Testing of automata: from experiments to representations by means of fragments

Igor S. Grunsky

Institute of Applied Mathematics and Mechanics,
National Academy of Sciences of Ukraine,

74, R. Luxemburg str., Donetsk, 340114, Ukraine
e-mail: math @ iamm.ac.donetsk.ua

Abstract

This paper discusses some developments and results of the theory of test generation from automata. These developments are driven by the needs to better understand the nature of automata testing and thus to make the testing theory more applicable to real systems. We provide an overview of some important results in automata testing recently obtained in the Soviet Union and in the countries that have arisen from it.

Keywords

Automaton (finite state machine), testing problems, checking experiments and sequences, checking test, test complexity, experiments on automata, automata fragment

1 INTRODUCTION

Automata or finite state machines have been widely used as mathematical models of discrete systems in diverse areas such as computer hardware and software and more recently, communication protocols [2,3,4,7,18,19,29,31]. In

A. Petrenko et al. (eds.), *Testing of Communicating Systems*

this paper, we study the fault-detection problems on automata. We are given the state diagram of an automaton A, and we have a «black box» automaton B which is supposed to implement A. We can test B by applying input sequences (tests) and observing the produced output sequences. We want to design the tests to determine whether or not B is an implementation of A. This problem has been referred to as the «fault-detection» or «checking problem» in automata theory. In the recent literature, this problem is also called conformance testing (of communication protocols).

There is an extensive literature on problems of automata testing. One may state that the theory of automata testing has been built over the last 40 years. There exist several excellent books [15,18,29] and surveys [2,20,31] in this theory, but many interesting results were omitted in these publications. The aim of this paper is to represent some important results recently obtained in the USSR and in the countries arisen from the Soviet Union (mainly, in Russia and Ukraine).

2 BACKGROUND

2.1 Basic notations

A deterministic finite state machine *(FSM)* or an automaton A is quintuple $A=(S,X,Y,\delta,\lambda)$, where S,X,Y are finite and nonempty sets of states, input symbols, and output symbols, respectively; δ: $Dom_A \to S$ is the state transition function and λ : $Dom \to Y$ is the output function; Dom_A is a specification domain of A, i.e. a subset of $S \times X$. If $Dom_A = S \times X$ then A is complete, otherwise it is partial. Let n,m,r be the cardinality of S,X,Y, respectively. If $p=x_1...x_k$ is an input word, then $\delta(s,p)$ is the state reached by A from state s when p is applied to A, and $\lambda(s,p)=y_1...y_k$ is the corresponding output word. The pair $(p, \lambda(s,p))$ is called an input-output word produced by the state s. We use $w=(x_1,y_1)...(x_k,y_k)$ to denote this word. The word w determines the unique partial «string» automaton $R(w)$ with the state set $\{0,1,...,k\}$, transition function Δ, and output function Λ: $\Delta(i,x_{i+1})=(i+1)$, $\Lambda(i,x_{i+1})=y_{i+1}$ for $0 \le i \le k-1$ and these functions are undefined in all other cases.

We denote λ_s and λ_s^k the sets of all input-output words produced by s which are of finite length and length equal to or less than k, respectively. The automaton A is reduced if $\lambda_s \neq \lambda_t$ for every pair of distinct states s and t. The reduced automaton is uniquely characterized by the set $L_A=\{\lambda_s\}$, $s \in S$. An important characterization of A is the set $\Phi_A=\cup_{s \in S}\lambda_s$.

Let s_0 be a designated initial state of A. The automaton A is called connected if for all states $s \in S$ exists an input word p such that $\delta(s_0,p)=s$. The automaton is called strongly connected if it is connected for all $s_0 \in S$.

Two automata A and B are said to be equivalent if $L_A=L_B$. If s_0 and t_0 are the initial states of A and B, then A and B are equivalent if $\lambda_{s0}=\lambda_{t0}$.

An input word p is a distinguishing word for A, if $s \neq t$, $s,t \in S$, implies $\lambda(s,p) \neq \lambda(t,p)$. The automaton A is called definitely diagnosable of order k if every p of length k is a distinguishing word (DDA-k, for short). It is known that k in this case is equal to or less than $n(n-1)/2$. A word p is a homing word for A if $\lambda(s,p)=\lambda$

(t,p) implies $\delta(s,p)\neq\delta(t,p)$. If every p of length k is a homing sequence for A, then A is the finite memory of order k automaton. The order k of finite memory is also equal to or less than $n(n-1)/2$.

2.2 Testing problems

In testing problems, we have a machine about which we lack some information, and we would like to deduce this information from input-output words obtained by experimenting with the machine. An experiment is a process when we apply input words to the machine, observe the produced output words and deduce the missing information about the machine. The applied input words are called a test suite and obtained input-output words are called an experiment.

We discuss the machine verification problem also known as the fault-detection or conformance testing problem. We are given a *specification* machine A, i.e. we have its state transition and output function. We are also given an *implementation* machine B that is a «black box» and we can only observe its input-output behaviour. We want to design a test (an experiment) to determine whether B conforms (is equivalent) to A. Obviously, without any assumption the problem is undecidable. There are a number of «classical» assumptions that are usually made in the literature:

a) the specification machine A is complete, reduced and strongly connected;
b) an implementation machine B has the same input alphabet as A;
c) B belongs to some known class F of «faulty machines».

The class F is often assumed to coincide with the class F_n of all machines with the number of states not more than n. The specification machines considered in the machine verification problem usually have a designated initial state (initialized machines).

Let $W=\{(p_i,q_i)\}$, $1\leq i\leq k$, be a set of input-output words. The set W is called a checking experiment, if $W\subseteq\lambda_{s_0}$ and, for all $B\in F_n$ and any state t of B, $W\subseteq\lambda_t$ implies that the machine B is equivalent to A. The parameter k is called the multiplicity of the experiment W, the length of the longest word from W is called the length of W, and the total length of all words in W is the size of W. The experiment is called simple when $k=1$ and multiple otherwise. A simple checking experiment *(p,q)* is called a checking sequence and p is called a checking test.

There is an extensive literature on testing automata, the fault-detection problem, in particular. It is convenient for us to distinguish three periods in this research: primary, classical, and modern periods. The primary period was opened by the famous Moore's paper [23] in which he studied a related, but harder problem of machine identification: given a machine with a known number of states, determine its state diagram. This period is characterised by the combinatorial nature of offered solutions, i.e. the obtained results are implied by counting the number of automata in a certain class.

Moore proved that there exists a multiple checking test for A with n states and F_n containing all input words of length $2n-1$. He also provided an exponential

algorithm for constructing a checking sequence of an exponential length and an exponential lower bound for this problem on the basis of the class F cardinality. Books [7,18,29] give a good exposition of the major results of the primary period.

The classical period was started by the influential Hennie's paper [14]. He shown that if the automaton A has a distinguishing input sequence of length l, then one can construct a checking sequence of a length polynomial in l and nm. Unfortunately, not every machine has a distinguishing sequence. Furthermore, only exponential algorithms are known for determining the existence such sequences. In [27] it is shown that the problem is *PS*-complete [1]. Rystsov [26] proved that for the length l of shortest distinguishing sequences the inequality $3^{n(1-\varepsilon)/6} < l < 3^{n(1+\varepsilon)/6}$ holds, where ε is any positive real number and $n \to \infty$. Nevertheless, the main Hennie's idea is most fruitful. It lies in embedding a distinguishing sequence in a test sequence in a special way to:

a) obtain the response of each state of A to the distinguishing sequence and,

b) check each transition of A by applying a proper input at the start state, observing the produced output, and verifying the tail state of the transition by using the distinguishing sequence.

Based on this idea, many papers were published in which various subsequences were used to verify the start and tail states of transitions, distinguishing sequences, adaptive distinguishing sequences, locating sequences, identifying sequences, homing sequences and others. To the classical period an important Vasilevskii's paper [30] belongs. He provided a polynomial algorithm for constructing a multiple checking experiment, and proved polynomial upper and lower bounds on the length of checking sequences. Books [3,4,15,18] and papers [20,31] give a good exposition of the major results of the classical period. During this period, many algorithms were constructed for special classes of specification automata, diagnosable, definitely diagnosable of order k [18] etc. In [25,28], the exact upper bound of adaptive distinguishing sequences was obtained. In [8], so-called checking sequences for state of A were investigated. The period has continued till recent days.

3 MODERN RESEARCHES

The results obtained during the primary and classical periods give a basis and a possibility to build a general theory of automata checking. A variant of the theory is stated below. Consider an automaton A and a (possibly partial) automaton $R=(T,X,Y,\Delta,\Lambda,D_R)$. A mapping $\varphi: T \to S$ is a homomorphism of R to A, if $\varphi(\Delta(t,x))=\delta(\varphi(t),x)$, $\Lambda(t,x)=\lambda(\varphi(t),x)$ for all $(t,x)\in D_R$. The automaton R is called a fragment of A if such a mapping exists. This fact is denoted $R \leq A$, and in the case, when φ is one-to-one mapping, it is denoted $R \subseteq A$. It is obvious, that if $w \in \Phi_A$, then $R(w)$ is a fragment of the automaton A.

Let F be a class of reduced complete automata over the alphabets X, Y and τ be a similarity relation on $F \cup \{A\}$. $\tau(A)$ is the class of automata similar to A. An automaton R is said to be a representation of A with respect to F and τ (a

representation of (A,F,τ), for short), if $R \leq A$ and if $R \leq B$, $B \in F$, implies $B \in \tau(A)$. It is obvious, that if w is a checking sequence for A and F, then the automaton $R(w)$ is the representation of (A,F,τ), where τ is the automata equivalence relation, that is $(A,B) \in \tau$ iff $L_A = L_B$. If W is a checking experiment, then the tree-like automaton $R(W)$ defined analogously to $R(W)$ is a representation of A, as well. The notion of a representation is a nontrivial and useful generalisation of checking experiments and it enables us to construct a unified profound theory of automata checking. The representation theory is widely stated in [11]. In the same book, the application of the representation theory to problems of technical diagnostics is discussed.

3.1 Existence of representations

Theorem 1 [11]

The following statements are equivalent:

1. a representation of (A,F,τ) exists,

2. the automaton A is a representation of (A,F,τ),

3. if $A \subseteq B$, $B \in F$, then $B \in \tau(A)$.

Given a checking experiment W, the tree-like automaton $R(W)$ is the representation. The existence condition for this important class of tree representations is given by:

Theorem 2 [11]

A tree representation of (A,F,τ) exists iff there exists a natural k such that for each $B \in F\text{-}\tau(A)$ there exists a state t of B such that $\lambda_t^k \neq \lambda_s^k$ for all $s \in S$.

The statement 3 of the Theorem 1 for a finite F may be checked effectively. The class F in Theorem 2 may be either finite or infinite. Any finite class F has this property but the converse is not always true. From this it follows that multiple checking experiments exist for any finite F.

In [11] several existence conditions are found for several types of representations of various (A,F,τ).

3.2 Representation structure

For the design of checking experiments, as it has been stated above, special sequences (distinguishing, locating etc.) play a significant role. Let us introduce a notion of state identifiers to generalize such sequences. A fragment R with a fixed state t is an identifier of state s of A if $R \leq A$ and if each homomorphism of R to A maps t to s. The fragment R is a state identifier of A, if it is an identifier of some state s. Let $R \leq A$ and I be a state identifier of A. The identifier I is said to be verified in R, if $R \leq B$, $B \in F$ implies that I is a state identifier of B.

Let J be a set of the state identifiers verified in R. Consider an equivalence relation on J: $(I_1,I_2) \in \sigma$ if I_1, I_2 are identifiers of the same state for all B

$\in F$, $R \leq B$. The pair (J,σ) generates on the state set of R a reflexive and symmetric relation: $(t_1,t_2) \in \beta$ iff there exist $I_1,I_2 \in J$ for some states t_1,t_2, respectively, $I_1,I_2 \leq R$, and $(I_1,I_2) \in \sigma$. The smallest congruence relation $\rho \supseteq \beta$ is called a closure of β. The closure ρ generates the fragment $[R]=R/\rho$ which is called a closure of R by (J,σ).

Theorem 3 [11]

If $[R]=A$ for (J,σ), then R is a representation of (A,F,τ).

An input-output word $w=(x_1,y_1)...(x_k,y_k)$, $w \in \Phi_A$ is called an initial (final) identifier of A if the fragment $R(w)$ with the state o (k, respectively) is a state identifier of A.

It follows from this definition that if p is a distinguishing (homing) word for A, then $R(p,\lambda(s,p))$ is an initial identifier of s (a final identifier of $\delta(s,p)$, respectively) of the automaton A. Taking into account this definition we may say that the results of Hennie and his successors are the corollaries to *Theorem 3*.

Consider the cases when the condition of Theorem 3 is both, sufficient and necessary. Let $F=F_n$, τ be an isomorphism relation, and A be a *DDA-k*.

Theorem 4 [11]

A word $w \in \Phi_A$ is a checking experiment for DDA-k, where $k \leq 11$, iff $[R(w)]=A$, where J is the set all initial state identifiers verified in $R(w)$.

Consider the case when A is *DDA-k*, $k=1$. Let Φ_A^1 be the set of all input-output words $w \in \Phi_A$ of length 1, and J be the set of all initial state identifiers of length 1 verified in R. Define a non oriented graph $G(R)=(V,E)$, where V is equal to the state set of $[R]$, and $(V_1,V_2) \in E$ if $V_1 \neq V_2$, $\Lambda(V_1,x) \neq \Lambda(V_2,x)$ for some $x \in X$. A mapping φ: V onto $\{1,2,...,n\}$ is called a colouring of $G(R)$, if $(V_1,V_2) \in E$ implies $\varphi(V_1) \neq \varphi(V_2)$. The graph $G(R)$ is said to be uniquely colourable iff all colourings of $G(R)$ are isomorphic.

V.A. Kozlovsky has proved the following important results.

Theorem 5 [11]

A fragment R is a representation of $(A,F_n,=)$ iff the following conditions hold:

1. $R \leq A$,
2. $\Phi_A^1 = \Phi_R^1$,
3. *$G(R)$ is uniquely colourable.*

On the basis of the Theorem 5 he has proved

Theorem 6 [11,16]

The problem «is a word w a checking experiment of $(A,F_n,=)$?» is NP-complete.

We note that the checking of conditions *1,2* of Theorem 5 may be performed by a polynomial algorithm. In [17] V.A. Kozlovsky has proved that this problem is *NP*-complete for a special class of machines, so-called group *DDA-1*.

3.3 Important subclasses of F_n.

The results discussed above indicate the fundamental difficulties in constructing tests for $(A,F_n,=)$. The difficulties stimulate investigation of subclasses of F_n for which the test derivation can be done more efficiently. A number of such classes are known [2, 11, 13, 16, 24]. We consider here the two classes: a locally generated class [11, 16] and a class generated by a fault-function [5, 6, 10, 11, 13, 31].

Given an automaton A, we define a class $F(A)$ generated by local transformations of A. The neighbourhood of state $s\in S$ in A is the set $O_A(s)$ of states t such that $\delta(s,x)=t$ or $\delta(t,x)=s$ for some $x\in X$. If $\delta(s,x)=z$ for some $s,z\in S$ and $x\in X$, then replacing z by some $t\in O_A(v)$ we obtain an automaton B which we call as the one directly generated by A. An automaton $B\in F(A)$ iff there exists a sequence of automata $B_0=A, B_1,...,B_k=B$ such that B_i is directly generated by B_{i-1}, $i=1,..,k-1$. Clearly, $F(A)\subseteq F_n$.

Theorem 7 [16]

A word $w=(x_1,y_1)...(x_n,y_n)$, $w\in\Phi_A$ is a checking experiment of $(A,F_n,=)$, where A is a DDA-1, iff $\Phi_A^{-1}=\{(x_1,y_1),...,(x_k,y_k)\}$, and (x_k,y_k) occurs in w at least twice.

Corollary 8 [16]

1. There is a polynomial algorithm solving the decision problem «is a word w a checking experiment of $(A,F(A),=)$»;

2. The length $d(w)$ of the shortest checking experiment w of $(A,F(A),=)$ satisfies the inequality $mn+1\le d(w)\le mn+(m-1)n(n-1)/2$, where both the lower and upper bounds are reachable.

In the book [11], a special case of the local transformations, so-called input faults on A, is considered. For these faults and a so called inversible A, an algorithm for constructing checking sequences of length at most $(2n-1)m$ is proposed.

Consider now the class generated by a fault-function [10]. Let S, X, Y be some alphabets of states, input, and output, respectively. Consider the pair of functions (f,g), where $f(p)\subseteq S$ and $g(p)\subseteq Y$ for each input word p. For an empty word e we define $f(e)=s_0\subseteq S$, $g(e)=e$. Let F be a class of automata in these alphabets. The pair (f,g) is said to be an evaluating function for F, if $\delta(s_0,p)\in f(p)$ and $\lambda(\delta(s_0,p),x)\in g(px)$ for each $x\in X$, input word p and each automaton $A\in F$ with the initial state s_0. Each pair (f,g) generates a maximal class F for which the pair is

an evaluating function. Assuming that an automaton A in F serves as a specification (it is a correct automaton), all the others are considered as faulty automata, and the pair (f,g) is called a fault function for A and F. In [13] a fault function with $g(p)=\lambda(s_0p)$ was considered. It is obvious, that such a fault function determines the class $F(A)$, if $f(e)=s_0$ and $f(px)=O_A(\delta(s_0p))$.

Fault functions are a powerful tool for defining automata classes. In [10,13,32] the methods for constructing multiple checking tests were proposed. These methods improve the Vasilevskii results [30] and may yield simpler tests. S. Yu. Boroday [5,6] has considered a subclass of the class F of automata with single transition faults, (automata whose transition and output functions differ from those of A only for one pair (s,x) or for one state s). As shown in [5,6], in these classes test derivation is simplified.

3.4 Finite-definable classes of automata

Consider now the testing problems when a class F can be infinite but it is defined by a finite means. In this section, we study two ways of defining an infinite class, by a set $M\subseteq X\times Y$ [9,12] and by nondeterministic automata [2,21,22,24].

Let $F(X,Y)$ be the class of all initialized reduced automata over inputs X and outputs Y. Given a subset $M\subseteq X\times Y$, consider a class $F(M)$ of all automata from $F(X,Y)$ in which every $(x,y)\in M$ can be produced by at most one state. Such M exists in practice (for example, protocol machines with a status message [19,31]). It is easy to see that the class $F(M)$ may be either finite or infinite. Let M_x be the set of all $(x,y)\in M$, $y\in Y$, and m_x be the cardinality of M_x.

Theorem 9 [12]

The following statements are equivalent:

1. The class $F(M)$ is finite,

2. $m_x=r$ for some $x\in X$,

3. there exists some $x\in X$ such that for each $A\in F(M)$, x is a distinguishing word.

The statement 2 can be checked by a polynomial algorithm. Let $G(M)\subseteq F(M)$ be a subclass of all automata in which every cycle of the transition graph has a state producing some $(x,y)\in M$.

Theorem 10 [12]

A multiple checking experiment of $(A,F(M),=)$ exists iff $A\in G(M)$.

The condition of Theorem 10 can be verified by a polynomial algorithm. The polynomial condition of checking sequence existence can be found in [12] as well.

Let R be a fragment of $A \in F(M)$. A closure $[R]_M$ of R by M is a (possible partial) automaton constructed from R by identifying all states of R producing the same $(x,y) \in M$ and its next states according to x.

Theorem 11 [9]

1. Let $n<r$ or $m_x<r$ for all $x \in X$. A fragment R is a representation of $(A,F(M),=)$ iff the closure $[R]_M$ is equivalent to A.

2. Let $n=r$ and $m_x=r$ for some $x \in X$. The fragment R is a representation of $(A,F(M),=)$ iff the following conditions hold:

a) $R \leq A$,

b) $\Phi_A^1 = \Phi_R^1$,

c) $[R]_M$ is uniquely colourable.

Another way of defining automata classes uses a nondeterministic automaton. Let $N=(V,X,Y,h,v_0)$ be an initialized nondeterministic observable automaton with h: $V \times (X \times Y) \to V$. Given state v and input-output word w, $v'=h(v,w)$ denotes a state reached when the automaton produces w. In the general case, the function h is partial, i.e. $h(v,w)$ may be undefined for some w. Let L_N be a set of all words w for which $h(v_0,w)$ is defined. An automaton N defines a class $F(N)$ of automata A from $F(X,Y)$ such that $\lambda s_0 \subseteq L_N$. The automaton $A \in F(N)$ is called an implementation of N. A state v of N is deterministic if for each p a unique output word q exists such that $h(v,(p,q))$ is defined. All deterministic states form a kernel of A.

Theorem 12 [22]

The class is finite iff the automaton N has a cycle outside of its kernel.

In [21] B.Lukyanov has found a sufficient condition for checking the test existence for $(A,F(N),=)$, where $A \in F(X,Y)$, and proposed a method for deriving a test. S. Yu. Boroday [6] has found a necessary and sufficient condition for checking the test existence and has given an algorithm. Moreover, he has given a method for checking whether the automaton is contained in $F(N_1)$ or in $F(N_2)$, where N_1, N_2 are nondeterministic automata.

4 CONCLUSION

The key issues in automata testing are the structure of experiments and the analysis of the experiments. The first problem consists in determining what kind of information about the specification machine must be present in checking experiments. The second problem consists in finding all automata generating a given experiment. Both problems are closely related and are very hard. This paper deals with the first problem. A framework has been introduced for this problem exploration on the base of the state identifiers. Note that in [11] some more types of identifiers are studied (input and output identifiers, for example). The results

presented in the paper (and in [11]) show us that the representations and the identifiers of non-observable components of automata are powerful means for this problem solution.

5 ACKNOWLEDGEMENTS

This research was partly supported by the Russian Found for Fundamental Research (grant 98-01-00113). The author wishes to thank Alex Petrenko (CRIM, Canada) for his stimulating help at all the stages of the preparation of this paper.

6 REFERENCES

[1] A. V. Aho, J.E. Hopcroft, J. D. Ullman, The Design and Analysis of Computer Algorithms. Reading, MA: Addison-Wesley, 1974.

[2] G. v. Bochmann and A. Petrenko, Protocol Testing: Review of Methods and Relevance for Software Testing, Proc. of the 1994 Int. Symp. on Software Testing and Analysis (ISSTA), Seattle, Wash., T. Ostrand Ed., pp. 109-124, 1994.

[3] A. M. Bogomolov, A. S. Barashko, I. S. Grunsky, Experiments on Automata. Kiev: Naukova Dumka, 1973 (in Russian).

[4] A. M. Bogomolov, I. S. Grunsky, D. V. Speransky, Checking and Transformation of Discrete Automata. Kiev: Naukova Dumka, 1976 (in Russian).

[5] S. Yu. Boroday, Checking of Single Faults of Finite State Machine Defined by a Fault Function, Kibernetica, no.6, pp 65-74, 1995.

[6] S. Yu. Boroday, Experiments on Effectively Defined Classes of Automata, Ph.D. Thesis. Saratov University, 1997.

[7] A. Gill, Introduction to the Theory of Finite-State Machines. New York: Mc Graw-Hill, 1962.

[8] S. M. Goberstein, Check Words for the States of Finite Automaton, Kibernetika, no 1, pp. 46-49, 1974.

[9] I. S. Grunsky, Structure of Fragments Uniquely Representing Automata, Intellectual Systems (Moscow), v.2, pp. 249-258, 1997 (in Russian)

[10] I. S. Grunsky, Checking of Automata Faults Defined with an Explicitly Given Fault Function, Proc. of Institute of Applied Math. and Mechanics, Ukrainian Acad. Of Sci. (Donetsk), vol.1, pp. 38-43, 1997.

[11] I. S. Grunsky, V. A. Kozlovsky, G. G. Ponomarenko, Finite Automata Representation by Means of Behaviour Fragments. Kiev: Naukova Dumka, 1990 (in Russian).

[12] I. S. Grunsky, I. I. Maximenko, On Experiments on Automata without an Upper Bound on the Number of their States, Dopovidi of Nat. Acad. Of Sci. (Kiev, Ukraine), no.6, pp. 31-35, 1996.

[13] I. S. Grunsky, A. Petrenko, Design of Checking Experiments on Automata Describing Protocols, Automatika I vychislitelnaya tehnika (Riga), no.4, pp. 7-14, 1988 (in Russian), Automatic Control and Computer Sciences, Allerton Press, Inc., USA, (in English).

[14] F. C. Hennie, Fault Detecting Experiments for Sequential Circuits, Proc. IEEE 5-th Ann. Symp. On Switch. Circuits Theory and Logical Design, 1964, pp. 95-110.

[15] Z. Kohavi, Switching and Finite Automata Theory. Mc Graw-Hill, 1970.

[16] V. A. Kozlovsky, On the Recognition of an Automaton Relative to a Locally Generated Class, Soviet Math. Dokl, vol. 23, no.3, pp. 625-628, 1981.

[17] V. A. Kozlovsky, On Representation of the Group Automata, Kibernetika i Sistemny Analiz, no.2, pp. 21-28, 1996.

[18] V. B. Kudryvtsev, S. V. Aleshin, A. S. Podkolzin, Introduction to Automata Theory, Moscow: Nauka, 1985 (in Russian).

[19] D. Lee, and D. Su, Modelling and Testing of Protocol Systems, Proc. IFIP TC6 10th Int. Workshop on Testing of Commun. Syst., Chapman & Hall, Myungchui Kim and Sungwon Kang Eds., pp. 339-364, 1997.

[20] D. Lee, M. Yannakakis, Principles and Methods of Testing Finite State Machines – A Survey, Proc. of the IEEE, vol. 84, no.8, pp. 1090-1123, Aug. 1996.

[21] B. D. Lukyanov, On Distinguishing and Checking Experiments on Nondeterministic Automata, Kibernetica, no.5, pp.69-76, 1995.

[22] B. D. Lukyanov, Deterministic Implementations of Nondeterministic Automata, Kibernetika, no.4, pp.34-50, 1996.

[23] E. F. Moore, Gedanken-Experiments on Sequential Machines, Automata Studies, Prinston, NY: Prinston Univ. Press, 1956.

[24] A. Petrenko, G. v. Bochmann, M. Yao, On Fault Coverage of Tests for Finite State Specifications, Computer Networks and ISDN Systems, vol. 29, pp. 81-106, 1996.

[25] I. K. Rystsov, A Proof of Accessible Bound of Adaptive Diagnostic Experiment Bound for Finite Automata, Kibernetika, no.3, pp. 20-22, 1976.

[26] I. K. Rystsov, On Asymptotic Bound of Diagnostic Word Length for Finite Automata, Kibernetika, no.2, pp. 31-35, 1980.

[27] I. K. Rystsov, An Investigation of Solutions Complexity for Finite Automata Theory Problems. Kandidat Degree dissertation, Kiev: Institute of Cybernetics Ukr. Acad. of Sci., 1980, 121p.

[28] M. N. Sokolovskii, Diagnostic Experiment with Automata, Kibernetika, no.6, pp. 44-49, 1971.

[29] B. A. Trakhtenbrot, Y. M. Barzdin, Finite Automata, Behaviour and Synthesis. Amsterdam: North-Holland, 1973.

[30] M. P. Vasilevskii, Failure Diagnosis of Automata, Kibernetika, no.4, pp. 98-108, 1973.

[31] M. Yannakakis, D. Lee, Testing Finite State Machines: Fault Detection, J. of Comput. and Syst. Sci., vol. 50, pp. 209-227, 1995.

[32] A. Petrenko, and N. Yevtushenko, Test Suite Generation for a FSM with a Given Type of Implementation Errors, IFIP Transactions Protocol Specification, Testing, and Verification XII (the Proceedings of IFIP TC6 12th International Symposium on Protocol Specification, Testing, and Verification) 1992, pp. 229-243.

7 BIOGRAPHY

Igor S. Grunsky received the Diploma degree in Radiophisics and Electronics from the University of Kiev, the USSR, in 1963 and Ph.D. (Candidate) degree in discrete mathematics and computer science from the Computing Centre of the Soviet Academy of Sciences, Moscow, in 1976. Since 1965, he has been with the Institute of Applied Mathematics and Mechanics of the Ukrainian Academy of Sciences. He is currently the Head of Applied Discrete Mathematics Laboratory of this institute. His research interests include discrete mathematics, automata theory, testing problems, checking and diagnostics of hardware, software and protocols.