Method for Unit Self-Diagnosis at System Level

Viktor Mashkov
Jan Evangelista Purkyně University in Ustí nad Labem, Ustí nad Labem, Czech Republic
E-mail: viktor.mashkov@ujep.cz

Volodymyr Lytvynenko
Kherson National Technical University, 73008 Kherson, Ukraine
E-mail: immun56@gmail.com

Abstract—This paper suggests unconventional approach to system level self-diagnosis. Traditionally, system level self-diagnosis focuses on determining the state of the units which are tested by other system units. In contrast, the suggested approach utilizes the results of tests performed by a system unit to determine its own state. Such diagnosis is in many respects close to self-testing, since a unit evaluates its own state, which is inherent in self-testing. However, as distinct from self-testing, in the suggested approach a unit evaluates it on the basis of tests that it does not performs on itself, but on other system units. The paper considers different diagnosis models with various testing assignments and different faulty assumptions including permanent and intermittent faults, and hybrid- fault situations. The diagnosis algorithm for identifying the unit's state has been developed, and correctness of the algorithm has been verified by computer simulation experiments.

Index Terms—System-level diagnosis, self-diagnosis, intermittent fault, hybrid-fault situations, computer simulation.

I. INTRODUCTION

System level self-diagnosis was introduced by Preparata et al. [1] and since then it has been deeply investigated in a number of researches [2,3]. It aims at diagnosing systems composed of units, with the requirement that they are able to test each other by exchanging data through available links. At this level of diagnosis, each particular test is considered as atomic, which means that the details of test are abstracted (i.e., not considered), and only the result of test is taken into consideration.

As a rule, decision about the state of a system unit is made on the basis of the results of tests performed on it by other system units. In view of this, testing assignment and the number of testing units are very important for system diagnosis [4].

The decision about the state of a system unit can also be made on the basis of comparison of the outcomes produced by one unit with the outcomes produced by other units [5]. It is worth noting that comparing testing, as a self-diagnosis tool, was developed only for homogeneous systems. In case of comparing testing, a test is a comparison of the outputs between paired units performing the same task. Comparing the outputs from two units, it allows only to detect, but not to diagnose a failure. Comparing the outputs from more than two units, it allows to diagnose certain number of faulty units (resp. fault-free units) depending on the chosen diagnosis technique.

Recently, a comparison-based diagnostic model has been widely used in mobile ad-hoc networks [6,7] and swarm systems. Swarm systems can be found in nature and engineering areas. Examples of engineering swarm systems are systems such as unmanned vehicle formation systems, multi-robot systems, sensor networks [8,9]. A swarm system consists of nodes and the topology of nodes. The nodes are intelligent which means that they can process data by themselves. Nodes can communicate by exchanging of messages. Nodes of swarm system are prone to failure due to energy depletion and to their possible deployment in a harsh or hostile environment. Faults of nodes and faults related to the topology have significant influences on the operation of a swarm system and may have serious consequences. In view of this, the task of fault diagnosis is very important and still requires developing of more effective diagnosis methods and algorithms.

Generally, fault diagnosis in the swarm systems can be performed either by external diagnostor or by inner system resources. Although the external diagnostor can perform more effective fault detection by comparing to the system nodes, communication between the nodes and the external diagnostor can be problematic [5]. That is why much attention should be given to the fault detection based on the testing performed by system nodes (i.e., system self-diagnosis). Such self-diagnosis employs one of the following three architectures: centralized, hierarchical and distributed architecture [8]. Centralized architecture uses one of the system nodes as an embedded diagnostor (or central node). Other nodes are not capable to perform diagnosis and they only send necessary messages to the diagnostor [10-12]. The main drawbacks of the centralized architecture are: strong dependence on reliability of embedded diagnostor; dependence on the state of communication between embedded diagnostor and system nodes; both communication load and computation

load are increasing rapidly as the number of nodes increases. A hierarchical architecture was introduced in order to improve the scalability and reliability of the centralized fault diagnosis. In the given case, local diagnosis information is sent to the next layer of architecture for further diagnosis [13,14]. This architecture also depends considerably on reliability of system nodes. Apparently, a lesser is more extent than the centralized architecture. Distributed architecture was proposed to improve both centralized and hierarchical architectures. In this architecture, all nodes are equipped with fault diagnosis algorithm. Fault diagnosis for a node is achieved in a collaborative way and the results of the algorithms in the neighbors of the monitored node are all needed [15,16]. It is worth noting that distributed architecture is much more reliable than the centralized one and the hierarchical one. As a limitation of this architecture, we can consider the fact that each node may use incorrect information received from other nodes (i.e., this architecture depends considerably on reliability of information which a node receives from its neighbors).

System units can be subject to sophisticated attacks which attempt to manipulate the outputs of each unit [17]. In case, when system units are deployed in a hash or hostile environment, probability of fault of each unit is generally expected to be high [5]. Therefore, related or multiple faults of units are very probable. At present, some problems of fault diagnosis in swarm systems still remain. First, this concerns the topologies that may fail or change dynamically and the heterogeneous units [8]. In this paper, we propose the new method of system self-diagnosis which can cope with all these situations (i.e., related or multiple faults of system nodes, changing topology and heterogeneity of nodes). Our method utilizes the results of series of tests performed by a node on other system nodes. The proposed self- diagnosis can deal with arbitrary topology (in the given case, it is called testing environment) and can be applied to heterogeneous systems in which a system node cannot be diagnosed by other system nodes. The proposed self-diagnosis can also cope with detecting and identifying the intermittent faults of nodes. An intermittent fault is a fault that produces negative effect only part of the time.

The key idea of this self-diagnosis consists in forcing the faults of a node to exhibit themselves during the testing which a node performs on other nodes. Particularly, the node faults could influence the results of tests. For each particular type of swarm system (e.g., multi-robot systems, sensor networks) evaluation of such influence can be done on the basis of results of special research. However, this task is beyond the scope of this paper. In the paper, we assume that a fault of a system node has effected the test results, and this fact is used for the fault detection. We also assume that each system node has a fault-tolerant part which always allows it to diagnose itself correctly. Similar assumption is made for self-testing of multiprocessors [18]. For the implementation of the method, it needs that a unit is capable to perform tests on its neighbors. The simulation performed on the basis of the proposed method enable evaluating the performance of the developed diagnosis algorithm and showing the efficiency of the suggested approach in the situations when majority of other approaches yield unsatisfactory results (e.g., in the situations when more than half of the system units are faulty).

The proposed diagnosis procedure is time consuming. In view of this, the paper focuses on the issue of determining the time needed for correct diagnosis. Particularly, we have determined how many times a unit must test the other system units in order to achieve highly credible result of diagnosis. The proposed self-diagnosis was first introduced in previous research work [19]. In this paper, we have modified and improved the diagnosis of intermittent faults and performed complete modeling of unit's self-diagnosis.

In the following sections, it will describe related works on the given topic in section 2. It will describe a proposed unit's self-diagnosis for different faulty assumptions section 3. The developed diagnosis algorithm is presented in section 4. The results of the performed computer simulation are shown in Section 5.

II. RELATED WORK

Comparison-based system level diagnosis has been researched in many papers, and it has been analyzed and structured in many surveys. This diagnosis was considered for possible implementation in wireless sensor networks, in multi-robot systems, in many-core processors and other areas.

In case of wireless sensor networks, researchers take into account the network topology (changed or fixed), types of possible node and communication faults (permanent or transient/soft/intermittent), protocol of communication among the nodes (one-to-one, one-to-many, one-to-all). Also, different comparing-based models were proposed (e.g., generalized, broadcast, probabilistic).

Application of comparison-based model in sensor networks allows a sensor node to identify its own status based on the information received from the neighbors [17, 20-22]. Alternatively, node state can be determined by the other system nodes [23, 24]. In case of multiple or related faults of sensor nodes, the unit diagnosis may be incorrect.

The developed comprehensive mutual diagnosis [25] is built on top of self- tests and presents a combination of self-testing and comparing approach. This diagnosis applied to multi-core arrays implies that each core first executes itself tests and generates a test signature. Then, it sends this signature to its four neighbors. Each core compares its own signature with those received from its neighbors. Comparing of signatures allows the core to detect faulty neighbors. The proposed in [25] mutual diagnosis also depends considerably on the state of the unit neighbors. The majority of comparison-based models impose an upper bound on the number of faulty units in the system. Some researchers consider faulty situations in which multiple and/ or related faults take place. Related faults or common faults consist of some system units

simultaneously become faulty due to the same reason. J. Xu in [26] proposed to combine comparison testing with $t/(n-1)$ - variant programming. It is worth noting that the adjudicator (i.e., diagnostor) in this scheme intends to detect only the correct variant. However, in practice it is also important to detect the faulty units (variants). In [27], the adjudicator with extended functionality was presented. This adjudicator allows to detect not only the correct units but also all the faulty ones. Both these papers consider independent and related faults. It is worth noting that these schemes were designed only for software systems to provide their fault-tolerance.

Mostly, the self-diagnosis performed on the basis of comparison-based model is used to detect permanently faulty system units [28, 22]. S. Chessa and P. Santi [24] have considered the problem of fault identification in ad-hoc networks and presented a comparison-based diagnostic model based on the one-to-many communication paradigm. In this paper, the authors showed how both hard and soft faults can be detected. M. Elhadef et al. [6] have presented distributed comparison-based self-diagnosis protocol for wireless ad hoc networks. The proposed protocol identifies hard and soft faults. P.M. Khilar [29] proposed a distributed fault diagnosis algorithm for wireless sensor networks to diagnose intermittently faulty sensor nodes. It is worth noting that these researches do not consider a model to describe the behavior of soft/intermittent faults. However, the different behavior of such faults can have different impact on the system. A more detailed consideration of system faults which are different from hard faults will allow to provide a more effective system recovery.

Using the Bayesian approach for system diagnosis is a common practice. B. Krishnamahari and S. Iyengar [30] presented Bayesian fault recognition algorithm to solve the fault-event disambignation problem in sensor networks. X. Luo et al. [31] have proposed a fault-tolerant energy-efficient event detection paradigm for wireless sensor networks and presented Bayesian detection method.

In our paper, we also use this well-known approach to diagnose the state of a system unit. However, in this case, the event that a fault unit is split into three events. Explanation of such splitting is given in Section 5.

III. BASICS OF UNIT`S SELF-DIAGNOSIS

Normally, system level self-diagnosis provides unit diagnosis on the basis of the results of tests performed on the diagnosed unit. In view of this, the number of fault-free testing units and, consequently, the system testing assignment are very important for correct unit diagnosis [4].

In contrast, we propose to deal with the results of tests which a unit performs on other system units. It is expected that test results produced by a fault-free unit will be different from those produced by a faulty unit. Quality of such diagnosis depends considerably on the number of tested units, on the number of tests performed on each tested unit and also on the states of tested units.

For elucidation of the main parts of the proposed self-diagnosis, we consider a simple example (Fig.1) with only one tested unit. It is assumed that unit u_i has performed m identical successive tests on unit u_j. It is also assumed that the states of testing and tested units do not change while all m tests are performed. We consider that each test result r_{ij} can take the value either 0 or 1, which depends on the states of tested and testing units and on the assumptions made in relation to the test results.
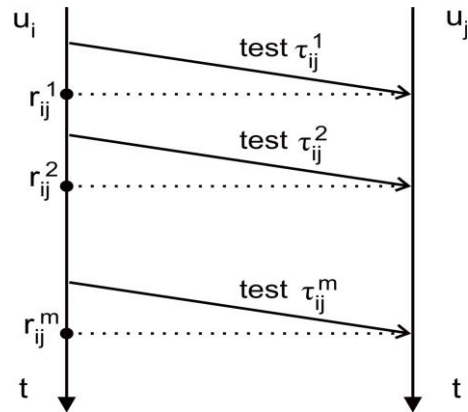

Fig.1. Repeated testing of unit u_j.

Assumptions made about test results and their probabilities can be expressed with the help of Fig.2.
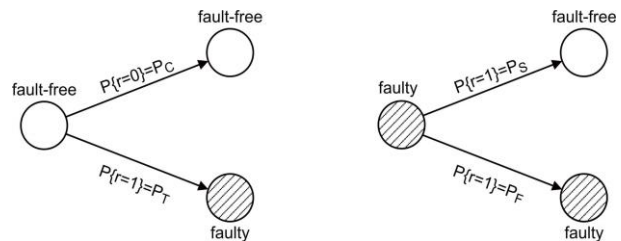

Fig.2. Test results and their probabilities.

In Fig.2, the following denotations are used:

P_C is the probability that a fault-free unit will correctly diagnose a tested fault-free unit. This probability takes into consideration the fact that connection between units can fail;

P_T is the probability that a fault-free unit will correctly diagnose a tested faulty unit. This probability reflects the quality of test (e.g., fault coverage);

P_S is the probability that a faulty unit will produce the test result equal to 1 when a tested unit is fault-free;

P_F is the probability that a faulty unit will produce the test result equal to 1 when a tested unit is faulty.

The assumptions made about test results can be specified and quantified. For example, PMC model [1] considers the following assumptions:

- Only permanent fault of a unit is possible;
- A fault-free unit always detects a fault in a tested unit (i.e., 100% fault coverage);
- Result of the test performed by a faulty unit is unpredictable not depending on the states of tested

units and can take the values either 0 or 1 with equal probability.

When the assumptions of PMC model are taken into consideration, the following specifications of the probabilities are possible: $P_C = 1$, $P_T = 1$ and $P_S = P_F = 0.5$. If the assumptions made in PMC model are applied to the test results in the considered example, then only two different tuples are possible (under the condition that unit u_i is fault-free). Each tuple has the length of m and consists of the results of single tests.

The first tuple $T_0 = (0, 0, .., 0)$ will be obtained when tested unit is fault-free, whereas the second one $T_1 = (1, 1, .., 1)$ will be obtained when tested unit is faulty. When unit u_i is faulty, the tuple will be obtained, with great probability, contain both 0 and 1. It is followed by the assumptions which made about test results. Thus, diagnosis algorithm consists in the following. If the obtained tuple is either T_0 or T_1, then the testing unit is fault-free. Otherwise, the testing unit is faulty. It is worth noting that faulty unit can also produce tuple equal to either T_0 or T_1. Probability of such result, P_{fn}, which can be determined as

$$P_{fn} = P_{T0} + P_{T1} \qquad (1)$$

where P_{T0} is the probability of obtaining tuple T_0, P_{T1} is the probability of obtaining tuple T_1.

If it assumes that all units have the same probability of fault, q, then

$$P_{T0} = (1-q)(1-P_S)^m + q(1-P_F)^m \qquad (2)$$

$$P_{T1} = (1-q)P_S^m + qP_F^m ; \qquad (3)$$

$$P_{fn} = (1-q)(1-P_S)^m + q(1-P_F)^m + \\ + (1-q)P_S^m + qP_F^m . \qquad (4)$$

Probability P_{fn} reflects Type II errors, and can be used for evaluating credibility of the performed diagnosis, D.

$$`D = 1 - P_{fn} . \qquad (5)$$

Fig. 3 and Fig. 4 give the idea of how parameters m, q, P_S and P_F can influence the credibility of diagnosis result.

The proposed diagnosis will slightly change if some of the assumptions made about test results are different from those that are accepted in PMC model. For example, we can accept the assumptions of BGM model [32] according to which a faulty testing unit will always produce the result equal to 1 when it tests any faulty unit (i.e., $P_F = 1$). Thus, one tested unit is used.
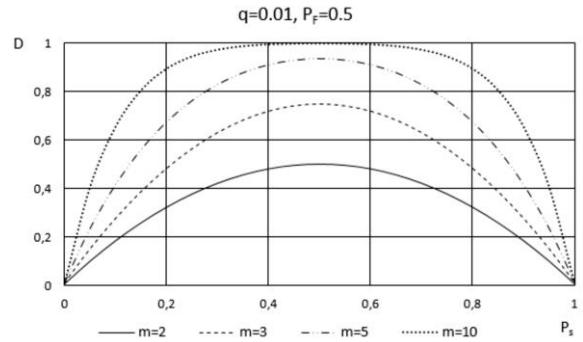

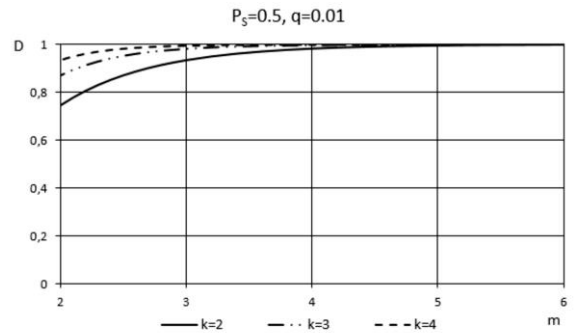Fig.3. Functional dependence D = f (PS).

Fig.4. Functional dependence $D = f(m)$.

Thus, the proposed diagnosis becomes less effective when only one tested unit is used. The effectiveness of the proposed diagnosis can be raised by using several tested units (Fig.5).
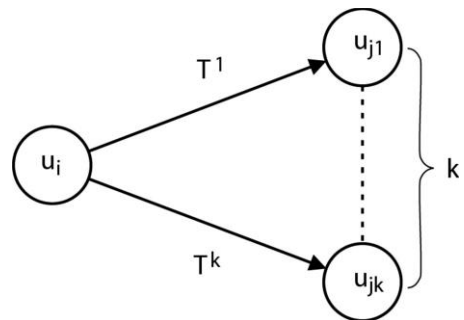

Fig.5. Repeated testing of k units.

When k tested units are used, we can expect, with great probability, that at least one of the tested units will be fault-free. This probability is equal to $1 - q^k$. Since a faulty testing unit after testing a fault free unit, it will get great probability (denoted as P_E), and produce the tuple different from T_1 and T_0, we can use this fact as the basis for diagnosis algorithm. Probability P_E can be computed as

$$P_E = 1 - P_S^m - (1-P_S)^m \qquad (6)$$

If testing unit is fault-free, all of the obtained tuples will be equal to either T_0 or T_1, which depends on the states of tested units. Thus, diagnosis algorithm is as follows. If each of the obtained tuples is either T_0 or T_1, then testing unit is fault-free. Otherwise, testing unit is faulty.

Credibility of such diagnosis can be evaluated on the basis of probability P_E. Diagnosis will be incorrect if testing unit is faulty and event A_i, $i \in [0, 1, .., k]$ takes place. Here A_i is the event when i tested units are fault-free, and each of the tuples is either of type T_0 or T_1. Probability of event A_i is determined as

$$P(A_i) = C_i^k q^{k-i} (1-q)^i P_d^i \qquad (7)$$

where probability $P_d = 1 - P_E$.

Probability of incorrect diagnosis, P_{ID} is equal to:

$$D = 1 - P_{ID} = 1 - \sum_{i=0}^{k} C_i^k q^{k-i} (1-q)^i P_d^i \qquad (8)$$

Thus,

$$P_{ID} = \sum_{i=0}^{k} P(A_i) = \sum_{i=0}^{k} C_i^k q^{k-i} (1-q)^i P_d^i. \qquad (9)$$

It is worth noting that the probability of incorrect diagnosis, P_{ID} is computed under the condition that testing unit is faulty. Fig.6 and Fig.7 shows the idea of how parameters m, q, k and P_S influence the credibility of proposed diagnosis.
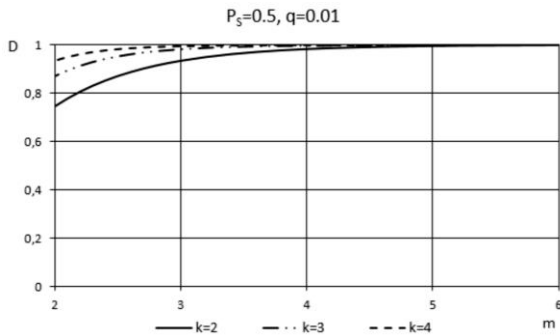

Fig.6. Functional dependence $D = f(m)$ for the best case.

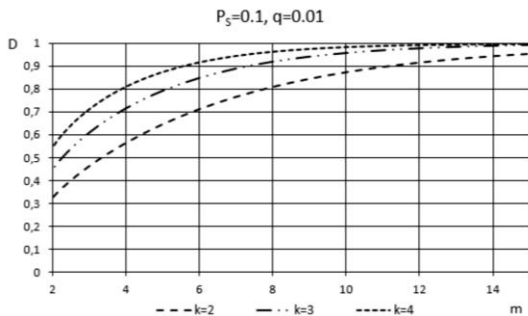Fig.7. Functional dependence $D = f(m)$ for the case of PS = 0.1.

Fig.6 presents the results for the case of $P_S = 0.5$. In the given case, credibility of diagnosis is greater than the credibility obtained for $P_S \neq 0.5$. Fig.7 depicts the credibility of diagnosis for $P_S = 0.1$. Even for this worse case, the number of test repetitions doesn't exceed a few dozens. For example, for $k > 2$, it is sufficient to repeat the tests 14 times to obtain highly credible result of diagnosis.

IV. DIAGNOSIS OF INTERMITTENT FAULT

Intermittent faults can be defined as the faults whose presence is bounded in time. More precisely, a unit can possess an intermittent fault but the effect of this fault is present only part of time. The amount of time of diagnosis procedure, t_d is important for diagnosis of intermittent faults. Depending on the amount of time t_d and on its position on the time axis (Fig. 8), the same fault may be identified as a permanent fault (case of t_d^1) and as an intermittent fault (case of t_d^2). There is also probability that the effect of intermittent fault will not be present during diagnosis procedure (case of t_d^3).
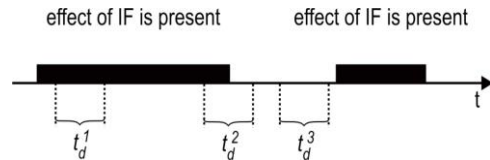

Fig.8. Diagnosis procedure and effect of intermittent fault.

Many researches have been done on the problem of diagnosis of intermittent faults. For instance, in [33], S. Kamal and V. Page considered the problem of the required number of test repetitions before a decision about the state of a digital circuit is made. At the beginning of testing, unit's state is indefinite. Testing procedure (i.e., repetition of tests) is terminated either when a fault is detected, or on the basis of a decision rule. The authors suggested several decision rules for termination of testing procedure with the outcome indicating unit's fault-free state. According to their research, the presence of intermittent fault in a unit can affect its behavior only part of the time. However, if the effect of an intermittent fault occurs during testing procedure, then such fault will be detected. Therefore, they described the behavior of intermittent faults (particularly, the occurrence of their effects) with the help of probability $P(S_i/w_i)$, where S_i denotes the state of the unit when it possesses intermittent fault w_i and the effect of the fault occurs.

Another approach to describe behavior of intermittent faults is presented in [34]. In this case, an intermittent fault has two states active (AS) and passive (PS). When an intermittent fault is in AS, the effect of intermittent fault is present. Whereas when an intermittent fault is in PS, its effect is not present. Transfers from one state to the other one are described with the corresponding intensities λ and μ (Fig.9).
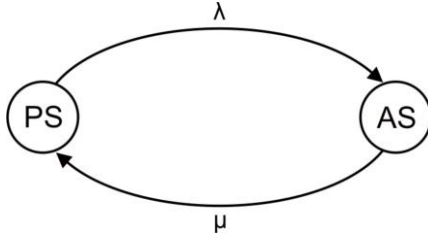
Fig.9. Model of intermittent fault.

The process of transfers between these two states can be described as continuous Markov chain, where the time of the process being in the given state is random value. This random value has exponential probability distribution. In our research, we have adopted this model of intermittent fault because it enables us to model and to examine a wide range of intermittent faults.

It is assumed that if a testing unit possesses an intermittent fault which testing is in the AS at the moment, then the result of the test will be affected by this fault, and such intermittent fault can be detected. In the sequel, this fault can be identified on the basis of the affected test result(s). It is evident that it is possible to detect different types of intermittent faults depending on the time allocated to testing procedure. Here, we introduce two classes of intermittent faults.

To the first class C_1, it refers to the intermittent faults which can frequently (more than once) appear in AS during testing procedure. To the second class C_2, we refer the intermittent faults which can rarely (not more than once) appear in AS during testing procedure. The introduced classification of intermittent faults is relative and depends considerably on the parameters of testing procedure. Such classification of intermittent faults is important for diagnosis based on limited time of testing procedure.

$$P\left(H_i/R\right)=\frac{P\left(H_i\right)P\left(R/H_i\right)}{\sum_{i=1}^{2}P\left(H_i\right)P\left(R/H_i\right)}, \quad (10)$$

V. DECISION RULE FOR UNIT`S SELF-DIAGNOSIS

As a rule, diagnosis algorithm is executed when checking procedure detects an error in system. The proposed unit's self-diagnosis adopts this approach, which means that if all test results are equal to 0, which suggests no diagnosis algorithms are executed. When at least one test result is equal to 1, diagnosis algorithm should figure out whether the unit is fault-free or not. This is done on the basis of the accepted decision rule.

We recommend using Bayes' rule

$$P(H_i/R)=\frac{P(H_i)P(R/H_i)}{\sum_{i=1}^{2}P(H_i)P(R/H_i)}$$

where R is a set of all test results (so-called, syndrome), H_1 is the hypothesis that a unit is fault-free, H_2 is the hypothesis that a unit is faulty.

The prior probabilities of the examined hypotheses can be determined as $P(H_1)=\xi$ and $P(H_2)=1-\xi$, where ξ is the probability that a unit is fault-free.

Decision about unit's state can be made by using likelihood ratio

$$\chi=\frac{P\left(H_1/R\right)}{P\left(H_2/R\right)} \quad (11)$$

and a chosen threshold ω. The value of ω can be set on the basis of Bayesian discriminant analysis, so as to minimize the average misjudgment cost [35].

Thus, if $\chi \geq \omega$, then hypothesis H_1 is accepted. Otherwise, additional measures should be undertaken to provide correct diagnosis.

The event that a fault unit can be split into three events P, C_1 and C_2, where

P is the event that a unit is permanently faulty;

C is the event that a unit possesses an intermittent fault of class C_1;

C is the event that a unit possesses an intermittent fault of class C_2.

For convenience, we denote with A the event that a unit is fault free, which corresponds to hypothesis H_1. With account of these events, the conditional probabilities $P(A|R)$, $P(P|R)$, $P(C_1|R)$ and $P(C_2|R)$ can be expressed as

$$P(A/R)=\frac{\xi P(R/A)}{\xi P(R/A)+P_P P(R/P)+P_{C_1}P(R/C_1)+P_{C_2}P(R/C_2)} \quad (12)$$

$$P(P/R)=\frac{P_P P(R/P)}{\xi P(R/A)+P_P P(R/P)+P_{C_1}P(R/C_1)+P_{C_2}P(R/C_2)} \quad (13)$$

$$P(C_1/R)=\frac{P_{C_1}P(R/C_1)}{\xi P(R/A)+P_P P(R/P)+P_{C_1}P(R/C_1)+P_{C_2}P(R/C_2)} \quad (14)$$

$$P(C_2/R)=\frac{P_{C_2}P(R/C_2)}{\xi P(R/A)+P_P P(R/P)+P_{C_1}P(R/C_1)+P_{C_2}P(R/C_2)} \quad (15)$$

where $\xi + P_P + P_{C1} + P_{C2} = 1$.

- P_P is the probability of the event that a unit is permanently faulty;
- P_{C1} is the probability of the event that a unit possesses an intermittent fault of class C_1.
- P_{C2} is the probability of the event that a unit possesses an intermittent fault of class C_2.

Given P_P, P_{C_1} and P_{C_2}, it is possible to compute conditional probabilities $P(A|R)$, $P(P|R)$, $P(C_1|R)$ and $P(C_2|R)$ and then it needs to choose two of them which have the greatest values.

VI. CONSIDERATIONS BEHIND THE DIAGNOSIS ALGORITHM

The situation when only one test result is not equal to 0, it means that the total number of "1" in the obtained syndrome R is equal to 1. It denotes such syndrome as R_1. This situation is possible only when performing test τ_{ij} either testing or tested unit at the moment, or both of them possess an intermittent fault in active state. In view of this, for examining this situation we suggest the following three hypotheses:

- h_1: unit u_i possesses an intermittent fault in AS and unit u_j is fault-free;
- h_2: unit u_j possesses an intermittent fault in AS and unit u_i is fault-free;
- h_3: both units u_i and u_j possess an intermittent fault in AS.

The probability of hypothesis h_3 is much lesser than the probabilities of hypotheses h_1 and h_2. In view of this, only hypotheses h_1 and h_2 are considered.

According to the Bayes' rule

$$P(h_1/(r_{ij}=1)) = \frac{P(h_1)P((r_{ij}=1)/h_1)}{\sum_{i-1}^{2} P(h_i)P((r_{ij}=1)/h_i)} \quad (16)$$

and

$$P(h_2/(r_{ij}=1)) = \frac{P(h_2)P((r_{ij}=1)/h_2)}{\sum_{i-1}^{2} P(h_i)P((r_{ij}=1)/h_i)}. \quad (17)$$

Consequently

$$\frac{P(h_1/(r_{ij}=1))}{P(h_2/(r_{ij}=1))} = \frac{P(h_1)P((r_{ij}=1)/h_1)}{P(h_2)P((r_{ij}=1)/h_2)}. \quad (18)$$

If information about the current state of unit u_j is not available, then $P(h_1) = P(h_2)$. If we assume that a unit possessing an intermittent fault in AS can produce test result 0 or 1 with equal probability, then $P((r_{ij} = 1)|h_1) = 0.5$. In case when $P_C = 1$ and $P_T = 1$, probability $P((r_{ij} = 1)|h_2) = 1$. Thus,

$$\rho = \frac{P(h_1/(r_{ij}=1))}{P(h_2/(r_{ij}=1))} = 0.5. \quad (19)$$

If unit u_j with certain probability η, it possesses an intermittent fault, then

$$\rho = \frac{P(h_1/(r_{ij}=1))}{P(h_2/(r_{ij}=1))} = \frac{1-\eta}{2\eta}. \quad (20)$$

The higher the value η is, the greater our confidence is.

unit u_i is fault-free. When the obtained syndrome contains more than one test results not equal to zero, it is necessary to compute conditional probabilities of receiving particular syndrome when a unit possesses certain type of fault (permanent, intermittent of class 1 or intermittent of class 2). Accurate determination of these probabilities presented in [4] which requires additional information about intermittent faults. In practice, obtaining this information is very difficult.

In view of this, we suggest the approximate method. This method consists in using approximate assessment of these conditional probabilities. Particularly, we assess these probabilities as either "very high" or "very small". In the latter case, we consider these probabilities equal to zero and omit them. This approach bears some resemblance to the techniques based on fuzzy logic [36, 37], as mentioned in [38] for an example.

Given P_C, P_T, P_S and P_F, each tuple T^α, $\alpha \in [1, .., k]$ can be expressed with the help of the table containing the total number of "1" in the tuple. For $P_C = 1$, $P_T = 1$, $P_S = 0.5$ and $P_F = 0.5$ such table takes the following view.

Table 1. Total number of "1" in the tuple

Total number of "1" in the resulting tuple, s		Tested unit			
		A	P	C_1	C_2
Testing unit	A	0	m	$1 \leq s < m$	0 U 1
	P	Chi{m/2}	Chi{m/2}	Chi{m/2}	Chi{m/2}
	C_1	$0 \leq s < m$	$1 \leq s < m$	$1 \leq s < m$	$0 \leq s < m$
	C_2	0 U 1	(m-1) U m	$1 \leq s < m$	-

In the table, the Chebyshev's inequality for the case when $mp = m/2$ is denoted as $Chi\{m/2\}$.

When a fault-free unit tests a fault-free unit (suppose m times), the resulting tuple will contain the total number of "1" equal to zero (in the table, see the intersection of the first row and the first column). When a tested unit is permanently faulty, the resulting tuple will contain the total number of "1" which is equal to m. When a tested unit possesses an intermittent fault of class C_1, the resulting tuple will contain "very high" probability, the total number of "1" satisfies the expression $1 \leq s < m$. If a tested unit possesses an intermittent fault of class C_2, then the resulting tuple will contain, with "very high" probability, the total number of "1" is equal to either zero or one (in Table 1, it is denoted as $0 \cup 1$).

Results produced by a permanently faulty unit are similar to tossing of fair coin. When a permanently faulty unit can produce result either 0 or 1 with equal probability, the total number of "1" in m tests (considered as Bernoulli trials) will follow the Chebyshev's inequality

$$p\{|s-mp| > \varepsilon\} < \frac{var(s)}{\varepsilon^2} \quad (21)$$

where $var(s)$ is the variance of s, i.e. $var(s) = mp(1-p)$.

Let's set the deviation of s from mp as three or more standard deviations, i.e.

$$\varepsilon = 3\sqrt{mp(1-p)}. \qquad (22)$$

Thus, Chebyshev's inequality will be formed as:

$$P\left\{ |s - mp| > 3\sqrt{mp(1-p)} \right\} \le \frac{1}{9}. \qquad (23)$$

For example, for $m=100$, it gets:

$$P\left\{ |s - 50| < 15 \right\} > \frac{8}{9}. \qquad (24)$$

In other words, when the probability is greater than or is equal to 0.888, the total number of "1" in 100 tests performed by a permanently faulty unit will be in the range $35 < s < 65$.

As distinct from permanently faulty units, a unit possessing an intermittent fault of class C_1 doesn't have such consistent pattern in producing tests results as a permanently faulty unit can produce. This fact allows us to discriminate between permanent and intermittent faults. In the given case, achieving correct diagnosis is not very important. As a rule, incorrect diagnosis results in a permanently faulty unit is diagnosed as a unit with an intermittent fault whose behavior is very similar to a permanently faulty unit.

We assume that the probability that both testing and tested units possess an intermittent fault of class C_2 and both these intermittent faults are in AS during the same testing procedure is so small that it can be neglected. In the Table, this situation is depicted as "-".

According to the above considerations, the sought conditional probabilities $P(R|A)$, $P(R|P)$, $P(R|C_1)$ and $P(R|C_2)$ can be determined approximately. More accurately these conditional probabilities can be determined by taking into account the fact that probability distribution of random variable s is not discrete uniform distribution.

Then, probabilities $P(A|R)$, $P(P|R)$, $P(C_1|R)$ and $P(C_2|R)$ can be determined by using expressions (12) to (15).

Having computed conditional probabilities $P(A|R)$, $P(P|R)$, $P(C_1|R)$ and $P(C_2|R)$, we can choose two most probable hypotheses and then compute the likelihood ratio χ.

In case when $\chi < \omega$, and testing procedure can be continued, we increase either m, or k, or both. After performing additional testing, the value of χ is computed anew.

VII. DIAGNOSIS ALGORITHM

The diagnosis algorithm summarizes and integrates all the results of the above considerations. The initial data for the algorithm are as follows:

- diagnosis model (i.e., assumptions about allowable faulty model, about the probabilities presented in Fig.2, about allowable classes of intermittent faults, about homogeneity of units, etc.);

- unit statistical characteristics (e.g., those related to reliability and intermittent fault behavior);
- requirements to credibility of diagnosis result;
- time available for diagnosis.

Flowchart of diagnosis procedure is shown in Fig.10. The value of m (i.e., the number of rounds of testing) can be set either on the basis of available time or on the basis of the results of simulation. For the given unit's reliability and the given intermittent fault parameters, simulation allows to determine the value of m which ensures high credibility of diagnosis result.

When information about possible intermittent faults is absent, the worst case can be modeled. In the given case, the "worst" parameters of intermittent faults are used. Diagnosis of intermittent faults with such "worst" parameters requires high values of m. In this case, for calculating the value of χ these worst parameters of intermittent faults should be used.

When information about possible intermittent faults is present, the interested party can use the known parameters and can set the acceptable threshold ω by taking into account the possible risk.

They can also make decision on possible changes of diagnosis procedure (e.g., value of m) which will allow to achieve better diagnosis results.

VIII. COMPUTER SIMULATION

With the aim to verify the correctness of the diagnosis algorithm, the computer simulation was performed. The simulation involved the following tasks:

- simulation of changes of units' states;
- simulation of testing procedure for obtaining syndrome;
- processing the obtained syndrome (i.e., performing diagnosis algorithm).

The first two tasks were solved by using Petri Nets, and the third one was fulfilled in the web application [14]. The main parts of this web application are shown in Fig.11.

In Fig.11, *z-score* means standardized value indicating the number of standard deviations above or below the mean. For the case under consideration, the mean is equal to $m/2$.

Petri Nets have proved as very efficient tool for providing modeling and simulation of tests performed by one unit on other ones [39]. We are going to elucidate the simulation performed by using Petri Net (PN) with a simple example. Assume that unit u_0 tests two other units, u_1 and u_2, in round-robin manner. The Petri net that depicts the units' states, tests and test results is shown in Fig.12.

Modeling of units' states is depicted at the top of Fig.12. The figure presents the case when all units possess intermittent faults.

Unit with intermittent fault has two states, AS and PS, (Fig.9) which are modeled by places P_{AS} and P_{PS}, and by

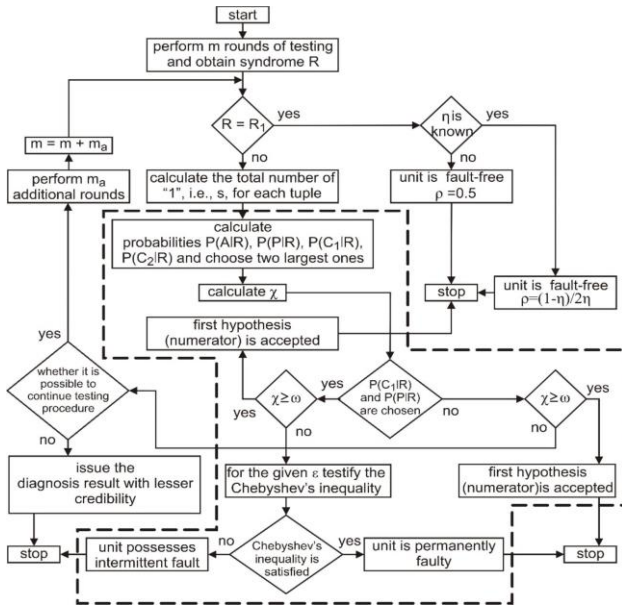two timed transitions, T_λ and T_μ.

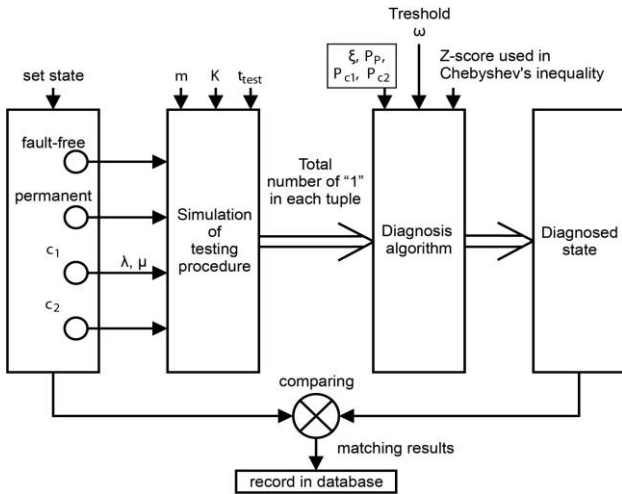Fig.10. Flowchart of diagnosis procedure.
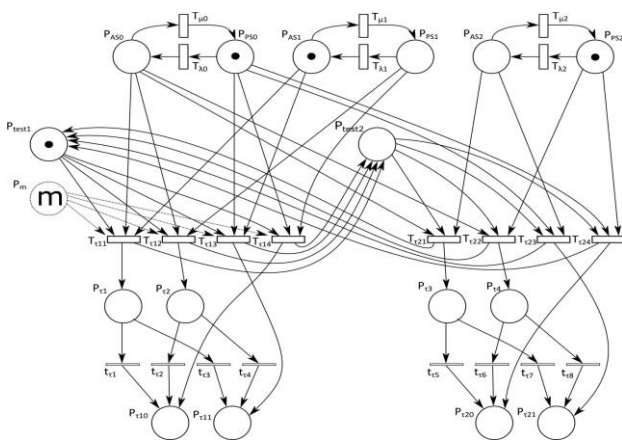

Fig.11. Main parts of application.

Fig.12. Petri Net used for simulation of testing procedure.

Duration of tests and the pairs of units which are involved in each test are modeled by timed transitions

$T_{\tau11}$, $T_{\tau12}$, $T_{\tau13}$, $T_{\tau14}$, $T_{\tau21}$, $T_{\tau22}$, $T_{\tau23}$, $T_{\tau24}$, and by places $P_{\tau1}$, $P_{\tau2}$, $P_{\tau3}$, $P_{\tau4}$ (in the middle of Fig. 12). And finally, the test results are modeled with the help of immediate transitions t_{r1}, t_{r2}, t_{r3}, t_{r4}, t_{r5}, t_{r6}, t_{r7}, t_{r8} and by places P_{r10}, P_{r11}, P_{r20}, P_{r21} (at the bottom of Fig. 12). PN has also additional places P_m, P_{test1} and P_{test2}. Place P_m is used for simulation of m rounds of testing. Places P_{test1} and P_{test2} are used to model the order of tests execution. In the given case, unit u_0 tests units u_1 and u_2 sequentially, not concurrently. The whole simulation procedure is presented on web site [40].

One of the main tasks of the simulation was to choose among the enabled transitions in each marking that transition actually fires. The choice is made on the basis of probability mass functions. Thus, we exploit the construct which is called a random switch [41].

At the beginning, the initial marking for the places that model unit's states is determined. This is done on the bases of the units' states which are set by researcher. When unit's state is set as having an intermittent fault, the probabilities of AS and PS for the first initial marking will be computed according to the following expressions:

$$P_{PS} = \mu/(\lambda + \mu) \text{ and } P_{AS} = \lambda/(\lambda + \mu).$$

Then, by using random switches, the resulting marking is computed. In the resulting marking, we are interested in the places that model the test results (for the above example these places are P_{r11} and P_{r21}).

The simulator clock is updated with the constant value equal to duration of test, t_{test}. The time needed for a testing unit to move the focus to the next unit for testing is very small and is neglected here. Every time after each clock update, new initial marking is determined only for the places modeling the unit states, and then the resulting marking is updated. As a result, the tokens are accumulated in the places which model test results.

For each updated initial marking, the probabilities P_{PS} and P_{AS} are determined as

$P_{AS} = 1 - e^{-\lambda t}$ if the previous clock update at the end, the intermittent fault was in P_S.

$P_{PS} = 1 - e^{-\mu t}$ if the previous clock update at the end, the intermittent fault was in AS, where $t = t_{test}$.

In the resulting marking, the places that model test results will contain the total number of tokens equal to the total number of "1" in the tuples, and, thus, give the input data for the diagnosis algorithm.

Hence, algorithm for simulation of testing procedure can be outlined as follows:

```
begin
    for i:=1 to (m × k) do
        begin
            compute initial marking;
            initial marking −→ resulting marking;
        end
    calculate the number of tokens in the places that
    model test results;
end
```

The main elements of diagnosis algorithm are shown in Fig.10 within the area bounded by dashed line. Diagnosis algorithm was written in PHP and is used by web application [40]. By exploiting this web application, a simulation of various faulty situations is performed to assess the quality of the developed diagnosis algorithm.

Fig.13 shows the probability of correct diagnosis, P_{det}.

During the simulation, the state of the testing unit was constantly set as permanent fault. For each execution of testing procedure, the state of each tested unit was arbitrarily set as either fault-free, or permanently faulty, or intermittently faulty (C_1 or C_2). Results of simulation have shown that reliability of the tested units has minor impact on diagnosis result. On the average, twelve rounds are sufficient to obtain P_{det} greater than 0.99 regardless of what faulty sets are allowed, even when all of the system units are faulty. The required number of rounds, m, reduces to 8 if the number of faulty units (with permanent or intermittent faults) is lesser than $(k + 1)/2$.
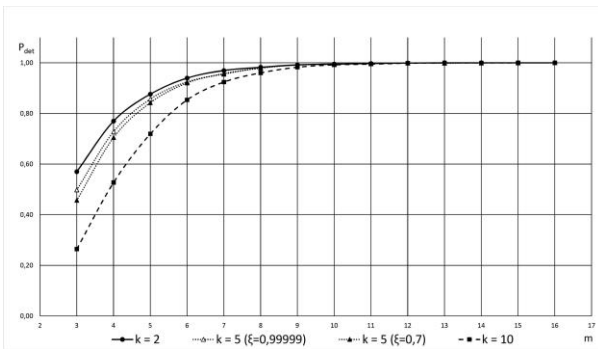

Fig.13. Probability of correct diagnosis of permanently faulty unit.
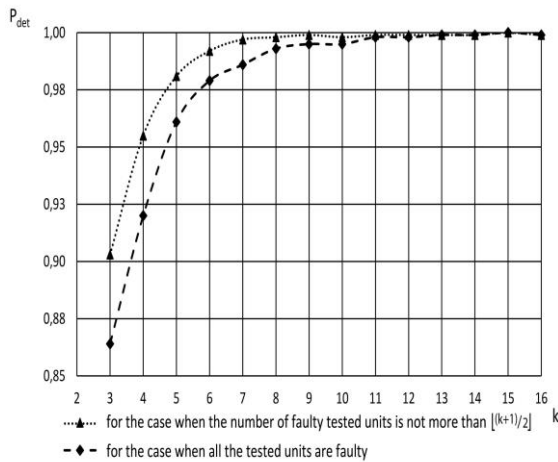

Fig.14. the probability of correct diagnosis of fault-free unit.

Fig.14 shows the probability of correct diagnosis. In the given case, incorrect diagnosis represents a Type I error (i.e., false positive). During the simulation, the state of the testing unit was constantly set as fault-free. The number of rounds and the reliability of tested units had a minor impact on diagnosis results. Fig. 14 depicts the case when $m = 2$ and $\xi = 0.9999$. If only faulty sets with not more than $(k + 1)/2$ arbitrarily faulty units are allowed, the probability P_{det} becomes greater than 0.99 when $k \geq 5$.

Fig.15 depicts possible intermittent faults expressed via parameters λ and μ. Each intermittent fault is presented as a point with coordinates (λ, μ). **A** denotes the subset of intermittent faults that do not belong to class C_1. **B** denotes the subset of intermittent faults that produce the effect similar to the effect produced by a permanent fault. We examined only intermittent faults that do not belong to subsets **A** and **B**. Basing on the simulation results, we have determined the subset of intermittent faults, denoted as **C**, in which every intermittent fault can be correctly diagnosed. Specifically, the probability of correct diagnosis of intermittent fault that belong to subset **C** is greater than 0.94 (for $k = 4$, $m = 16$ and $z\text{-}score = 0.5$).
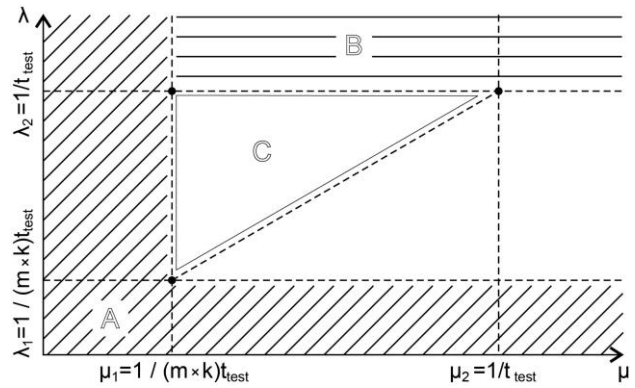

Fig.15. Subsets of intermittent faults.

The impact of $z\text{-}score$ on the probability of correct diagnosis was investigated for intermittent faults belonging to subset **C**, and is shown in Fig.16.

It should be noted that the proposed method allows increasing the probability of correct diagnosis if statistical characteristics of possible intermittent faults are taken into account.

IX. CONCLUSION

The paper presents a novel approach to system level self-diagnosis. Traditionally, system level self-diagnosis uses the results of tests performed on a unit for the diagnosis of its state. Unlike this traditional approach, we proposed diagnosis which uses a set of test results for diagnosis of a testing unit. This approach does not impose strict requirements on testing assignment. Diagnosis can be performed for different types of faults and different models of test result interpretation. It enables achieving the required credibility of diagnosis result by way of increasing the time of execution of testing procedure.

The distinctive feature of the proposed diagnosis consists in the fact that correct diagnosis can be obtained in case of multiple on related faults in the system and in case of heterogeneous systems. The main deficiency of the suggested diagnosis is somehow time-consuming, which may restrict its applicability. When time is crucial factor, the suggested diagnosis could be used as a supplementary facility for the traditional system level diagnosis, and could be performed in background mode (i.e., as a daemon).

Simulation results of the proposed unit's self-diagnosis give proof of feasibility of this new technique. Conventionally, for self-testing and self-diagnosis it is implicitly assumed that a unit includes some fault-free subsystem capable of executing the diagnosis algorithms correctly. This limitation can be overcome if a unit sends its test results to other system units (e.g., to its neighbors in multi-core arrays). In this case, the diagnosis algorithm will be executed by these neighbors.

Finally, the proposed diagnosis is intended to be applicable to complex systems such as, for example sensor networks, multi-robot system, many-core processors, multi-agent systems [42, 43], and possibly, in other fields.
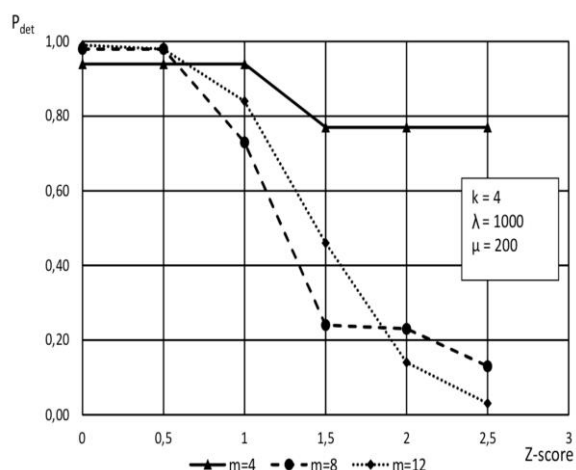

Fig.16. Impact of *z-score* on the probability of correct diagnosis.

REFERENCES

[1]   F. Preparata, G. Metze, R. Chien, "On the connection assignment problem of diagnosable systems", IEEE Transactions on Electronic Computers (EC-16), 6 (Dec.), pp. 848-854, 1967.

[2]   V. Mashkov, O. Barabash, "Self-checking of modular systems under random performance of elementary checks", Engineering Simulation, Vol.12, pp. 433-445, 1995.

[3]   V. Mashkov, O. Barabash, "Self-testing of multimodule systems based on optimal check-connection structures", Engineering Simulation, Vol.13, pp. 479-492, 1996.

[4]   V. Mashkov, "Selected problems of system level self-diagnosis", Lviv, Ukrainian Academic Press, 2011, ISBN 978-966-322-365-0.

[5]   M. Malek, "A comparison connection assignment for diagnosis of multiprocessor systems", in Proceedings of the 7th Annual Symposium on Computer Architecture, pp. 31-36, 1980.

[6]   M. Ding, D. Chen, K. Xing, X. Cheng, "Localized fault-tolerant event boundary detection in sensor networks", IEEE Infocom, pp. 902-913, 2005.

[7]   A. Russoniello, E. Gamess, "Evaluation of Different Routing Protocols for Mobile AdHoc Networks in Scenarios with High-Speed Mobility", International Journal of Computer Network and Information Security (IJCNIS), Vol.10, No.10, pp.46-52, 2018.

[8]   M. N. Riaz, "Clustering Algorithms of Wireless Sensor Networks: A Survey", International Journal of Wireless and Microwave Technologies (IJWMT), Vol.8, No.4, pp. 40-53, 2018.

[9]   N. V. Dinh, N. X. Thao, "Some Measures of Picture Fuzzy Sets and Their Application in Multi-attribute Decision Making", I.J. Mathematical Sciences and Computing (IJMSC), Vol.4, No.3, pp. 23-41, 2018.

[10]   N. Meskin, K.Khorasaniy, "Fault detection and isolation of discrete-time Markovian jump linear systems with application to a network of multiagent systems having imperfect communication channels", Automatica, 45(9), pp. 2032-2040, 2009.

[11]   R. Micalizio, P. Torasso, G. Torta, "On-line monitoring and diagnosis of multi-agent systems: A model based approach", 16th European Conference on Artificial Intelligence (ECAI), Valencia, Spain, Vol. 16, p. 848, 2004.

[12]   C. Wang, W. Shang, D. Sun, "Monitoring malfunction in multirobot formation with a neural network detector". Journal of Systems and Control Engineering, Vol. 225, pp. 1163-1172, 2011.

[13]   F. Barsi, F. Grandoni, P.Maestrini, "A theory of diagnosability of digital systems", IEEE Transactions on Computers, Vol. C-25, No.6, pp.585-593, 1976.

[14]   N. Meskin, K. Khorasani, C. A. Rabbath, "A hybrid fault detection and isolation strategy for a network of unmanned vehicles in presence of large environmental disturbances", IEEE Transactions on Control Systems Technology, 18(6), pp. 1422-1429, 2010.

[15]   G. Cueva-Fernandez, J. Pascual Espada, V. Garcia-Diay, R. Gonzalez-Crespo, "Fuzzy decision method to improve the information exchange in a vehicle sensor tracking system", Applied Soft Computing, Vol.35, pp. 708-716, 2015.

[16]   N. Lchevin, C. A. Rabbath, E. Earon, "Towards decentralized fault detection in uav formations", Control Conference, ACC07, New York City, USA, pp. 5759-5764, 2007.

[17]   A.Bobbio, "System modelling with Petri Nets", In: A.G. Colombo and A. Saiz de Bustamante (eds.). System Reliability Assessment, pp.102-143, 1990.

[18]   A. Apostolakis, D. Gizopoulos, M. Psarakis, A. M. Paschalis, "Software-Based Self-Testing of Symmetric Shared-Memory Multiprocessors", IEEE Transactions on Computers, Vol. 58, No. 12, pp.1682-1694, 2009.

[19]   V. Mashkov, "New approach to system level self-diagnosis", in Proceedings of IEEE 11th International Conference on Computer and Information Technology, CIT2011, Cyprus, pp.579-584, 2011.

[20]   M. Elhadef, A. Boukerche, H.Elkadiki, "Performance analysis of a distributed comparison-based self-diagnosis protocol for wireless ad hoc networks", in Proceedings of the 9th ACM International Symposium on Modeling Analysis and Simulation of Wireless and Mobile Systems, pp.165-172, 2006.

[21]   S. Jangale, D.Hadsul, "Detection of faulty sensor nodes in wireless sensor network", Computer technology and Applications, Vol.4, No.1, pp. 150-154, 2009.

[22]   M. H. Lee, Y. H. Choi, "Fault detection on wireless sensor networks", Computer Communications, Vol. 31, 2008.

[23]   L. Albini, J. Duarte, R. Ziwich, "A generalized model for distributed comparison-based system-level diagnosis", J. Brazi, Comput. Soc., Vol. 10, No. 3, pp. 44-56, 2005.

[24]   J. Chen, S. Kher, A.Somani, "Distributed fault detection of wireless sensor network", in Proceedings of the International Conference on Mobile Computing and Networking, New York, USA, pp. 65-72, 2006.

[25]   S. Chessa, P.Santi, "Comparison-based system-level fault

diagnosis in ad hoc network", in 20th Symp. Reliable Distributed Systems, pp. 257-266, 2001.

[26] J. Xu., "The t/(n-1) diagnosability and its application to fault tolerance", Technical report, No. 340, University of Newcastle upon Tyne, 1991.

[27] V. Mashkov, J. Pokorny, "Scheme for comparing results of diverse software versions", in Proc. of ICSOFT Conference, Barcelona, Spain, pp.341- 344, 2007.

[28] M. J. Daigle, X. D. Koutsoukos, G. Biswas, "Distributed diagnosis in formations of mobile robots", IEEE Transactions on Robotics, Vol.23, No.2, pp. 353-369, 2007.

[29] P. M. Khilar, "Performance analysis of distributed intermittent fault diagnosis in wireless networks using clustering", in Proceedings of 5th International Conference on Industrial and Information Systems, ICIIS, pp. 13-18, 2010.

[30] B. Krishnamachari, S.Iyengar, "Distributed Bayesian algorithms for fault-tolerant event region detection in wireless sensor networks", IEEE Transaction on Computers, Vol.53, No.3, pp. 241-250, 2004.

[31] X. Luo, M. Dong, Y. Huang, "On distributed fault-tolerant detection in wireless sensor networks", IEEE Transactions on Computers, Vol.55, No.1, pp. 58-70, 2006.

[32] A. Auddy, S. Mukhopadhyay, "Modelling Online Admission System: A MultiAgent Based Approach", International Journal of Modern Education and Computer Science (IJMECS), Vol.6, No.5, pp. 26-32, 2014.

[33] P. Jiang, "A new method for fault detection in wireless sensor networks", in Proceeding of ISSN 1424-8220, Hangzhou Dianzi Unversity, 2009.

[34] S. Mallela, G. Masson, "Diagnosable systems for intermittent faults", IEEE Transactions on Computers, Vol.C-27, 6 (June), pp. 560-566, 1978.

[35] L. Qin, X. He, D. H. Zhou, "A survey of fault diagnosis for swarm systems". Systems Science and Control Engineering: An Open Access Journal, Vol.2, pp. 13-23, 2014.

[36] M. Dhar, H.K. Baruah, "Theory of Fuzzy Sets: An Overview", I.J. Information Engineering and Electronic Business (IJIEEB), Vol.5, No.3, pp.22-33, 2013.

[37] S. Kamal, C. V. Page, "Intermittent faults: A model and a detection procedure", IEEE Transactions on Computers, Vol.23, Iss.7, pp. 713-719, 1974.

[38] J. Collet, P. Zajac, M. Psarakis, D. Gizopoulos, "Chip self-organization and fault-tolerance in massively defective multicore arrays", IEEE Transactions on Dependable and Secure Computing, Vol.8, No.2, pp.207-217, 2011.

[39] V. Mashkov, J. Barilla, P. Simr, "Applying Petri Nets to Modeling of Many-core Processor Self-testing when Tests are Performed Randomly", Journal of Electronic Testing, Vol.29, No.1, pp.25-34, 2013.

[40] PNsimulator. Available at http://vtan.ujep.cz/PNsimulator.

[41] A. Barua, K. Khorasani, "Intelligent model-based hierarchical fault diagnosis for satellite formations". IEEE international conference on Systems, Man and Cybernetics, ISIC, Montreal, Quebec, Canada, 2007, pp. 3191-3196.

[42] V. Mashkov, "Task allocation among agents of restricted alliance", in Proc. of IASTED ISC2005 Conference, Cambridge, MA, USA, pp.13-18, 2005.

[43] V. Mashkov, "Restricted alliance and coalition formation", Proc. of IEEE/WIC/ACM International Conference on Intelligent Agent Technology, China, pp. 329-332, 2004.

Authors' Profiles

Viktor Mashkov Doctor of Science in Engineering Docent Department of IT at the University of J.E. Purkyne in Usti nad Labem (Czech Republic). His major research focuses on dependability of computer systems, software fault tolerance, system level self-diagnosis and multi-agent systems

Volodymyr Lytvynenko: Kherson National Technical University. D.Sc. (Eng. hab.), Professor, Head of the Department Informatics & Computer Sciences. Research interests: data mining of complex data, dependability of computer systems, software fault tolerance, time series forecasting.