ANNALES
UNIVERSITATIS MARIAE CURIE – SKŁODOWSKA
LUBLIN – POLONIA

VOL. LV, 4 SECTIO A 2001

ZDZISŁAW GRODZKI and JERZY MYCKA

n-dimensional Markov - like algorithms

Abstract. New class $\mathcal{MA}_n^{k_1,\cdots,k_n}, n \geq 1$, of n-dimensional Markov-like algorithms is introduced. The equivalence of this class of algorithms and the class $\mathcal{MNA}$ of Markov normal algorithms is discussed.

1. Introduction. The intensive studies on the formalization of the notion of algorithm were conducted from 1930 on [2,7,10,12]. The majority of classical algorithms, such as partial recursive functions, Turing machines, Herbrand-Gödel computability, Markov normal algorithms are collected in Mendelson's monograph [9]. The equivalence of particular classes of algorithms were shown earlier by several authors [1,4,7] but all results are collected in Mendelson's monograph [9]. The next class of algorithms, for example the unlimited register machines ($\mathcal{URM}$), was also introduced[3]. The equivalence of the class $\mathcal{URM}$ and the class $\mathcal{PRF}$ of partial recursive functions was shown in [3].

A few classes of Markov-like algorithms were introduced by the authors in [5] where also the equivalence of these algorithms to the class $\mathcal{MNA}$ of Markov normal algorithms were shown. Only a few papers related to

1991 *Mathematics Subject Classification.* Primary 03D10, Secondary 68Q05.
Key words and phrases. Effective computabilty, Markov normal algorithms, equivalence of classes of algorithms.

algorithms of higher dimension were published [6,10]. The equivalence of the class of two-dimensional Markov-like algorithms and the class $\mathcal{MNA}$ was shown in [6].

This paper deals with the class $\mathcal{MA}_n^{k_1,\dots,k_n}$ of n-dimensional Markov-like algorithms with respect to the order $x_{k_1},\dots,x_{k_n}$ of axes. Every algorithm of $\mathcal{MA}_n^{k_1,\dots,k_n}$ is defined by means of a set $\{P_1,\dots,P_m\}$ of n-dimensional equally shaped productions which are labelled by elements of any set L (for simplicity we assume that $L=\{1,\dots,m\}$). The succession of the use of n-dimensional productions to the transformed words is almost the same as for classical Markov normal algorithms, but the manner of use of the productions depends on the choice of the subwords in the transformed words. We choose a production $P_i : x_i \longrightarrow (\cdot)y_i$, with the least label $i \le m$, such that its left-hand side word x_i occurs in a transformed n-dimensional word t_1. If such a production exists then we replace the first occurrence of x_i with respect to the order $x_{k_1},\dots,x_{k_n}$ of axes by y_i of P_i. If a production P_i is final then the algorithm stops, otherwise we should proceed with the newly obtained word t_2 analogously as with t_1.

Notice that every labelled set of n-dimensional productions determines $n!$ different algorithms of the classes $\mathcal{MA}_n^{k_1,\dots,k_n}$ with respect to the choice of the orders $x_{k_1},\dots,x_{k_n}$ of axes.

In this paper only the concept of the proof of a theorem relating to the equivalence of the above class of n-dimensional Markov-like algorithms to the class of $\mathcal{MNA}$ of Markov normal algorithms is given. The complete proof is very long and troublesome. Therefore we omit the proof.

This paper is the first step in the description of n-dimensional Markov-like algorithms.

The following reasons motivate the introduction of this class of algorithms:

(1) This paper is the first step of developments on n-dimensional formal algorithms, which can be used to study the complexity problems of n-dimensional structures.

(2) One is able to define n-dimensional partial recursive functions by analogy with word or graph recursive functions;

(3) The formalism used here allows to introduce other classes of n-dimensional Markov-like algorithms, for example parallel algorithms;

(4) One can define n-dimensional (not necessarily Markov-like) algorithms by a slight modification of the transformation and control functions. These algorithms may be useful for the description of real processes (biological, chemical, physical, medical and some others).

2. n-dimensional words and productions. Let Σ be a nonempty (finite) alphabet. By a *n-dimensional word* in an alphabet Σ we mean a partial function $\psi : N^n \longrightarrow \Sigma$ (N is the set of all nonnegative integers) satisfying the following conditions:

(1) $0 < Dom(\psi) < \infty$, where $Dom(\psi)$ denotes the domain of ψ;

(2) $(0, i_2, \ldots, i_n) \in Dom(\psi)$, $(j_1, 0, j_3 \ldots j_n) \in Dom(\psi), \ldots, (k_1, \ldots, k_{n-1}, 0) \in Dom(\psi)$;

(3) for arbitrary $(i_1, \ldots, i_n) \in Dom(\psi)$ and $(k_1, \ldots, k_n) \in Dom(\psi)$ there exists a sequence $(j_1^s, \ldots, j_n^s) \in Dom(\psi), 1 \leq s \leq m)$ where $(j_1^1, \ldots, j_n^1) = (i_1, \ldots, i_n)$, $(j_1^m, \ldots, j_n^m) = (k_1, \ldots, k_n)$, and for every $s \in \{1, \ldots, m-1\}$ and for all $t \in \{1, \ldots, n\}$ we have: $(j_t^s = j_t^{s+1}$ or $j_t^s = j_t^{s+1} \pm 1)$.

Statements given in (1) and (2) mean that we consider only finite n-dimensional words having at least one coordinate on particular axis, condition (3) means that every point of a word is connected with another arbitrary one.

Let Σ_n^* denote a class of all n-dimensional words in an alphabet Σ including the empty n-dimensional word λ_n (the function ψ describing λ_n has the empty domain).

In the majority of cases the n-dimensional words of Σ_n^* will be denoted by lower case Latin letters t, u, v, w, x, y, z (possibly with subscripts). If a word t is described by a function ψ then we will write $t(i_1, \ldots, i_n)$ or $t_{i_1, \ldots, i_n}$ instead of $\psi(i_1, \ldots, i_n)$ and $Dom(t)$ instead of $Dom(\psi)$.

A n-dimensional word t will be called over an alphabet Σ iff t is in an alphabet Σ', where Σ is a subset of Σ'.

Let us define a shape of a n-dimensional word $t \in \Sigma_n^*$. The n-tuple $(m_1, \ldots, m_n)$ is said to be *a shape* of a n-dimensional word t ($sh(t)$) iff

$$m_j = \sup\{i_j \in N : \exists (i_1 \ldots i_j \ldots i_n) \in Dom(t)\} + 1$$

for every $1 \leq j \leq n$

We assume the $sh(\lambda_n) = (0, \ldots, 0)$.

Let us consider two arbitrary n-dimensional words u and v of Σ_n^* and let

$$P = \{(i_1, \ldots, i_n) \in Dom(v) : \exists (i'_1, \ldots, i'_n) \in Dom(u) \exists (k_1, \ldots, k_n) \in N^m$$

$$\forall (j \leq n)[i_j = i'_j + k_j]\}.$$

Then a restricted function $v|_P$ is said to be *an occurrence* of u in v.

A restricted function $v|_P$ is said to be *the first occurrence of* u *in* v *with respect to the order of axes* $x_{k_1}, \dots , x_{k_n}$ iff the following conditions are satisfied[1]:

(1) $(k_1, \dots , k_n)$ is any permutation of $(1, \dots , n)$;

(2) $v|_P$ is an occurrence of u in v;

(3) For every $P' \subset Dom(v)$ such that $v|_{P'}$ is an occurrence of u in v there exists $(i_1, \dots , i_n) \in P$ such that for every $(i'_1, \dots , i'_n) \in P$ we have:

$$i_{k_1} < i'_{k_1}$$

or if there exists $m \leq n$ such that

$$i_{k_j} = i'_{k_j}, \text{ for all } 1 \leq j < m, \text{ then } i_{k_m} < i'_{k_m} \ .$$

Example 2.1. Let us consider the 2-dimensional word v in the following form:[2]

$$\begin{array}{ccccc} & & & a & \\ & & a & b & \\ b & a & b & & a \\ & a & b & a & b \end{array} .$$

Then the word u such that $u(0, 0) = a$, $u(1, 0) = b$, $u(1, 1) = a$ has three occurrences in the word v. Namely, the restricted sequences $v|_P$, $v|_{P'}$, and $v|_{P''}$ are the occurrences of u in v, where $P = \{(1, 1), (2, 1), (2, 2)\}$, $P' = \{(2, 2), (3, 2), (3, 3)\}$ and $P'' = \{(3, 0), (4, 0), (4, 1)\}$.

$v|_P$ is the first occurrence of u in v with respect to the order of axes (x_1, x_2) but $v|_{P'}$ is the first occurrence of u in v with respect to the order of axes (x_2, x_1).

Now let us define the concatenation of n-dimensional words u and v of shapes $(p_1, \dots, p_n)$ and $(q_1, \dots, q_n)$, respectively.

By *a concatenation* $u \circ_j v$ of the words u and v in the direction of j-th axis we mean a n-dimensional word w which is defined as follows:

(4) $sh(w) = (\max(p_1, q_1), \dots , p_j + q_j, \dots , \max(p_n, q_n))$;

(5) For every $(i_1, \dots , i_j, \dots i_n) \in Dom(u)$ we have $(i_1, \dots , i_j, \dots i_n) \in Dom(w)$ and $u_{i_1,\dots,i_j,\dots,i_n} = w_{i_1,\dots,i_j,\dots,i_n}$;

(6) For every $(s_1, \dots , s_j, \dots , s_n) \in Dom(v)$ we have $(s_1, \dots , s_j + p_j, \dots , s_n) \in Dom(w)$ and $v_{s_1,\dots,s_j,\dots,s_n} = w_{s_1,\dots,s_j+p_j,\dots,s_n}$.

[1]The axes of the n-dimensional Cartesian space N^n will be denoted by $x_1, \dots , x_n$

[2]We assume the convention that all n-dimensional words and productions will be written without axes.

Example 2.2. Let us consider the 2-dimensional word v of Example 2.1. Then the concatenations $u_1 = v \circ_1 v$ and $u_2 = v \circ_2 v$ have the forms:

$$u_1 = \begin{array}{cccccccccc} & & & a & & & & & a & \\ & & a & b & & & & a & b & \\ b & a & b & & a & b & a & b & & a \\ & a & b & a & b & & a & b & a & b \end{array}, \qquad u_2 = \begin{array}{ccccc} & & & a & \\ & & a & b & \\ b & a & b & & a \\ & a & b & a & b \\ & & & a & \\ & & a & b & \\ b & a & b & & a \\ & a & b & a & b \end{array}.$$

Now let us introduce a notion of n-dimensional production in an alphabet Σ.

By *an n-dimensional production* in an alphabet Σ we mean a pair (x, y) of n-dimensional words x, y in an alphabet Σ, $(m_1, \dots, m_n)$ is a shape of x and $(p_1, \dots, p_n)$ is a shape of y, with the properties:

$Dom(x) = \{(i_1, \dots, i_n) : 0 \le i_k \le m_k, 1 \le k \le n\}$

$Dom(y) = \{(j_1, \dots, j_n) : 0 \le j_k \le p_k, 1 \le k \le n\}$

and $m_k \le p_k$ for all $1 \le k \le n$ or $m_k \ge p_k$ for all $1 \le k \le n$.

The above conditions mean that we consider only such productions, whose left-hand side word x and right-hand side word y are "full" n-dimensional cubes and domain of one word contains the domain of the second one.

The set of all n-dimensional productions in an alphabet Σ will be denoted by $\mathcal{P}^n_\Sigma$. Let us distinguish a nonempty subset $\overline{\mathcal{P}^n_\Sigma}$ whose elements are called *final* whereas of $\mathcal{P}^n_\Sigma \backslash \overline{\mathcal{P}^n_\Sigma}$ -nonfinal ones.

As in Markov's monograph [8] the elements of $\overline{\mathcal{P}^n_\Sigma}$ will be denoted by $x \longrightarrow \cdot y$ (possibly with subscripts) while of $\mathcal{P}^n_\Sigma \backslash \overline{\mathcal{P}^n_\Sigma}$ - by $x \longrightarrow y$. Regardless of the fact that a production is final or nonfinal it will be written in the form $x \longrightarrow (\cdot)y$.

Now let us define *an extending function* $ex_\delta : \Sigma^*_n \longrightarrow \{\Sigma \cup \delta\}^*_n, \delta \notin \Sigma$ as follows:

for arbitrary words $u \in \Sigma^*_n, v \in \{\Sigma \cup \delta\}^*_n$ we have:

$$ex_\delta(u) = v$$

iff the following conditions hold:

(7) if $(i_1, \dots, i_n) \in Dom(u)$ then $v_{i_1, \dots, i_n} = u_{i_1, \dots, i_n}$;

(8) if $(i_1, \dots, i_n) \notin Dom(u)$ then $v_{i_1, \dots, i_n} = \delta$ for all $(i_1, \dots, i_n) : 0 \le i_k \le m_k, 1 \le k \le n$, $(m_1, \dots, m_n)$ is a shape of u.

Now let us define *a cuting function* $ct_\delta : \{\Sigma \cup \delta\}_n^* \longrightarrow \Sigma_n^*$ as follows: for arbitrary words $v \in \Sigma_n^*, u \in \{\Sigma \cup \delta\}_n^*$ we have:

$$ct_\delta(u) = v$$

iff the following conditions hold:

(9) if $(i_1, \dots, i_n) \in Dom(u)$ and $u_{(i_1,\dots,i_n)} \neq \delta$ then $u_{i_1,\dots,i_n} = v_{i_1,\dots,i_n}$;

(10) if $(i_1, \dots, i_n) \in Dom(u)$ and $u_{i_1,\dots,i_n} = \delta$ then $(i_1, \dots, i_n) \notin Dom(v)$, for all $(i_1, \dots, i_n) \in Dom(u)$.

Example 2.3. Let us consider the 2-dimensional word

$$t = \begin{array}{cccccc} b & a & b & a & a & a \\ & a & b & a & b & a \\ & & b & a & b & a \\ b & a & a & b & & \end{array}.$$

Then $$ex_\delta(t) = \begin{array}{cccccc} b & a & b & a & a & a \\ \delta & a & b & a & b & a \\ \delta & \delta & b & a & b & a \\ b & a & a & b & \delta & \delta \end{array} \quad \text{and} \quad ct_\delta(ex_\delta(t)) = t.$$

Now let us define *a resulting function* $Res_n^{k_1 \dots k_n} : \mathcal{P}_\Sigma^n \times \Sigma_n^* \longrightarrow \Sigma_n^*$ as follows:
For arbitrary n-dimensional production $x \longrightarrow (\cdot)y \in \mathcal{P}_\Sigma^n$ and a n-dimensional word $t \in \Sigma_n^*$ we have:

$$Res_n^{k_1 \dots k_n}(x \longrightarrow (\cdot)y, t) = \begin{cases} u & \text{if } x \text{ occurs in } t \\ t & \text{otherwise,} \end{cases}$$

where u is a n-dimensional word of Σ_n^* which is obtained from t in such a way that

$$u = ct_\delta({t'}^p_{k_1} \circ_{k_1} {t'}^p_{k_2} \circ_{k_2} \dots {t'}^p_{k_n} \circ_{k_n} y \circ_{k_n} {t'}^s_{k_n} \circ_{k_{n-1}} \dots \circ_{k_1} {t'}^s_{k_1})$$

where

$$t' = ex_\delta(t)$$

$$t' = {t'}^p_{k_1} \circ_{k_1} {t'}^p_{k_2} \circ_{k_2} \dots {t'}^p_{k_n} \circ_{k_n} t|_P \circ_{k_n} {t'}^s_{k_n} \circ_{k_{n-1}} \dots \circ_{k_1} {t'}^s_{k_1},$$

${t'}^p_{k_i}, {t'}^s_{k_i}$ for all $1 \le i \le n$ are maximal "prefix" and "sufix" subwords and $t|_P$ is the first occurrence of x in t with respect to the order of the axes $(x_{k_1}, \dots, x_{k_n})$, $\delta \notin \Sigma$.

We can distinguish in the above construction of u a few steps:
(1) extending the word t to the "full" n-dimensional cube t';

(2) distinguishing all ${t'}^p_{k_i}, {t'}^s_{k_i}$, which are "prefix" and "sufix" subwords of t' in all n dimensions surrounding the occurrence of x in t';
(3) replacing the occurrence x by y;
(4) creating u by cutting all additonal symbols in the word which is result of step (3).

In the first case (if x occurs in t) the production is said to be *effectively used*, whereas in the second one *noneffectively used* to a word t.

Example 2.4. Let us present here an example of action of the function $Res_n^{k_1\ldots k_n} : \mathcal{P}_\Sigma^n \times \Sigma_n^* \longrightarrow \Sigma_n^*$. Let us consider the 2-dimensional word t from Example 2.3 and the production:

$$P : \begin{array}{cc} a & b \\ a & b \end{array} \longrightarrow \begin{array}{ccc} c & c & c \\ c & c & c \\ c & c & c \end{array}.$$

Then we have

$${t'}^p_1 = \begin{array}{ccc} \delta & a & b \\ \delta & \delta & b \end{array}$$

$${t'}^s_1 = \begin{array}{c} a \\ a \end{array}$$

$${t'}^s_2 = \begin{array}{cccccc} b & a & b & a & a & a \end{array}$$

$${t'}^p_2 = \begin{array}{cccccc} b & a & a & b & \delta & \delta \end{array}$$

Hence

$$Res_2^{2,1}(P,t) = ct_\delta({t'}^p_2 \circ_2 {t'}^p_1 \circ_1 y \circ_1 {t'}^s_1 \circ_2 {t'}^s_2) = \begin{array}{ccccccc} b & a & b & a & a & a & \\ & & & c & c & c & \\ & a & b & c & c & c & a \\ & & b & c & c & c & a \\ b & a & a & b & & & \end{array}.$$

3. The n-dimensional Markov-like algorithms. New class $\mathcal{MA}_n^{k_1,\ldots,k_n}$, $n \geq 1$ of n-dimensional Markov-like with respect to the order $(x_{k_1}, \ldots, x_{k_n})$ of the axes will be introduced.

By *a n-dimensional Markov-like algorithm* of the class $\mathcal{MA}_n^{k_1,\ldots,k_n}$ (the order of the axes $x_{k_1}, \ldots, x_{k_n}$ is fixed) in an alphabet Σ we mean a sixtuple

$$A = (P_\Sigma^n, L, L_i, L_f, Contr\ {}_n^{k_1,\ldots,k_n}, Tr\ {}_n^{k_1,\ldots,k_n}),$$

where

P_Σ^n is a finite (nonempty) subset of $\mathcal{P}_\Sigma$ of n-dimensional productions in an alphabet Σ,

L is *a set of labels* of P^n_Σ (we assume $L = \{1, \dots |P_\Sigma|\}$);

$L_i = \{1\}$ and L_f are the subsets of L whose elements are called *initial* and *final* labels, respectively[3].

A partial function $Contr\,{}^{k_1,\cdots,k_n}_n : \Sigma^*_n \times L \mapsto L$, called *a control* of A, and a total function $Tr\,{}^{k_1,\cdots,k_n}_n : \Sigma^*_n \times L \mapsto \Sigma^*_n$, called *a transformation* of A, are defined as follows:

For arbitrary n-dimensional words $t, u \in \Sigma^*_n$ and $i \in L$ we have[4]:

$$Contr\,{}^{k_1,\cdots,k_n}_n(t,i) = \begin{cases} 1 & \text{if } x_i \text{ occurs in } t \text{ and } i \notin L_f \\ i+1 & \text{if } x_i \text{ doesn't occur in } t \text{ and } i \leq |L| \\ \text{undefined} & \text{if } x_i \text{ occurs in } t \text{ and } i \in L_f \\ & \text{or } x_i \text{ doesn't occur in } t \text{ and } i = |L|, \end{cases}$$

$$Tr\,{}^{k_1,\cdots,k_n}_n(t,i) = \begin{cases} Res^{k_1,\cdots,k_n}_n(x_i \longrightarrow (\cdot)y_i, t) & \text{if } x_i \text{ occurs in } t \\ t & \text{otherwise.} \end{cases}$$

We denote some production from P^n_Σ by P_i iff $\phi(i) = P_i$, where ϕ is one-to-one mapping of P^n_Σ onto L.

Thus if a production P_i has been effectively used to a word t and it is nonfinal then $Contr\,{}^{k_1,\cdots,k_n}_n(t,i) = 1$ or if x_i doesn't occur in t and $i < |L|$ then $Contr\,{}^{k_1,\cdots,k_n}_n(t,i) = i+1$. $Contr\,{}^{k_1,\cdots,k_n}_n$ is undefined if a production P_i has been effectively used to a word t and it is final ($i \in L_f$) or if x_i doesn't occur in t and $i = |L|$.

If a production $P_i : x_i \longrightarrow (\cdot)y_i$ has been effectively used to a word t then $Tr\,{}^{k_1,\cdots,k_n}_n$ transforms a word t in a such way that the first occurrence with respect to the order $x_{k_1}, \dots, x_{k_n}$ of the axes of x_i of a production P_i in t is replaced by y_i. If a production P_i has been noneffectively used to a word t then $Tr\,{}^{k_1,\cdots,k_n}_n(t,i) = t$ and we continue a computation in every case (if P_i is final or nonfinal) with such only a restriction that $i < |L|$. Let us observe that the process stops iff the effectively used lately production is final or if the used lately production P_j has been noneffectively used and $j = |L|$.

In some examples there is a need to introduce some additional letters (separators, parenthesis). This leads to the following definition.

A n-dimensional Markov-like algorithm is said to be over an alphabet Σ iff it is an algorithm in some alphabet Σ' such that $\Sigma \subset \Sigma'$.

Now let us introduce a notion of a computation of a n-dimensional Markov-like algorithm.

A sequence $T = t_1, t_2, \ldots \in (\Sigma^*_n)^\omega$ (finite or infinite) is said to be *a computation* of a n-dimensional algorithm $A = (P^n_\Sigma, L, L_i, L_f, Contr\,{}^{k_1,\cdots,k_n}_n,$

[3]We will sometimes identify labelled productions with their labels.

[4]Statement x_i occurs in t means that exists $P \subset Dom(t)$ such that $t|_P$ is an occurrence of x_i in t

$Tr\ _n^{k_1,\cdots,k_n}$) of the class $\mathcal{MA}_n^{k_1,\cdots,k_n}$ iff there exists a sequence $I = i_1, i_2, \ldots \in L^\infty$, called *a trace* of T, such that the following conditions hold:

(1) Both sequences T and I are infinite, $i_1 \in L_i$, and for every $j \geq 1$ we have: $t_{j+1} = Tr\ _n^{k_1,\cdots,k_n}(t_j, i_j)$ and $i_{j+1} = Contr\ _n^{k_1,\cdots,k_n}(t_j, i_j)$;

(2) Both sequences T and I are finite of lengths equal to m for some $m > 1$, $i_1 \in L_i$ and for every $1 \leq j < m$ we have: $t_{j+1} = Tr\ _n^{k_1,\cdots,k_n}(t_j, i_j)$ and $i_{j+1} = Contr\ _n^{k_1,\cdots,k_n}(t_j, i_j)$. We additionally assume that $i_m = |L| + 1$ and this label indicates the fact that a computation T stops.

Two cases imply that T is finite. The first one is when a production $P_{i_{m-1}}$ with the label i_{m-1} has been effectively used to a word t_{m-1} and it is final or if $P_{i_{m-1}}$ has been noneffectively used to a word t_{m-1} and $i_{m-1} = |L|$.

The set of all computations of a n-dimensional Markov-like algorithm A is said to be its *computation set* and denoted by $\mathcal{C}(A)$.

As for all algorithms of $\mathcal{MA}_n^{k_1,\cdots,k_n}$ the control $Contr\ _n^{k_1,\cdots,k_n}$ and transformation $Tr\ _n^{k_1,\cdots,k_n}$ are defined in the same manner therefore to define an algorithm $A \in \mathcal{MA}_n^{k_1,\cdots,k_n}$ it is sufficient to define a sequence of productions in (or over) an alphabet Σ by omitting their labels (such a sequence will be called *a schema of productions*).

Example 3.1. Let us consider the 2-dimensional algorithms $A \in \mathcal{MA}_2^{1,2}$ and $A' \in \mathcal{MA}_2^{2,1}$ in the alphabet $\Sigma = \{a, b, c, d\}$ with the same schema of productions:

$$P_1 : \begin{matrix} d & c \\ d & c \end{matrix} \longrightarrow \cdot \begin{matrix} a & a \\ a & a \end{matrix}$$

$$P_2 : \begin{matrix} c & c \\ d & d \end{matrix} \longrightarrow \begin{matrix} a & a \\ a & a \end{matrix}$$

$$P_3 : \begin{matrix} c & c \\ a & b \end{matrix} \longrightarrow \begin{matrix} d & d \\ d & d \end{matrix}$$

Let us consider the word v of the following form: $\begin{matrix} c & c & c & c \\ a & b & c & c \\ & & a & b \end{matrix}$.

Then the computation of A with the initial word v has the form:

$$\begin{matrix} c & c & c & c \\ a & b & c & c \\ & & a & b \end{matrix}\ ,\quad \begin{matrix} d & d & c & c \\ d & d & c & c \\ & & a & b \end{matrix}\ ,\quad \begin{matrix} d & a & a & c \\ d & a & a & c \\ & & a & b \end{matrix}\ .$$

The computation of A' with the same initial word v has the form:

$$\begin{array}{cccc} c & c & c & c \\ a & b & c & c \\ & & a & b \end{array}\;, \quad \begin{array}{cccc} c & c & c & c \\ a & b & d & d \\ & & d & d \end{array}\;, \quad \begin{array}{cccc} c & c & a & a \\ a & b & a & a \\ & & d & d \end{array}\;, \quad \begin{array}{cccc} d & d & a & a \\ d & d & a & a \\ & & d & d \end{array}\;.$$

4. The permutation n-dimensional words and algorithms. Let us give at the beginning the necessary definitions connected with permutations of axis of n-dimensional words and productions.

A word v' of Σ_n^* will be called *a* $(j_1, \dots, j_n)$*-permutation word* of the word $v \in \Sigma_n^*$ (and denoted $v^{j_1,\cdots,j_n}$) iff the following conditions are satisfied:

(1) for all $(i_1, \dots, i_n) \in Dom(v)$ we have $(i_{j_1}, \dots, i_{j_n}) \in Dom(v')$ and $v_{i_1,\dots,i_n} = v'_{i_{j_1},\dots,i_{j_n}}$;

(2) for all $(i_{j_1}, \dots, i_{j_n}) \in Dom(v')$ we have $(i_1, \dots, i_n) \in Dom(v)$ and $v_{i_1,\dots,i_n} = v'_{i_{j_1},\dots,i_{j_n}}$;

Example 4.1. Let us consider the 2-dimensional word v from Example 3.1. Then the word v' which is (2,1)-permutations word of v has the form:

$$v' = \begin{array}{ccc} b & c & c \\ a & c & c \\ & b & c \\ & a & c \end{array}\;.$$

Let (x, y) be an arbitrary n-dimensional production and let $(j_1, \dots, j_n)$ be a permutation of $(1, \dots, n)$.

Then a production $P' = (x', y')$ is said to be *a n-dimensional* $(j_1, \dots, j_n)$*-permutation production* in an alphabet Σ of the production $P = (x, y)$ in an alphabet Σ (P' is denoted as $P^{j_1,\cdots,j_n}$) iff $x' = x^{j_1,\cdots,j_n}$ and $y' = y^{j_1,\cdots,j_n}$.

A n-dimensional algorithm $A' \in \mathcal{MA}_n^{k_{j_1},\cdots,k_{j_n}}$ in an alphabet Σ is said to be *a n-dimensional* $(j_1, \dots, j_n)$*-permutation algorithm* of the algorithm $A \in \mathcal{MA}_n^{k_1,\cdots,k_n}$ in an alphabet Σ iff $L = L'$, $L_i = L'_i$, $L_f = L'_f$, $|P^n_\Sigma| = |P'^n_\Sigma|$ and $P'_i = P_i^{j_1,\cdots,j_n}, 1 \le i \le |P^n_\Sigma|$ (*Contr* and *Tr* of A' are the same as for the whole class of algorithms $\mathcal{MA}_n^{k_{j_1},\cdots,k_{j_n}}$).

Example 4.2. Let us consider the algorithm $A \in \mathcal{MA}_2^{1,2}$ from the Example 3.1. Then the schema of productions of $2,1$-permutation algorithm $A' \in \mathcal{MA}_2^{2,1}$ in the alphabet $\Sigma = \{a, b, c, d\}$ has the form:

$$P'_1 : \begin{array}{cc} c & c \\ d & d \end{array} \longrightarrow \cdot \begin{array}{cc} a & a \\ a & a \end{array}$$

$$P'_2 : \begin{array}{cc} d & c \\ d & c \end{array} \longrightarrow \begin{array}{cc} a & a \\ a & a \end{array}$$

$$P_3' : \begin{array}{cc} b & c \\ a & c \end{array} \longrightarrow \begin{array}{cc} d & d \\ d & d \end{array} .$$

Then computation of A' for the word v' from Example 4.1 has the form:

$$\begin{array}{ccc} b & c & c \\ a & c & c \\ & b & c \\ & a & c \end{array} \quad , \quad \begin{array}{ccc} b & c & c \\ a & c & c \\ & d & d \\ & d & d \end{array} \quad , \quad \begin{array}{ccc} b & c & c \\ a & a & a \\ & a & a \\ & d & d \end{array} \quad .$$

Let us point that the computation of A' for the initial word v' is "symmetrical" to the computation of A for the initial word v.

So, we can give here the following lemma.

Lemma 4.4. *For every $A \in \mathcal{MA}_n^{k_1,\cdots,k_n}$ and the word $v \in \Sigma_n^*$ we have*

$$A'(v^{j_1,\cdots,j_n}) = [A(v)]^{j_1,\cdots,j_n}$$

where $(j_1,\ldots,j_n)$ is a permutation of $(1,\ldots,n)$ and $A' \in \mathcal{MA}_n^{k_{j_1},\cdots,k_{j_n}}$ is an permutation algorithm of A.

Proof is obvious.

5. The equivalence of the classes $\mathcal{MA}_n^{k_1,\ldots,k_n}$ and $\mathcal{MNA}$. We will use in this section the notion of a representation of n-dimensional word t by $n-1$ dimensional word u. Intuitively, to create a representation of a given word $t \in \Sigma_n^*$, we will place t in a n-dimensional cube v whereas in empty places any element outside the alphabet Σ will be located. Then a cube covering a word t is cut into $(n-1)$-dimensional layers with respect to direction of the x_{k_n} axis. The representation of a word t will be equal to the concatenation $\overline{v} = \theta_1 v_1, \ldots, \theta_n v_n$ (v_i is the i-th layer of v, θ_i- $(n-1)$-dimensional separators).

Of course, a notion of representation can be inductively extended to representation of n-dimensional word by 1-dimensional word.

We can also represent $n-1$-dimensional word t by n-dimensional word u by extending word t to u in such a way that the $(n-1)$-dimensional layer of u with respect to direction of the x_{k_n} axis on the zero coordinate is equal t and other layers of u with respect to x_{k_n} are empty.

Now we can say about equivalence of two classes $\mathcal{A}_1$, $\mathcal{A}_2$ of n-dimensional algorithms and m-dimensional algorithms, when:

1) for every algorithm $A_1 \in \mathcal{A}_1$ there exists $A_2 \in \mathcal{A}_2$ such that for each n-dimensional word t the m-dimensional representant of $A_1(t)$ is equal to the value of A_2 for m-dimensional representant of t

and

2) for every algorithm $A_2 \in \mathcal{A}_2$ there exists $A_1 \in \mathcal{A}_1$ such that for each m-dimensional word u the n-dimensional representant of $A_2(u)$ is equal to the value of A_1 for n-dimensional representant of u.

Theorem 5.1. *For every order $x_{k_1}, \dots, x_{k_n}$ of the axes the classes $\mathcal{MA}_n^{k_1,\dots,k_n}$ and $\mathcal{MNA}$ (where $\mathcal{MNA}$ denotes the class of Markov normal algorithms) are equivalent.*

We shall give only a short outline of the proof of a lemma relating to the equivalence of the classes $\mathcal{MA}_n^{k_1,\dots,k_n}$ and $\mathcal{MA}_{n-1}^{k_1,\dots,k_{n-1}}$ of Markov-like algorithms, which is a main part in the inductive proof of Theorem 5.1.

This proof is supported of two lemmas.

Lemma 5.2. *For arbitrary algorithm $A \in \mathcal{MA}_n^{k_1,\dots,k_n}$ in an alphabet Σ there exists an algorithm $B \in \mathcal{MA}_{n-1}^{k_1,\dots,k_{n-1}}$ over an alphabet Σ such that $\overline{A(v)} = B(\overline{v})$, for every $v \in \Sigma_n^*$ where $\overline{v}$ is a representation of n-dimensional word $v \in \Sigma_n^*$ in $(n-1)$-dimensional word of Σ_{n-1}^*. Analogously $\overline{A(v)}$ denotes a representation of n-dimensional word $A(v)$ in $(n-1)$-dimensional word of Σ_{n-1}^*;*

Lemma 5.3. *For arbitrary algorithm $B \in \mathcal{MA}_{n-1}^{k_1,\dots,k_{n-1}}$ in an alphabet Σ there exists an algorithm $A \in \mathcal{MA}_n^{k_1,\dots,k_n}$ over an alphabet Σ such that $\overline{B(v')} = A(\overline{v'})$ for every $v' \in \Sigma_{n-1}^*$.*

The Lemma 5.3 can be easily proved by transformation of every word $v' \in \Sigma_{n-1}^*$ into a word $w \in \Sigma_n^*$ by adding a new x_{k_n} coordinate and by extending word v to v' by locating v on the zero coordinate of the new x_{k_n} axis.

We proceed analogously with $(n-1)$-dimensional productions.

The proof of Lemma 5.2 is more complicated. Two problems should be solved:

(1) Representation of all n-dimensional words of Σ_n^* by means of $(n-1)$-dimensional words over Σ_{n-1}^*, and analogously we follow n-dimensional productions;

(2) We have to transform a schema of n-dimensional productions into a schema of $(n-1)$-dimensional productions.

The problem (1) can be solved in the following way. A given word $t \in \Sigma_n^*$ is placed in a n-dimensional cube whereas in free places any element outside the alphabet Σ is located. Then a cube covering a word t is cut into $(n-1)$-dimensional layers with respect to direction of the x_{k_n} axis. Then we assign to a pair of n-dimensional cubes corresponding to n-dimensional production $\mathcal{P}_i : x_i \longrightarrow (\cdot)y_i$ the sequence of $(n-1)$-dimensional cubes corresponding to the successive layers. The transformed n-dimensional word t should be replaced by a concatenation $\overline{v} = \theta_1 v_1, \dots, \theta_n v_n$ (v_i is the i-th layer of v, θ_i-$(n-1)$-dimensional separators).

The problem (2) can be solved by adding to productions of a sequence $\overline{\mathcal{P}}_i (1 \leq i \leq m)$ new symbols outside Σ and some additional productions transforming these additional symbols such as following conditions hold:

(2.1) If a sequence $\overline{\mathcal{P}_i}$ has been effectively used to a transformed word t then we must return to first element of $\overline{\mathcal{P}_1}$ if $i \notin L_F$ or an algorithm stops if $i \in L_F$;

(2.2) If a sequence $\overline{\mathcal{P}_i}$ corresponding to a production $\mathcal{P}_i$ has not been effectively used to a transformed word then we have to go to $\overline{\mathcal{P}_{i+1}}$ (if $i < m$).

Let us add that the complete proof of theorem relating to the equivalence of 2-dimensional Markov-like algorithms and Markov normal algorithms has been given in [6].

Open problems. Let us put forward some open problems relating to n-dimensional algorithms:

(1) One is able to define classes $\mathcal{MA}_{j,n}^{k_1,\dots,k_n}$ of n-dimensional Markov-like j-algorithms for which the j-th left-hand side occurrence of the left side of the productions in the transformed words is replaced by the right side of the respective productions (taking into account some order of axes);

(2) A class of n-dimensional weighed Markov-like algorithms can be introduced that to every production $\mathcal{P}_i$ a weight w_i is assigned, indicating that the w_i-occurrence of the left-hand side of productions with respect to some order of axes is replaced by the right-hand side of $\mathcal{P}_i$.

(3) One is able to introduce other classes of n-dimensional algorithms (not necessarily Markov-like) only insignificantly modifying the transformation and control functions.

(4) There is a need to study different aspects of complexity of n-dimensional algorithms.

(5) By analogy with n-dimensional algorithms one is able to define n-dimensional word recursive functions.

References

[1] Asser, G., *Turing Maschinen und Markovsche Algorithmen*, Z. Math. Logik Grundl. Math. **5** (1959), 326–359.

[2] Church, A., *An unsolvable problem for elementary number theory*, J. of Math. **58** (1936), 345–363.

[3] Cutland, N.J., *Computability and introduction to recursive function theory*, Cambridge University Press, Cambridge, London, New York, Sydney, Melbourne, 1980.

[4] Dietlows W.K., *Equivalence of Markov normal algorithms and recursive functions*, Trudy Mat. Inst. Steklov. **IV** (1952), 66–69.

[5] Grodzki, Z., J. Mycka, *The equivalence of some classes of algorithms*, Ann. Univ. Mariae Curie-Skłodowska Sect. A **49** (1995), no. 6, 85–99.

[6] Grodzki, Z., J. Mycka, *Two-dimensional Markov-like algorithms*, Ann. Univ. Mariae Curie-Skłodowska Sect. A **50** (1996).

[7] Kleene, S.C., *λ-definability and recursivennes*, Duke Math. J. **2** (1936), 340–358.

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

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

[10] Priese, L., *A note of asynchronous cellular automata*, J. Comput. System Sci. **17** (1978), 237–252.

[11] Robinson J., *General recursive functions*, Proc. Amer. Math. Soc. **I** (1950), 703–718.

[12] Turing A., *On computable numbers with an application to the Entscheidungsproblem*, Proc. London Math. Soc. **42** (1936), 230–265, (correction ibid., **43** (1937), 544–546).

Department of Applied Mathematics
Technical University of Lublin
ul. Bernardyńska 13
20-950 Lublin, Poland

received November 10, 1999

Institute of Mathematics
M. Curie–Skłodowska University
pl. M. Curie–Skłodowskiej 1
20-031 Lublin, Poland