On Determining the State of The Processors of a Multiprocessor System During its Testing

Alexei M. Romankevich[a], Kostiantyn V. Morozov[a], Vitaliy A. Romankevich[a]

[a] *National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute", 37, Prosp. Peremohy, Kyiv, 03056, Ukraine*

Abstract

The work is devoted to assessing the completeness of the test for a given limit on the maximum number of faulty processors in the case of applying a formal method for determining the state of the processors of a multiprocessor system. This method is based on the results of performing mutual test checks, which consists in forming a specialized boolean equation based on the results of the tests, as well as its subsequent transformation. The above limitation is determined by the structure of the test or, in other words, the system's capabilities in terms of mutual testing of processors, which can be reflected in the form of a directed graph. A number of assertions are proved and criterions are formulated that are used to determine the maximum number of processors, the failure of which still guarantees the determination of the state of all processors in the system in the case of applying the method. Presented examples demonstrate the practical application of these criterions for estimating the maximum allowable number of faults. An important feature of the method is the possibility of applying it to systems corresponding to graphs of arbitrary topology, and not only to complete graphs.

Keywords

Multiprocessor systems, Mutual testing of processors, PMC-model, *M*-diagnosable systems

1. Introduction

Modern control systems (CS) for various objects are increasingly built on the basis of microprocessors. For the so-called systems of critical application, the failure of which control objects may lead to significant losses (control systems for transport, electricity generation, production facilities, military equipment, etc.), often put forward increased requirements for both reliability and performance [1-8]. This problem can be effectively solved by building a CS based on fault-tolerant multiprocessor systems (FTMS) [9, 10]. In such systems, the failure of some processors does not lead to the failure of the system as a whole. This effect is achieved, in particular, through the use of various types of redundancy, which makes it possible to reconfigure the system [11-15], and as a result, tasks are redistributed between serviceable processors.

For efficient system reconfiguration, it is important to have information about the current state (healthy/failed) of each of its processors, which can be obtained as a result of system testing [16, 17].

There are two main approaches to testing fault-tolerant multiprocessor systems: by using a specialized subsystem that performs testing, or by mutual testing of processors with each other.

The first approach is characterized by the relative simplicity of implementing the testing procedure (in particular, the procedure for determining the state of each of the system processors based on the test results is trivial), however, its disadvantage is the need to introduce an additional node into the system, which is also prone to failures (moreover, its failure leads to the impossibility of performing the testing

IntelITSIS'2022: 3rd International Workshop on Intelligent Information Technologies and Systems of Information Security, March 23–25, 2022, Khmelnytskyi, Ukraine

EMAIL: romankev@scs.kpi.ua (Alexei Romankevich); mcng@ukr.net (Kostiantyn Morozov); zavkaf@scs.kpi.ua (Vitaliy Romankevich)
ORCID: 0000-0001-5634-8469 (Alexei Romankevich); 0000-0003-0978-6292 (Kostiantyn Morozov); 0000-0003-4696-5935 (Vitaliy Romankevich)

process). This approach can be used, in particular, to implement system testing at the production and service stages.

The second approach is based on the performance of mutual test checks by the processors of the system and analysis of the test results. It should be noted that the organization of this approach is much more complicated, in particular, the task of determining the state of each of the processors of the system, depending on the results of test checks, becomes much more complicated [18]. This approach can be very effective for application during the operation phase of the system, and it is this approach that is considered in this article.

Processor testing can be carried out by executing a certain set of test tasks and comparing their results with reference values: if they match, the processor is considered to be serviceable, and otherwise, it is faulty. It should be noted that in the approach under consideration, two processors are involved in testing: the one that is testing and the one that is being tested, each of which, in turn, can be serviceable or faulty. It is assumed that the testing of the system is carried out in accordance with the Preparata-Metze-Chien (PMC-model) [19], in which the result of the test experiment will be 0 (no errors found) if the processor under test is serviceable or 1 (errors are found) if the tested processor is faulty. The above is true only if the testing processor is in good condition. Otherwise, the result can be either 0 or 1, regardless of the state of the processor under test.

To reflect the possibilities of conducting mutual test checks of the processors of the system, a digraph can be constructed, the vertices of which will correspond to the processors, and the presence of an arc from vertex *a* to vertex *b* is the possibility of carrying out a test check of the processor corresponding to vertex *b* by the processor corresponding to vertex *a*. This graph will actually reflect the capabilities of the system in terms of diagnosing. An example of such a graph is shown in Figure 1.

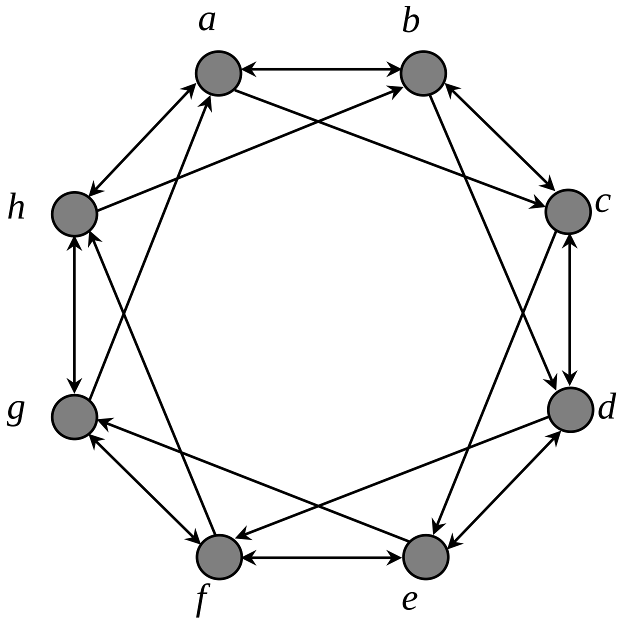

Figure 1: Graph reflecting the possibilities of mutual testing of processors

Special consideration may be granted for the case when the above graph is complete, i.e. it is possible to test each of the processors with any other processor. Such a case is considered, in particular, in [19], where it is shown that the state of the system can be uniquely determined if less than half of all processors in the system fail. It was shown in [20] that this is almost always possible even with a larger number of faults, and in [21] an algorithm was proposed that makes it possible to uniquely determine this state by conducting no more than $N + 2p$ test checks, where N is the number of system processors, and p is the number of truly faulty processors. However, in practice it is far from always possible to implement the possibility of testing processors on a one-to-one basis.

2. Formal method for determining the state of processors in a multiprocessor system

In [22], a formal method was proposed for determining the state of a multiprocessor system based on the results of the test checks, which allows an arbitrary topology of a graph that reflects the capabilities of the system in terms of diagnosing. Note that the topology of this graph affects the allowable number of processor failures, in which the state of the system can still be determined.

The concept of *M*-diagnosability is also used in the literature [23–25]. Thus, a system is *M*-diagnosable if it is possible to establish the state of all its processors in case of failure of at most *M* of them.

According to the method proposed in [22], each of the processors of the system is associated with some boolean variable x_i, which takes the value of 1 if it is in good condition and 0 if it is out of order. Any state of such a system corresponds to some constituent of one – an elementary conjunction containing all x_i, moreover, without inversions if they correspond to serviceable processors and with inversions – if they are faulty. Note that for the real state of the system and only for it, such a constituent is equal to one.

It is also assumed that the system is *M*-diagnosable, i.e. its state can be established only in case of a failure of at most *M* processors, and therefore such situations are considered.

According to [22], each of the results of test checks is associated with an expression constructed in a special way, containing the variables x_i. For the test, which consists in testing the processor corresponding to the variable x_j by the processor corresponding to the variable x_i, as a result of which the value r_{ij} was obtained, the expression will be built

$$R_{ij}(x_i, x_j) \triangleq \begin{cases} x_i x_j \vee \bar{x}_i \equiv x_j \vee \bar{x}_i, if\ r_{ij} = 0 \\ x_i \bar{x}_j \vee \bar{x}_i \equiv \bar{x}_j \vee \bar{x}_i, if\ r_{ij} = 1 \end{cases}. \quad (1)$$

With such a construction, the equality will be valid: $R_{ij}(x_i, x_j) = 1$.

Based on the results of *K* tests, an expression V_K can be constructed.

$$V_K \triangleq \bigwedge_{R \in S_K} R, \quad (2)$$

where S_K is a set of expressions constructed in accordance with (1) for each of the conducted test checks. It is also easy to show that based on this construction

$$V_K = 1. \quad (3)$$

To determine the state of the system, in accordance with [22], the expression V_K is transformed into a perfect disjunctive normal form (PDNF), i.e. disjunction of the constituents of one. In this case, all conjunctions containing more than *M* inversions are excluded at any stage of the transformation, which, as was shown in [22], does not lead to the exclusion of the conjunction corresponding to the real state of the system, if no more than *M* processors actually failed in it.

Thus, after performing the above transformations, V_K will be represented in the following form:

$$V_K = \bigvee_{l=1}^{L} C_l, \quad (4)$$

where each of C_l – some constituent of one (containing no more than *M* inversions), and *L* is their number. Taking into account (3) we get:

$$\bigvee_{l=1}^{L} C_l = 1. \quad (5)$$

All possible equalities of the form (5) can be divided into three cases: $L = 1$, $L > 1$, $L = 0$.

In the first case, the real state of the system corresponds to a single constituent and, as shown above, can be easily determined: the processors corresponding to the variables included in the constituent without inversions are considered to be serviceable, and those that enter it with inversions are considered to be faulty.

In the second case, equality contains more than one constituent, among which there is one corresponding to the real state of the system. Additional tests can be done to determine it. It is also worth noting that if some variable is included in all constituents without inversions, then the processor corresponding to it is unambiguously good, and if some variable is included in them with inversions, then the processor corresponding to it is unambiguously faulty. The state of all other processors (that is, those that correspond to variables included in some of the constituents with inversions, and in others without inversions) is not determined. Based on this information, the choice of subsequent test checks can be made. For example, it makes sense to choose such tests in which the testing processor is known to be good, but the state of the tested one is not defined. And the checks in which the testing processor is known to be faulty can be excluded. We note that all of the above is true only when no more than *M* processors actually failed in the system.

The third case, where the equality does not contain any constituents, is possible only if more than *M* processors are faulty in the system, i.e. conjunctions containing more than *M* inversions were illegitimately excluded. However, it is worth noting that the converse is generally not true, since in this case, depending on the results of the test checks, in principle any of the three situations is possible.

3. Test completeness assessment

As already mentioned, in order to apply the method considered above, it is necessary to set the maximum allowable number of faulty processors *M*. This value obviously depends on the organization of the system, namely, the possibilities of mutual testing of its processors, which, as shown above, can be reflected in the form of a directed graph. Thus, the problem of determining the value of M depending on the structure of such a graph becomes relevant.

Let the diagnostics of a system containing *N* processors occur in accordance with the PMC-model and correspond to graph *G*. Let, in case of failure of no more than *M* arbitrary processors, as a result of a certain set of tests, it is possible to unambiguously determine the state of each of the processors using the method described above.

Let us formulate and prove several assertions for such a system.

Assertion 1.

If for some group of *n* processors, it is possible to establish the state of each of them by conducting mutual test checks between the processors that are only in this group in all cases when no more than any *m* of them fail, then

$$m < \frac{n}{2} \tag{6}$$

Proof:

Let's say, $m \geq \frac{n}{2}$. We arbitrarily divide the considered set of processors into two subsets: *A* and *B*, with capacities respectively *m* and *l* = *n* – *m*. Since $m \geq \frac{n}{2}$, then $l = n - m \leq \frac{n}{2}$, and hence, *l* ≤ *m*.

Consider two possible cases.

Case 1:

All processors from set *A* are operational, and all processors from set *B* are faulty. However, faults in these processors are such that always lead to inversion of test results by these processors of other processors, i.e. if such a processor tests another faulty processor, the test result will be 0, and if it tests a working processor, then 1 (as before it is said, according to the PMC-model, in case of failure of the testing processor, the results of test checks can take arbitrary values). Thus, in this situation we will have *l* ≤ *m* faulty processors.

As a result of testing some processor *y* with another processor *x* we get the following results:

1. 0, if $x \in A$ and $y \in A$;
2. 1, if $x \in A$ and $y \in B$;
3. 1, if $x \in B$ and $y \in A$;
4. 0, if $x \in B$ and $y \in B$.

Case 2:

The situation is the opposite: all processors with *A* are faulty (and, as in the previous case, the test results are inverted), and all processors with *B* are operational. In this case we will have *m* faulty processors.

As a result of testing a processor *y* with a processor *x* we get the following results:

1. 0, if $x \in A$ and $y \in A$;

2. 1, if $x \in A$ and $y \in B$;
3. 1, if $x \in B$ and $y \in A$;
4. 0, if $x \in B$ and $y \in B$.

As you can see, in both cases the results of all possible tests coincide. At the same time, obviously, the states of the processors in each of these cases are different. That is, there is ambiguity, as a result of which it is not possible to determine the reliable state of the system based on test results.

Thus, when $m \geq \frac{n}{2}$, it is not always possible to unambiguously determine the state of the system.■

Note that for the case of a set that includes all processors of the system, *Assertion 1* will take the form:

$$M < \frac{N}{2}, \tag{7}$$

which corresponds to the results obtained in [19].

Assertion 2.

According to the proposed method, the reliable state of the system in case of failure of no more than *M* of its arbitrary processors with a given set of tests corresponding to diagnostic graph *G*, can be established if and only if for any set *F* of vertices of graph *G* with power *K*, $T > M - \left\lceil \frac{K}{2} \right\rceil$ vertices of the graph G exist, which are not contained in *F*, such that from each of them there is an arc in at least one of the vertices from the set *F*.

Proof:

Consider some set *F* of vertices of the graph *G* with power *K*. It corresponds to the set of processors F_p.

Let the set *E* contain all vertices of the graph *G* that are not contained in *F*, and from each of which there is an arc in at least one of the vertices of the set *F*. *T* is the power of the set *E*. This set corresponds to the set of processors E_p.

Prove the necessity:

Assume that $T \leq M - \left\lceil \frac{K}{2} \right\rceil$. Let all processors from E_p (*T* pieces) in the system fail, as well as $\left\lceil \frac{K}{2} \right\rceil$ processors from F_p. That is, the number of faulty processors in the system: $T + \left\lceil \frac{K}{2} \right\rceil \leq M$.

Since $\left\lceil \frac{K}{2} \right\rceil \geq \frac{K}{2}$, then, based on *Assumption 1*, testing processors from F_p only with processors from F_p (i.e. only on their own) will not allow in all cases to unambiguously establish the state of all of them.

Other test checks on processors from F_p can only be performed by processors from E_p (because only vertices on set *F* have arcs on vertices on set *E*), but all of these processors are faulty, i.e. test results can be any, regardless of the state of the tested processors and therefore they will not help to establish the state of processors from F_p.

So if $T \leq M - \left\lceil \frac{K}{2} \right\rceil$, it is not always possible to unambiguously establish the state of all processors in the system.

Prove sufficiency.

Suppose that after performing a set of test checks, according to the method described above, the following equation is obtained:

$$C_1 \vee C_2 \vee \ldots \vee C_S = 1. \tag{8}$$

We assume that when applying the method, the value of *M* was used, which meets the criteria of the assertion that is being proved. Thus, the number of inversion variables (corresponding to faulty processors) in each of the C_i conjunctions in expression (7) cannot exceed *M* (since such conjunctions are excluded from the equation).

Let C_r correspond to the real state of the system's processors. We prove that for each C_i, $i \neq r$, there will be a test that will exclude it.

The expressions C_i and C_r can be represented as follows:

$$C_r = A_i \wedge B_i, \tag{9}$$

$$C_i = A_i \wedge \tilde{B}_i, \tag{10}$$

where A_i – conjunction of all variables that coincide in C_i and C_r (i.e. in both cases are inverted or in both cases are inverted), and B_i and $\tilde{B}_i$ – conjunctions of all variables that differ in them, and it is obvious that all variables in $\tilde{B}_i$ will be inverted relative to the corresponding variables in B_i.

Let $A_i = A_i^w \wedge A_i^f$, where A_i^w contains only variables without inversions, and A_i^f – only variables with inversions. In addition, the number of variables in A_i^f is denoted as L.

Let $B_i = B_i^w \wedge B_i^f$, where B_i^w contains only variables without inversions, and B_i^f – only variables with inversions. In addition, the number of variables in B_i^w and B_i^f is denoted by P and Q, respectively.

Let also $\tilde{B}_i = \tilde{B}_i^w \wedge \tilde{B}_i^f$. Moreover, $\tilde{B}_i^w$ contains only variables without inversions. These are all variables that are part of B_i^f and therefore their number is Q. At the same time, $\tilde{B}_i^f$ contains only variables with inversions, and these are all variables that are part of B_i^w, so their number is P.

Let F_p – the set of all processors to which the variables from B_i and $\tilde{B}_i$ correspond. Its power: $K = Q + P$. This set corresponds to some set F of vertices of the graph G. Let E be the set of vertices of the graph not contained in F and such that from each of them there is an arc in at least one of the vertices of the set F. The power of this set is denoted by T. The vertices will correspond to the set from E_p. Obviously, the sets E_p and F_p do not intersect, and therefore the processors from the set E_p correspond to variables only from A_i.

Note that L, Q and P – the number of inversions in A_i, B_i and $\tilde{B}_i$ respectively, and hence the number of inversions in C_r: $L + Q$, and in C_i: $L + P$. Recall that the number of inversions in C_r, as in C_i can not exceed M, so that $L + Q \leq M$ and $L + P \leq M$, and therefore, $Q \leq M - L$ and $P \leq M - L$. Then: $K = Q + P \leq 2 \cdot (M - L)$, that is $2L \leq 2M - K$, respectively $L \leq M - \frac{K}{2}$, and given that L and M are integers, the following is true: $L \leq M - \left\lceil \frac{K}{2} \right\rceil$.

If $T > M - \left\lceil \frac{K}{2} \right\rceil \geq L$, then there is at least one processor from E_p, that does not correspond to any of the variables from A_i^f, and therefore it corresponds to the variable from A_i^w, i.e. one that is inverted in both C_r and C_i. Let this processor, which we will call j-th, correspond to the vertex $v_j \in E$ of the graph G, from which there is an arc to some vertex $v_k \in F$, which corresponds to the processor, which we will call k-th. Thus, it is possible to test the k-th processor with the j-th processor, resulting in the expression: $R = \tilde{x}_k \vee \bar{x}_j$, where x_k is inverted or non-inverted depending on the test result. Because, as already mentioned, x_i is non-inverted in both C_r and C_i, then $R \wedge C_r \equiv \tilde{x}_k \wedge C_r$, and $R \wedge C_i \equiv \tilde{x}_k \wedge C_i$. On the other hand, since the k-th processor corresponds to the vertex from the set F, then x_k is included in B_i (with or without inversion) and, accordingly, in $\tilde{B}_i$ (but in the opposite state). Thus, according to the test results, one of the conjunctions will be excluded. Obviously, this will be C_i, because C_r corresponds to the real state of the system and as shown in [22] can not be excluded).

Note that in the same way we can prove the existence of a test that excludes C_i for any $i \neq r$. Therefore, all C_i except C_r will be excluded and, thus, the real state of the system will be established.

In particular, for the case of a set F with power $K = 1$, i.e. for a separate vertex of the graph G the condition of *Assertion 2* can be reformulated as follows: the number of arcs T, entering each of the vertices must be at least M, i.e., the following condition must be satisfied: $T \geq M$.

Assertion 3.

Satisfaction of the condition of *Assertion 1* for the set containing all processors of the system, i.e. $M < \frac{N}{2}$ it is sufficient to fulfill the condition of *Assertion 2* for the set F containing all the vertices of the graph G.

Proof.

Obviously, if the set F contains all the vertices of the graph G, its power is equal to the number of processors in the system, i.e. $K = N$. On the other hand, there are no other vertices in the graph in principle, therefore, $T = 0$. Thus, the condition of *Assertion 2* takes the form $0 > M - \left\lceil \frac{N}{2} \right\rceil$. Taking into account that $M < \frac{N}{2}$, we have $M - \left\lceil \frac{N}{2} \right\rceil < \frac{N}{2} - \left\lceil \frac{N}{2} \right\rceil \leq 0$, i.e. the condition is met.

Assertion 4.

The condition of *Assertion 2* will always be satisfied for sets of vertices with power $K > 2M$.

Proof.

It is easy to see that for $K > 2M$, we have $\left\lceil \frac{K}{2} \right\rceil > \left\lceil \frac{2M}{2} \right\rceil = M$. Therefore, $M - \left\lceil \frac{K}{2} \right\rceil < 0$ and the condition of *Assertion 2* will be satisfied for any $T \geq 0$.

Thus, the choice of the value M for application in the method [22] can be carried out by checking all possible sets of vertices of the graph G for compliance with the conditions of *Assertion 2*. In addition, according to the statements proved above, somewhat simpler criteria for choosing this value can be formulated.

1. The value of M must be less than half the number of processors in the system.
2. In the graph G, the number of arcs entering each of the vertices must be at least M.
3. For each set F of vertices of the graph G with power K ($2 \leq K \leq 2M$), there must be $T > M - \left\lceil \frac{K}{2} \right\rceil$ vertices not contained in F and from which arcs that enter some vertices from the set F emanate.

4. Examples

Let's look at a few examples demonstrating the application of the above *criteria.*

Example 1.

Let us consider a system consisting of 5 processors corresponding to the graph G shown in Figure 2. From each vertex of the graph, arcs emanate to two neighboring vertices. Let us show that the state of all processors of such a system can be established based on the results of mutual test checks of processors in the event of failure of no more than 2 arbitrary processors, i.e. $M = 2$.

It should be noted that the graph shown in Figure 2 is symmetrical, which makes it possible to significantly simplify the analysis by reducing the enumeration of all possible groups of vertices.

Let us check the fulfillment of the *criterions* formulated above. The value $M = 2$ is indeed less than half the number of processors in the system, which is consistent with *criterion 1*. For any of the vertices of the graph, there are two vertices from which arcs emanate to the given one, which corresponds to *criterion 2* for $M = 2$.

Consider different mutual arrangements of pairs of vertices (Figure 3 a, b) up to rotation, i.e. $K = 2$. As you can see, in these situations there are 2 or 3 vertices from which arcs emanate in them, i.e. $T = 2$ or $T = 3$. In both cases, the condition of *criterion 3* for $K = 2$ and $M = 2$ is satisfied.

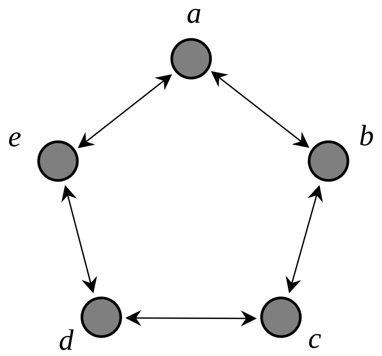

Figure 2: The structure of the graph G for the system considered in *Example 1*

Next, consider the various relative positions of triples of vertices (Figure 3 c, d), i.e. $K = 3$. As we see, in both situations there are 2 vertices from which the arcs in them emanate, i.e. $T = 2$ and the condition of *criterion 3* for $K = 3$ and $M = 2$ is satisfied.

For groups of four vertices ($K = 4$), there is only one possible mutual arrangement of vertices (Figure 3 e), and only one vertex from which arcs emanate to some of them ($T = 1$). In this case, the condition of *criterion 3* for $K = 4$ and $M = 2$ is fulfilled.

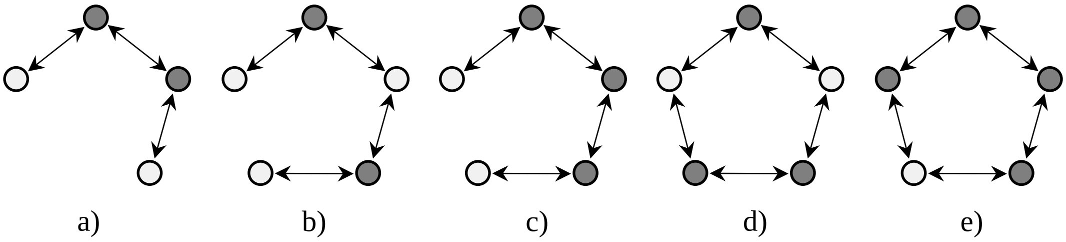

Figure 3: Various arrangements of vertex groups (highlighted in dark color), as well as vertices from which the arcs that enter them come from for *Example 1*

As we can see, the conditions of all criteria are met, therefore, for this system, the use of the value $M = 2$ in the method described in [22] is admissible. Note that this result corresponds to the result obtained in [19] for a system of 5 processors, however, it does not require the graph G to be complete.

Example 2.

Let the system consist of processors organized in the form of a matrix $A \times B$, $A \geq 3$, $B \geq 3$. We denote the processors of this system as $x_{i,j}$, where i – the row number ($i = 1, 2, \ldots, A$), j – the column number ($j = 1, 2, \ldots, B$). In this case, the processor $x_{i,j}$ can participate in testing its neighboring processors, i.e. processors $x_{(i+1) \bmod A+1,j}$, $x_{(i-1) \bmod A+1,j}$, $x_{i,(j+1) \bmod B+1}$, $x_{i,(j-1) \bmod B+1}$ (Figure 4). Let us prove that the state of all processors of such a system can be determined from the results of mutual test checks of processors in the event of failure of no more than 4 arbitrary processors.

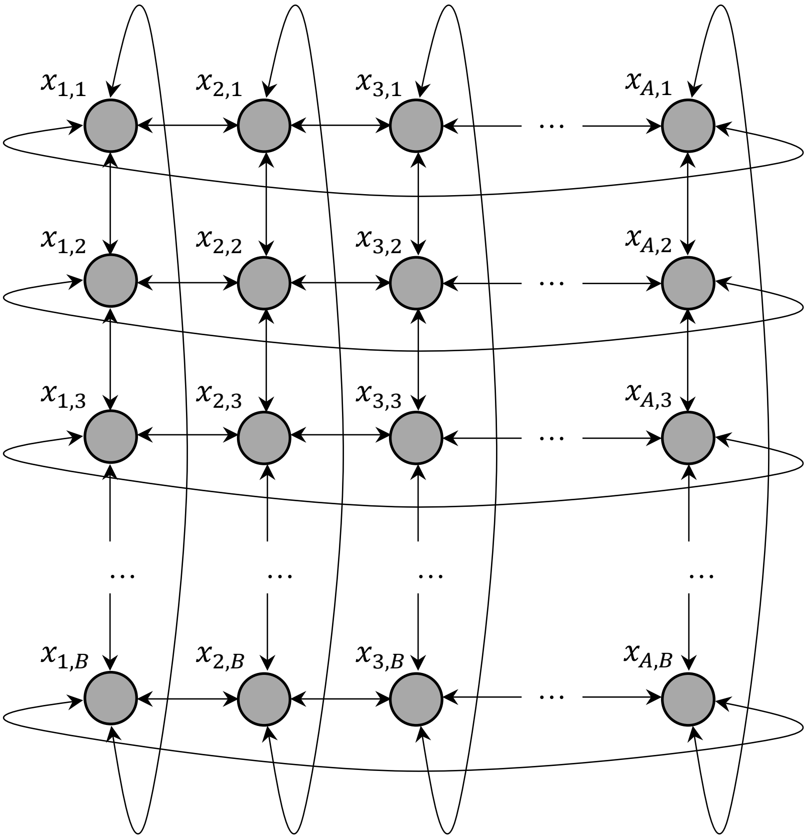

Figure 4: The structure of the graph G for the system considered in *Example 2*

Let us show that for any set of vertices of the graph G, the *criteria* formulated above for $M = 4$ will be satisfied.

Recall that, according to the condition $A \geq 3$ and $B \geq 3$. Therefore, the number of processors $N = A \cdot B \geq 9$. Thus, the value $M = 4$ is indeed less than half the number of processors in the system, i.e., the condition of *criterion 1* is met.

It is also easy to see that for any vertex of the graph, there are exactly 4 incoming arcs, which corresponds to *criterion 2* for $M = 4$.

Next, we consider various options for the mutual arrangement of pairs of vertices of the graph G, as well as the vertices from which the arcs that enter them proceed (Figure 5 a-d). As we see, the least number of such vertices is $T = 6$, which satisfies the condition of *criterion 3* for $K = 2$ and $M = 4$.

A similar operation can be performed for various triples of vertices ($K = 3$). As a result, for this situation, the minimum value $T = 7$ will be found (Figure 5 e), which, as it is easy to see, also corresponds to the condition of *criterion 3* for $K = 3$ and $M = 4$. Further, the same can be done for all possible groups of vertices with power 4, 5, ... 8, which is omitted in order to reduce the volume of the article.

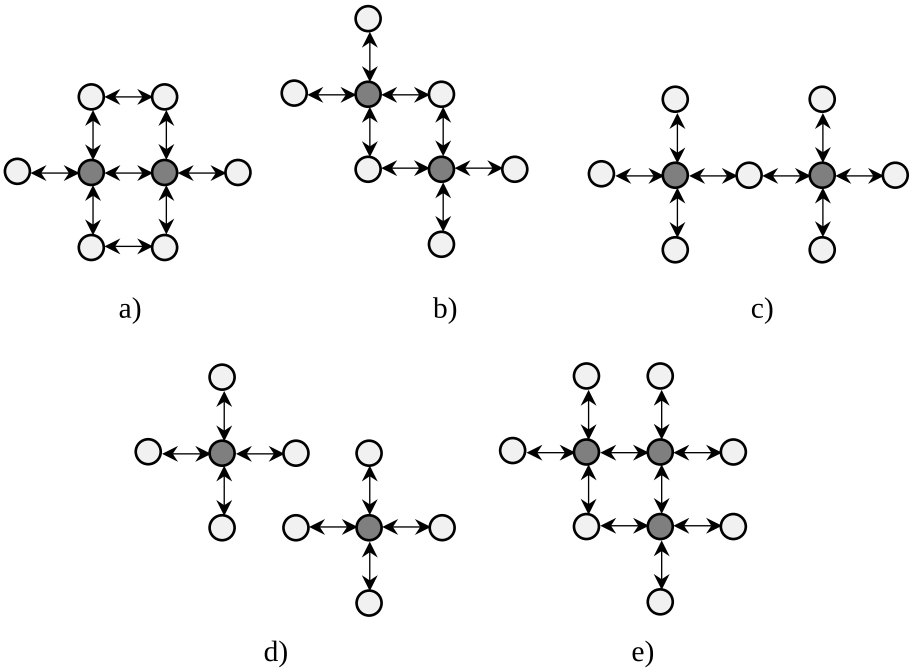

Figure 5: Various arrangements of vertex groups (highlighted in dark color), as well as vertices from which the arcs that enter them come from for *Example 2*

Example 3.

Consider a system consisting of 5 processors corresponding to the graph G shown in Figure 6. In this case, unlike the previous examples, the graph is not symmetrical. Let us show that, as in *Example 1*, the state of all processors of such a system can be established based on the results of mutual test checks of processors in the event of failure of no more than 2 arbitrary processors, i.e. $M = 2$. To do this, we check the fulfillment of the *criteria* formulated above.

The condition of *criterion 1* is satisfied, because the value $M = 2$ is indeed less than half the number of processors in the system. Let us check the fulfillment of the conditions of *criterion 2*. Table 1 shows the number of arcs entering each of the vertices of the graph G, as well as the set E of vertices from which these arcs originate. As we can see, for each of the vertices there are exactly two incoming vertices, which corresponds to the condition of *criterion 2* for $M = 2$.

To check the fulfillment of the conditions of *criterion 3*, the condition of *Assertion 2* must be checked for each possible set of vertices of the graph G, which contains from 2 to 4 vertices. Due to the lack of symmetry in the graph G, in contrast to the previous examples, one has to perform a direct enumeration of all such sets.

Let us construct Table 2, in which for each set F of vertices of the graph G with power $K = |F|$, where $2 \leq K \leq 4$, there is a set E of vertices that are not included in F and from which arcs emanate to any of the vertices of the set F, as well as the power of this set $T = |E|$. As we see, for $K = 2$ two cases are possible $T = 2$ or $T = 3$, for $K = 3$ in all cases $T = 2$, and for $K = 4$ always $T = 1$. It is easy to see that all these situations satisfy the condition *criterion 3* for $M = 2$, i.e. $T > M - \left\lceil \frac{K}{2} \right\rceil$.

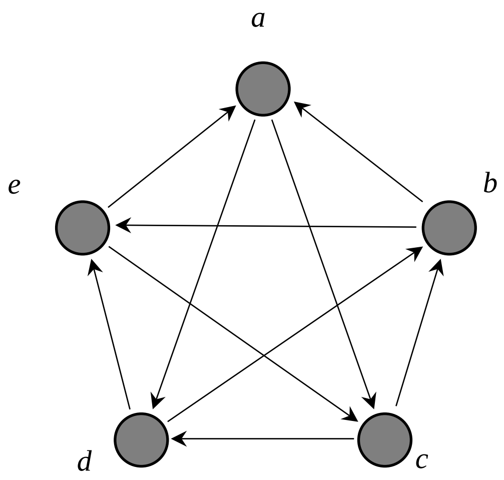

Figure 6: The structure of the graph *G* for the system considered in *Example 3*

Table 1
The number of incoming arcs for each of the vertices of the graph, as well as the set *E* of vertices from which these arcs originate

Vertex	Set of vertices *E*	Number of incoming arcs
a	{*b*, *e*}	2
b	{*c*, *d*}	2
c	{*a*, *e*}	2
d	{*a*, *c*}	2
e	{*b*, *d*}	2

Thus, the system under consideration satisfies all three *criterions* for $M = 2$; therefore, the state of all processors of such a system can indeed be established from the results of mutual test checks of processors in the event that no more than 2 of its arbitrary processors fail.

5. Conclusions

When developing a self-testing multiprocessor system, a separate task is the development of a test, i.e. set of mutual checks of processors in order to determine their state. This task can be quite difficult. At the same time, one of the important requirements for such a test is its completeness, i.e. guarantee of determining the state of the processors. In this case, the presence of certain restrictions is usually assumed, among which, in particular, sets of acceptable processor faults can be distinguished. In practice, instead of such sets, it is more convenient to use a simpler restriction, namely, the maximum allowable number of processors, upon failure of which the completeness of the test is preserved.

For an already developed test, it is important to be able to check its completeness for given constraints. This test actually reflects the capabilities of the system in terms of self-diagnosis, which can be described using a directed graph. In the article, a number of statements were proved, on the basis of which the criteria were formulated, which allow determining the maximum number of faults that allow the effective use of the method described in [22]. An important feature of the method under consideration is the possibility of applying it to systems corresponding to graphs of arbitrary topology, and not only to complete graphs.

Table 2
Sets F of vertices of the graph G, as well as sets E of vertices not included in F and from which arcs emanate to any of the vertices of the set F

Set of vertices F	$K = \|F\|$	Set of vertices E	$T = \|E\|$
$\{a, b\}$	2	$\{c, d, e\}$	3
$\{a, c\}$	2	$\{b, e\}$	2
$\{a, d\}$	2	$\{b, c, e\}$	3
$\{a, e\}$	2	$\{b, d\}$	2
$\{b, c\}$	2	$\{a, d, e\}$	3
$\{b, d\}$	2	$\{a, c\}$	2
$\{b, e\}$	2	$\{c, d\}$	2
$\{c, d\}$	2	$\{a, e\}$	2
$\{c, e\}$	2	$\{a, b, d\}$	3
$\{d, e\}$	2	$\{a, b, c\}$	3
$\{a, b, c\}$	3	$\{d, e\}$	2
$\{a, b, d\}$	3	$\{c, e\}$	2
$\{a, b, e\}$	3	$\{c, d\}$	2
$\{a, c, d\}$	3	$\{b, e\}$	2
$\{a, c, e\}$	3	$\{b, d\}$	2
$\{a, d, e\}$	3	$\{b, c\}$	2
$\{b, c, d\}$	3	$\{a, e\}$	2
$\{b, c, e\}$	3	$\{a, d\}$	2
$\{b, d, e\}$	3	$\{a, c\}$	2
$\{c, d, e\}$	3	$\{a, b\}$	2
$\{a, b, c, d\}$	4	$\{e\}$	1
$\{a, b, c, e\}$	4	$\{d\}$	1
$\{a, b, d, e\}$	4	$\{c\}$	1
$\{a, c, d, e\}$	4	$\{b\}$	1
$\{b, c, d, e\}$	4	$\{a\}$	1

In the general case, the verification of criterions conditions is associated with the enumeration and evaluation of all possible subsets of graph vertices, each of which corresponds to one of the system processors. However, in the case of symmetric and homogeneous structures, enumeration can be significantly reduced, as shown in the examples given in the article.

6. References

[1] A.Drozd, J.Drozd, S.Antoshchuk, V.Antonyuk, K.Zashcholkin, M.Drozd, O.Titomir. Green Experiments with FPGA, in book: Green IT Engineering: Components, Networks and Systems Implementation, V. Kharchenko, Y. Kondratenko, J. Kacprzyk, Eds., Vol. 105. Berlin, Heidelberg: Springer International Publishing, 2017, pp. 219–239.

[2] J.Avizˇienis, B.Laprie, Dependability and its threats: a taxonomy, Building the Information Society (2004) 91–120.

[3] Ushakov, I. (ed.): Reliability of Technical Systems: Handbook. Radio i Sviaz, Moskov (1985) (in Russian).

[4] W.Kuo, M. J.Zuo: Optimal reliability modeling: principles and applications. John Wiley & Sons Inc., New Jersey, USA (2003).

[5] Kuldeep Singh Kaswan, Sunita Choudhary, Kapil Sharma. Software Reliability Modeling using Soft Computing Techniques: Critical Review. IJITCS, vol.7, no.7, pp.90-101, 2015

[6] Ritika Wason, A. K. Soni, M. Qasim Rafiq. Estimating Software Reliability by Monitoring Software Execution through OpCode. IJITCS, vol.7, no.9, pp.23-30, 2015.

[7] Michael Onuoha Thomas, Babak Bashari Rad, Reliability Evaluation Metrics for Internet of Things, Car Tracking System: A Review, International Journal of Information Technology and Computer Science(IJITCS), Vol.9, No.2, pp.1-10, 2017.

[8] A. Keshtgar Saeid, B. Arasteh Bahman. Enhancing Software Reliability against Soft-Error using Minimum Redundancy on Critical Data. International Journal of Computer Network and Information Security(IJCNIS), Vol.9, No.5, pp.21-30, 2017.

[9] A.Avizienis. Fault-tolerance: The Survival Attribute of Digital Systems. Proc. IEEE, 1978, vol. 66, no. 10, pp. 1109-1126.

[10] Danial Rahdari, Amir Masoud Rahmani, Niusha Aboutaleby, Ali Sheidaei Karambasti. A Distributed Fault Tolerance Global Coordinator Election Algorithm in Unreliable High Traffic Distributed Systems. IJITCS, vol.7, no.3, pp.1-11, 2015. DOI: 10.5815/ijitcs.2015.03.01.

[11] E.Nazemi, T.Talebi, H.Elyasi. Self-Healing Mechanism for Reliable Architecture with Focus on Failure Detection [J]. I.J. Information Engineering and Electronic Business, 2015, 3, 32-38

[12] Sinha B., Singh A.K., Saini P. A Failure Detector for Crash Recovery Systems in Cloud [J]. I.J. Information Technology and Computer Science, 2019, 7, 9-16

[13] V.A.Vedeshenkov Organization of diagnostics of digital systems with the structure of a symmetric bipartite graph[J]. Control Science, 2009, 6: 59 67.

[14] A.Drozd, S.Antoshchuk, J.Drozd, K.Zashcholkin, M.Drozd, N.Kuznietsov, M.Al Dhabi, V.Nikul. Checkable FPGA Design: Energy Consumption, Throughput and Trustworthiness, in book: Green IT Engineering: Social, Business and Industrial Applications, Studies in Systems, Decision and Control, V. Kharchenko, Y. Kondratenko, J. Kacprzyk (Edits), Vol. 171. Berlin, Heidelberg: Springer International Publishing, pp. 73 – 94,

[15] M.F.Karavaj, V.S. Podlazov. Extended generalized hypercube as a fault-tolerant system network for multiprocessor systems [J]. LSS, 2013, 45: 344 371.

[16] M.A.Mikeladze. Development of Basic Self-Diagnosis Models for Complex Engineering Systems, Autom. Remote Control, 1995, vol. 56, no. 5, pp. 611–623.

[17] V.Y.Grishin, A.V.Lobanov, V.G.Sirenko. Distributed System Diagnosis of Byzantine Failures in Partially Connected Multicomputer Systems, Autom. Remote Control, 2005, vol. 66, no. 2, pp. 304–312.

[18] P.P.Parkhomenko. Checking Multiprocessor Computer Systems for Serviceability by Analyzing Their Syndrome Graphs, Avtom. Telemekh., 1999, no. 5, pp. 126-135.

[19] F.P.Preparata, G.Metze, R.T.Chien. On the Connection Assignment Problem of Diagnosable Systems[J]. IEEE Trans. Electron. Comput, 1967, ES-16(6): 848 854.

[20] A.M.Romankevich, V.A.Romankevich. Diagnosis of Multiprocessor Systems Under Failure of More Than Half Processors[J]. Autom. Remote Control, 2017, 78(9): 1614 1618.

[21] V.E.Belyavskii, V.N.Valuiskii, A.M.Romankevich, V.A.Romankevich. Self-Diagnosable Multimodular Systems: Some Estimates of Testing[J]. Autom. Remote Control, 1999, 60(8): 1179 1183.

[22] A. M.Romankevich, K. V.Morozov, V. A.Romankevich. A formal method for determining the state of processors in a multiprocessor system under testing[J]. Autom. Remote Control, 2021, 82(3): 460 467.

[23] S.L.Hakimi, A.T.Amin. Characterization of Connection Assignment of Diagnosable Systems[J]. IEEE Trans. Comput, 1974, C-23(1): 86 88.

[24] Y.K.Dimitriev On t-diagnosability of Multicore Systems with Symmetric Circulant Structure // Autom. Remote Control. 2013. V. 74. No. 1. P. 105–112.

[25] Y.K.Dimitriev. Necessary and sufficient conditions for t-diagnosability of multiprocessor computer systems for various models of nonreliable testing established using the system graph-theoretical model. Autom. Remote Control, 2015, vol. 76, no. 7, pp. 1260–1270.