The Formal Language Theory Column

by

Giovanni Pighizzini

Dipartimento di Informatica
Università degli Studi di Milano
20135 Milano, Italy
pighizzini@di.unimi.it

Some Results on Regular Events for Multitape Finite Automata: A Preliminary Report

Alexander Godlevsky
National Academy of Sciences of Ukraine,
Glushkov Institute of Cybernetics
a.godl49@gmail.com

Hayk Grigoryan
IT Educational and Research Center,
Yerevan State University
hgrigorian@gmail.com

Tigran Grigoryan
IT Educational and Research Center,
Yerevan State University
tigran.grigoryan1995@gmail.com

Samvel Shoukourian
IT Educational and Research Center,
Yerevan State University
samshouk@sci.am

Abstract

A new representation of languages for multitape finite automata is considered based on a special binary coding of elements in a free partially commutative semigroup. The mentioned coding was used previously for solution of several problems in theory of automata. An overview of these results, as well as, related works are presented.

Some new results based on this representation, which are currently under review, are formulated. They include: a new characterization for commutation classes of free partially commutative semigroups (trace monoids); a metric, based on the introduced characterization, and a metric space for regular events over a free partially commutative semigroup; the synthesis of a multitape finite automaton for a given regular expression; and a method for the approximate calculation of distance between regular events for multitape finite automata.

1 Introduction

A new representation of languages for multitape finite automata (MFA) and some new results for the representation, which are currently under review, are adduced.

In 1959 M. O. Rabin and D. S. Scott introduced deterministic (DMFA) and nondeterministic (NMFA) multitape finite automata and considered their equivalence problem [1].

The equivalence problem for NMFAs turned out to be undecidable [2].

The equivalence problem of deterministic two-tape finite automata was proved to be solvable in 1973 by M. Bird [3].

In 1991 T. Harju and J. Karhumäki proved the solvability of the equivalence problem for DMFA without any restriction on the number of tapes via a purely algebraic technique [4].

In 2010, a combinatorial proof of solvability was given [5], which led to a polynomial algorithm for the problem [6]. The proof of solvability is similar to the solution suggested by M. Bird, but instead of the transformation of source automata to a commutative diagram, the commutativity assumptions are taken into account via a multidimensional tape used for coding execution traces. The referred combinatorial proof also led to a solution of the equivalence problem for processes in specific object-oriented environments [7].

Automata with multidimensional tapes, where the motion of the heads is monotone in all directions (no backward motion), were introduced earlier for the solution of the equivalence problem of program schemata with nondegenerate operators [8, 9, 10, 11, 12].

It was shown that the equivalence problem for program schemata with nondegenerate basis of rank unity is reducible to the problem of automata with multidimensional tapes. Later, in [13], it was proven that the equivalence problem of multidimensional multitape automata can be reduced to the equivalence problem of DMFA, thus, solving the open problem.

Coming back to the equivalence problem of DMFA, in 2013, J. Worrell proved that a probabilistic polynomial bound exists for the problem. The proof is basing on consideration of matrix algebras [14]. Directly compared, the used techniques in [5] and [14] are different.

For the proof of solvability in [5], a special binary representation/coding of an element in a free partially commutative semigroup is used, which brings to a new characterization of commutation classes of free partially commutative semigroups, also called Cartier-Foata commutation monoids [15], or trace monoids after Mazurkiewicz [16] and his followers in computer science [17, 18, 19, 20, 21], the connection of which with free partially commutative semigroups was first observed in [17, 18]. This characterization is stronger than the already known one [21] in the sense that it requires less information to characterize commutation classes.

Problems, formulated in [22] as a synthesis problem, aka synthesizing a finite automaton from a given regular expression, and an analysis problem, aka building a regular expression for the set of words accepted by a given finite automaton, were solved for one-tape automata by 1960 [23]. Then a natural question arises: is there a way to have a similar to regular expressions representation for the set of n-tuples of words accepted by a given MFA? In the case of a positive answer, not less natural continuation is to consider a distance between regular events and a metric space of regular events, similarly to the case of one-tape automata.

Some notable, from our point of view, attempts to consider in that way languages accepted by multitape automata are briefly discussed below.

In [24], B. G. Mirkin has considered a special coding for the sets of words tuples accepted by multitape automata. The proofs and discussions are only for the case of $n = 2$ and there are no explanations/proofs on how to extend this to the case of $n > 2$. Meantime, the considerations in [5] show that a direct use of the representation from [24] will not be sufficient for the case $n > 2$.

Another paper [25] by P. H. Starke is dedicated to the following result: "An n-ary relation R over $W(X)$ is representable by a finite deterministic n-tape automaton if and only if there exists an admissible regular expression T such that $R = Val_n(T)$". In the same paper, the author mentions that "Unfortunately there are non-admissible regular expressions T such that $Val_n(T)$ is representable by a deterministic automaton".

The coding considered in [5] was used in [26] to define regular expressions and regular events for multitape automata. The existing notation of regular expressions for one-tape automata [22] was re-used as a notation for languages accepted by MFA via interpreting differently the "concatenation" operation.

For a given alphabet with n letters, a free semigroup G with a unit, $Y = \{y_1, \ldots, y_n\}$ generators, where each letter corresponds to a unique generator of G and a finite set of relations of type $y_i y_j = y_j y_i$ is considered. Thus, every word in the given alphabet has a corresponding element in the semigroup G. A homomorphism is defined from G to n-element vector space of binary strings. The concatenation of images of two words on n-element vector space of binary strings is performed by pairwise concatenating i-th component of the second n-element vector to the *left* of i-th component of the first n-element vector for all $i = 1, \ldots, n$. The formal definition of this new interpretation for the "concatenation" operation is given in Section 2.

Results adduced in [26], allow us to state that already existing metrics defined for regular languages for one-tape automata are applicable in the case of regular languages over a free partially commutative semigroup. Some of them are presented below.

In 1965 V. G. Bodnarchuk has introduced a metric on the linear space of events [27], which was later called the Bodnarchuk metric [28]. V. Vianu has investigated many topological properties of this metric and its link with the topology of the learning space [28].

The edit distance, specifically, Levenstein distance [29, 30], can also be applied to the regular languages for multitape automata. Though, in this case some adjustment is necessary for $d(w_1, w_2) = 0 \Leftrightarrow w_1 \equiv w_2$ axiom to take place. For this reason, we redefine the edit distance to be L_2 norm of a vector, in which at each position i, the original edit distance between the sub-words of w_1 and w_2, having all and only the letters from i-th tape, are written.

Other known metrics, e.g. [31], can also be considered in our case.

However, from the practical point of view, it is impossible to calculate these metrics. In general case, it is impossible to answer the question: when is the distance between two regular languages for multitape automata equal to 0? This is due to the fact, that the intersection problem for MFA, hence the equivalence problem as well, is not solvable [1, 2].

Due to that, a new approach is developed, taking the advantage of the introduced coding in defining the distance function. The first attempt was to use the fact that the introduced binary coded tuples were equivalent to some sub-semigroup of n-dimensional space $\mathbb{N}^n$ [6]. This fact allows to consider the distance between words over a free partially commutative semigroup as an Euclidean distance between them. Yet, the considered distance is inadequate in some boundary cases.

A new metric presented in this work, is basing on the newly introduced characterization vectors of commutation classes.

An approximate method is proposed for calculating this distance, which in some cases, precisely calculates the distance between regular events. In other cases, it calculates the approximate distance, by performing those calculations on finite subsets of regular events extracted from the algorithm for determining the equivalence between DMFAs, adduced in [6]. The complexity of the later one is bounded by $k \cdot s^{n+1}$, where k is some constant, n is the number of tapes and s is the total number of states in the considered automata.

In Section 2, the mentioned and other results on regular events over a free partially commutative semigroup and multitape automata are presented. Results which are currently in the review process for publication are highlighted specially.

2 Formulation of Results on Regular Events for Multitape Finite Automata

First, recall some basic definitions.

If X is an alphabet, then the set of all words in the alphabet X, including the empty word, will be denoted X^*, and the set of all n-element tuples of words will be denoted $(X^n)^*$.

Let G be a semigroup with a unit, generated by the set of generators $Y = \{y_1, y_2, \ldots, y_n\}$. G is called free partially commutative semigroup, if it is defined by a finite set of definitive assumptions of type $y_i y_j = y_j y_i$.

Let $K : Y^* \to (\{0,1\}^n)^*$ be a homomorphism over the set Y^*, which maps words from Y^* to n-element vectors in binary alphabet $\{0,1\}$. The homomorphism K over the set of symbols of the set Y is defined by the equation:

$$K(y_i) = (a_{1i}, \ldots, a_{ni}), \text{ where } a_{ij} = \begin{cases} 1, & i = j \\ e, & y_i y_j = y_j y_i \\ 0, & y_i y_j \neq y_j y_i \end{cases}$$

At the same time $K(e) = (e, \ldots, e)$.

$K(y_i y_j),\ i \neq j$ can be defined in any 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: $K(y_i y_j) = (a_{1j} a_{1i}, \ldots, a_{nj} a_{ni})$.

In [5, 6], the left concatenation was used, which allows to consider the n-tuples of binary coded words as vectors in $\mathbb{N}^n$. Left concatenation will be used throughout this paper as well, and will be called just concatenation.

Any element in $K(G)$ can be represented in the form $k_{i_1} \ldots k_{i_m}$, where $\forall j, k_{i_j} \in K(Y)$ [12]. Note that an element may have multiple representations of this form. Hence, an equivalence ρ_K is defined over $K(Y)^*$ - the set of all words in $K(Y)$, i.e., if $p, q \in K(Y)^*$, then $p \rho_K q$ if and only if p and q are the representations of the same element in $K(G)$. Denote the equivalence class of p by $[p]$.

Using the mapping K with the left concatenation, a regular event and a regular expression over a free partially commutative semigroup of binary coded n-tuples are defined, similarly to [22].

For a free partially commutative semigroup G with the set of generators Y, a regular event over $K(Y)$ is defined as follows [26]:

1. $\emptyset$ (empty set) is a regular event in $K(Y)$.
2. $\tilde{E} = \{[e, \ldots, e]\}$ is a regular event in $K(Y)$.
3. $\forall y \in Y, \{[K(y)]\}$ is a regular event in $K(Y)$.

4. If P and Q are regular events in $K(Y)$, then so are
   (a) $P + Q = P \cup Q$,
   (b) $PQ = \{[s] \,|s = pq, [p] \in P, [q] \in Q\}$, where pq is the concatenation of p and q as defined above,
   (c) $P^* = \bigcup_{n\geq 0} P^n$, where $P^0 = \tilde{E}$, $P^n = PP^{n-1}$, for $n \geq 1$.
5. There are no any other regular events in $K(Y)$.

The above definition of regular event not only is different from the known definition of regular events by the interpretation of the concatenation operation, but it defines a regular event as a set of equivalence classes. For simplicity, we will use the phrase "word p belongs to the regular event S" to indicate that "the equivalence class $[p]$ belongs to S".

A regular expression over $K(Y)$ is defined as follows [26]:

1. $\emptyset$ is a regular expression, denoting the regular event $\emptyset$.
2. $K(e) = (e, \ldots, e)$ is a regular expression, denoting the regular event $\tilde{E}$.
3. $\forall y \in Y, K(y)$ is a regular expression, denoting the regular event $\{[K(y)]\}$.
4. If p and q are regular expressions in $K(Y)$, denoting regular events P and Q correspondingly, then so are
   (a) $p + q$, denoting the regular event $P + Q$,
   (b) pq, denoting the regular event PQ,
   (c) $p^* = \bigcup_{n\geq 0} p^n$, where $p^0 = K(e)$, $p^n = pp^{n-1}$, for $n \geq 1$, denoting the regular event P^*.
5. There are no any other regular expressions in $K(Y)$.

2.1 A New Characterization for Commutation Classes

A new characterization for commutation classes of free partially commutative semigroups (trace monoids [16]), which is stronger than the already known one (Projection Lemma [21]) in the sense that it requires less information to characterize commutation classes, is presented in this sub-section. This result is currently under review.

Lemma 2.1 (Projection Lemma [21]). *Let u and v be two traces of the trace monoid $M(\Sigma, I)$, then we have $u = v$ if and only if $\pi_{\{y,y'\}}(u) = \pi_{\{y,y'\}}(v)$ for all $yy' = y'y$, where π_A is a canonical homomorphism, which erases all the letters not belonging to A from a trace.*

We define $Ocr_y(w)$ as the number of occurrences of the letter y in the word w, and for a set of letters $Z \subset Y$, $Ocr_Z(w) = \sum_{y\in Z} Ocr_y(w)$. Also, define $\sigma_{y,i,j}(w), i < j$ as the sub-word of the word w such that the first letter of that sub-word is the i-th occurrence of the letter y in the word w and the last letter is j-th occurrence of the letter y in w. For the case $i = 0$, $\sigma_{y,i,j}(w)$ is a prefix of the word w, and for the case $j > Ocr_y(w)$, $\sigma_{y,i,j}(w)$ is a suffix of w.

Let w be a word on a partially commutative alphabet Y. For an element $y \in Y$, define the vector of numbers $C_y(w) = (n_0, n_1, \ldots, n_k)$ such that $n_i = Ocr_{\{y' \in Y | yy' \neq y'y\}}(\sigma_{y,i,i+1}(w))$, where $k = Ocr_y(w)$ and $i = 0, \ldots, k$. In other words, $C_y(w)$ is a vector having length equal to the number of occurrences of the symbol y in the word w plus 1. It is also greater by 1 from the number of 1s in the binary word corresponding to the symbol y in $K(w)$. The elements of the vector $C_y(w)$ are the numbers of 0s between each pair of successive 1s, except the first and last elements, which are the maximum numbers of consecutive 0s as a suffix and as a prefix correspondingly.

Theorem 2.2. *Let g_1 and g_2 be elements of partially commutative semigroup G with generators $\{y_1, y_2, \ldots, y_n\}$, then $g_1 = g_2$ if and only if*

$$C_{y_i}(g_1) = C_{y_i}(g_2) \textit{ for all } i = 1, \ldots, n.$$

Theorem 2.2 shows that the vectors $C_{y_i}(w)$ characterize commutation classes.

Lemma 2.3. *The characterization vectors C_y can be directly obtained from the projections $\pi_{\{y,y'\}}$.*

Lemma 2.3 means that the characterization of commutation classes given in Theorem 2.2 is stronger than the one given by Projection Lemma.

2.2 A Metric Space for Free Partially Commutative Semigroups

Next, basing on this characterization, we define a metric space. The results on this metric are currently under review.

For a letter y and semigroup elements g_1, g_2, the vectors $C_y(g_1)$ and $C_y(g_2)$ might have different lengths. To be able to perform vector operations on them, we add -1-s in the end of the smaller vector. Denote the new vectors by $C'_y(g_1)$ and $C'_y(g_2)$. Define the distance between $g_1 \in G$ and $g_2 \in G$, where G is a free partially commutative semigroup with the set of generators $\{y_1, y_2, \ldots, y_n\}$, as follows:

$$d(g_1, g_2) = \left\| \left(\left\| C'_{y_1}(g_1) - C'_{y_1}(g_2) \right\|_2, \ldots, \left\| C'_{y_n}(g_1) - C'_{y_n}(g_2) \right\|_2 \right) \right\|_2,$$

where $\|v\|_2$ is L_2 norm of the vector v.

Lemma 2.4. *The distance function d is metric for any free partially commutative semigroup.*

Lemma 2.4 allows us to define Hausdorff distance [32] on the set of all, except empty, regular events over a free partially commutative semigroup. Let $E_1 \neq \emptyset$ and $E_2 \neq \emptyset$ be regular events over a free partially commutative semigroup. d_H distance between E_1 and E_2 is the quantity

$$d_H(E_1, E_2) = max\left\{\sup_{r_1 \in E_1} \inf_{r_2 \in E_2} d(r_1, r_2), \sup_{r_2 \in E_2} \inf_{r_1 \in E_1} d(r_1, r_2)\right\}.$$

The calculation of d_H, generally, is not solvable. For this reason, a method for the approximate calculation of d_H is introduced in Section 2.4.

2.3 Synthesis of Multitape Finite Automata

We consider two models of MFAs: the Rabin-Scott model [1] and the one called mixed-state model [33]. The difference between them is that unlike in Rabin-Scott model (MFA), each state of automata in mixed-state model (MFA-MSM) is not bound to a specific tape and can have transitions using symbols from different tapes. The formal definitions of these two models are given below.

Let Q be a finite set of states, X be an input alphabet, $\delta : Q \times X \to 2^Q$ be the transition function, $q_0 \in Q$ be the initial state and $F \subseteq Q$ is the set of final states. Suppose that X can be divided into disjoint, ordered subsets $X = X_1 \cup \ldots \cup X_n$, such that $X_i \bigcap_{i \neq j} X_j = \emptyset$ and $\forall x, x' \left(x \in X_i, x' \in X_j (i \neq j), xx' = x'x\right)$. Each subset X_i corresponds to i-th tape.

Definition 2.5 (Rabin-Scott model [1]). *Let $T : Q \to \{1, \ldots, n\}$ be a tape function, which associates each state from Q with a certain tape. An n-tape automaton is called a tuple $A = (Q, T, X, \delta, q_0, F)$, where $Q = \bigcup_{i=1}^{n} Q_i$, such that $Q_i = \{q | q \in Q, T(q) = i\}$ $\forall i = 1, \ldots, n$.*

Definition 2.6 (Mixed-state model [33]). *An n-tape automaton is called a tuple $A = (Q, X, \delta, q_0, F)$.*

In order to calculate the approximate distance between two regular events over a free partially commutative semigroup R_1 and R_2 using the polynomial equivalence algorithm [6], we first need to transform those regular expressions to DMFAs accepting similar languages to the ones recognized by R_1 and R_2.

Theorem 2.7 (On synthesis of multitape automata). *There exists an algorithm which, for a given regular expression over a free partially commutative semigroup R, synthesizes a NMFA with ε transitions A (NMFA-ε), such that the language accepted by A is the same language recognized by R.*

The proof of the theorem is straightforward. It is necessary to apply "Thompson's construction" [34] and verify at each step that the constructed automaton accepts the same language.

Lemma 2.8. *For every NMFA-ε there exists an equivalent NMFA-MSM.*

Indeed, after applying the "Subset construction" algorithm [34] on an NMFA-ε, the resulting automaton is an equivalent NMFA-MSM. Its worth noticing, that after this transformation the resulting NMFA-MSM does not have any two transitions from the same state with the same letter.

Let $A = (Q, X, \delta, q_0, F)$ be an n-tape NMFA-MSM obtained from "Subset construction". We introduce a new set of symbols $X_{n+1} = X_\varepsilon = \{\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_n\}$ as a new $n + 1$-th tape. We call this tape *control tape*, and the other n tapes - *information tapes*. Each symbol in the control tape (ε_i) corresponds to an information tape (X_i).

The introduced tape allows us to transform the given NMFA-MSM A to a DMFA $A^{(D)}$, so that the language accepted by $A^{(D)}$ differs from the language accepted by A only by additional symbols of the $(n + 1)$-th tape. The transformation is done as follows: for each state $q \in Q$, if there exist transitions from q with letters from different tapes $(X_{i_1}, \ldots, X_{i_k})$, create new states for each such tape $(q_{i_1}, \ldots, q_{i_k})$. For each (q, x, q') transition from q, such that $x \in X_j$, add transition (q_j, x, q') and then remove (q, x, q'). Afterwards, for each added state q_j add a transition (q, ε_j, q_j).

2.4 An Approximate Method for Calculating the Distance Between Regular Events over a Free Partially Commutative Semigroup

The proposed method presented in this sub-section, currently, is under review as well.

Let R_1 and R_2 be regular expressions over a free partially commutative semigroup, and E_1, E_2 be regular events denoted by R_1 and R_2 correspondingly. Let A_1, A_2 be NMFA-MSMs and $A_1^{(D)}, A_2^{(D)}$ be DMFAs constructed from R_1 and R_2 correspondingly, using the techniques described in Section 2.3.

Apply the algorithm "Congruence Builder" from [6] on $A_1^{(D)}$ and $A_2^{(D)}$ by taking the union of their final states as F and the union of all their states as Q. This algorithm checks the equivalence between DMFAs. After the algorithm completes, if the starting states of $A_1^{(D)}$ and $A_2^{(D)}$ belong to the same equivalence class, then $A_1^{(D)}$ and $A_2^{(D)}$ are equivalent, hence R_1 and R_2 recognize the same language and the distance between them is 0 ($d_H(E_1, E_2) = 0$). Otherwise, if the starting states belong to different equivalence classes, the following method is used to approximately calculate $d_H(E_1, E_2)$ distance.

1. During the execution process, the "Congruence Builder" algorithm stores the $(n+1)$-tuples of words accepted by the automata (not all accepted tuples) as coordinates of the equivalence classes containing the starting states q_{10} of $A_1^{(D)}$ and q_{20} of $A_2^{(D)}$ are found. These tuples can be obtained from the coordinates of the classes [6]. Denote the finite sets of these $(n + 1)$-tuples of words corresponding to q_{10} state by $P_1^{(D)}$, and to q_{20} - by $P_2^{(D)}$
2. Delete the last words from each tuple in $P_1^{(D)}$ and $P_2^{(D)}$, obtaining sets of n-tuples of words P_1 and P_2 correspondingly. The deleted words contained only letters ε_i added in the process of constructing the DMFAs.
3. For every $p \in P_1 \setminus P_2$, if p is accepted by A_2, add the tuple p to P_2.
4. For every $p \in P_2 \setminus P_1$, if p is accepted by A_1, add the tuple p to P_1.
5. Compute the distance $d_H(P_1, P_2)$ for the finite sets P_1 and P_2.

3 Conclusion

In this work, some results in theory of automata are presented, which are achieved basing on a special binary coding for elements of a partially commutative semigroup. This coding leads to a new more effective characterization of commutation classes.

It is shown, that an MFA can be synthesized from a given regular expression over a free partially commutative semigroup.

A new metric and metric space are introduced for measuring the distance between regular events for MFA. An approximate method for calculating this distance is described.

References

[1] Michael O. Rabin and Dana S. Scott. Finite Automata and Their Decision Problems. *IBM J. Res. Dev.*, 3(2):114-–125, 1959. doi: 10.1147/rd.32.0114.

[2] Timothy V. Griffiths. The Unsolvability of the Equivalence Problem for Lambda-Free Nondeterministic Generalized Machines. *J. ACM*, 15(3):409–413, 1968. doi: 10.1145/321466.321473.

[3] Malcolm Bird. The Equivalence Problem for Deterministic Two-Tape Automata. *J. Comput. Syst. Sci.*, 7(2):218–236, 1973. doi: 10.1016/S0022-0000(73)80045-5.

[4] Tero Harju and Juhani Karhumäki. The Equivalence Problem of Multitape Finite Automata. *Theor. Comput. Sci.*, 78(2):347–355, 1991. doi: 10.1016/0304-3975(91)90356-7.

[5] Alexander A. Letichevsky, Arsen S. Shoukourian, and Samvel K. Shoukourian. The Equivalence Problem of Deterministic Multitape Finite Automata: A New Proof of Solvability Using a Multidimensional Tape. In Adrian-Horia Dediu, Henning Fernau, and Carlos Martín-Vide, editors, *Language and Automata Theory and Applications, 4th International Conference, LATA 2010, Trier, Germany, May 24-28, 2010. Proceedings*, volume 6031 of *Lecture Notes in Computer Science*, pages 392–402, Springer, 2010. doi: 10.1007/978-3-642-13089-2_33.

[6] Hayk A. Grigoryan and Samvel K. Shoukourian. Polynomial Algorithm for Equivalence Problem of Deterministic Multitape Finite Automata. *Theor. Comput. Sci.*, 833:120–132, 2020. doi: 10.1016/j.tcs.2020.05.044.

[7] Hayk A. Grigoryan and Arsen S. Shoukourian. Equivalence of Processes in Partially Commutative Object-Oriented Environments. *Fundamenta Informaticae*, 105(4):417–434, 2010. doi: 10.3233/FI-2010-372.

[8] Y. I. Ianov. The Logical Schemes of Algorithms. *Problems of Cybernetics*, I, 82–140, 1958.

[9] M. Paterson. Equivalence Problems in a Model of Computation. Doctoral Dissertation, Cambridge University, 1967.

[10] David C. Luckham, David Michael Ritchie Park, and Mike Paterson. On Formalised Computer Programs. *J. Comput. Syst. Sci.*, 4(3):220–249, 1970. doi: 10.1016/S0022-0000(70)80022-8.

[11] Alexander. A. Letichevskii. Functional Equivalence of Discrete Converters. Part 2. *Kibernetika* (in Russian), No. 2, 14–28, 1970.

[12] A. B. Godlevskii, A. A. Letichevskii, and S. K. Shukuryan. Reducibility of Program-Scheme Functional Equivalence on a Nondegenerate Basis of Rank Unity to the Equivalence of Automata With Multidimensional Tapes. *Cybern. Syst. Anal.*, 16:793–799, 1980. doi: 10.1007/BF01069053.

[13] Haik Grigorian and Samvel K. Shoukourian. The Equivalence Problem of Multidimensional Multitape Automata. *J. Comput. Syst. Sci.*, 74(7):1131–1138, 2008. doi: 10.1016/j.jcss.2008.02.006.

[14] James Worrell. Revisiting the Equivalence Problem for Finite Multitape Automata. In Fedor V. Fomin, Rusins Freivalds, Marta Z. Kwiatkowska, and David Peleg, editors, *Automata, Languages, and Programming - 40th International Colloquium, ICALP 2013, Riga, Latvia, July 8-12, 2013, Proceedings, Part II*, volume 7966 of *Lecture Notes in Computer Science*, pages 422–433. Springer, 2013. doi: 10.1007/978-3-642-39212-2_38.

[15] P. Cartier, D. Foata. Problemes Combinatoires de Commutation et Rearrangements. Volume 85 of *Lecture Notes in Mathematics*, Springer-Verlag Berlin Heidelberg, 1969. ISBN 978-3-540-36094-0. doi: 10.1007/BFb0079468.

[16] Antoni W. Mazurkiewicz. Concurrent Program Schemes and their Interpretations. *DAIMI Report Series*, 6(78), 1977. doi: 10.7146/dpb.v6i78.7691.

[17] Alberto Bertoni, M. Brambilla, Giancarlo Mauri, and Nicoletta Sabadini. An Application of the Theory of Free Partially Commutative Monoids: Asymptotic Densities of Trace Languages. In Jozef Gruska and Michal Chytil, editors, *Mathematical Foundations of Computer Science 1981, Strbske Pleso, Czechoslovakia, August 31 - September 4, 1981, Proceedings*, volume 118 of *Lecture Notes in Computer Science*, pages 205–215, Springer, 1981. doi: 10.1007/3-540-10856-4_86.

[18] Alberto Bertoni, Giancarlo Mauri, and Nicoletta Sabadini. Equivalence and Membership Problems for Regular Trace Languages. In Mogens Nielsen and Erik Meineche Schmidt, editors, *Automata, Languages and Programming, 9th Colloquium, Aarhus, Denmark, July 12-16, 1982, Proceedings*, volume 140 of *Lecture Notes in Computer Science*, pages 61–71, Springer, 1982. doi: 10.1007/BFb0012757.

[19] Volker Diekert. Combinatorics on Traces. Volume 454 of *Lecture Notes in Computer Science*, Springer, 1990. ISBN 978-3-540-53031-2. doi: 10.1007/3-540-53031-2.

[20] Antoni W. Mazurkiewicz. Introduction to Trace Theory. In Volker Diekert and Grzegorz Rozenberg, editors, *The Book of Traces*, pages 3–41, World Scientific, 1995. doi: 10.1142/9789814261456_0001.

[21] Volker Diekert and Yves Métivier. Partial Commutation and Traces. In Grzegorz Rozenberg and Arto Salomaa, editors, *Handbook of Formal Languages, Volume 3: Beyond Words*, pages 457–533, Springer, 1997. doi: 10.1007/978-3-642-59126-6_8.

[22] V. M. Glushkov. The Abstract Theory of Automata. *Russian Mathematical Surveys*, 16(5):1–53, 1961. doi: 10.1070/rm1961v016n05abeh004112.

[23] Robert McNaughton and Hisao Yamada. Regular Expressions and State Graphs for Automata. *IRE Trans. Electron. Comput.*, 9(1):39-47, 1960. doi: 10.1109/TEC.1960.5221603.

[24] Boris G. Mirkin. On the theory of multitape automata. *Cybern. Syst. Anal.*, 2:9–14, 1966. doi: 10.1007/BF01073664.

[25] Peter H. Starke. On the Representability of Relations by Deterministic and Nondeterministic Multi-Tape Automata. In Jirí Becvár, editor, *Mathematical Foundations of Computer Science 1975, 4th Symposium, Mariánské Lázne, Czechoslovakia, September 1-5, 1975, Proceedings*, volume 32 of *Lecture Notes in Computer Science*, pages 114–124, Springer, 1975. doi: 10.1007/3-540-07389-2_186.

[26] Tigran A. Grigoryan. Some Results on Regular Expressions for Multitape Finite Automata. *Proceedings of the YSU: Physical and Mathematical Sciences*, 53(2):82–90, 2019. doi: 10.46991/PYSU:A/2019.53.2.082.

[27] V. G. Bodnarchuk. The Metrical Space of Events. Part I. *Cybern. Syst. Anal.*, 1:20–24, 1965. doi: 10.1007/BF01071439.

[28] Victor Vianu. The Bodnarchuk Metric Space of Languages and the Topology of the Learning Space. In Jozef Gruska, editor, *Mathematical Foundations of Computer Science 1977, 6th Symposium, Tatranska Lomnica, Czechoslovakia, September 5-9, 1977, Proceedings*, volume 53 of *Lecture Notes in Computer Science*, pages 537–542, Springer, 1977. doi: 10.1007/3-540-08353-7_177.

[29] V. I. Levenshtein. Binary Codes Capable of Correcting Deletions, Insertions, and Reversals. *Soviet Physics Doklady*, 10(8):707–710, 1966.

[30] Horst Bunke. Edit Distance of Regular Languages. In *5th Annual Symposium on Document Analysis and Information Retrieval, Las Vegas, Nevada, April 15-17, 1996, Proceedings*, pages 113–124, 1996.

[31] Austin J. Parker, Kelly B. Yancey, and Matthew P. Yancey. Regular Language Distance and Entropy. In Kim G. Larsen, Hans L. Bodlaender, and Jean-François Raskin, editors, *42nd International Symposium on Mathematical Foundations of Computer Science, MFCS 2017, August 21-25, 2017 - Aalborg, Denmark*, volume 83 of *LIPIcs*, pages 3:1–3:14, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017. doi: 10.4230/LIPIcs.MFCS.2017.3.

[32] F. Hausdorff. Set Theory. *Chelsea Pub Co, 4 edition*, 1991. ISBN 978-0-8284-1119-6.

[33] H. Tamm. On Minimality and Size Reduction of One-Tape and Multitape Finite Automata. Ph.D. thesis, Department of Computer Science, University of Helsinki, Finland, 2004.

[34] Alfred V. Aho, Ravi Sethi, and Jeffrey D. Ullman. Compilers: Principles, Techniques, and Tools. *Addison-Wesley series in computer science / World student series edition*, Addison-Wesley, 1986. ISBN 0-201-10088-6.