REPORTS ON MATHEMATICAL LOGIC
30 (1996), 83–99

Zdzisław GRODZKI and Jerzy MYCKA

THE NEW CLASSES OF MARKOV-LIKE K-ALGORITHMS

A b s t r a c t This paper continues the line initiated in [1] of uniform formalization of some classes of formal algorithms. The successive new classes $\mathcal{RMA}_k$ and $\overline{\mathcal{RMA}_k}$ of right-hand side Markov-like k-algorithms are introduced. The classes $\mathcal{RMA}_k$ and $\overline{\mathcal{RMA}_k}$ of algorithms are "symmetric" to the classes $\mathcal{MA}_k$ and $\overline{\mathcal{MA}_k}$ of left-hand side Markov-like k-algorithms which have been introduced in [1]. The equivalence of the classes $\mathcal{RMA}_k$ and $\overline{\mathcal{RMA}_k}$ and the class $\mathcal{MNA}$ of Markov normal algorithms is shown here. This implies the closure properties of the above classes under the same operations as of $\mathcal{MNA}$.

1. Introduction

In this paper two new classes $\mathcal{RMA}_k$ and $\overline{\mathcal{RMA}_k}$ of right-hand side Markov-like k-algorithms are introduced and briefly characterized. Every algorithm A of $\mathcal{RMA}_k \cup \overline{\mathcal{RMA}_k}$ is defined, analogously as of $\mathcal{MNA}$, by means of an indexed set $P = \{(x_i, y_i), 1 \leq i \leq n\}$ of productions in an

Received June 19, 1995

alphabet Σ in which some final productions are distinguished. The succession of use of the productions of P to a transformed word $t \in \Sigma^\star$ is the same for algorithms of $\mathcal{RMA}_k$ (and slightly different for algorithms of $\overline{\mathcal{RMA}_k}$) as of $\mathcal{MNA}$. The difference is only im the usage of productions to the transformed words. For algorithms of $\mathcal{RMA}_k$ we replace the j-th right-hand side subword of the left sides of productions, for maximal $j \leq k$, by their right sides (for algorithms of $\mathcal{MA}_k$ we replace the j-th left-hand side subword of the left sides of productions, for maximal $j \leq k$, by their right sides).

For algorithms of $\overline{\mathcal{RMA}_k}$ we choose a production (x_i, y_i) of P with the least index $i \leq |P|$ such that x_i occurs at least k times in a transformed word t. If so then we replace the k-th right-hand side subword of x_i in t by y_i, otherwise we choose a production (x_j, y_j) with the least index $j \leq |P|$, such that x_j occurs in $t \quad (k-1)$ times and so on.

The equivalence of the classes $\mathcal{RMA}_k$, $\overline{\mathcal{RMA}_k}$ and $\mathcal{MNA}$ is shown.

The following reasons motivate the introduction of the classes $\mathcal{RMA}_k$ and $\overline{\mathcal{RMA}_k}$ of algorithms:

(1) We verify a transformed word by algorithms of $\mathcal{RMA}_k$ from the right to the left side. The sentences of some natural languages (Jewish, Chinese, Arabic) are also written from the right to the left side. This idea suggests the introduction of the right-hand side algorithms;

(2) The operations of addition, subtraction and multiplication of positive integers (written in binary notation) are performed from the right to the left side. Therefore the complexity of computations of algorithms of $\mathcal{RMA}_k$ can be - for some problems - less time consuming than of $\mathcal{MNA}$;

(3) Inspiration of the use of the k-th subword of some words in the transformed ones is based on genuine processes (biological, chemical, medical). Accumulation of some attributes implies the growth of organisms (for example illness);

(4) The algorithms of $\mathcal{MA}_k$ and $\mathcal{RMA}_k$ (as well as $\overline{\mathcal{MA}_k}$ and $\overline{\mathcal{RMA}_k}$) differ only in the direction of verification of the transformed words.

Therefore both classes can be used exchangeably with respect to the kind of computations.

2. The designated algorithms

The notion of designated algorithm introduced here is based on the notion of iterative algorithms (Mazurkiewicz [3]) but the formulation is quite different. Mazurkiewicz algorithms are defined by means of relations and they are formal models of programs without procedures. In this paper we use the notion of designated algorithm to define two new classes of right-hand side Markov-like k-algorithms. These classes are "symmetric" to the classes of left-hand side Markov-like k-algorithms introduced in [1].

By *a designated algorithm* DA we mean a tuple $(P, P_i, P_f, V, L, \mathit{Contr}, \mathit{Tr})$, where V is a nonempty set of objects, $P = \{(x_i, y_i) \in V^2 : 1 \leq i \leq n\}$ is an indexed set of productions, P_i and P_f are distinguished (nonempty) subsets of P (not necessarily disjoint) of initial and final productions, $L = \{1, \ldots, n\}$ is the set of labels of P, $\mathit{Contr} : P \times L \times V \mapsto P \times L$ is a partial function, called *a control* of DA and $\mathit{Tr} : P \times L \times V \mapsto V$ is a total function, called *a transformation* of DA.

Informally, we say that a production (x_i, y_i) of designated algorithm DA is effectively used to an object v if $\mathit{Tr}((x_i, y_i), i, v) \neq v$ for $x_i \neq y_i$.

More precisely, a production (x_i, y_i) of a designated algorithm $DA = (P, P_i, P_f, V, L, \mathit{Contr}, \mathit{Tr})$ is said to be *effectively used* to an object v if $\mathit{Contr}((x_i, y_i), i, v)$ is an initial production for $(x_i, y_i) \notin P_f$, otherwise (if $(x_i, y_i) \in P_f$) for algorithm $DA' = (P, P_i, P_f - \{(x_i, y_i)\}, V, L, \mathit{Contr}, \mathit{Tr})$ the result of $\mathit{Contr}((x_i, y_i), i, v)$ is an initial production, too.

A sequence $\mathbf{v} = v_1, v_2, \ldots \in V^\infty$ (finite or infinite) is said to be a computation of a designated algorithm $DA = (P, P_i, P_f, V, L, \mathit{Contr}, \mathit{Tr})$ iff there exists a sequence of productions with their labels (*trace*) of P of the form $\mathbf{p} = ((x_{i_1}, y_{i_1}), i_1), ((x_{i_2}, y_{i_2}), i_2) \ldots$ satisfying the conditions:

(2.1) $l(\mathbf{v}) = l(\mathbf{p}) + 1$, where $l(\mathbf{v})$ and $l(\mathbf{p})$ denote the lengths of $\mathbf{v}$ and $\mathbf{p}$, or are both infinite;

(2.2) $(x_{i_1}, y_{i_1}) \in P_i$;

(2.3) For every j, $1 \leq j < l(\mathbf{p})$, we have $v_{j+1} = Tr((x_{i_j}, y_{i_j}), i_j, v_j)$ and $((x_{i_{j+1}}, y_{i_{j+1}}), i_{j+1}) = Contr((x_{i_j}, y_{i_j}), i_j, v_j)$;

(2.4) For every $m > 1$, $l(\mathbf{v}) = m$ iff a production $(x_{i_{m-1}}, y_{i_{m-1}})$ is final and $(x_{i_{m-1}}, y_{i_{m-1}})$ is effectively used to v_{m-1}. In this case $((x_{i_{m-1}}, y_{i_{m-1}}), i_{m-1}, v_{m-1}) \notin$ $\blacksquare$ Dom($Contr$), where Dom(ϕ) denotes the domain of ϕ.

A partial function $\pi : V \to V$ is said to be computable by an algorithm DA iff for every $v \in \mathrm{Dom}(\pi)$ there exist a computation and a trace of the form:

$$\mathbf{v} = v_1, \dots, v_p$$

$$\mathbf{p} = ((x_{i_1}, y_{i_1}), i_1), \dots, ((x_{i_{p-1}}, y_{i_{p-1}}), i_{p-1})$$

such that $v = v_1$, $((x_{i_{p-1}}, y_{i_{p-1}}), i_{p-1}, v_{p-1}) \notin \mathrm{Dom}(Contr)$ and $\pi(v) = v_p$.

3. Markov-like k-algorithms

A new class $\mathcal{RMA}_k$ of right-hand side Markov-like k-algorithms is introduced. Every algorithm A of $\mathcal{RMA}_k$ is defined, analogously as of $\mathcal{MNA}$, by means of an indexed set $P = \{(x_i, y_i), 1 \leq i \leq n\}$ of productions[1] in any alphabet Σ which are labelled by numbers of $L = \{1, \dots, n\}$. The set of labelled productions is called a schema of productions. Practically schema of productions will be written as sequence of productions. The succession of use of the productions of P to a transformed word $t \in \Sigma^\star$ is the same as for algorithms of $\mathcal{MNA}$, but the manner of use of the productions is slightly different. We choose a production (x_i, y_i), with the least index $i \leq n$, such that x_i is a subword of t. Then we replace the j-th right-hand side subword of x_i by y_i, for maximal $j \leq k$. If (x_i, y_i) is final then we stop, otherwise we follow analogously with the new obtained word t_1 as with t.

At the beginning let us introduce some notations.

[1] We will also write non-final productions in the form: $x_i \longrightarrow y_i$, final productions in the form: $x_i \longrightarrow \cdot y_i$ and in a general case $x_i \longrightarrow (\cdot) y_i$.

For an alphabet Σ let $\Sigma^\star$ denote the set of all finite sequences over Σ including the empty sequences ε.

For every $v = v_1 \ldots v_m \in \Sigma^\star$ and $1 \leq i \leq j \leq m$ $v_{[i,j]}$ denotes a sequence $v_i \ldots v_j$. We additionally assume that if $i > j$ then $v_{[i,j]} = \varepsilon$.

For every words $u, v \in \Sigma^\star$ a word u is said to be a subword of v, $u \preceq v$, iff there are words $w, z \in \Sigma^\star$ (possibly empty) such that $v = wuz$.

For $u, v \in \Sigma^\star$ let us set

$$\mathrm{Occ}_{v;u} = \{j \in N : v_{[j,j+p-1]} = u\} \text{ and } \mathrm{Occ}^q_{v;u} = \{j \in \mathrm{Occ}_{v;u} : j \leq q\}$$

where p is the length of u.

We additionally assume that $\mathrm{Occ}_{v;u} = \emptyset$ if $\neg(u \preceq v)$.

A word u is said to be *the right-hand side i-th subword of u in v*, $u \preceq^r_i v$, iff $u = v_{[j,j+p-1]}$, $j = \min_q(|\mathrm{Occ}^q_{v;u}| = s - i + 1)$, where $s = |\mathrm{Occ}_{v;u}|$.

We additionally assume that if $u = \varepsilon$ than by $\varepsilon \preceq^r_i v$ we understand $\varepsilon \preceq^r_1 v$.

A word $u = v_{[j,j+p-1]}$ is said to be *at most right-hand side i-th subword of u in v*, $u \preceq^r_{\leq i} v$, iff $u \preceq^r_i v$, or there exists $1 \leq l < i$ such that $u \preceq^r_l v$ and $\neg(u \preceq^r_{l+1} v)$.

Example 3.1. Let $u = 01$ and $v = 1010111011$. Then $\mathrm{Occ}_{v;u} = \{2, 4, 8\}$ and $\mathrm{Occ}^0_{v;u} = \mathrm{Occ}^1_{v;u} = \emptyset; \mathrm{Occ}^2_{v;u} = \mathrm{Occ}^3_{v;u} = \{2\}; \mathrm{Occ}^i_{v;u}$ for $i = 4, 5, 6, 7$ is equal to $\{2, 4\}$ and for $j \geq 8$ $\mathrm{Occ}^j_{v;u} = \{2, 4, 8\}$.

Example 3.2. For the words u, v from Example3.1 the 1-st right-hand side subword of u in v has initial position equal to $\min_k(|\mathrm{Occ}^k_{v;u}| = 3 - 1 + 1 = 3) = 8$, the 2-nd right-hand side subword of u in v has initial position equal to $\min_k(|\mathrm{Occ}^k_{v;u}| = 3 - 2 + 1 = 2) = 4$, the 3-rd right-hand side subword of u in v has initial position equal to $\min_k(|\mathrm{Occ}^k_{v;u}| = 3 - 3 + 1 = 1) = 2$, 3-rd right-hand side subword is also at most right-hand side i-subword, for $i \geq 4$.

Let us assume the following convention. A schema of productions: $xa_1y \longrightarrow (\cdot)xy, xa_2y \longrightarrow (\cdot)xy, \ldots, xa_ny \longrightarrow (\cdot)xy$, where $\Sigma = \{a_1, \ldots, a_n\}$ ∎

will be briefly denoted by $x\alpha y \longrightarrow (\cdot)xy, (\alpha \in \Sigma)$. The additional symbols $(\notin \Sigma)$ will be denoted by $\gamma, \xi, \chi, \psi, \lambda$ (with or without subscripts).

Let us introduce the class $\mathcal{RMA}_k$.

Definition 3.3. By *a Markov-like k-algorithm $A \in \mathcal{RMA}_k$ in alphabet* Σ we mean the designated algorithm:

$$A = (P, P_i, P_f, V, L, \mathit{Contr}_1, \mathit{Tr}_1)$$

such that $V = \Sigma^\star$, $P = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$, where $x_i, y_i \in \Sigma^\star$, for all $i, 1 \leq i \leq n$, $x_n = \varepsilon, y_n = \varepsilon$ and $(x_n, y_n) \in P_f$, $P_i = \{(x_1, y_1)\}$, $L = \{1, \ldots, n\}$,

$$\mathit{Contr}_1((x_i, y_i), i, v) = \begin{cases} ((x_1, y_1), 1) & \textit{if } x_i \preceq^r_{\leq k} v \textit{ and } (x_i, y_i) \notin P_f \\ ((x_{i+1}, y_{i+1}), i+1) & \textit{if } \neg(x_i \preceq^r_{\leq k} v) \\ \textit{undefined} & \textit{if } x_i \preceq^r_{\leq k} v \textit{ and } (x_i, y_i) \in P_f \end{cases}$$

$$\mathit{Tr}_1((x_i, y_i), i, v) = \begin{cases} v_{[1,j-1]} y_i v_{[j+m,l(v)]} & \textit{if } v_{[j,j+m-1]} \textit{ is at most} \\ & k \textit{ subword of } x_i \textit{ in } v \\ v & \textit{otherwise} \end{cases}$$

An algorithm A of $\mathcal{RMA}_k$, $(k \geq 1)$ is said to be an algorithm over an alphabet Σ iff it is an algorithm in some alphabet $\Sigma_1 \supset \Sigma$.

The notions of *computation* of k-algorithm A as well as *computable function* by A can be introduced analogously as for designated algorithms.

4. Equivalence of the classes $\mathcal{RMA}_1$ and $\mathcal{MNA}$

Let $\mathcal{A}$ and $\mathcal{A}_1$ be the classes of algorithms (in or over an alphabet Σ) which are considered in this paper and let $A \in \mathcal{A}$, $A_1 \in \mathcal{A}_1$. Then A and A_1 are said to be equivalent with respect to Σ ($A \equiv_\Sigma A_1$) iff for every $x \in \Sigma^\star$, the results $A(x)$ and $A_1(x)$ of application of both algorithms to the initial word x are the same or are both undefined.

The equivalence of the classes of algorithms $\mathcal{A}$ and $\mathcal{A}_1$ should be understood as the equivalence with respect to an alphabet Σ. In more detail for every algorithm $A \in \mathcal{A}$ there exists algorithm $A_1 \in \mathcal{A}_1$ such that $A \equiv_\Sigma A_1$ and conversely for every $A_1 \in \mathcal{A}_1$ there exists $A \in \mathcal{A}$ such that $A_1 \equiv_\Sigma A$.

Theorem 4.1. *The classes $\mathcal{RMA}_1$ and $\mathcal{MNA}$ are equivalent.*

Proof. At the beginning we shall prove the following statement:

(1) For every algorithm $A \in \mathcal{RMA}_1$ in an alphabet Σ there exists an equivalent algorithm $M \in \mathcal{MNA}$ over Σ.

Let $P = \{(x_i, y_i) : 1 \leq i \leq n\}$ be an indexed set of productions of an algorithm $A \in \mathcal{RMA}_1$. Let us define the auxiliary Markov normal algorithms M' and M'' with the production sets $P' = \{P'_1, \ldots, P'_6\}$[2] and $P'' = \{P''_1, P''_2, \ldots, P''_n\}$ of the form:[3]

$P'_1 = (\eta\eta \longrightarrow \xi) \qquad (\eta, \xi \notin \Sigma)$

$P'_2 = (\xi\beta \longrightarrow \beta\xi)$

$P'_3 = (\xi\eta \longrightarrow \xi)$

$P'_4 = (\xi \longrightarrow \cdot\varepsilon)$

$P'_5 = (\eta\alpha\beta \longrightarrow \beta\eta\alpha)\ (\alpha, \beta \in \Sigma)$

$P'_6 = (\varepsilon \longrightarrow \eta)$

and $P''_j = (\hat{x}_j \longrightarrow (\cdot)\hat{y}_j)$ for all productions $P_j = (x_j \longrightarrow (\cdot)y_j)$[4]. Let us see that M' inverts every word $x \in \Sigma^\star$, i. e. $M'(x) = \hat{x}$. Let us put $M = M' \circ M'' \circ M'$. As the algorithm M is the composition of three Markov normal algorithms M', M'' and M' then it is also Markov normal algorithm (see [4], Chapter 5, p.214).

It follows immediately from the construction that $M(x) = A(x)$ for every $x \in \Sigma^\star$.

The following statement:

(2) For every Markov normal algorithm $M_1 \in \mathcal{MNA}$ in an alphabet Σ there exists an equivalent algorithm $A_1 \in \mathcal{RMA}_1$ over an alphabet Σ;

[2] The above algorithm is recalled after [4]

[3] A production P_j of the form $x_j \longrightarrow (\cdot)y_j$ can be written in the form $P_j = (x_j \longrightarrow (\cdot)y_j)$

[4] If $x = x_1 x_2 \ldots x_{n-1} x_n$ then $\hat{x} = x_n x_{n-1} \ldots x_2 x_1$.

can be analogously proved as of (1).[5]

The validity of Theorem 4.1 follows from statements (1) and (2).

5. Equivalence of the classes $\mathcal{RMA}_1$ and $\mathcal{RMA}_k$

Lemma 5.1. *For every algorithm $A \in \mathcal{RMA}_k$ in an alphabet Σ there exists an algorithm $M \in \mathcal{RMA}_1$ over an alphabet Σ such that $A \equiv_\Sigma M$.*

Proof. At the beginning let us introduce some notation. For $x = x_1 \ldots x_p$ $(x_j \in \Sigma, 1 \leq j \leq p)$ let $In(x, a) = x_1 a \ldots x_p$. For each symbol α of Σ, let α^j be a new symbol, and let Σ^j be the alphabet consisting of these α^j's $(1 \leq j \leq k$, α^0 will denote the symbol α without subscripts)[6]. Let us put $\Sigma'' = \Sigma \cup \Sigma^1 \cup \Sigma^2 \cup \cdots \cup \Sigma^n \cup \Delta^1 \cup \Delta^2 \cup \cdots \cup \Delta^n \cup \Psi(= \{\psi_1, \ldots, \psi_n\}) \cup \Gamma(= \{\gamma_1, \ldots, \gamma_n\}) \cup \Lambda(= \{\lambda_1, \ldots, \lambda_n\}) \cup \mathcal{X}(= \{\chi_0, \chi_1, \ldots, \chi_n\}) \cup \mathcal{N}(= \{\nu_1, \ldots, \nu_n\}) \cup Z(= \{\zeta_1, \ldots, \zeta_n\})$ where $\Delta^j = \{\delta_j{}^0, \delta 2_j{}^1, \ldots, \delta_j{}^{k-1}\}$.

We have to prove that $M \equiv_\Sigma A$. Let us give at the beginning an idea of the construction.

The production set $Sr_j^k(\Sigma'')$ which is assigned to every production $P_j : x_j \longrightarrow (\cdot)y_j$ consists of k additional symbols $\delta_j^i, i = 0, \ldots, k-1$, where δ_j^0 is an initial symbol. We replace the i-th right-hand side subword of x_j in a transformed word t by $\delta_j^i x_j$. Then $\delta_j^{k-1} x_j$ denotes that the k-th right-hand side subword of x_j in t has been found. If so then we replace $\delta_j^{k-1} x_j$ by $y_j \lambda_j$. Symbol λ_j denotes the fact that the k-th right-hand side subword of x_j in t is replaced by y_j. If there is not the k-th right-hand side subword of x_j in t then the symbol δ_j^i is moved to the end of t and is replaced by ψ_j. Then the production $\psi_j \varepsilon \longrightarrow \chi_j$ introduces new symbol χ_j which is then moved in left-hand side direction so long as we find the last right-hand side subword of x_j in t. Is so then we replace this subword by y_j. If there is not the i-th right-hand side subword of x_j in t for $1 \leq i < k$ then we introduce

[5] Proof of closure of $\mathcal{RMA}_1$ under composition is identical - with replacing in the additional productions $x \longrightarrow y$ by $\hat{x} \longrightarrow \hat{y}$ - to the proof of closure $\mathcal{MNA}$ under composition in [4].

[6] The same notations is in Mendelson's monograph [4].

new symbol ζ_j at the beginning of t. Now the transformed word has the form $\zeta_j t$.

An auxiliary schema Sa_j transforms the words in alphabet Σ^j into the words in alphabet Σ^{j+1} introducing as prefix symbol δ^0_{j+1}, if there exists the symbol ζ_j in the transformed word, or into the words in alphabet Σ, if x_j is at most k-th right-hand side subword of t (in the transformed word there exists the symbol ν_j). If P_j belongs to the set P_f and x_j is at most k-th right-hand side subword of t then Sa_j use the final production to stop algorithm.

For each production P_j of $A \in \mathcal{RMA}_k$ we create a set of productions: ${Sr_j}^k(\Sigma'')$:

$$
\begin{array}{ll}
{\delta_j}^{k-2}{x_j}^j \longrightarrow In({x_j}^j, {\delta_j}^{k-1}) & \\
{\delta_j}^{k-3}{x_j}^j \longrightarrow In({x_j}^j, {\delta_j}^{k-2}) & \\
\vdots & \\
{\delta_j}^{0}{x_j}^j \quad \longrightarrow In({x_j}^j, {\delta_j}^{1}) & \\
{\delta_j}^{k-1}{x_j}^j \longrightarrow {y_j}^j\lambda_j & \\
\chi_j{x_j}^j \longrightarrow {y_j}^j\lambda_j & \\
\chi_j\alpha \longrightarrow \alpha\chi_j & \\
\chi_j\varepsilon \longrightarrow \zeta_j & \\
\varepsilon\psi_j \longrightarrow \chi_j & \\
\alpha\delta_j \longrightarrow \delta_j\alpha & (\alpha \in \Sigma \cup \Sigma^1 \cup \Sigma^2 \cup \cdots \cup \Sigma^n,\ \delta_j \in \Delta^j) \\
\varepsilon\delta_j \longrightarrow \psi_j & \\
\alpha\lambda_j \longrightarrow \lambda_j\alpha & \\
\varepsilon\lambda_j \longrightarrow \nu_j. &
\end{array}
$$

Let us assign to each schema ${Sr_j}^k(\Sigma'')$ $(1 \leq j \leq n)$ an auxiliary schema $Sa_j(\Sigma'')$ of the form:

$$
\begin{array}{ll}
\gamma_j\alpha \longrightarrow \alpha\gamma_j & \\
\alpha\zeta_j \longrightarrow \zeta_j\alpha^{j+1} & (\alpha^j \in \Sigma^j) \\
\varepsilon\zeta_j \longrightarrow \gamma_j & \\
\gamma_j\varepsilon \longrightarrow \delta^0_{j+1} &
\end{array}
$$

$\nu_j\alpha \longrightarrow \alpha^0\nu_j \qquad (\alpha^0 \in \Sigma)$

$\nu_j\varepsilon \longrightarrow *\varepsilon$ where $* := \varepsilon$ if $P_j = (x_j \longrightarrow y_j)$ or $* := \cdot$ if $P_j = (x_j \longrightarrow \cdot y_j)$

if $1 \leq j \leq n-1$ or

$\gamma_j\alpha \longrightarrow \alpha\gamma_j$

$\alpha\zeta_n \longrightarrow \zeta_n\alpha^0 \qquad (\alpha^0 \in \Sigma,\ \alpha \in \Sigma \cup \Sigma^1 \cup \Sigma^2 \cup \cdots \cup \Sigma^n)$

$\varepsilon\zeta_n \longrightarrow \cdot\varepsilon$

$\nu_n\alpha \longrightarrow \alpha^0\nu_n$

$\nu_n\varepsilon \longrightarrow *\varepsilon$ where $* := \varepsilon$ if $P_n = (x_n \longrightarrow y_n)$ or $* := \cdot$ if $P_n = (x_n \longrightarrow \cdot y_n)$

if $j = n$.

One additional set, initial $Si(\Sigma'')$, has the form:

$Si(\Sigma'')$:

$\chi_0\alpha \longrightarrow \alpha^1\chi_0$

$\chi_0\varepsilon \longrightarrow \delta_1^0$

$\varepsilon\alpha \longrightarrow \chi_0\alpha$

The construction of Sr_j^k guarantees that this schema simulates P_j. The fact that $\Sigma^i \cap \Sigma^j = \emptyset$ for $i \neq j$ implies that none collision between Sr_j^k and Sr_i^k can take place. The proof can be simply completed by induction on the number n of productions.

This determines a Markov-like 1-algorithm M with the indexed set of productions $P = (Sa_1, Sa_2, \ldots, Sa_n, {Sr_1}^k, {Sr_2}^k, \ldots, {Sr_n}^k, Si)$

over Σ such that for arbitrary $x \in \Sigma^\star$ the results $M(x)$ and $A(x)$ of application of both algorithms M and A to x are the same, $M(x) = A(x)$, or both results are undefined.

Lemma 5.2. *For every algorithm $M_1 \in \mathcal{RMA}_1$ in an alphabet Σ and a number $k \geq 1$ there exists an algorithm $A_1 \in \mathcal{RMA}_k$ over an alphabet Σ such that $A_1 \equiv_\Sigma M_1$.*

Proof. Let M_1 be a Markov-like 1-algorithm in an alphabet Σ with an indexed set of productions $P = \{P_1, P_2, \ldots, P_n\}$.

Let us assign to every production P_j $(1 \le j \le n)$ an indexed productions set $Sr_j(\Sigma')(\Sigma' = \Sigma \cup \{\xi_1, \dots, \xi_{n+1}\} \cup \{\gamma_1, \dots, \gamma_n\})$ of the form:

$\xi_j x_j \longrightarrow (\cdot) y_j$

$\alpha\xi_j \longrightarrow \xi_j\alpha \qquad (\alpha \in \Sigma)$

$\varepsilon\xi_j \longrightarrow \gamma_j$

$\gamma_j\alpha \longrightarrow \alpha\gamma_j \qquad (\alpha \in \Sigma)$

$\gamma_j\varepsilon \longrightarrow \xi_{j+1}$

Let $A_1 \in \mathcal{RMA}_k$ be an algorithm with indexed sets of productions $(Sr_1, Sr_2, \dots, Sr_n, \xi_{n+1} \longrightarrow \cdot\varepsilon, \varepsilon \longrightarrow \xi_1)$.

Let us give some comments on the above construction.

Each schema Sr_j is assigned to P_j. As every production of algorithm A_1 transforms at most k-th subword of x_j then it is sufficient to use in Sr_j two singular symbols ξ_j, γ_j. Introduction of these symbols implies that each at most k-th subword of $\xi_j x_j$ is in reality first one of $\xi_j x_j$. To find x_j we will move ξ_j so long as the production $\xi_j x_j \longrightarrow y_j$ will be used. The symbol γ_j denotes fact, that the transformed word has not a subword x_j. Because the above additional symbols are different for every $Sr_i, Sr_j, i \neq j$, none collision between Sr_i and Sr_j can occurr.

Proof, can be completed by induction on the number of productions of M_1.

The above consideration implies that $M_1 \equiv_\Sigma A_1$.

Theorem 5.3. *The classes $\mathcal{RMA}_1$ and $\mathcal{RMA}_k (k \ge 1)$ are equivalent.*

Proof immediately follows from previous lemmas.

Theorem 5.4. *The classes $\mathcal{MNA}$ and $\mathcal{RMA}_k (k \ge 1)$ are equivalent.*

Proof. Theorems 4.1 and 5.3 determine the equivalence $\mathcal{RMA}_k \equiv_\Sigma \mathcal{MNA}$. ∎

Theorem 5.5. *Every class $\mathcal{RMA}_k (k \ge 1)$ is closed under composition, ramification, propagation and iteration operations.*

Proof. $\mathcal{MNA}$ is closed under all the above operations (see Mendelson [4], pp. 214–218) and the classes $\mathcal{RMA}_k (k \ge 1)$ and $\mathcal{MNA}$ are equivalent.

6. A class $\overline{\mathcal{RMA}_k}$ of right-hand side Markov-like k-algorithms

New class $\overline{\mathcal{RMA}_k}$ of right-hand side Markov-like k-algorithms introduced here is "symmetric" to the class $\overline{\mathcal{MA}_k}$ of left-hand side k-algorithms introduced in [1]. The difference is only such that we verify the transformed words by using the algorithms of $\overline{\mathcal{RMA}_k}$ from the right to the left, whereas by means of algorithms of $\overline{\mathcal{MA}_k}$ from the left to the right.

Every k-algorithm of $\overline{\mathcal{RMA}_k}$ is defined by means of an indexed set of productions $P = \{(x_i, y_i) : 1 \leq i \leq kn+1\}$ in an alphabet Σ which are labelled by the numbers $\{1, \ldots, kn+1\}$. Some final productions are also distinguished. The indexed set of productions consists of k identical subsets $\{(x_i, y_i) : (p-1)n+1 \leq i \leq pn\}, 1 \leq p \leq k$ with n productions, but for every subset productions are connected with different subword $(k+1-p)$ left side of production in the transformed word.

For brevity we shall write a schema of productions without repetitions in the form $P = \{(x_i, y_i) : 1 \leq i \leq n\}$

Given word $t \in \Sigma^\star$ we choose a production (x_j, y_j) with the least index $j \leq n$ such that x_j occurs in t at least k times. If so then we replace the k-th right-hand side subword of x_j in t by y_j. If (x_j, y_j) is final then we stop, otherwise we follow analogously with the new obtained word t_1 as with t. If there does not exist a production (x_j, y_j) as above, then we repeat verification for $k-1$ and so on.

We give a formal definition of the class $\overline{\mathcal{RMA}_k}$.

Definition 6.1. By *a Markov-like k-algorithm $A \in \overline{\mathcal{RMA}_k}$ in the alphabet* Σ we mean a designated algorithm

$$A = (P, P_i, P_f, V, L, \mathit{Contr}_2, \mathit{Tr}_2)$$

such that $V = \Sigma^\star$, $P = \{(x_1, y_1), (x_2, y_2), \ldots, (x_{kn+1}, y_{kn+1})\}$, where for all $i, 1 \leq i \leq kn+1, x_i, y_i \in \Sigma^\star$, and for all $j (n < j < kn+1) x_j = x_{j-n}$ and $y_j = y_{j-n}$ and $x_{kn+1} = \varepsilon, y_{kn+1} = \varepsilon$, $(x_{kn+1}, y_{kn+1}) \in P_f$ and

$P_i = \{(x_1, y_1)\}$, $L = \{1, \ldots, kn+1\}$ and the functions $Contr_2$ and Tr_2 which follows:

$$Contr_2((x_i, y_i), i, v) = \begin{cases} ((x_1, y_1), 1) & x_i \preceq^r_l v,\ l = k - entier(\frac{i-1}{n}), \\ & and\ (x_i, y_i) \notin P_f \\ ((x_{i+1}, y_{i+1}), i+1) & \neg(x_i \preceq^r_l v),\ l = k - entier(\frac{i-1}{n}) \\ undefined & x_i \preceq^r_l v,\ l = k - entier(\frac{i-1}{n}), \\ & and\ (x_i, y_i) \in P_f \end{cases}$$

$$Tr_2((x_i, y_i), i, v) = \begin{cases} v_{[1,j-1]} y_i v_{[j+m,l(v)]} & if\ v_{[j,j+m-1]}\ is\ l\ right\text{-}hand\ side \\ & \ subword\ of\ x_i\ in\ v \\ & where\ l = k - entier(\frac{i-1}{n}) \\ v & otherwise \end{cases}$$

An algorithm $A \in \overline{\mathcal{RMA}_k}$ is said to be an algorithm over an alphabet Σ iff it is an algorithm in some alphabet $\Sigma_1 \supset \Sigma$.

Now we explain on the example an activity of algorithms of this class.

Example 6.2. Let us define a right-hand side 2-algorithm $A \in \overline{\mathcal{RMA}_2}$ in the alphabet $\Sigma = \{0, 1\}$ by means of set of productions $P = \{P_1 = (01, 10), P_2 = (111, 010), P_3 = (01, 10), P_4 = (111, 010), P_5 = (\varepsilon, \varepsilon)\}$ (first two productions are used for second right-hand side subword of left sides of productions in transformed words, next two productions are used for first right-hand side subword of left sides of productions in transformed words, this set of productions without repetition has the form $\{(01, 10), (111, 010)\}$) and $P_f = \{(\varepsilon, \varepsilon)\}$. For $v = 1111$ the computation and trace of A have the form:

$$\mathbf{v} = ((1111), (1111), (0101), (1001), (1001),$$

$$(1001), (1010), (1010), (1010), (1100), (1100),$$

$$(1100), (1100), (1100))$$

$$\mathbf{p} = ((P_1, 1), (P_2, 2), (P_1, 1), (P_1, 1), (P_2, 2),$$

$$(P_3, 3), (P_1, 1), (P_2, 2), (P_3, 3), (P_1, 1), (P_2, 2),$$

$$(P_3, 3), (P_4, 4), (P_5, 5))$$

The result of application of A to the word 1111 is equal $Tr_{\overline{\mathcal{RMA}_2}}((\varepsilon,\varepsilon),5,1100) =$ 1100.

Theorem 6.3. *The classes $\overline{\mathcal{RMA}_k}$ and $\mathcal{MNA}$ are equivalent.*

Proof. First let us prove the equivalence of the classes $\overline{\mathcal{RMA}_k}$ and $\mathcal{RMA}_1$. The proof of equivalence $\overline{\mathcal{RMA}_k}$ and $\mathcal{RMA}_1$ presented here is the simple modification of the proof of equivalence $\overline{\mathcal{MA}_k}$ and $\mathcal{MNA}$ given in [1](Theorem 5.2.).

We prove this equivalence in two steps.

(1) For every k-algorithm $A \in \overline{\mathcal{RMA}_k}$ in an alphabet Σ there exists an equivalent algorithm $M \in \mathcal{RMA}_1$ over an alphabet Σ such that $A \equiv_\Sigma M$;

(2) For every algorithm $M \in \mathcal{RMA}_1$ in an alphabet Σ there exists an equivalent algorithm $A_2 \in \overline{\mathcal{RMA}_k}$ over an alphabet Σ such that $A_2 \equiv_\Sigma M$.

First we prove (1).

Let $\Sigma' = \Sigma \cup \{\gamma_j{}^{l,i}, \lambda^i\})$ where $1 \le i \le n$ and $1 \le j,l \le k$. For an algorithm $A \in \overline{\mathcal{RMA}_k}$ with the schema of productions $(x_1,y_1),\ldots,(x_n,y_n)$ (productions are without repetition) we create for each production (x_i,y_i) an auxiliary block of productions $Sr_i{}^l$ of the form:

$\gamma_j{}^{l,i}x_i \longrightarrow In(x_i, \gamma_{j+1}{}^{l,i})$ for $j < k$

$\gamma_k{}^{l,i}x_i \longrightarrow y_i$ this production is final iff (x_i,y_i) is final

$\alpha\gamma_j{}^{l,i} \longrightarrow \gamma_j{}^{l,i}\alpha$ for $j < k$

$\varepsilon\gamma_j{}^{l,i} \longrightarrow \lambda^{l,i}$ for $j < k$

$\lambda^{l,i}\alpha \longrightarrow \alpha\lambda^{l,i}$

$\lambda^{l,i}\varepsilon \longrightarrow \gamma_1{}^{l,i+1}$ if $i+1 = n$ then right side $=\gamma_1{}^{l-1,1}$

Let $M \in \mathcal{RMA}_1$ be an algorithm with a schema of productions $(Sr_1{}^k, Sr_2{}^k,\ldots,Sr_n{}^k, Sr_1{}^{k-1}, Sr_2{}^{k-1},\ldots,Sr_n{}^{k-1},\ldots,Sr_1{}^1,Sr_2{}^1,\ldots,Sr_n{}^1,(\gamma_1{}^{0,1} \longrightarrow \varepsilon),(\varepsilon \longrightarrow \gamma_1{}^{k,1}))$, $((\gamma_1{}^{0,1} \longrightarrow \varepsilon) \in P_f)$. One can easily see that $A \equiv_\Sigma M$.

To prove (2) let us consider an algorithm M of $\mathcal{RMA}_1$ in the alphabet Σ with the schema of productions of the form: $(x_1,y_1),\ldots,(x_n,y_n)$.

Let us construct an algorithm $A_2 \in \overline{\mathcal{RMA}_k}$ in the alphabet ($\Sigma' = \Sigma \cup \{\xi_1, \ldots, \xi_{k+1}\} \cup \{\gamma_1, \ldots, \gamma_k\}$) as follows.

Let us assign to the j-th production (x_j, y_j) $(1 \leq j \leq n)$ the following schema of productions Sr_j:

$$\begin{array}{ll}
\alpha\xi_j \longrightarrow \xi_j\alpha & \\
\beta_j\alpha \longrightarrow \alpha\beta_j & \\
\varepsilon\xi_j \longrightarrow \gamma_j & (\alpha \in \Sigma) \\
\beta_j\varepsilon \longrightarrow \varepsilon & \\
\gamma_j\alpha \longrightarrow \alpha\gamma_j & \\
\gamma_j\varepsilon \longrightarrow \xi_{j+1} &
\end{array}$$

Let $A_2 \in \overline{\mathcal{RMA}_k}$ be an algorithm with a schema of productions (without repetition) of the form: $((\xi_1 x_1 \longrightarrow \zeta_1), \ldots, (\xi_n x_n \longrightarrow \zeta_n), Sr_1, Sr_2, \ldots, Sr_n, (\varepsilon\gamma_n \longrightarrow \varepsilon) \in P_f, (\varepsilon \longrightarrow \xi_1))$; where $\zeta_i = y_i\beta_i$ if $i \notin P_f$ and otherwise $\zeta_i = y_i$ and this production is final.

One can easily see that $A_2 \equiv_\Sigma M$.

The above constructions of new algorithms use schemas Sr_j, Sr_j^l, where each Sr_j or Sr_j^l simulate one production P_j from given algorithm. The additional symbols guarantee that algorithms replace respective k-th subwords of x_j in the transformed word and preserve from collisions between different Sr_j, Sr_i. Because the above proof of equivalence $\overline{\mathcal{RMA}_k}$ and $\mathcal{RMA}_1$ is modification of the proof of equivalence $\overline{\mathcal{MA}_k}$ and $\mathcal{MNA}$ given in [1] we omit here any explanation. Formal proof can be done by induction on the number of productions.

Theorem 4.1 shows that the classes $\mathcal{MNA}$ and $\mathcal{RMA}_1$ are equivalent. Hence the classes $\overline{\mathcal{RMA}_k}$ and $\mathcal{MNA}$ are equivalent, too.

Theorem 6.4. *Every class* $\overline{\mathcal{RMA}_k}$ $(k \geq 1)$ *is closed under the following operations: composition, ramification, propagation and iteration.*

Proof follows from equivalence of $\overline{\mathcal{RMA}_k}$ and $\mathcal{MNA}$ and from closure $\mathcal{MNA}$ under above operations.

7. Final remarks

The equivalence of the classes $\mathcal{MA}_k$ and $\overline{\mathcal{MA}_k}$ of left-hand side Markov-like k-algorithms and the class $\mathcal{MNA}$ of Markov normal algorithms has been shown in [1]. In this paper the equivalence of the classes $\mathcal{RMA}_k$ and $\overline{\mathcal{RMA}_k}$ and $\mathcal{MNA}$ is shown. Therefore every of the classes $\mathcal{MA}_k$, $\overline{\mathcal{MA}_k}$, $\mathcal{RMA}_k$, $\overline{\mathcal{RMA}_k}$ and $\mathcal{MNA}$ can be used exchangeably for computing of some functions (for example Boolean) or for solution of some decidable problems.

Let us also add that the mentioned above classes of k-algorithms are only partially worked out. The majority of problems, especially a problem of complexity, have remained open till now.

References

[1] Grodzki Z., Mycka J. *The equivalence of some classes of algorithms*, Annales Universitatis Mariae Curie-Sklodowska, vol XLIX, 6(1995), 85–99.

[2] Markov, A. *The Theory of Algorithms*, (in Russian), Trudy Mat. Inst. Steklov. XLII, (1954).

[3] Mazurkiewicz, A *Podstawy teorii programowania* (*Foundation of the programming theory*), Problemy przetwarzania informacji, vol.2 (1974), 37–93,(in Polish).

[4] Mendelson, E., *Introduction to Mathematical Logic*, The University Series in Mathematics, Princeton, (1964).

Institute of Management and Foundation of Technics,

Department of Applied Mathematics,

Technical University of Lublin

ul. Bernardynska 13

20-950 Lublin, Poland.

Institute of Mathematics,

M. Curie-Sklodowska University
pl. M. Curie-Sklodowskiej 1
20-031 Lublin, Poland.