A MATHEMATICAL TOOL FOR SUPPORT OF FAULT-TOLERANT EMBEDDED SYSTEMS DESIGN

S. Frenkel, A. Pechinkin, V. Chaplygin, I.Levin[1]
Institute of Informatics Problems RAS, Moscow,
Russia, 119333, Vavilova 44, kor.2, Moscow, Russia
Tel-Aviv University, Ramat-Aviv, 69978, Israel,
E-mails: (slf-ipiran@mtu-net.ru, Apechinkin@ipiran.ru, ilia1@post.tau.ac.il)

Abstract Designers of fault-tolerant computer systems need methodological and software framework which would support their efforts in analysis and optimization of new design solutions, based on new and forthcoming hardware and software technologies, embedded systems, in particular These new and advanced technologies - high-performance and self-reconfigurable systems, nanotechnologies- lead to unprecedented challenges. For example, often as a result of transient faults, reconfigurations in FPGA-based high-performance systems become unsafe. Therefore, designers have to make decisions concerning the systems' reliability at the various design levels, and the performance and safety abilities of the systems as well. Some timing characteristics of possible failure detection and recovery may be very important in the decision making process.
In this paper a concept of principal pieces of such framework will be considered. Namely, possible tools matching the current state-of-the art of mathematical modeling of self-recovering features of the fault-tolerant Smart systems are considered.

KEYWORDS
embedded system modeling, fault-tolerant computing, self-checking, fault latency, finite-state machines, Markov chains

1. Introduction

Computing systems for many applications must be *fault-tolerant* to be able to continue operating despite limited failures of portions of their hardware or software [4].

The fault-tolerant properties are provided by using of various types of redundancy (time, hardware, information) which may, in particular, provide fault-tolerance via achieving the phenomena self-healing, self-stabilization, self-reconfiguration [1,2,4].

Self-healing refers to the system's ability to detect failures in any of its components or interaction protocols to correct them so that the work is not interrupted. The mechanism of self-healing enables the system to continue operating properly on the event of the failure of some of its components, to determine the errors and to recover from them. For example, a concept of a partially monotonic Finite State Machine (FSM), where the transitions are computed by partially monotonic Boolean functions is used to provide self-healing properties. In particular, if we consider a self-checking digital circuit design, the different properties of logical functions may provide self-healing properties of the circuit [1].

The self-healing notion can be very interested for reliability modeling of embedded systems in presence of transient faults .
The architecture that supports the self-healing property of the FSM is a well-known self-checking architecture [1], that uses output self-checking checker.

In along with self-healing, *self-stabilization* also belongs to important fault-tolerant computing aspects. A system is said to be *self-stabilizing*, if starting from any state, it is guaranteed that the system will eventually reach a correct state (*convergence*). Given that the system is in a correct state, it is guaranteed that it will be stayed in a correct state, provided that no further fault happens (*closure*). For example, a distributed algorithm *A* is self-stabilizing if, whatever the initial configuration it starts from, it reaches within a finite time a set *L* of "legal" configurations, i.e, configurations satisfying a desired property [5].

Mostly important numerical characteristics of both these phenomena are the time for the recovery from arbitrary state disruptions. For example, if we explore a self-stabilizing algorithm *A, it may be* the expected time of reaching a set *L* of the legal configurations [5], which are usually a subset of states or processes, proper from some viewpoint of a target system design [6].
For the FSM, it can be the average number of

clocks required for the FSM healing.

Forms of fault tolerance can be divided into masking fault tolerance and non-masking fault tolerance [2]. For non-masking fault tolerant systems, the time of the fault detection is very important attribute since it can help avoid a failure if a fault-tolerance mechanism is available. One may try to minimize the *fault detection latency*, i.e., the difference between the first time the error is activated and the first time it is detected.

Since, in general, both input data and faults appear randomly, this time is a random value. Therefore, the self-healing models should be probabilistic ones. A natural mathematical model of this phenomena is a random transition from one state to another, thereby some states can be amalgamated in "*configurations"* mentioned above, which than consider as new states. The semantics of this transitions may be considered in terms of finite state machines(FSM) [6].

This formalism may be used for various applications, from HW circuits (of sequential automata) to a program implementation of distributed algorithms, where interacted processes of computation may be expressed as FSM.

Since in order to define the recovery time of a system we should compare behavior of both fault-free and faulty systems, we need a mathematical model for such comparison, which can be characterized as a model of expression of a "coupling" conditions of the models. As for FSM-based models, both the direct product of both automata and Kronecker forms are used in order to define a probabilistic state of self-healing time model [7, 8]. That is the model of the FSMs product behavior under random input vectors is a Markov chain, whose state space can be constructed as a direct product of fault-free and faulty state transitions, as well as Kronecker product of the states. Another example of the coupling is, a *coupling of Markov chains* method as a way of such pairs modeling for a timing characterization of self-stabilizing systems [5].

In this paper we consider mostly first mathematical conception of model coupling for fault detection latency analysis.

The paper is organized as follows. Section 2 describes the basic definitions and assumptions. The latency distribution functions for faults in the FSM are discussed in Section 3. Concludes are presented in Section 4.

2. Principal requirements to the design tools

To predict and estimate timing characteristics of fault-tolerance features at early steps of the design process, designers need a set of tools to compute some metrics at different design levels to guide the process of fault-tolerant computer systems design. These metrics should characterize self-healing time, fault manifestation latency, self-stabilizing distributed algorithms, etc. Taking into account that both logical and numerical-probabilistic tools are used by designers of fault-tolerant systems, a theoretical-methodological framework for coordination of these two aspects should be developed. The tools should allow designers to consider both permanent and transient faults. For example, from the reliability point of view, in the case of designing at the level of finite state machine (FSM), for the permanent faults it is reasonable to maximize the probability of transition from a state, where the fault is in a *latent mode* [1], that is where the presence of a fault cannot be detected since a test vector detecting the fault has not yet appeared at the circuit's inputs, to an erroneous mode. But for transient faults it would be more reasonable to maximize the probability of the FSM transition from the latent mode to the fault-free mode. It is clear that this transition in the fault free mode is a self-healing of the FSM under transient fault effect. Some corresponding Markov models are considered.

As it mentioned above, the designers need both the logical-algebraic (LA) and probabilistic models-based interacted tools to support the fault-tolerant system design. The logic-semantic aspects of fault-tolerant computer systems design are usually concerned with the use of various abstractions in the modelling of the fault detection and fault robustness properties. These aspects deal, for example, with logical analysis for detecting conditions of both temporary and permanent fault effects, dependability and data integrity, liveness, safety, synchronization, etc. In particular, a concept of a partially monotonic FSM, where the transitions are computed by partially monotonic Boolean functions, is used to provide self-healing properties, that is if we consider a self-checking digital circuit design, the different properties of logical functions may provide self-healing properties of the circuit [1]. Therefore, the designers need a generalized model of systems, represented in terms of interacting processes and interacting FSMs, which should provide the designers with possibilities of choosing the design solution into all current suit of fault-tolerant paradigms. Namely, it would be desirable to have the models that will consider every possible system's configuration and all possible transitions between configurations. They should provide the possibility to specify the target systems in such way that all configurations belonging to both fault-free/faulty classes would be characterized by local predicates defined over the states of the individual processes..

A relationship between LA and probabilistic methods is reached by defining a Markov chain or semi-Markov processes over the space of the

configurations and their transitions. The numerical probabilistic Markov models of the systems can be used on the state space, generated by these models of a qualitative nature.

One very important area of research is the development of adequate Markov models which, in contrast to existing models, will allow for the expression of fault-latency and self-healing probabilistic characteristics in terms of the characteristics of components of an FSM composition (network). The tools should allow the designer to

- calculate the transition probability matrix for transitions due to a single step of FSM.
- calculate the transition probability matrix for transitions due to fault steps, and then obtain a transition probability matrix representing an appropriate Markov chain of a product of fault-free and faulty FSMs.

In our model, as we mentioned above, we will estimate the fault latency by comparing both fault-free and faulty FSMs, which is performed implemented by a direct product of the two version of FSMs, which firstly has been considered in [7]. First of all, let us outline an alternative approach that is not based explicitly on the product of the two FSMs version. This approach was studied in [9]. The main idea of the approach is to divide the whole set of possible trajectories of the Markov chain describing behavior of the FSM in presence of a stuck-at-fault, into two subsets. The first subset does not contain trajectories manifesting a particular fault. The second subset of trajectories includes the fault manifesting states. Then, the probability of the fault manifestation at the *t*-th step is equal to the probability that the process moves along the trajectories from the first subset over *t*-1 steps and then, falls into the second subset at the *t*-th step. Consequently, the fault is detected exactly at the *t*-th step. The transition probability matrix of the Markov chain is computed by the known probabilities of input vectors and transition tables of the FSM.

It is essentially, however, that using this method it is possible to compute directly just the *fault* cumulative distribution function but not latency probability distribution of various errors, which can be caused by the physical fault (Another problem of this approach deals with some difficulties of formal definition of the trajectories, which can prevent suitable automation of the computations.

3 FSM-base fault manifestation model

As it mentioned above, in order to define the time of an observable event after a fault manifestation we need to compare the behavior both fault-free and faulty FSMs.

Let FSM be a Mealy machine, with the state set $\{a_1,\ldots,a_n\}$, the input set ,x_t,=$\{x_1(t),..x_m(t)\}$ and the output set $y_t=\{y_1(t),..y_n(t)\}$. Functions δ and λ are multiple-output Boolean functions which are a relation between the (input state, present state) pairs and the next states (δ), and between the (input state, present state) pairs and the output states (λ). Let the input words of the FSM be a randomly generated test input vector sequence. In this case there are two possible general models to compare the behavior of the fault-free and faulty FSM defined either in a product of transition spaces of both FSMs or on some pairs of states of Markov chains, corresponding to random transitions of both FSMs. We denote the first type as a "FSMs product-based model", while the latter we consider as a coupling time for corresponding Markov chains with a given states of initial state distributions.

3.1 Fault-Free and Faulty Behavior Models

1. *Fault model.* Not specifying exactly all fault models considered here, note, that we consider any faults which are resulted in a change of the FSM states (and, correspondingly, provoke an error). So, principally they may be both *permanent* (that remain in existence indefinitely if no corrective action is taken), and *transient* ones (that appear and disappear quickly). However, to provide a complete analysis for transient fault, we have to add the latency model by distributions of the fault duration and faults occurrence while in our analysis we assume that the probability of occurrence of second fault during effect of the first fault is negligibly small.

The model [7] *assumption* is statistical independence of random input vectors from one time step to another, and that their probability distribution is fixed so that, at any given time step, input takes place with probability $Prob(x_k=1)=p_k$.

3.2 FSMs Product Model

To compute fault latency and fault manifestation, both FSMs are considered as operating in parallel. These functioning can be represented as a direct product of both. The following assumptions are stated for the FSMs product model:

1) The signals with the same input probabilities distributions $Pr(x_i)$, i=1,..M are numbers of input variables, are applied simultaneously to the both faulty and fault–free FSMs. For example, the transition from a_1 to a_2 in Table 1 has place with

the probability $Pr(x_2)(1-Pr(x_3)$ since this is a Boolean conjunction of the Boolean variables x_2 and the negation of x_3. Initial states of thy FSMs are equal.

The product A_p of fault - free and faulty two FSMs, is a table of all $n^2 \times n^2$ pairs of the FSMs states, that is:

$$A_p = \{(s_i, s_j^f)\} \quad (2)$$

where s_i is a state of the fault-free FSM, s_j^f is a state of f –faulty FSM, i,j=1,…n.

Note that if FSM A_p it is not irreducible, the fault step probability matrix A^f may be transformed to make its irreducible.

The stochastic behavior of the product of two FSMs under independent random input vectors is also the homogeneous Markov chain with the state space $X=\{ (S, S^f) \cap s_A\}$, where s_A is an absorbing state of the Markov chain, which corresponds to the state when the outputs of both FSM become unequal. That is the state space of the Markov chain of the FSM defined on the $n^2 \times n^2$ pairs plus an absorbing state, corresponding to lack of coincidence of the FSMs outputs at a given pair of states, that is $Y(s_m,s_n) \neq Y(s_p^f, s_q^f)$. So, we may consider any errors in FSM behavior, provoked by a fault f, as a such pair (s_i, s_j^f), and also we may characterize the latency properties of a FSM as a number of clocks between the moment of hit in the state (s_i, s_j^f) of the product FSM and the moment of its observation on its output. For example, for the FSM Table 1, the pair (1,2) means that fault-free FSM is in state 1 while faulty one is in state 2. (The format of the table in the figure 1 has produced by a software describing below. Numbers 1,2 are indexes of states a_1, a_2. " -- " means all bits are 1s). Correspondingly, transition (1,2) → (2,2) in the product FSM (figure 1) means that the fault-free FSM transits from state 1 to state 2 while the faulty one leaves in state 2. Since the transitions of the product FSM are determined by (time-and-space) independent input random values, the product FSM can be considered as a Markov chain. The transition probabilities are computed by known probabilities of input variables $x_1,\ldots x_7$.. So, the size of the transition matrix of the Markov chain is $(n^2+1)\times(n^2+1)$.

The transition probabilities matrix of the Markov chain is

$$PTM=\{p(a_i, a_j)\} \quad (3)$$

where a_i, a_j (i,j =1,..,n^2+1) corresponding to the states $\{s_l, s_m^f\}$, $\{s_p, s_q^f\}$, l,m, p,q = 1,…n.

The latency L_f of a fault f may be represented as

$$Prob(L_f=k)=Prob(S_{Abs}(t+k)/S_m(t))$$

$$Prob(L_f=k) = \{p_{S,, Abs}\}^k \quad (4)$$

where $S_m(t)$ is a state (indexed m) of the faulty FSM which where corrupted by the fault f at the clock t ("fault manifestation at clock t"), k is the number of clocks between the fault manifestation and the fault detection, that is clock of time when first $Y(s_m,s_n) \neq Y(s_p^f, s_q^f)$, and corresponding states are considered as an "absorbing state"($S_{Abs}(t+k)$). The Markov chain gets from a state s to its absorbing state (Abs) for k clocks, that is the matrix entry (s, S_{Abs}) of the k-th power of the transition probability matrix (3). This conditional probability allows to analyze latency regarding to a specific state of the FSM.

To find the probability distribution function of the fault latency $Prob(L_f \leq t)$ we should compute the probabilities $Prob(L_f = k)$, k=1,2,…, which are defined by computing the state probability vector **P**(k) for k-th step of the Markov chain transitions:

$$\mathbf{P}(k) = \mathbf{P}(0)(PTM)^k \quad (5)$$

Table 1. The transition table of the FSM

a_m	a_s	$X(a_m,a_s)$	$Y(a_m,a_s)$
a_1	a_1	$\bar{x}_2$	5,4
	a_2	$x_2\bar{x}_3$	3,2
	a_3	x_2x_3	4,2
a_2	a_4	x_7	4,6
	a_5	$\bar{x}_7$	4,6
a_3	a_4	——	——
a_4	a_1	$x_1x_2x_3$	1
	a_2	$x_1\bar{x}_3$	1
	a_3	$\bar{x}1$	7,4,5
	a_5	$x_1\bar{x}_2x_3$	1
a_5	a_1	$\bar{x}3\ \bar{x}9$	4,6
	a_2	x_3	4,6
	a_3	$\bar{x}3x_9$	4,6

	(1,1)	(1,2)	(2,1)	(2,2)	Absorb State
(1,1)	0,80000000	0,00000000	0,00000000	0,06000000	0,14000000
(1,2)	0,00000000	0,00000000	0,00000000	0,00000000	1,00000000
(2,1)	0,00000000	0,00000000	0,00000000	0,00000000	1,00000000
(2,2)	0,37000000	0,00000000	0,63000000	0,00000000	0,00000000
Absorb State	0,00000000	0,00000000	0,00000000	0,00000000	1,00000000

Figure 1. Product matrix of a FSM

where **P**(0) is n^2+1-bit vector of initial states probabilities distributions, thereby, only components {i,i}, i=1,…2n+1 can be non-zero,

because of assumption about equality of initial states of faulty and fault-free FSMs.

3.3 Description of the software

The approach for computation of the probability distribution function has been implemented as a specific software. The input of the software is an FSM description (like in the Table 1), as well as set of stuck-at-faults description (it may be input, memory, and outputs faults), which is used to produce automatically corresponding Markov chains both fault-free and faulty, and the Markov chain of the product of these two FSMs. The output of this program is:

- transition table of faulty FSM,
-probability transition tables of all the Markov chains,
- probabilities to move from a pair $=\{(s_i, s_j^f)\}$ to the absorbing state for k steps (k=1,2..), which is the probability distribution function of the error (s_i,s_j^f) latency. This error is "the fault-free FSM is in the state s_i, the actual faulty FSM is in the state s_j^f and the outputs of both FSMs are different".

As example, the figure 2 shows the probability distribution function of the error (6,1) of FSM described in table 2 caused by the stuck-at-fault x4≡1 (marked as x4/1 on the picture).

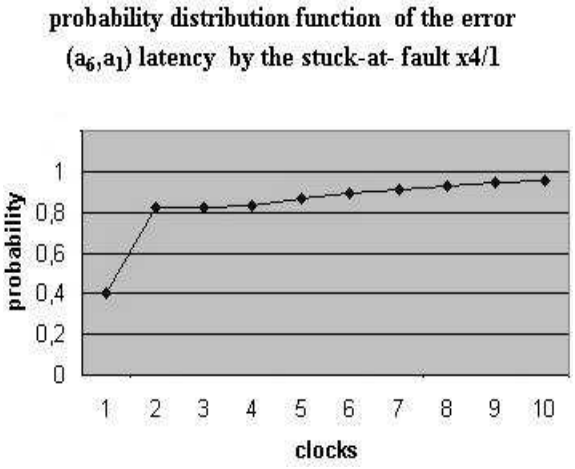

Figure 2.Error latency probability distribution function

3.4 Fault detection latency of the FSMs NETWORK

Let a FSM C was decomposed unto a network of its components FSMs $C^1,..,C^k$ in such a way, that at each clock the only one of them is in a working mode, that is, the output of the FSM is observable and it is used as a functional output of the target system, while others are in testing one under the working input signals (that is in *concurrent self-testing* mode).

The "testing mode" means that each of the component FSMs of the network are tested by entering working vectors to its input poles, but its outputs are observed the only to detect a possible faulty behavior not using their bits as an output of the target FSM. The observations are performed by a test monitor. The components are controlled by a supervisor, which switches the modes. In general, it is similar to well-known decomposition techniques like [11].

To illustrate this possibility let us consider FSM It would be highly desirable to model the FSM fault latency in terms of each of the components probabilistic characteristics, eventually, in terms of probabilistic transition matrix of corresponding Markov chains. Unfortunately, in general, the state transitions of the components are not Markovian ones since the transitions in each of them depend not only on the independent random inputs but also on transitions

Table 2. FSM transition table

a_m	a_s	$X(a_m,a_s)$	$Y(a_m,a_s)$	h
a_1	a_1	$x_3x_4x_5$	-	1
	a_5	$x_3x_4\bar{x}_5$	y_1y_2	2
	a_7	$x_3\bar{x}_4$	y_1y_3	3
	a_9	$\bar{x}_3$	y_2	4
a_2	a_1	$x_5x_6x_7$	$y_2y_3y_4$	5
	a_7	$x_5x_6\bar{x}_7$	y_3	6
	a_6	$x_5\bar{x}_6$	y_2y_3	7
	a_6	$\bar{x}_5$	y_2	8
a_3	a_6	x_5x_6	y_2y_3	9
	a_7	$x_5\bar{x}_6$	y_3y_4	10
	a_8	$\bar{x}_5\bar{x}_6$	y_3y_4	11
	a_2	$\bar{x}_5x_6$	y_2	12
a_4	a_3	x_6x_7	y_2	13
	a_5	$x_6\bar{x}_7$	$y_2\ y_3$	14
	a_7	$\bar{x}_6x_7$	y_3	15
	a_8	$\bar{x}_6\ \bar{x}_7$	y_3y_4	16
a_5	a_2	x_1x_2	y_3y_4	17
	a_3	$x_1\bar{x}_2$	y_4y_5	18
	a_4	$\bar{x}_1$	y_3y_5	19
a_6	a_2	x_4x_5	y_1y_3	20
	a_5	$x_4\bar{x}_5$	y_2	21
	a_7	$x_3\bar{x}_4$	y_1y_3	22
a_7	a_4	x_3x_2	y_2	23
	a_5	$x_3\bar{x}_2$	y_1y_3	24
	a_7	$\bar{x}_3$	y_3y_4	25
a_8	$a_{(8)}$	x_3x_4	y_2	26
	a_6	$x_3\bar{x}_4$	y_1y_3	27
	a_7	$\bar{x}_3x_5$	y_1y_3	28
	a_8	$\bar{x}_3\bar{x}_5$	-	29
a_9	a_3	x_1x_3	y_3	30
	a_4	$x_1\bar{x}_3$	y_3y_5	31
	a_5	$\bar{x}_1$	y_3y_4	32

in either components, which generate corresponding signals for the predefined states mentioned above. Therefore, the main problem is to define the Markov model of this network (or semi-Markov one; the only such models may be used as a analytical tool for the probabilistic modeling). Note that in [10] a FSM is decomposed into a network of smaller component FSMs, which describe the random transitions from one mode to another. However, only one component can be active in this decomposition model, while others are waiting for their activation, not being tested as above.

3.5 Markov Model of Network of Component FSMs

Let us consider, like as in previous Section, a states space of the fault-free and a faulty decomposed FSM as $S^p = \{(S_{i,j,k}(n), S^f_{i,}j,k(n))\}$, where, however, all these states are a tuple of the components (both fault-free and faulty) states:

$$S_{i,j,k}(n) = \{ a_i, a_j , a_k\}_n$$
$$S^f_{i,j,k}(n) = \{ a_i^f, a_j^f, a_k^f\}_n \qquad (6)$$

where n is a clock number, the number of a position in the tuples is the number of the component (that is, 1,2, or 3), $i =(i_1,\ldots i_M)$, $j=(j_1,\ldots,j_N)$, $k=(k_1,\ldots,k_Q)$ are a numeration of the states of components of both fault-free and faulty network. Note, that we consider the same faults that in the initial FSM (we do not consider, say, a supervisor's faults).

It may be shown, that the random transitions in S^p under random inputs signals X_n on the poles of the component FSMs may be considered as a Markov chain with the transition probabilities matrix like (2-3).

Since all outputs both of the working and the tested components are observed, the latency also can be modeled as a hit of the Markov chain in an absorbing state, and this absorbing state of the Markov chain is defined relatively to all the outputs.

Figure 3 represents the fault latency probabilistic distribution functions of FSM described by table 2, computed and FSM decomposed by the technique mentioned below, when the fault $x_4 \equiv 1$ is manifested in state a_6 .

We can see some reduction of the latency due to the decomposition. Although this reduction seems not very significant, in some design situations even such reduction may be reasonable. Note, that the degree of latency reduction via decomposition depends considerably of the distribution of input signals, the transition relationships, and the faults. We have some simulation results, showing the possibility of essential latency reduction due to the composition , but the topic of the paper is not the methods of the reduction but rather the methods of the latency computation in a network of FSMs.

3.6 Kronecker Product-base Approach

Obviously, in general case the number of possible states of the Markov chain can be very large

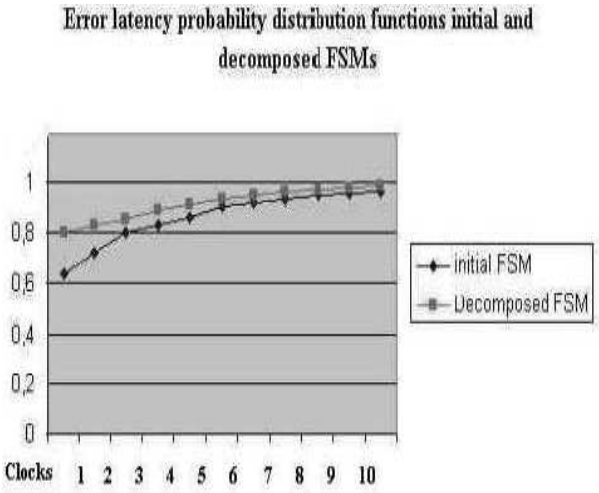

Figure 3. An example of error latency reduction by a FSM decomposition

because we are forced to deal with the combination of the tuples (6). So, the practical way of the using of the Markov model is a reducing of the state space. In general, it may be achieved if there are unreachable states in the product matrix, which should be avoided from the consideration.

In general, the number of states of the component network is $\prod n(i)$. However, due to the transitions obey the "synchronizing conditions" with some predefined state, not all of these states must be reachable. In general, such analysis deals with constructing and analyzing of the states reachability tree, which is used, for example, in the area of circuits design verification on the base of probabilistic Model-checking [12]. In general, as it practices often in the area of using of stochastic automata network (SAN) tools [12], it would be attractive to use the Kronecker product –based approach. This technique could provide representing a large state space as a cross product of small state spaces. However, since of the transitions of all components are dependent due to dependencies of input signals of each components, we will have to consider the products of pair of component's matrices, that is dimension of the problem will not be reduced.

Nevertheless, in some specific cases it is possible to represent the decomposed product in more

compact way to accelerate the computations. As example, let us consider the case when the BIST techniques [3] is used. In this case, due to BIST systems include a test pattern generator which applies a sequence of test patterns to a circuit with the BIST mechanism. Therefore, if the FSM has such built-in mechanism, the components in test mode can be tested by the embedded test generator, while the component in working is under corresponding working input vectors. Note that the test vectors are applied to the normal inputs of the components (there are not any additional test inputs).

The main idea of such decomposition scheme would be the optimization of testing process (e.g., enhancing fault coverage) thanks to use the input test vectors generated in a specific way [3] which, however, also can be modeled as random sequences with known probabilistic distribution.

Again let us consider three-components decompositions. In this case, the random transitions into the components in the testing mode can be considered as independent of the transitions of working mode, but the transitions are mutual dependent in two components in the instance when takes place a change the modes of the components (when transitions of a working component reaches a predefined state of type $a_{(l)}$. Therefore, we may structurize the state space of the overall components network, taking into account that in each of clock, at least behavior of one of components is independent of two other.

First of all, let us remark, that all transition trajectories, which are generated by the components FSM S_i, i=1,2,3 in the working modes will cover all transitions in the initial FSM (table 2), that is, if we copy out sequentially all transitions of FSMs when they are in the working mode, it turns out that the transitions will coincide with all transitions of the initial automaton. In other words, it can be proved that the working area of the decomposed FSM preserves the behavior of the initial one.

It is possible to prove that the probability transition matrix of the Markov chain describing the product of overall of fault-free and faulty 3-component networked FSMs can be represented as

$$P = \begin{pmatrix} P_{11}P_{12} & P_{13} \\ P_{21}\,P_{22} & P_{23} \\ P_{31}\,P_{32} & P_{33} \end{pmatrix} \qquad (7)$$

where P_{ij}, i,j =1,3, is a transition probability matrix of a Markov chains corresponding the state spaces of combinations of various pairs of fault-free/faulty components, namely:

$$\begin{aligned} P_{11} &= A_{11} \otimes B_{22} \otimes B_{33} \\ P_{22} &= B_{11} \otimes A_{22} \otimes B_{33} \\ P_{33} &= B_{11} \otimes B_{22} \otimes A_{33} \end{aligned} \qquad (8)$$

$A_{ii,}$ i=1,2,3, is the transition probability matrix of the product of fault-free and faulty component i in working mode, which expresses the probability, that under fault f all transitions in the fault-free component i in the working mode are resulted in the same output that the faulty component, that is probability of transitions pair $\{(s_i, s_j^f), (s_k, s_m^f)\}$, that means the transition of fault-free component is $s_i \rightarrow s_k$ while if the component is faulty, that under the same input vector the transition will be $s_k^f \rightarrow s_m^f$.

Note, that the states (s_k, s_m^f) may belong to other components, as in working mode a transition from a transient state $a_{(l)}$ may lead to a state of any of the three components (subsection 4.1).

B_{ii} , j=1,2,3, is similar matrix of transition probabilities of product fault-free and faulty component j in testing mode inside this component.

As for non-diagonal probabilities matrixes P_{ij}, they are expressed in terms of some transition probability matrixes Q_{ij} which represent the probabilities, that following transitions are resulted with some equal outputs of fault-free and faulty components:

-the transitions of fault-free component i being in the working mode in an inner state j_1 caused by transition in a predefined state of type $a_{(l)}$,

-the transition in the component j being in that time in a testing mode in a state j_2 to a state k_2, in this component,

- the transition of the corresponding faulty components, taking into account, that predefined state $a_{(l)}$ is equal for both fault-free and faulty component i due to the principal assumption mentioned above (Section 3) that initial states of both fault-free and faulty FSMs are equal.

For example, the probability matrix

$$P_{12} = Q_{12} \otimes B_{33}$$

where Q_{12} is the probabilities of the following pairs of transitions of $\{i_1,j_1, i_2,j_2\} \rightarrow \{k_1,l_1,k_2,l_2\}$, where

- i_1 is the state of fault-free component 1 in the working mode, j_1 is the state of faulty component 1 in the working mode,
- i_2, j_2 are correspondingly the fault-free and faulty component 2,
- $k_1=l_1$ is the predefined state in the component 1 (the state type $a_{(l)}$),
-k_2 is the state of fault-free component 2 where moved component 1 (correspondingly, 2 became in the working mode),

- l_2 is the state of the faulty component 2 to which the faulty working component has moved (becoming in this time in a testing mode), that is l_2 is the state to which moved the j_1),

The other transitions probabilities P_{km} (P_{21}, $P_{31,}$ etc.) are computed also using the transition probabilities Q_{km} of similar pairs of transitions, taking into account the non-comminicavity of Kronecker product [13]

The probability of k clocks latency Pr(L=k) can be computed as :

$$Pr(L=k)= 1 - p_kP \times (1)^T$$

where $p_k = p_{k-1}P$ is the state probability vector of the network on k-th step, T means the 1-vector transposition.

4. Conclusion, discussion and future work

In this paper we analyze a possibility to use some probabilistic model of fault detection latency, based on representation of FSM transitions as a Markov chain, as a tool of self-testing systems design. We consider both fault latency and error latency characteristics. In this paper we analyze a possibility to use well-known probabilistic models of fault detection latency, based on representation of FSM transitions as a Markov chain, as a tool of self-testing systems decompositional design. They are strongly related to the concept of the potential and the real latencies, which are applicable for self-checking systems [1]). The latency model that has been used is based on both fault-free and faulty FSMs transition tables. On the one hand, this model is more precise in comparison with others, and contains maximum possible information (in terms of target FSM behavior) about the latency. On the other hand, since the computational complexity is quadratic on the number of states, it may be very time-consuming, especially for the analysis of the decomposed FSM. In order to implement this approach for rather large size state spaces, it is necessary to use some advanced methods of search such sets of unreachability, which, in fact, may require to involve some additional search-in-trees tolls (e.g., BDD–based tools), which are used, for example, in probabilistic Model-checking [12]. Such combination may be reasonable (at least, from an economic point of view) if such tool are used for target systems verification reasons. It would mean inclusion the latency analysis into the general roadmap of computer-aided design.

References

[1] Levin I., Ostrovsky V., et al., Self-checking Sequential Circuits with Self-healing Ability. Proceedings of the 12-th ACM Great Lake Symposium on VLSI, New York, 71-76, 2002.

[2] A. Dhama, O. Theel, T. Warns, Reliability and Availability Analysis of Self-Stabilizing Systems, *Proc. of SSS 2006* . Dallas, TX, USA, Nov. 17-19, 2006, pp. 244-261.

[3] Al-Asaad et al, Online BIST for embedded systems, *IEEE Des. and Test of Comp.*, V.15, No. 4, pp. 17-24.

[4] D. Buel, High-Performance Reconfigurable Computing, *IEEE Computer*, March 2007, pp.3.

[5] Laurent Fribourg · St´ephane Messika · Claudine Picaronny Coupling and self-stabilization, *Distrib. Comput*. DOI 10.1007/s00446-005-0142-7, Special issue: DISC 04 (2005).

[6] Abhishek Dhama, Oliver Theel, and Timo Warns, Reliability and Availability Analysis of Self-Stabilizing Systems, *Proceedings* of 8th *Int. Symposium SSS2006 on Stabilization, Safety and security of Distributed systems*, 2006.

[7] Shedletsky J., McCluskey E., 1976 ,"The Error Latency o f Fault in a Sequential Digital Circuit", *IEEE Transaction on Computers*, vol. 25, No 6, 1976, pp. 655-659.

[8] Hadjicostis. Chr. N,Aliasing Probability Calculations for Arbitrary Compaction under Independently Selected Random Test Vectors, *IEEE Trans. on Computers*, Vol. 54, no. 12 (DEC), 2005.

[9] Goot R., Levin I., Ostanin S., 2002, Fault Latencies of Concurrent Checking FSMs, *Proceedings of the Euromicro Symposium on Digital System Design* (DSD'02).

[10]Goot, R., Levin, I., Ostanin, S., Statistical Analysis of Decomposition Automata. *Automatic Control and Computer Science*. Vol. 37, No. 4, 2003, pp. 6-13.

[11] Lim, K.. Devadas S., 1990, Performance-oriented decomposition of sequential circuits. On *IEEE International Symposium on circuits and Systems* vol.4 pp. 2642 - 2645.

[12]Buchholz P.,et al., Model-checking Large Structured Markov Chains, *Journal of Logic and Algebraic Programming*, 2003, vol. 56, p. 69-97.

[13] Graham A., Kronecker Products and Matrix Calculus with Applications, John Wiley and Sons, New York, 1981.

***Declaration**:*

All necessary clearances for the publication of this paper have been obtained. If accepted, the author will present the paper at the conference and will prepare the final manuscript in time for inclusion in the conference proceedings