q', i+1, h+m-1\rangle y_2 \ldots y_{m-1}\langle q', i+m-1, h+m-1\rangle y_m$ else $r'\colon \langle p, i+1, h\rangle \rightarrow \langle q', i+1+m-1, h+m-1\rangle\}$
$\mid i \leq h \leq h_{max}\}$
$\cup$
(iii) $\{\langle p, j, h\rangle \rightarrow \langle q', j+m-1, h+m-1\rangle,$
$\{r' \mid$ if $j = h$, then $r'\colon \langle q'', i, h+m-1\rangle \rightarrow y_0\langle q', i, h+m-1\rangle y_1\langle q', i+1, h+m-1\rangle y_2 \ldots y_{m-1}\langle q', i+m-1, h+m-1\rangle y_m$ else $r'\colon \langle p, j+1, h\rangle \rightarrow \langle q', j+1+m-1, h+m-1\rangle\}$
$\mid i < j \leq h, i \leq h \leq h_{max}\}$
$\cup$
(iv) $\{\langle q'', i, h+m-1\rangle \rightarrow y_0\langle q', i, h+m-1\rangle y_1\langle q', i+1, h+m-1\rangle y_2 \ldots y_{m-1}\langle q', i+m-1, h+m-1\rangle y_m,$
$\{r' \mid r'\colon \langle q', 1, h+m-1\rangle \rightarrow \langle q, 1, h+m-1\rangle\}$
$\mid i \leq h \leq h_{max}\}$
$\cup$
(v) $\{\langle q', j, h+m-1\rangle \rightarrow \langle q, j, h+m-1\rangle,$
$\{r' \mid$ if $j < h+m-1$, then $r'\colon \langle q', j+1, h+m-1\rangle \rightarrow \langle q, j+1, h+m-1\rangle$ else $r'\colon \langle \tilde{p}, 1, h+m-1\rangle \rightarrow \langle \tilde{q}', 1, h+m-1+\tilde{m}-1\rangle$, where $\tilde{p}\ {}_{\tilde{i}}\# \rightarrow \tilde{q}\tilde{y}_0\#\tilde{y}_1 \ldots\ \tilde{y}_{\tilde{m}-1}\#\tilde{y}_{\tilde{m}} \in R$, $\tilde{y}_0, \tilde{y}_1, \ldots, \tilde{y}_{\tilde{m}} \in T^*$, if $\tilde{i} = 1$, then $\tilde{q}' := \tilde{q}''\}$
$\mid 1 \leq j \leq h+m-1, i \leq h \leq h_{max}\}$.

Basic Idea. Inside of every nonterminal $\langle p, i, h\rangle$ in programmed grammar occurring in a sentential form, we record

(1) p–current state of simulated string-partitioning system (the same in first and last simulation stage);
(2) i–the position of the bounder occurrence in the sentential form
(3) h–the total number of all bounders in the simulated sentential form.

From these three pieces of information and the set $g(p)$ associated with p, we find out whether p is applicable in the next step, and if so, we simulate the step by rules introduced in 4^{th} step of the above construction as follows:

(a) inside of all nonterminals in the sentential form, change h to $h+m-1$, where m is the number of nonterminals occurring on the right-hand side of p, so $h+m-1$ is the number of nonterminals after the application of p (see (i) through (iii));
(b) in the nonterminals that follow the rewritten nonterminal, change their position so it corresponds to the position after the application of p (see (iii));
(c) apply p and select a rule label q from p's set of labels $g(p)$ to be applied in the next step (see (iv));
(d) some auxiliary steps in G to finish the simulation of one derivation step from string-partitioning system H (see (v)).

Claim 5 *If $\langle\sigma\rangle\# \Rightarrow^c \langle\vartheta\rangle y_0\#y_1\dots y_{n-1}\#y_n$ in H, then $S \Rightarrow^* y_0A_1y_1\dots y_{n-1}A_ny_n$ in G for some $c \geq 0$.*

Proof. Basis: Let $c = 0$. For $\langle\sigma\rangle\# \Rightarrow^0 \langle\sigma\rangle\#$ in H there exists $S \Rightarrow^0 S$ in G.

Induction Hypothesis: Suppose Claim 5 holds for all derivations of length c or less for some $c \geq 0$.

Induction Step: Consider $\langle\sigma\rangle\# \Rightarrow^c \langle\vartheta\rangle y_0\#y_1\dots y_h\ [r_1r_2\dots r_c]$ in H, $r_t \in lab(R)$, $1 \leq t \leq c$ and r_{c+1}: $\langle\vartheta\rangle_i\# \rightarrow \langle\omega\rangle x_0\#x_1\dots x_{m-1}\#x_m \in R$, $x_0,\dots,x_m \in T^*$ so that $\langle\vartheta\rangle y_0\#\dots\#y_h \Rightarrow \langle\omega\rangle y_0\#y_1\#\dots\#y_{i-1}x_0\#x_1\#\dots\#x_my_i\#y_{i+1}\#\dots\#y_h\ [r_{c+1}]$. Based on Claim 5 there exists also a derivation $D_{1*}: y_0A_1\dots A_hy_h \Rightarrow^* y_0A_1y_1\dots y_{i-1}x_0B_1x_1\dots B_mx_my_iA_{i+1}\dots A_hy_h$ in G. It is shown such a derivation exists based on the construction part of the proof.
Let us have a form $y_0A_1y_1\dots A_hy_h$. Rename nonterminals A_t to $\langle\vartheta,t,h\rangle$ for $1 \leq t \leq h$ and get a base form $y_0\langle\vartheta,1,h\rangle y_1\dots y_{h-1}\langle\vartheta,h,h\rangle y_h$ which starts the simulation of the D_{1*} derivation. This simulation must come out of continuous application of construction's 4^{th} item.

(4i) $\forall j : 1 \leq j < i$ apply rules of the form $\langle p,j,h\rangle \rightarrow \langle q',j,h+m-1\rangle$:
$F_1 = y_0\langle\vartheta,1,h\rangle y_1\dots y_{h-1}\langle\vartheta,h,h\rangle y_h \Rightarrow y_0\langle\omega',1,h+m-1\rangle y_1\langle\vartheta,2,h\rangle y_2\dots y_{h-1}$ $\langle\vartheta,h,h\rangle y_h \Rightarrow^{i-2} y_0\langle\omega',1,h+m-1\rangle y_1\dots y_{i-2}\langle\omega',i-1,h+m-1\rangle y_{i-1}\langle\vartheta,i,h\rangle$ $y_i\dots y_{h-1}\langle\vartheta,h,h\rangle y_h = F_2$

(4ii) apply $\langle p,i,h\rangle \rightarrow \langle q'',i,h+m-1\rangle$:
$F_2 \Rightarrow y_0\langle\omega',1,h+m-1\rangle y_1\dots y_{i-1}\langle\omega'',i,h+m-1\rangle y_i\dots y_{h-1}\langle\vartheta,h,h\rangle = F_3$.
If $i = h$, then $F_4 := F_3$ and continue with [4iv] otherwise with [4iii].

(4iii) $\forall j : i < j \leq h$ apply rules of the form $\langle p,j,h\rangle \rightarrow \langle q',j+m-1,h+m-1\rangle$:
$F_3 \Rightarrow y_0\langle\omega',1,h+m-1\rangle y_1\dots y_{i-2}\langle\omega',i-1,h+m-1\rangle y_{i-1}\langle\omega'',i,h+m-1\rangle y_i$ $\langle\omega',i+m,h+m-1\rangle\ y_{i+1}\langle\vartheta,i+2,h\rangle y_{i+2}\dots y_{h-1}\langle\vartheta,h,h\rangle y_h \Rightarrow^{h-i-1} y_0\langle\omega',1,h+m-1\rangle$ $y_1\dots y_{i-1}\langle\omega'',i,h+m-1\rangle y_{i+1}\dots y_{h-1}\langle\omega',h+m-1,h+m-1\rangle y_h = F_4$

(4iv) apply $\langle q'',i,h+m-1\rangle \rightarrow y_0\langle q',i,h+m-1\rangle y_1\dots y_{m-1}\langle q',i+m-1,h+m-1\rangle y_m$:
$F_4 \Rightarrow y_0\langle\omega',1,h+m-1\rangle y_1\dots y_{i-1}x_0\langle\omega',i,h+m-1\rangle x_1\dots x_{m-1}\langle\omega',i+m-1,h+m-1\rangle x_my_i\dots y_{h-1}\langle\omega',h+m-1,h+m-1\rangle y_h = F_5$

(4v) $\forall j : 1 \leq j \leq h+m-1$ apply rules of the form $\langle q',j,h+m-1\rangle \rightarrow \langle q,j,h+m-1\rangle$:
$F_5 \Rightarrow^{h+m-1} y_0\langle\omega,1,h+m-1\rangle y_1\dots y_{i-1}x_0\langle\omega,i,h+m-1\rangle x_1\dots x_{m-1}\langle\omega,i+m-1,h+m-1\rangle x_my_i\dots y_{h-1}\langle\omega,h+m-1,h+m-1\rangle y_h = F_6$ (Final form)

Rename all nonterminals of the form $\langle\omega,t,h+m-1\rangle$ in F_6 to A_t for $1 \leq t < i$, $\langle\omega,t,h+m-1\rangle$ to B_{t-i+1} for $i \leq t \leq i+m-1$, $\langle\omega,t,h+m-1\rangle$ to A_{t-m+1} for $i+m \leq t \leq h+m-1$. We have obtained $y_0A_1y_1\dots y_{i-1}x_0B_1x_1\dots B_mx_my_iA_{i+1}\dots A_hy_h$.

Claim 6 *If $\langle\sigma\rangle\# \Rightarrow^z \langle\rangle y$ in H, then $S \Rightarrow^* y$ for some $z \geq 0$.*

Proof. This claim follows from Claim 5 with $n = 0$ and $y = y_0$.

Theorem 7. *Infinite hierarchy $\mathcal{L}_k(SPS) \subset \mathcal{L}_{k+1}(SPS)$ holds for every $k \geq 1$.*

Proof. $\mathcal{L}_k(P,CF) = \mathcal{L}_k(SPS)$ follows from Lemma 1 and Lemma 4. Then, Theorem 7 follows from $\mathcal{L}_k(P,CF) = \mathcal{L}_k(SPS)$ and theorem $\mathcal{L}_k(P,CF) \subset \mathcal{L}_{k+1}(P,CF)$, for every $k \geq 1$ which is an analogy to Theorem 3.1.7: $\mathcal{L}_k(M,CF) \subset \mathcal{L}_{k+1}(M,CF)$ in [1], page 161.

5 Conclusion

There was presented a basic variant of the string-partitioning system of index k as a completely new concept of rewriting mechanism. The entire system uses only special symbol (called bounder) and rewriting rules contain an index specifying which bounder in the sentential form will be rewritten. The family of languages generated by string-partitioning systems of index k was described and classified.

We would like to mention here also some other variants of the string-partitioning systems as an open field for the next investigation.

5.1 Deterministic variant

A string-partitioning system of index k is called *deterministic*, if for every two rules $r_1\colon p_1\,{}_i\# \rightarrow q_1x$ and $r_2\colon p_2\,{}_j\# \rightarrow q_2y$ holds if $p_1 = p_2$, then $i \neq j$. We can also mention the *strict deterministic* form, which suppose, that for every two rules is $p_1 \neq p_2$.

5.2 Accepting variant

Let $H = (Q, \Sigma, s, R)$ be a string-partitioning system of index k. H is called *accepting string-partitioning system*, if accepts given language through series of reductions. H makes a *reduction step* from $quxv$ to $pu\#v$ by using $r\colon p\,{}_n\# \vdash qx$, symbolically written $quxv \vDash pu\#v\ [r]$ in H or simply $quxv \vDash pu\#v$. Let $\vDash^*$ denote the transition and reflexive closure of $\vDash$, respectively.

The *language reduced* by H, $L(H)$, is defined as

$$L(H) = \{w \mid qw \vDash^* s\#,\ q \in Q, w \in (\Sigma - \{\#\})^*\}.$$

Consider $M = (\{s, p, q, f\}, \{a, b, c, \#\}, s, R)$ from Example 1. Example of accepting variant is given in reduction of string $aaabbbccc$: $faaabbbccc \vDash faaabbb\#cc\ [5] \vDash paa\#bb\#cc\ [4] \vDash qaa\#bb\#c\ [3] \vDash pa\#b\#c\ [2] \vDash qa\#b\#\ [3] \vDash p\#\#\ [2] \vDash s\#\ [1]$.

5.3 Parallel variant

Let I be a set of positive integers $\{1, 2, \ldots, k\}$. A *parallel string-partitioning system* is a quintuple $H = (Q, \Sigma, s, P, R)$, where Q, Σ and s are defined in the same manner as previously, $P \subseteq I \times \Sigma^*$ is a finite relation containing items called *simple rules*, that are written in the form ${}_n\#\,{}_s\!\rightarrow x$, $n \in I$, $x \in \Sigma^*$, hereafter, $R \subseteq Q \times 2^P \times Q$ is a finite relation with a condition: for each rule $(p, F, q) \in R$, $p, q \in Q$, $F \in 2^P$ holds, that for every two simple rules $c, d \in F$, $c\colon\ {}_{i_c}\#\,{}_s\!\rightarrow x_c$, $d\colon\ {}_{i_d}\#\,{}_s\!\rightarrow x_d$ is $i_c \neq i_d$.

A rule $t = (p_t, \{r_1, \ldots, r_m\}, q_t) \in R$ is applicable to the configuration px, $p \in Q$, $x \in \Sigma^*$, if and only if $p = p_t$, $occur(x, \#) \geq i_j$, for $1 \leq j \leq m$, where $r_j\colon\ {}_{i_j}\#\,{}_s\!\rightarrow y_j$.

H makes a derivation step from pu to qv by using $t = (p, \{r_1, \ldots, r_m\}, q)$, symbolically written $pu\ {}_{pd}\!\Rightarrow qv\ [t]$ in H, if t is applicable to the pu and basic rules $r_1, \ldots, r_m$ are applicated to u and state p is changed onto q.

Consider parallel string-partitioning system of index k as a direct analogy of basic variant of string-partitioning system of index k, then the paragraph descripting the rule-applicability has to be extended with the condition $occur(x, \#) - m + \sum_{l=1}^{m} occur(y_l, \#) \leq k$.

Let ${}_{pd}\!\Rightarrow^*$ denote the transitive and reflexive closure of ${}_{pd}\!\Rightarrow$. The *language derived* by H, $L(H, {}_{pd}\!\Rightarrow)$, is defined as

$$L(H, {}_{pd}\!\Rightarrow) = \{w \mid s\#\ {}_{pd}\!\Rightarrow^* qw,\ q \in Q, w \in (\Sigma - \{\#\})^*\}.$$

Acknowledgements

The authors acknowledge the support of FRVŠ MŠMT grant FR1909/2006/G1.

Bibliography

1. J. Dassow, Gh. Păun: *Regulated Rewriting in Formal Language Theory*, Springer, New York, 1989.
2. A. Meduna: *Automata and Languages, Theory and Applications*, Springer, London, 2000.
3. T. Kasai: A Hierarchy Between Context-Free and Context-Sensitive Languages. In: *Journal of Computer and System Sciences*, volume 4, 1970.