XI Workshop
Program Semantics, Specification and Verification:
Theory and Applications
PSSV-2020

Hayk A. Grigoryan and Samvel K. Shoukourian

A Polynomial Algorithm for Equivalence Problem of Deterministic Multitape Finite Automata

Email: samshouk@sci.am

Moscow

In memory of
Professor Alexander Letichevsky

1979

Yerevan State University
Scientific Leader
Information Technologies Educational and Research Center

Samvel K. Shoukourian
Doctor of Sciences in Physics and Mathematics,
Professor,
Member of the National Academy of Sciences

Received the Doctor of Science degree in Physics and Mathematics in 1990 and the Academic Rank of Professor in Computer Science and Software Engineering in 1993 from the Supreme Certification Board (Moscow). In 1996, was elected a Full Member of the National Academy of Sciences of Armenia.

Since 1994, simultaneously with the work at YSU, he has been Chief Scientific Advisor and Development Director in different international companies. Since 2000, he is directing the Department of Embedded Test and Repair at Virage Logic Corporation (USA) and since 2010, the same department at Synopsys, Inc (USA). Was awarded the State Prize of Armenia in 2013, has authored more than 100 papers, and holds 1 Russian and 11 U.S. patents.

The main topics of current research interests include formal models of distributed systems, testing of electronic devices and systems, information technologies and architectures for multimedia virtual environments.

Outline

Hayk Grigoryan, Samvel K. Shoukourian: Polynomial algorithm for equivalence problem of deterministic multitape finite automata. Theor. Comput. Sci. 833: 120-132 (2020)

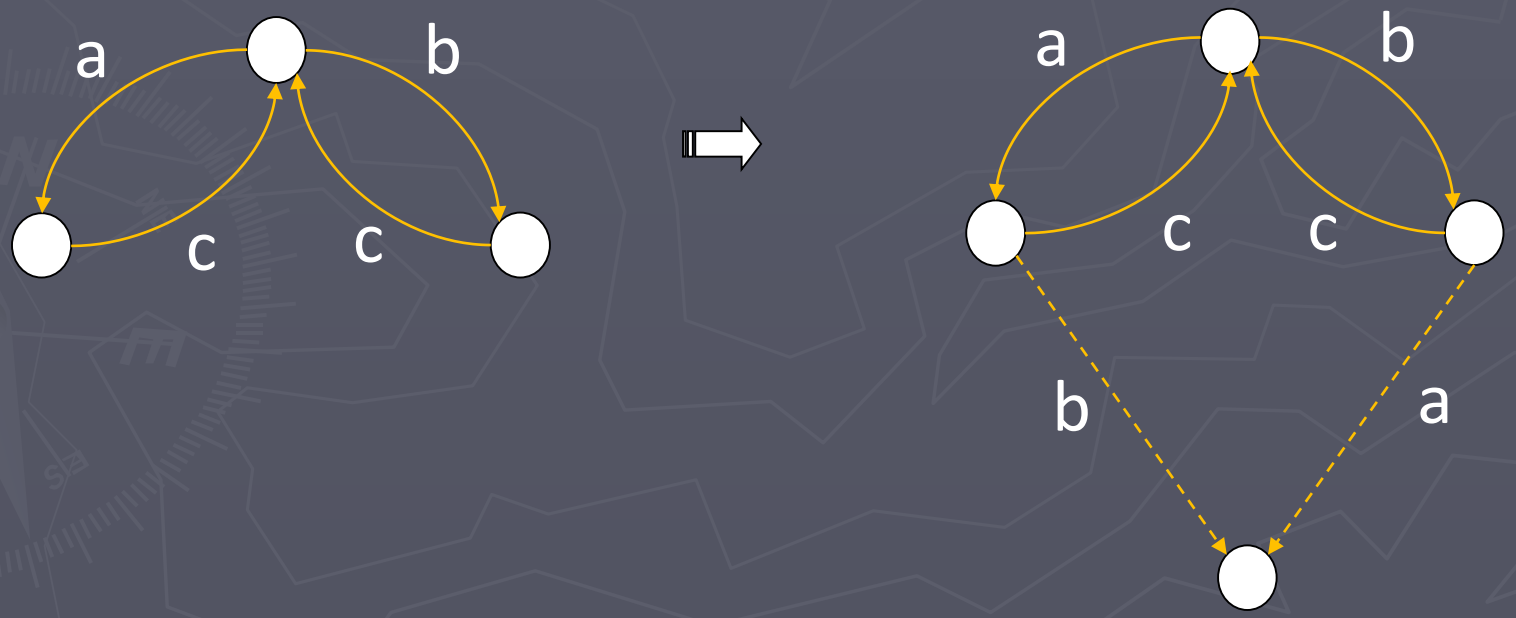
- Some History
- Multidimensional Tape as a Storage for Execution Traces
- The Idea of the Combinatorial Proof
- Some Related Definitions
- Sketch of the Solution
- Estimation of a Complexity
- Notes Behind the Solution

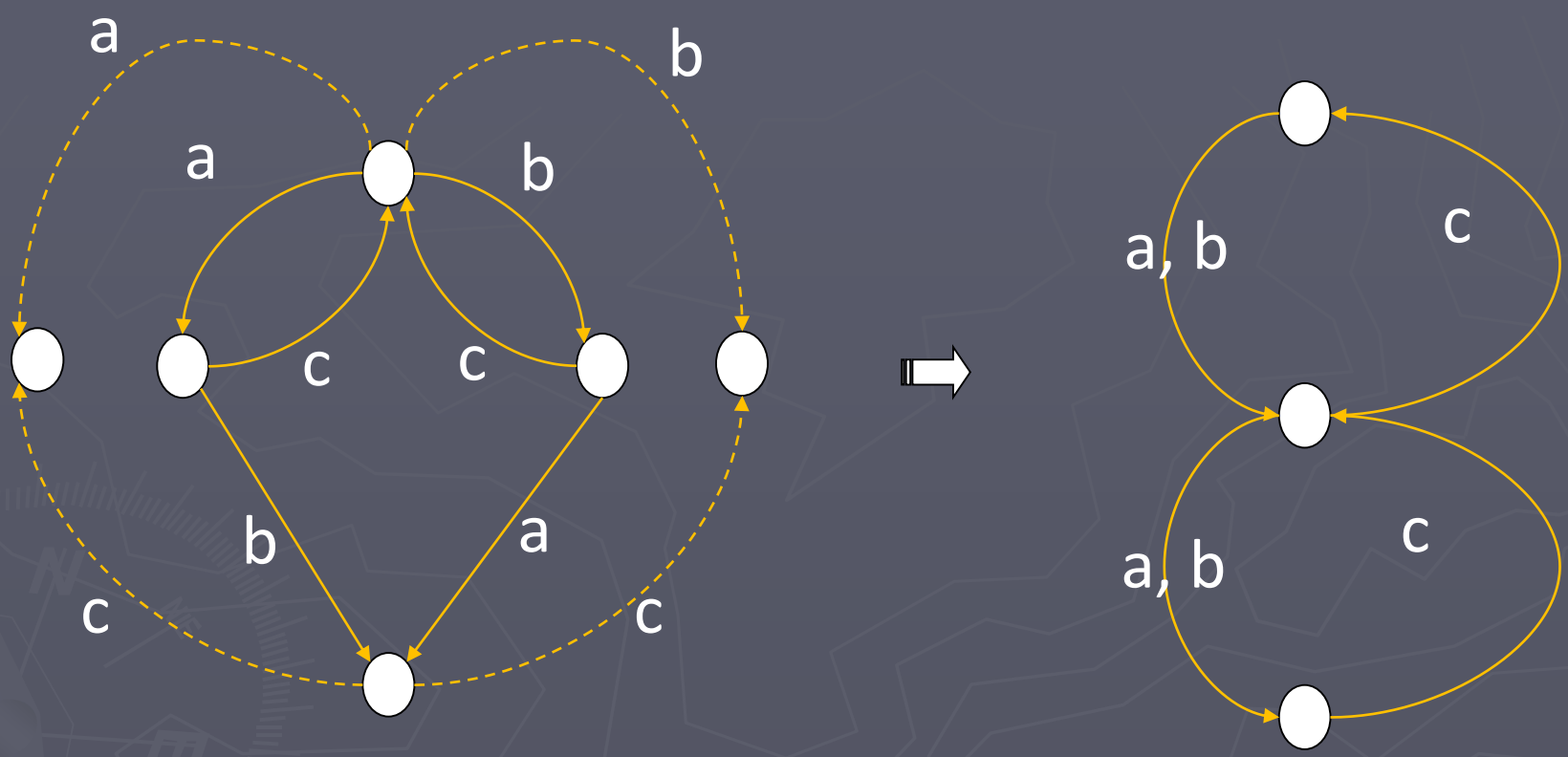
Some History

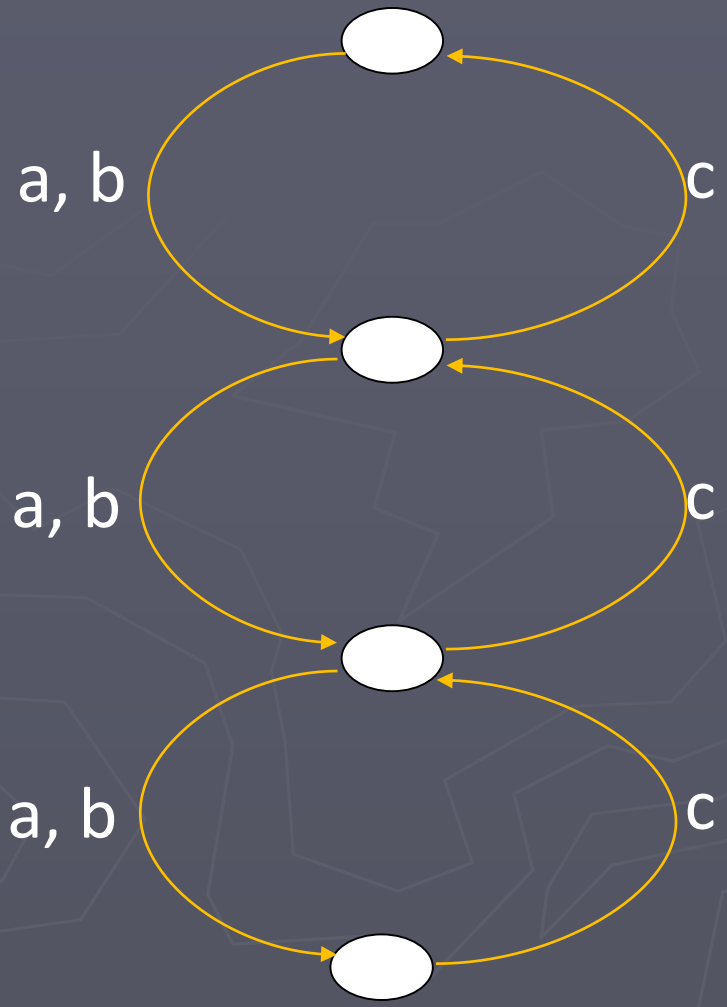
M.O.Rabin, D.Scott. Finite Automata and Their Decision Problems (1959)

M.Bird. The equivalence problem for deterministic two-tape automata (1971)

A combinatorial technique




a
b
a
b
c
c
b
a
c
c
a, b
c
a, b
c


a, b
c
a, b
c
a, b
c

Some History

T.Harju, J.Karhumaki. The equivalence problem of multitape finite automata (1991)

A purely algebraic technique

Per T. Harju

”the main argument of the proof uses an embedding result for ordered groups into division rings”.

Partially Commutative Semigroups

Godlevskii, A.B., Letichevskii, A.A. and Shukuryan, S.K., Reducibility of program-scheme functional equivalence on a nondegenerate basis of rank unity to the equivalence of automata with multidimensional tapes (1980).

G is a semigroup with a unit, generated by $Y=\{y_1, \ldots, y_n\}$.

G is called a **free partially commutative semigroup**, if it is described by a set of definitive assumptions of type $y_i y_j = y_j y_i$.

A homomorphism $K : F_Y \to F^n_{\{0,1\}}$ is defined

$$K(e) = (e, \ldots, e)$$

$$K(y_i) = (\alpha_{1i}, \ldots, \alpha_{ni})$$

$$\alpha_{ij} = \begin{cases} 1, i = j \\ e, y_i y_j = y_j y_i \\ 0, y_i y_j \neq y_j y_i \end{cases}$$

Partially Commutative Semigroups (Cont.)

The concatenation operation can be defined in one of the two alternative ways:

1. Right concatenation: $K(y_i y_j) = (a_{1i}a_{1j}, \ldots, a_{ni}a_{nj})$
2. Left concatenation: $K(y_i y_j) = (a_{1j}a_{1i}, \ldots, a_{nj}a_{ni})$

The second way is used in this work.

Concatenation for semigroup elements $a = y_{i1}\ldots y_{ik}$ and $b = y_{j1}\ldots y_{jl}$ is defined in the following way:

$$K(\mathrm{ab}) = \mathrm{K}\big((a_{1i_k}\ldots a_{1i_1}, \ldots, a_{\mathrm{n}i_k}\ldots a_{\mathrm{n}i_1}), (a_{1j_l}\ldots a_{1j_1}, \ldots, a_{\mathrm{n}j_l}\ldots a_{\mathrm{n}j_1})\big)$$
$$= (a_{1j_l}\ldots a_{1j_1}a_{1i_k}\ldots a_{1i_1}, \ldots, a_{\mathrm{n}j_l}\ldots a_{\mathrm{n}j_1}a_{\mathrm{n}i_k}\ldots a_{\mathrm{n}i_1})$$

Partially Commutative Semigroups (Cont.)

Lemma 1. y_i, y_j – generators of G, $y_i \neq y_j$ $g_1 = y_i y_j$, $g_2 = y_j y_i$ – elements of G. $g_1 = g_2 \Leftrightarrow K(y_i y_j) = K(y_j y_i)$

Lemma 2. < - is a linear (lexicographical) order on Y . Any two different representations of the same element of the semigroup G are comparable in terms of the order < .

Lemma 3. Any free partially commutative semigroup of n generators is isomorphic to some sub-semigroup of Cartesian product of n free semigroups with two generators.

Binary Coding

$Y = \{y_1, y_2\}$

$y_1 y_2 \neq y_2 y_1$

$K : F_Y \rightarrow F^2_{\{0,1\}}$

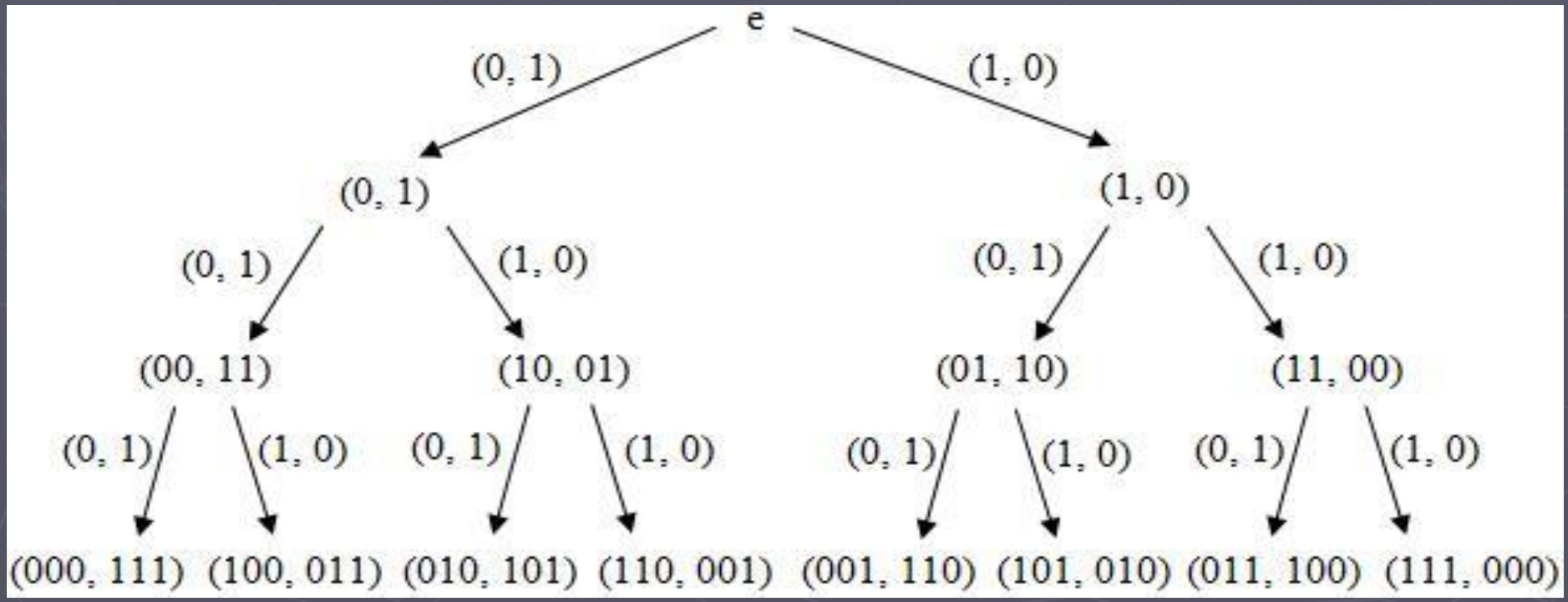
$K(e) = (e, e)$, $K(y_1)=(1, 0)$, $K(y_2)=(0, 1)$,

$K(y_1y_1)=(11, 00)=(3, 0)$,

$K(y_1y_2)=(01, 10)=(1, 2)$,

$K(y_2y_1)=(10, 01)=(2, 1)$,

$K(y_2y_2)=(00, 11)=(0, 3)$



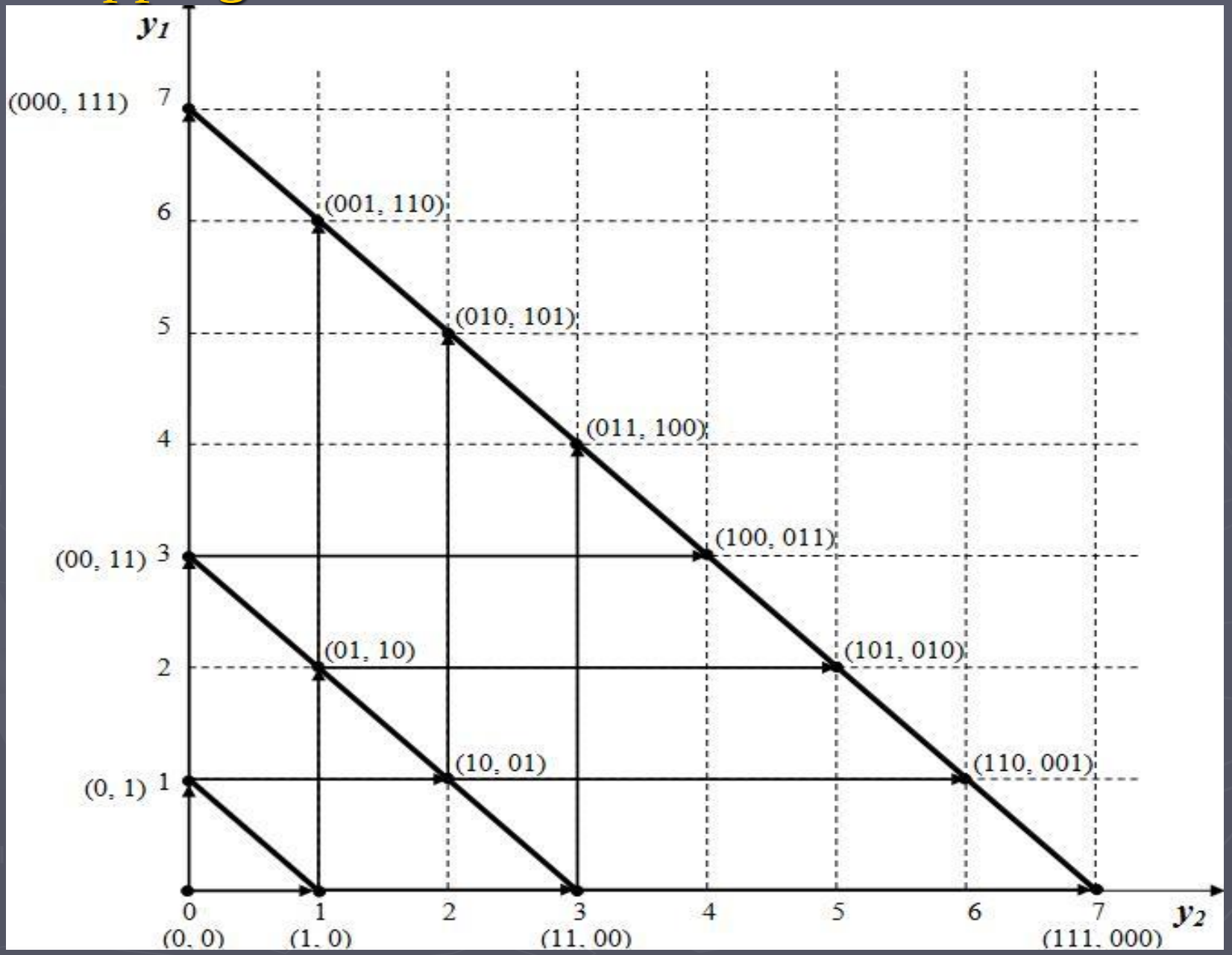
Partially Commutative Semigroups (Cont.)

We can consider an n-element tuple of integers for the given c_g (binary coded canonical form). It will be denoted $Num(c_g)$, where $Num : C_G \rightarrow N^n$ is a function, and will be used instead of c_g.

Lemma 4. From a known $T=Num(c_g)$ the corresponding $c_g=Num^{-1}(T)$ can uniquely be constructed (the mapping Num between the binary word tuples and the integer tuples is one-to-one).

Lemma 5. Any free partially commutative semigroup of n generators is isomorphic to some sub-semigroup of the n-dimensional space, where the semigroup operation is the left concatenation of integer tuples defined above.

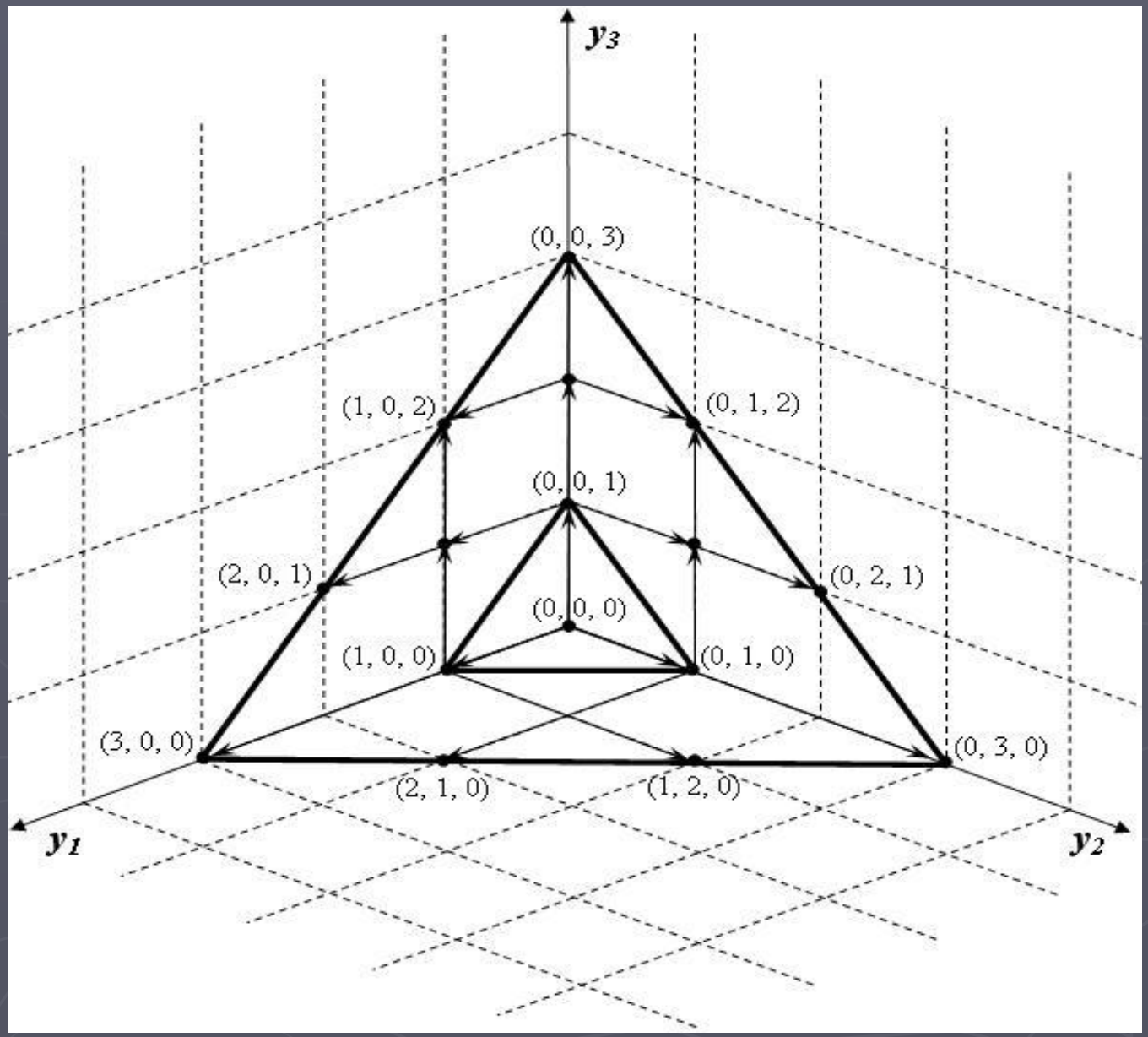
Mapping of Non-Commutative Elements to N^2



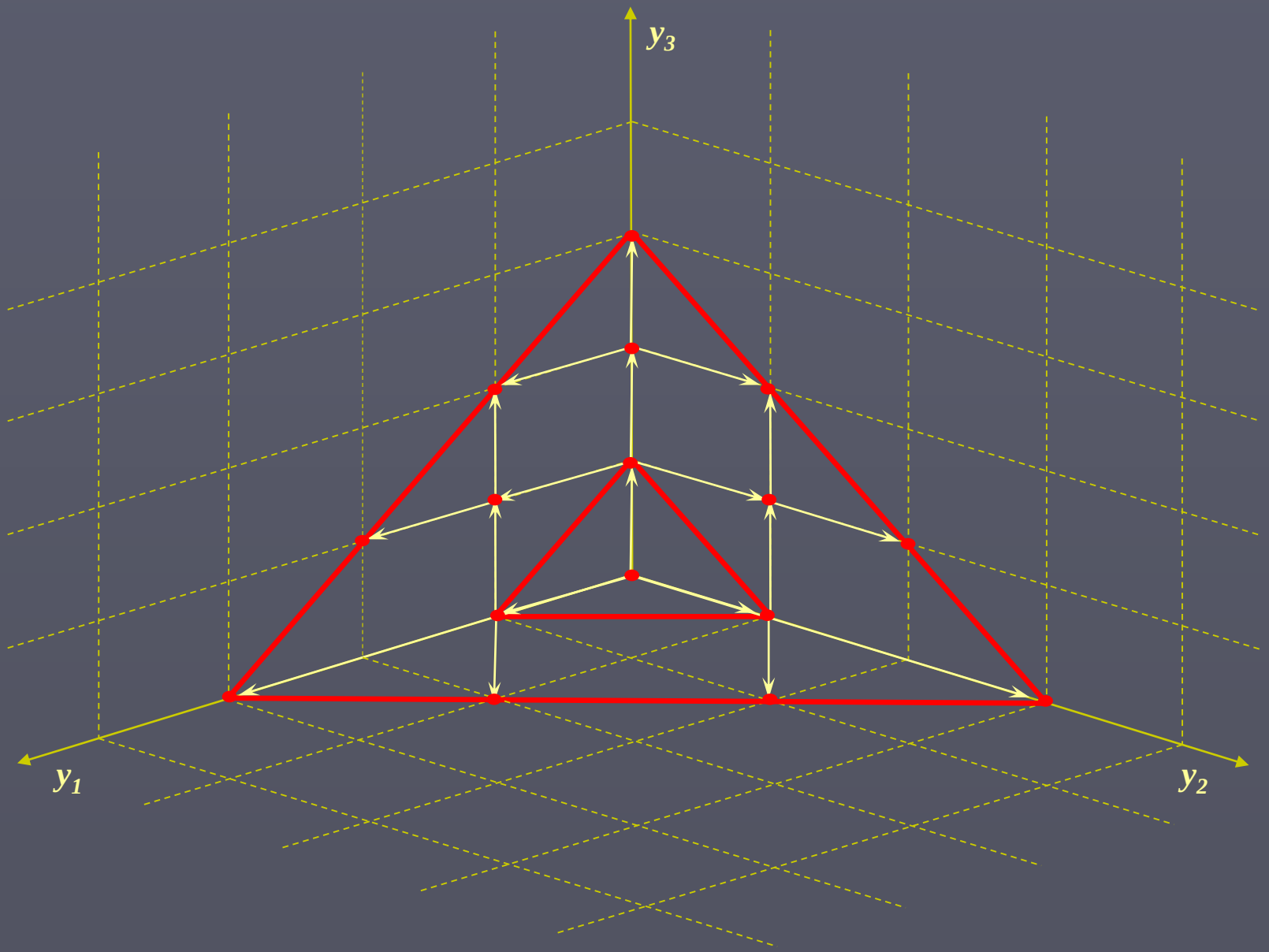
Mapping of a Partially Commutative Semigroup to N^3

$$Y = \{y_1, y_2, y_3\}$$

$$y_1 y_3 = y_3 y_1$$
$$y_2 y_3 = y_3 y_2$$
$$y_1 y_2 \neq y_2 y_1$$



Mapping of a Partially Commutative Semigroup to N^3 (Cont.)

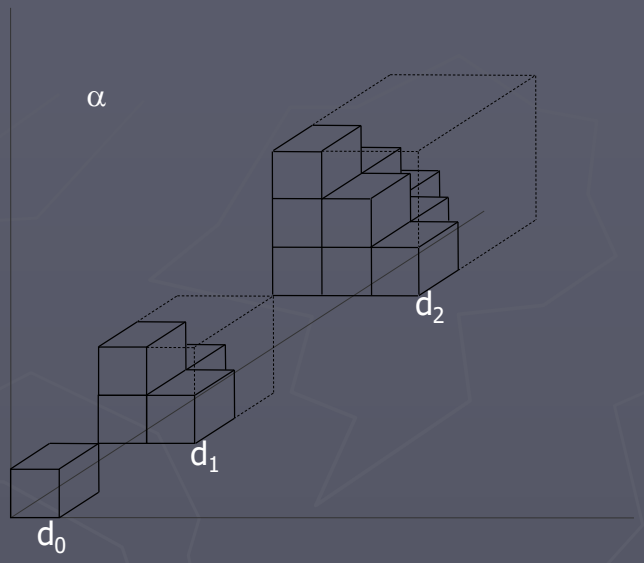


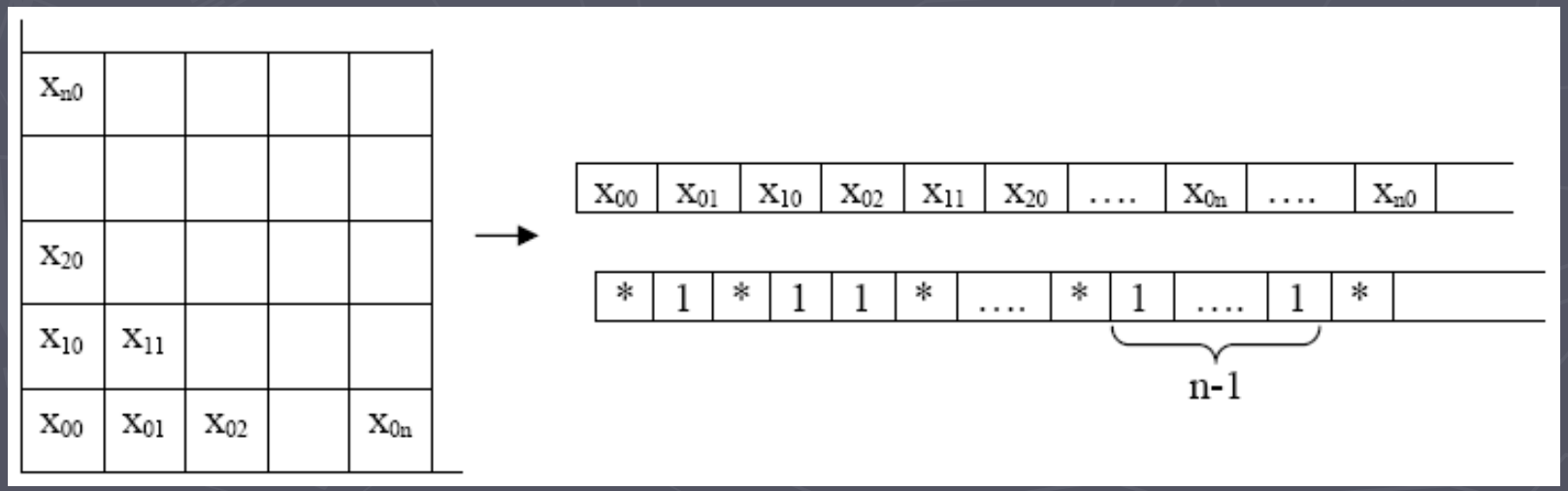
Multidimentional Tapes

Haik Grigorian, Samvel Shoukourian:
The equivalence problem of multidimensional multitape automata. J. Comput. Syst. Sci. 74(7): 1131-1138 (2008)

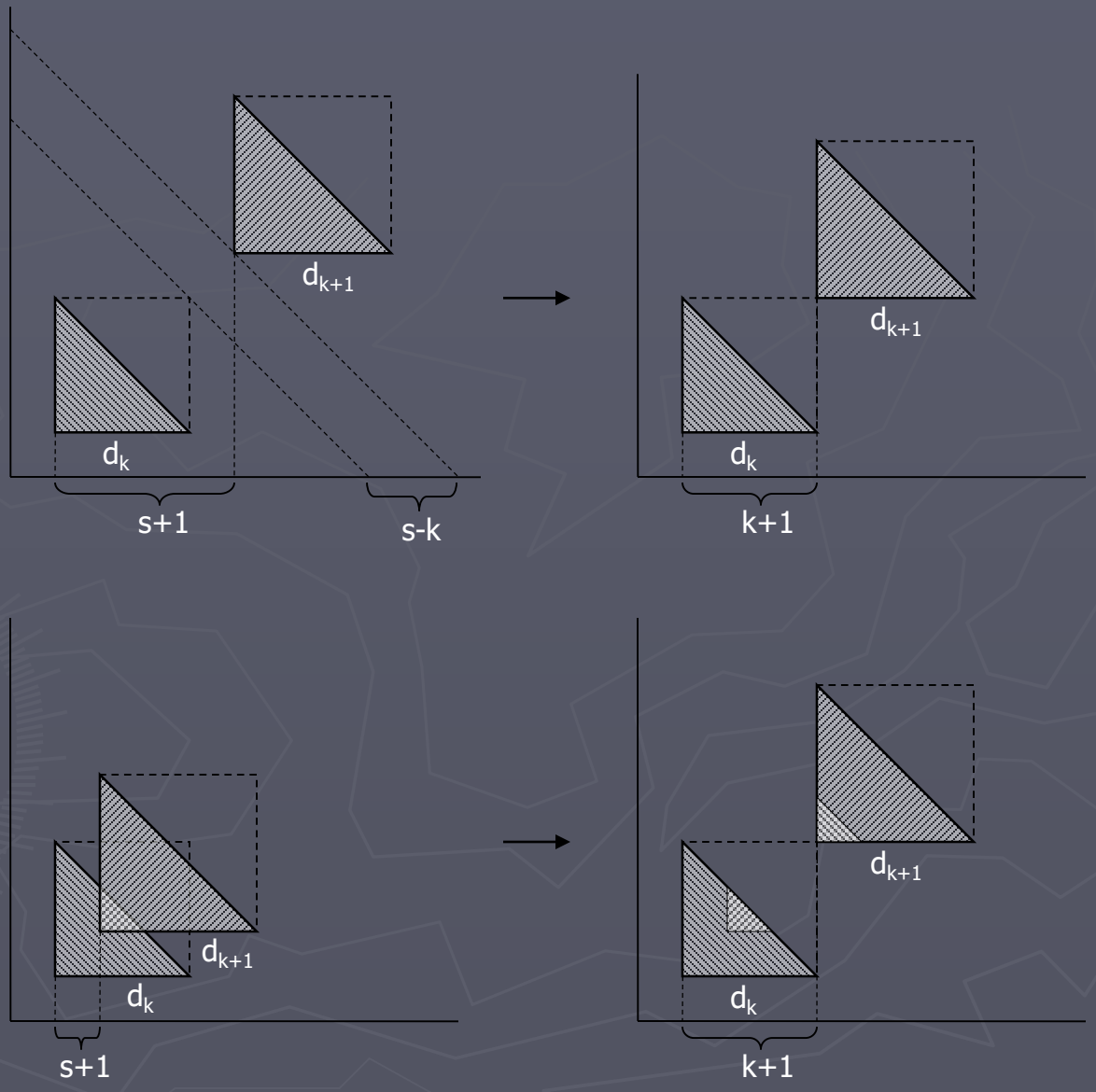
- A notion of diagonal
- A combinatorial proof
- A polynomial algorithm for equivalence problem of program schemata with non-degenerate operators

Coding of Multidimentional Tapes





Coding of Multidimentional Tapes (Cont.)



A New Combinatorial Proof for Equivalence Problem of Deterministic Multitape Finite Auromata

Alexander A. Letichevsky, Arsen S. Shoukourian, Samvel K. Shoukourian:

The Equivalence Problem of Deterministic Multitape Finite Automata: A New Proof of Solvability Using a Multidimensional Tape. LATA 2010: 392-402 (2010)

- A notion of essential diagonal
- Left concatenation
- A combinatorial algorithm for equivalence problem of deterministic multitape finite automata

Deterministic p-tape finite automaton

$A=<Y, S, \delta, F, s_0>$ - p-tape automaton in alphabet Y,

$$Y = \bigcup_{i=1}^{p} Y_i, Y_i \bigcap_{i \neq j} Y_j = \varnothing$$

S – set of states, $S = \bigcup_{i=1}^{p} S_i, S_i \bigcap_{i \neq j} S_j = \varnothing$

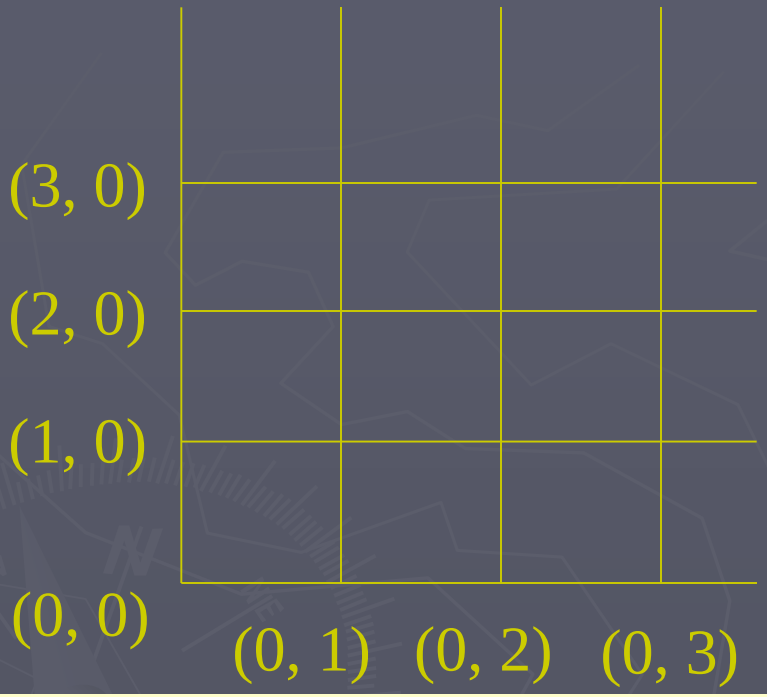
δ – transition function

F – set of final states

s_0 – initial state

Multidimensional Tape

n – positive integer, $N = \{0, 1, \ldots\}$, N^n – tape



(0, 0)- initial cell

$a_1 = (0, 0)$, $a_2 = (0, 1)$

$\pi(a_1, a_2)$ = true

$\pi_2(a_1, a_2)$=true

Cell $a_1=(\alpha_{11}, \ldots, \alpha_{1n})$ is called a **predecessor** of cell $a_2=(\alpha_{21}, \ldots, \alpha_{2n})$ if $\alpha_{2j}=\alpha_{1j}+(L+1)$, $\alpha_{2i}=\alpha_{1i}$, $i \neq j$, $L = \alpha_{11} + \ldots + \alpha_{1n}$

Execution Traces

$\varphi_A: N^n \rightarrow 2^S$ – the set of **all execution traces** of automaton A if and only if

1) $\varphi_A(0, \ldots, 0) = \{s_0\}$

2) $\varphi_A(a) = (\varphi_A(a^{(j)}_{pred}), y_j)$

Lemma. If for a given automaton A and a given cell a from $N^n \setminus \{(0, \ldots, 0)\}$ $\varphi_A(a)$ is defined then there exists a unique cell a' such that $\varphi_A(a')$ is defined and $\pi(a', a)$ is true.

Execution Traces

A part of a set of execution traces, where the sum of coordinates for each cell $\leq k - 1$ is called a **trace word** of the length *k*.

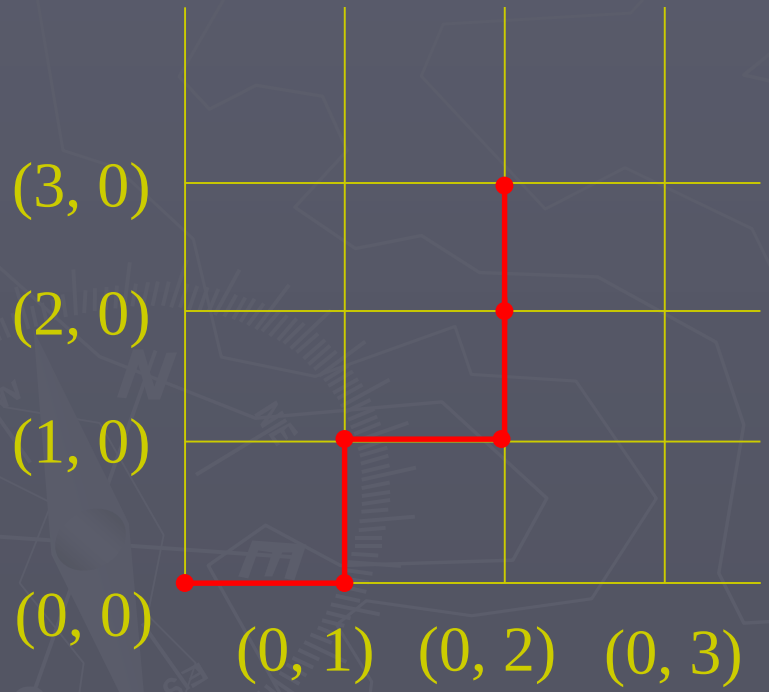
A part of a trace word having the sum of coordinates equal to *k* is called a **diagonal** *k*.

Its length is equal to k+1.

Diagonals of length one less than powers of two are called **essential**.

Paths in Trace Words

Path $p = a_{p_1} \ldots a_{p_m}$ for trace word ω is a sequence of cells that $\pi(a_{p_i}, a_{p_{i+1}})$ is true.



The path is **complete** if $a_{p_1} = (0,0)$

The path is **accepted (AP)** if $\varphi_A(a_{p_m})$ contains final state

Coordinates of the last cell - (3, 2) – **canonical form** (CF) of complete path

Characteristics of a Path

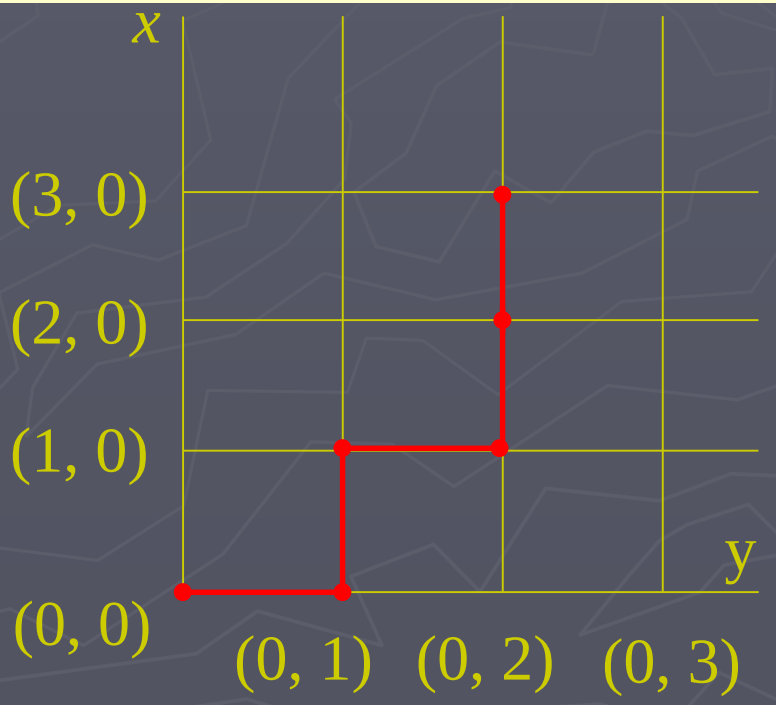
Characteristics $\chi = y_{p_1} \ldots y_{p_{m-1}}$ for path $p = a_{p_1} \ldots a_{p_m}$

$y_{p_j} \in Y_i,\ i \in \{1, \ldots, n\},\ j \in \{1, \ldots, m-1\} \Leftrightarrow$

coordinate i increases by 1 in cell $a_{p_{j+1}}$ in comparison with cell a_{p_j}

p = (0,0)(0,1)(1,1)(1,2)
(2,2)(3,2)

χ = yxyxx



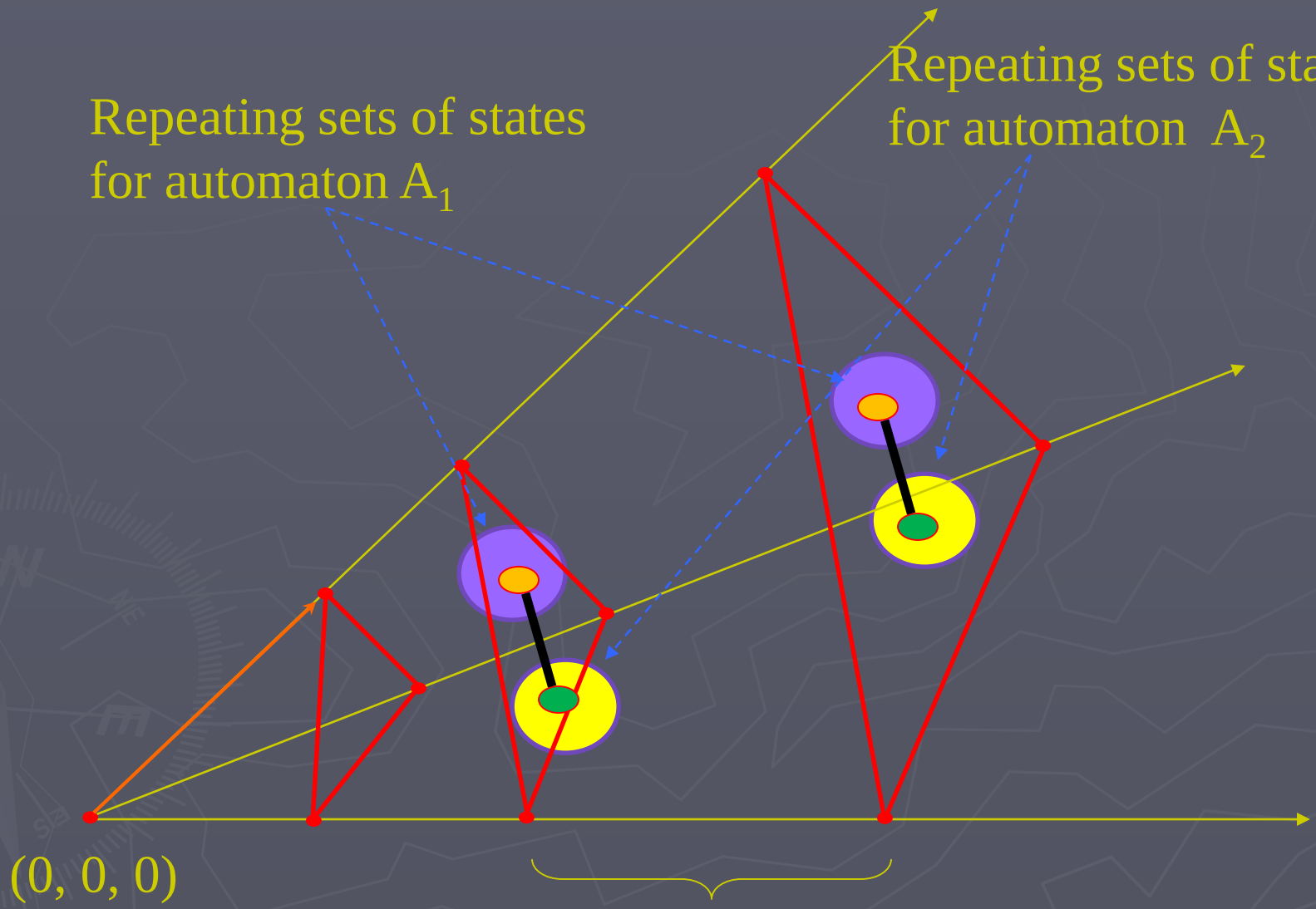
Sets of States on Essential Diagonals

$$\psi_A : D^{(E)} \rightarrow 2^{N^n \times S}$$

$$\psi_A(d_h^{(E)}) = \bigcup_{a \in U_{d_h^{(E)}}} (a, \varphi_A(a)), \forall d_h^{(E)} \in D^{(E)}$$

$\xi_A(d_h^{(E)}) \subseteq \psi_A(d_h^{(E)})$ - contains only final states

The Idea: Distance is Unchanged for Equivalent Automata



Equivalence Problem of Multitape Automata

A_1 and A_2 - deterministic multitape automata

ω_1 and ω_2 - trace words of length k.

$k = s^2 \times (2^s - 1)$, $s = |S_1| + |S_2|$.

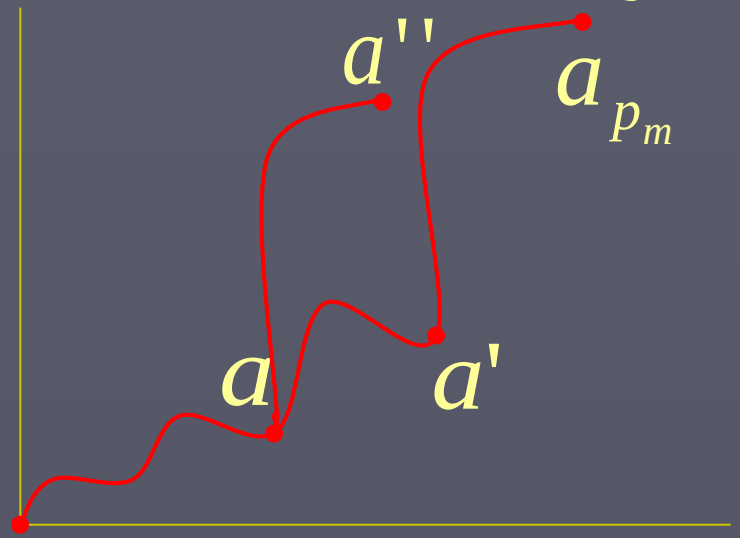
Lemma.

$$CF_{AP_{A_1}(\omega_1)} = CF_{AP_{A_2}(\omega_2)} \Leftrightarrow A_1 \sim A_2$$

Theorem. The equivalence problem of deterministic multitape automata is solvable.

Proof

$p\ p = a_{p_1} p_1 \ldots a_{p_m}, a_{p_g} > k \ldots a' \ldots a_{p_m}, m > k$



$\varphi_{A_j}(a) = \varphi_{A_j}(a')$

$p' = a' \ldots a_{p_m}$

$p'' = a \ldots a''$ → $\chi_{p'}$

$p''' = a_{p_1} \ldots a_{p_g}$

$p_{new} = p''' p'' = \underbrace{a_{p_1} \ldots a_{p_g} a \ldots a''}_{<=k}$

Algorithm for Merging Equivalent States

- Let r_1, $r_2 \in R$ be equivalence relations. The relation r_1 is called less than the relation r_2, $\boldsymbol{r_1 < r_2}$, if and only if every equivalence class of r_1 is included in some equivalence class of r_2.

- The relation r is named ***congruent*** if and only if for any a_1, $a_2 \in Q$, belonging to the same equivalence class of the relation r, the following statement is true: $\forall p \in F_X$ if $\delta(a_1, p)$, $\delta(a_2, p)$ are defined then these states belong to the same equivalence class of the relation r.

- To build an automaton without equivalent states for a given automaton A means to build a congruence relation on the set of states Q, which is the maximal relation in the sense of the relation $<$.

Algorithm for Merging Equivalent States (cont.)

- For the considered congruence relation and for any equivalence relation, which is more than this relation in the sense of <, the following property is true: for any a_1, $a_2 \in Q$, belonging to different equivalence classes of the equivalence relation there exists a word p over the alphabet X such that only one of the states $\delta(a_1, p)$, $\delta(a_2, p)$ belongs to F. Any equivalence relation, which has this property, is named ***correct***.

- To build the congruence relation:
  - Start from any correct equivalence relation.
  - Build a chain of reducing in the sense of < correct equivalence relations until the congruence relation is found.

A.A. Letichevskii, A.B. Godlevskii,Optimization of Algorithms during the Process of Their Design via a Method of Formalized Technical Specifications,Design Automation of Computers and Their Components, Academy of Sciences of Ukraine, Institute of Cybernetics, Kiev, 1977

А. А. Летичевский, А. Б.Годлевский, Оптимизация алгоритмов в процессе их проектирования методом формализованных технических заданий, Автоматизация проектирования ЭВМ и их компонентов, Акад. наук УССР, Ин-т кибернетики, Киев, 1977.

Algorithm for Merging Equivalent States (cont.)

- The result of a *splitting operation* on (L, c) by a pair (K, x) where $x \in X_i$ for some $i \in \{1, ..., n\}$ are $(L^+, c \bullet x)$ and $(L^-, (0, ..., 0))$, where:
- 1. $C(r)$ is the set of all equivalence classes for the equivalence relation r will be denoted
- 2. $K \in C(r)$
- *3.* $L^+ = \{a \mid a \in L, \gamma(a) = i, \delta(a, x) \in K\}$, $L^- = L \setminus L^+$
- *4.* $c \bullet x$ is the vector of coordinates for the concatenation

- A *splitting of a relation v by the pair* (K, x) is an operation which results in splitting of all elements of $C(v)$ paired with the corresponding coordinates, by this pair. The result of splitting of a relation is a relation which is named a *quotient* of the splitting operation and denoted $v/(K, x)$.

- **Lemma 19.** *For any equivalence relation r and any two subsets of Q :* $K, K' \subseteq Q$ *the following equation is true:*

$$(r/(K, x))/(K', x') = (r/(K', x'))/(K, x)$$

- **Lemma 20.** *Let r be a correct equivalence relation, r′ be an equivalence relation and* $r < r'$, $K \in C(r')$. *Then the equivalence relation* $r/(K, x)$ *is correct for any* $x \in X$.

- **Lemma 21.** *Let r be an i-equivalence relation,* $C(r) \times X = \{(K, x) \mid x \in X, K \in C(r)\}$. *Then the equivalence relation* $[r/C(r) \times X]_i$ *is an* i +1*-equivalence relation.*

- **Lemma 22.** *If* $r/C(r) \times X = r$ *then r is a congruence relation.*

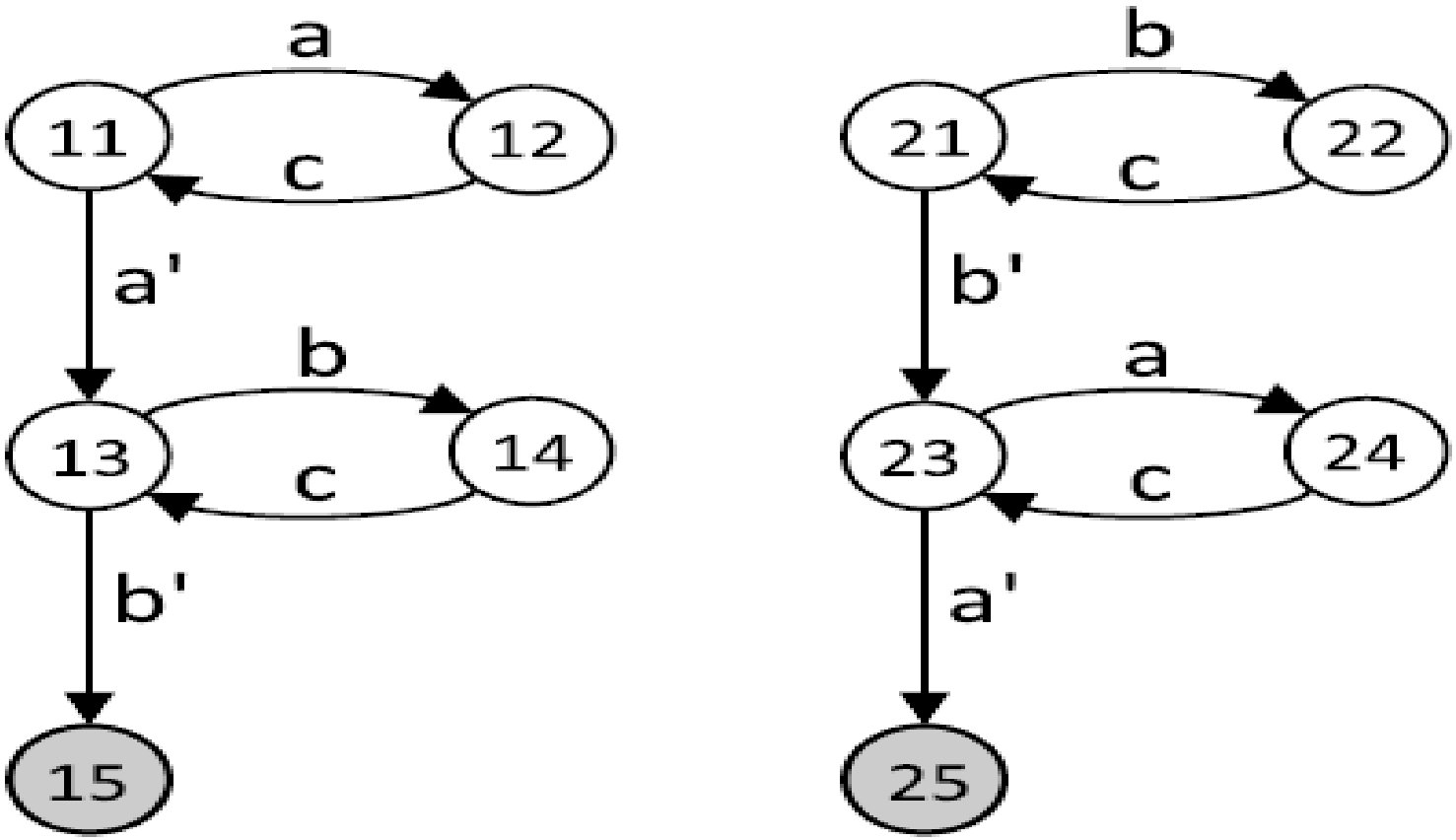
Algorithm for Merging Equivalent States (cont.)

Algorithm CB (Congruence Builder):

set r' equal to $\{(F, (0, \ldots, 0)), (Q \setminus F, (0, \ldots, 0))\}$
set i equal to 0
CLOSURE: **set** r equal to r'
set *SPLITTER* equal to $C(r) \times X$
NEW_RELATION: **while** $SPLITTER \neq \emptyset$ **do**
(**extract** $s \in SPLITTER$
set i equal to $i + 1$
set r' equal to $[r'/s]_i$)
if $r \neq r'$ **then go to** CLOSURE
merge equivalence classes

extract is interpreted as selecting an element from a given set and removing it from the set.
merge is interpreted as a union of all the equivalence classes that have the same coordinate.

Example of Equivalent MDFAs



S.K. Shukuryan, Recognition of Properties for Discrete Transformers and Multitape Multihead Automata, Thesis submitted for fulfillment of require-ments to the degree of Candidate of Science in Physics and Mathematics in specialty 01.01.09 “Mathematical Cybernetics”, National Academy of Sciences of Ukraine, Institute of Cybernetics, Kiev, May 1977.

Complexity Estimation of the Algorithm

- **Theorem**. *The complexity of the algorithm for determining the equivalence of states of a deterministic multitape finite automaton is $O(s^{n+1})$, where s is the number of states of the automaton (consequently, if we are checking the equivalence of 2 automata, then $s = s_1 + s_2$) and n is the number of tapes.*

- There are 2 loops in Algorithm CB – the inner loop NEW_RELATION is nested in the outer loop CLOSURE. The loop NEW_RELATION implements the closure operation for the set SPLITTER and builds a new r' relation which is less than the current relation r. The loop CLOSURE iterates until the built relation does not differ from the current one.

- Let us estimate the number of steps of the inner loop of the algorithm – the number of elements of the set SPLITTER. There are 3 possible cases.

Complexity Estimation of the Algorithm (cont.)

Case 1 - movement from one essential diagonal to another, i.e. when read symbols are commutative with each other. In this case, at most one symbol from each tape can participate (possibly multiple times) in a considered word:

$$p = x_{p_1}^{j_1} \dots x_{p_k}^{j_k}, x_{p_1} \in X_{p1}, \dots, x_{p_k} \in X_{pk},\ j_i \geq 1,\ i = 1, \dots, k,\ k \leq n,\ p_i \in \{1, \dots, n\}$$

Then, the number of words to be considered for building of equivalence classes for a given iteration will be:

$$\left|C(r) \times X\right| = s \times (s - j_1) \times \left(s - (j_1 + j_2)\right) \times \dots \times \left(s - (j_1 + \dots + j_{k-1})\right) \times \overline{X_{p1}} \times \dots \times \overline{X_{pk}} \leq s^k \times M^k \leq M^k \times s^n$$

Case 2 - movement between essential diagonals, i.e. when read symbols are not commutative with each other. It is sufficient to consider sequences of symbols which have a length less than or equal to the number of automaton states, because consideration of words with more length can be reduced to the consideration of the mentioned set. So we get:

$$p = x_{p_1}^{j_1} \dots x_{p_k}^{j_k}, x_{p_1}, \dots, x_{p_k} \in X_c,\ c \in \{1, \dots, n\},\ j_i \geq 1,\ k \leq n$$

So the estimation will be

$$|C(r) \times X| = s \times (s - j_1) \times (s - (j_1 + j_2)) \times \dots \times (s - (j_1 + \dots + j_{k-1})) \times \underbrace{\overline{X_c} \times \dots \times \overline{X_c}}_{k} \leq s^k \times M^k \leq M^k \times s^n$$

Complexity Estimation of the Algorithm (cont.)

Case 3 - movement crossing essential diagonals, when some of the read symbols are commutative with each other and some are not.

$$p = x_{p_1}^{j_1} \ldots x_{p_k}^{j_k}, x_{p_1} \in X_{p1}, \ldots, x_{p_k} \in X_{pk}, j_i \geq 1, i = 1, \ldots, k, k \leq n, p_i \in \{1, \ldots, n\}$$

The only difference here from case 1 is that $p_1, \ldots, p_k$ need not be distinct. So the same estimation of $O(s^n)$ will also hold for this case.

- It follows from the finiteness of the set of states that the number of algorithm iterations for the outer loop CLOSURE (the number of considered essential diagonals) for determination of the congruence relation is restricted by *s*.
- The overall complexity of the suggested algorithm can be estimated as $O(s^{n+1})$.

Instead of a Conclusion: Behind the Obtained Result

1. To use the introduced coding for representation of sets of word tuples (languages) accepted by multitape finite automata. To use the existing notation of regular expressions for one-tape automata as a notation for these languages accepted by MFA via interpreting the "concatenation" operation differently - as it is defined in the paper.

2. To formulate and prove theorems of analysis and synthesis for multitape finite automata.

3. To consider a metric space of regular events for multitape finite automata

Thank You

Q & A