DOI: https://doi.org/10.15276/aait.08.2025.30
UDC 004.05

A method for constructing GL-models of behavior under failure flow for complex non-basic fault-tolerant multiprocessor systems

Vitaliy A. Romankevich[1)]
ORCID: https://orcid.org/0000-0003-4696-5935; zavkaf@scs.kpi.ua. Scopus Author ID: 57193263058
Kostiantyn V. Morozov[1)]
ORCID: https://orcid.org/0000-0003-0978-6292; mcng@ukr.net. Scopus Author ID: 57222509251
Daniil V. Halytsky[1)]
ORCID: https://orcid.org/0009-0004-4421-3443; zipper135401@gmail.com. Scopus Author ID: 58553487600
Ihor A. Yermolenko[1)]
ORCID: https://orcid.org/0009-0008-5298-4888; yermolenkomail@gmail.com
Lefteris Zacharioudakis[2)]
ORCID: https://orcid.org/0000-0002-9658-3073; l.zacharioudakis@nup.ac.cy. Scopus Author ID: 57422876200
[1)] National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute", 37, Peremogy Ave. Kyiv, 03056, Ukraine
[2)] Neapolis University Pafos, 2, Danais Ave. Pafos, 8042, Cyprus

ABSTRACT

The paper proposes a method for constructing GL-models of the behavior of complex non-basic fault-tolerant multiprocessor systems under a failure flow. The aim of the study is to develop a universal approach that enables the formation of an integrated GL-model for systems characterized by multiple independent or weakly coupled operability conditions. Such models can be used, in particular, to evaluate the reliability parameters of the systems under consideration using statistical simulation methods. The study focuses on systems whose operability is determined by the simultaneous fulfillment of several relatively simple conditions, for each of which established methods for constructing GL-models are available (for example, a condition limiting the number of failures within a certain subset of processors). These include, in particular, hierarchical systems composed of multiple subsystems with their own levels of fault tolerance, as well as systems containing specialized processors of different types. The proposed method involves the preliminary construction of auxiliary GL-models for each operability condition; followed by their integration into a unified model through the sequential merging of their graphs via selected vertices (the merged vertices form a single vertex, while the remaining vertices and edges are copied). The order of model merging and the choice of corresponding vertices can be defined arbitrarily, providing flexibility in the structure of the resulting GL-model. Examples of the method's application are presented, illustrating various options for determining the sequence of merging auxiliary models and selecting the connecting vertices of their graphs, as well as the use of different methods for constructing these models. The scientific novelty of the work lies in the generalization and formalization of the sequential GL-model merging procedure, which makes it possible to combine models of arbitrary structure and type into a unified model of a complex system without compromising the correctness of its behavior. Experimental results confirm that, despite structural differences in the graphs of the obtained models, their behavior on identical input vectors coincides completely and accurately reflects the operation of the fault-tolerant multiprocessor system under a failure flow. It is also shown that the method imposes no restrictions on the construction techniques of GL-models for individual conditions: models of different types can be combined, the conditions do not necessarily correspond to basic systems, and the model graphs may be other than a cycle graph. Furthermore, the paper provides complexity estimates for GL-models constructed by the proposed method, including the number of vertices and edges in their graphs and the overall complexity of edge functions, depending on the characteristics of the corresponding auxiliary models. The practical value of the method is that it enables automated construction of comprehensive models for systems with complex operability conditions and supports efficient reliability evaluation of real multiprocessor systems.

Keywords: Fault-tolerant multiprocessor systems; GL-models; non-basic systems; control systems; reliability evaluation; statistical experiments

For citation: **Romankevich V. A., Morozov K. V., Halytsky D. V., Yermolenko I. A., Zacharioudakis L. "A method for constructing GL-models of behavior under failure flow for complex non-basic fault-tolerant multiprocessor systems".** ***Applied Aspects of Information Technology.*** **2025; Vol.8 No.4: 465 –480. DOI: https://doi.org/10.15276/aait.08.2025.30**

INTRODUCTION

In recent decades, the automation of various processes has become increasingly widespread [1], [2]. On the one hand, this makes it possible to relieve humans from performing monotonous tasks, and on the other, to reduce the impact of the human factor on the execution of such tasks. Some tasks cannot be performed by humans at all due to the limitations of physiological capabilities (for example, reaction speed), or because of the undesirability or impossibility of their direct presence at the site (for instance, space missions, military unmanned systems, etc.). One of the key

components of such objects and systems is their control system (CS), which, based on data obtained from various sensors, generates the corresponding control signals for actuating devices [1], [2].

There exist systems whose malfunction may lead to significant adverse consequences, such as considerable material losses, threats to human health or life, and risks to the welfare and stability of a state [3], [4], [5]. Such systems in general – and their control systems (CS) in particular – are therefore subject to increased reliability requirements. Moreover, the control of these systems often involves solving problems of substantial computational complexity. Hence, it is reasonable to implement the control systems of such objects using so-called fault-tolerant multiprocessor systems (FTMS), which consist of a large number of processors (allowing, in particular, for high performance levels) and remain operational even in the event of failures of some processors (thus ensuring high reliability) [6], [7], [8].

LITERATURE REVIEW AND PROBLEM STATEMENT

For a developer of fault-tolerant multiprocessor systems, it is important to be able to assess the reliability level of the system being designed. This task is not always straightforward, particularly because FTMSs used in control systems may have complex and heterogeneous architectures, consisting of processors of different types that perform specialized tasks, and so forth.

Methods for calculating the reliability parameters of FTMSs can be conventionally divided into two groups [9], [10]. The first group comprises methods based on the derivation of complex analytical expressions, which, on the one hand, often enable highly accurate evaluation of the reliability parameters of an FTMS, but on the other hand, are not universal: for each new type of system, a new method usually has to be developed [11], [12], [13], [14], [15], [16]. The second group includes methods that allow calculating FTMS reliability parameters by conducting statistical experiments using models of their behavior under a failure flow [17], [18], [19] [20]. These methods are universal; however, the accuracy of the obtained results generally depends on the number of experiments performed. Therefore, reducing the complexity of an experiment (in particular, through simplification of the model) makes it possible, on the one hand, to decrease the computation time and, on the other hand, to increase the accuracy of the results.

As models of FTMS behavior under a failure flow, GL-models [20], [21] can be employed, which combine the properties of graphs and Boolean functions. A GL-model represents an undirected graph in which each edge is associated with a Boolean edge function that depends on the so-called system state vector – a Boolean vector whose elements correspond to the states of the system's processors (1 indicates that a processor is operational, 0 indicates that it has failed). If an edge function evaluates to zero, the corresponding edge is removed from the graph. The connectivity of the graph for a given vector corresponds to the system's state under a specific configuration of processor states: a connected graph represents an operational system, whereas a disconnected graph indicates a system failure. The construction of GL-models for FTMSs can be performed using various approaches [22], [23], [24], [25].

Of particular interest are the so-called basic systems, which are capable of remaining operational provided that no more than a certain number of their processors have failed. A basic FTMS, denoted as $K(m, n)$, consists of n processors and is tolerant to the failure of no more than m arbitrary processors. GL-models of basic systems can be constructed on the basis of cycle graphs [22], [23], which, in particular, allows for a reduction in the complexity of the connectivity evaluation procedure. It is worth noting that the state of a basic FTMS under a failure flow can be easily determined even without constructing a GL-model – for example, by simply counting the number of zeros in the system state vector.

However, control FTMSs are often non-basic, meaning that they remain tolerant to certain failures of a given multiplicity while being intolerant to other failures of the same multiplicity. Such systems include, in particular, consecutive k-out-of-n [14], [16], [26], [27], [28], consecutive k-within-m-out-of-n [29], [30], consecutive k-out-of-r-from-n [31], [32], m-consecutive-k-out-of-n [33], [34], [35], (n, f, k) [36], [37], $\langle n, f\, k\rangle$ [36], [38], consecutive-(k, l)-out-of-n [15], m-consecutive-k,l-out-of-n [39], [40], k_c-out-of-n [35], (r, s)-out-of-(m, n) [12], [41], [42], consecutive-k_r-out-of-n_r [43], as well as other, potentially even more complex, systems. In such cases, the determination of the system state can no longer be reduced to a simple count of zeros in the state vector.

ISSN 2617-4316 (Print)
ISSN 2663-7723 (Online)

PROBLEM STATEMENT

The construction of a GL-model for a non-basic system can be performed by modifying the model of a certain basic system, in particular, by altering the structure of its graph (for example, by introducing additional edges) and/or by changing the expressions of its edge functions. This approach is especially convenient in cases where the FTMS does not differ significantly from the basic one – that is, it behaves as a basic system in most situations and deviates only in certain specific cases (for particular combinations of operational and failed processors), either becoming non-operational or remaining functional [24], [44], [45].

However, some real FTMSs may differ significantly from basic ones. For example, each processor type in a system may have its own maximum allowable failure multiplicity. In addition, additional constraints may also be present – for instance, a maximum total failure multiplicity, or a maximum number of allowable failures within a certain subset of processors. This is particularly relevant for systems composed of several distinct subsystems: each subsystem may behave as a basic system, while the overall system behavior may differ considerably from that of a basic one. In [25], a method for constructing GL-models for such hierarchical systems was proposed. However, the models obtained by this method are themselves hierarchical and, consequently, rather complex: first, calculations must be performed for several auxiliary models, after which the calculation for the system's GL-model is carried out.

A relevant problem is the construction of GL-models for complex non-basic fault-tolerant multiprocessor systems for which existing modeling methods are ineffective, particularly when their application leads to a significant increase in the complexity of the edge-function expressions in the resulting models. This issue is especially pronounced for systems whose operability is determined by several independent or weakly coupled conditions, each of which must be modeled and combined within a unified framework.

RESEARCH AIM AND OBJECTIVES

The aim of this study is to develop a method for constructing GL-models of complex non-basic FTMSs of a special type – namely, those whose behavior under a failure flow can be described by a set of relatively simple conditions (for example, the failure of no more than a certain number of processors within a specific subset of the system's processors), for each of which a separate GL-model can be constructed by one method or another. It is assumed that the operability of the system is maintained only when all of these conditions are satisfied simultaneously.

To achieve this goal, the following objectives have been defined:

1) to develop a method for constructing GL-models of non-basic FTMSs by combining several auxiliary models formed for individual operability conditions of the system;

2) to design, based on the proposed method, an algorithm for constructing such GL-models;

3) to perform an experimental validation of the correctness of the GL-models constructed using the proposed method.

METHOD FOR CONSTRUCTING A GL-MODEL OF A NON-BASIC FTMS

Let us consider a non-basic FTMS that remains operational only if a certain set of conditions C_1, C_2, …, C_k are simultaneously satisfied. For each of these conditions C_i, a corresponding GL-model M_i can be constructed in some way (these models will be referred to as *auxiliary* models). Thus, the satisfaction of condition C_i corresponds to the connectivity of the graph of model M_i.

Since the system is operational only when all conditions C_i are satisfied, the graph of the GL-model M of this FTMS must remain connected if the graphs of all models M_i are connected, and it must become disconnected if the graph of at least one of the models M_i becomes disconnected.

Let us now consider two arbitrary graphs, G_1 and G_2. Let graph G_1 contain vertices $\alpha_1, \alpha_2, \dots, \alpha_{n_1}$, and graph G_2 contain vertices $\beta_1, \beta_2, \dots, \beta_{n_2}$. Select two arbitrary vertices, α_i and β_j, belonging to graphs G_1 and G_2, respectively. Perform the merging of graphs G_1 and G_2 through vertices α_i and β_j, i.e., merge these vertices into one, while copying the remaining vertices and all edges of both graphs. As a result of this merging, a new graph G is obtained.

The connectivity of graph G_1 means that there exists a path between any pair of its vertices. Conversely, the lack of connectivity of graph G_1 means that there is at least one pair of vertices with no path between them. Similarly, the connectivity of graph G_2 implies that a path exists between any pair of its vertices, while the absence of connectivity in graph G_2 indicates that there is at least one pair of vertices that are not connected by a path.

Let us show that graph G is connected if and only if both graphs G_1 and G_2 are connected, and

that it becomes disconnected if at least one of the graphs G_1 or G_2 loses connectivity. Assume that graphs G_1 and G_2 are connected. Consider an arbitrary pair of vertices of graph G. The following three cases are possible.

1. Both vertices belong to the set $\{\alpha_1, \alpha_2, \dots, \alpha_{n_1}\}$. In this case, the existence of a path between these vertices follows from the connectivity of graph G_1.

2. Both vertices belong to the set $\{\beta_1, \beta_2, \dots, \beta_{n_2}\}$. Similarly, the existence of a path between these vertices follows from the connectivity of graph G_2.

3. One of the vertices (denoted as α_k) belongs to the set $\{\alpha_1, \alpha_2, \dots, \alpha_{n_1}\}$, and the other (denoted as β_l) belongs to the set $\{\beta_1, \beta_2, \dots, \beta_{n_2}\}$. From the connectivity of graph G_1, it follows that there exists a path between vertices α_k and α_i; from the connectivity of graph G_2, it follows that there exists a path between vertices β_j and β_l. Therefore, since in graph G the vertices α_i and β_j are merged into a single vertex, there also exists a path between vertices α_k and β_l.

Thus, if graphs G_1 and G_2 are connected, graph G will also be connected.

Next, we show that graph G will be disconnected if at least one of the graphs G_1 or G_2 is disconnected. Let graph G_1 be disconnected. In this case, there exists at least one pair of vertices from the set $\{\alpha_1, \alpha_2, \dots, \alpha_{n_1}\}$ between which no path exists. It is easy to see that in this situation there will also be no path from at least one of these vertices to vertex α_i (otherwise, a path between the previously considered vertices would exist through vertex α_i). Consequently, in graph G, there will also be no path from this vertex to any vertices from the set $\{\beta_1, \beta_2, \dots, \beta_{n_2}\}$, since only vertices α_i and β_j were merged and no additional edges were added. Thus, graph G will be disconnected.

Similarly, it can be shown that if graph G_2 is disconnected, then graph G will also be disconnected.

The merging procedure described above can be extended to an arbitrary number of graphs. Let us apply it to merge the graphs of the auxiliary models $M_1, M_2, \dots, M_k$.

It is easy to see that the GL-model M obtained in this way will indicate the operable (fault-free) state of the system if and only if each of the conditions $C_1, C_2, \dots, C_k$ is satisfied, that is, when the graphs of all models $M_1, M_2, \dots, M_k$ remain connected. Indeed, on the one hand, the connectivity of each of the graphs of models $M_1, M_2, \dots, M_k$ ensures the connectivity of the graph of model M. On the other hand, merging the graphs of the models through common vertices does not create any additional paths that could preserve the connectivity of the graph of model M in the event that it is lost by at least one of the graphs of models $M_1, M_2, \dots, M_k$.

It should also be noted that the method allows for an arbitrary choice of both the order in which the models are merged and the vertices through which their graphs are joined. Therefore, for the same FTMS, a large number of alternative GL-model variants can be obtained. This, in particular, makes it possible to construct a more convenient graph structure (for example, to accelerate the connectivity evaluation procedure or to simplify further model modifications, if required).

ALGORITHM FOR CONSTRUCTING A GL-MODEL OF A NON-BASIC FTMS

According to the proposed method, the algorithm for constructing the GL-model of the above-described non-basic FTMS can be formulated as follows.

1. For each of the conditions $C_1, C_2, \dots, C_k$, construct separate GL-models (denote them as $\Omega = \{M_1, M_2, \dots, M_k\}$).

2. Remove an arbitrary model M_i from the set Ω; let $M = M_i$ be the model under construction.

3. If the set Ω is empty, proceed to Step 8.

4. Remove an arbitrary model M_j from the set Ω.

5. Select an arbitrary vertex α in the graph of model M and an arbitrary vertex β in the graph of model M_j.

6. Merge the graphs of models M and M_j through vertices α and β, resulting in a new model M.

7. Return to Step 3.

8. The resulting model M is the desired GL-model of the system.

EXAMPLES AND EXPERIMENTAL RESULTS

Example 1. Let us construct a GL-model of a system consisting of 12 processors, which is operable if and only if the following conditions are simultaneously satisfied (Fig. 1).

1. Among processors 1…6, there are no more than two faulty ones.

2. Among processors 7…10, there is no more than one faulty processor.

3. Among processors 4…8, there is no more than one faulty processor.

ISSN 2617-4316 (Print)
ISSN 2663-7723 (Online)

4. Among processors 1, 3, 7, 8, and 9, there are no more than two faulty ones.

5. In the entire system, there are no more than three faulty processors.

Each of the above conditions corresponds to the behavior under a failure flow of a certain basic system. Let us construct GL-models for each of them, namely: model $K_1(2, 6)$ for the set of processors 1…6, model $K_2(1, 4)$ for processors 7…10, model $K_3(1, 5)$ for processors 4…8, model $K_4(2, 5)$ for processors 1, 3, 7, 8, 9, and model $K_5(3, 12)$ for processors 1…12. The elements of the system state vector will be denoted as x_i, where i is the index of the corresponding processor. To construct these models, we will use the method described in [22].

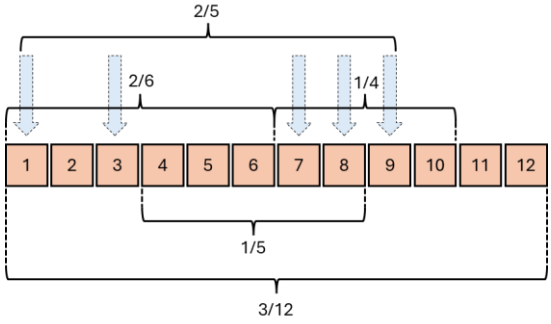

Fig. 1. **The FTMS considered in *Example 1***
Source: **compiled by the authors**

Model $K_1(2, 6)$ is based on a cycle graph with five vertices (denoted as α_1, α_2, α_3, α_4, α_5) and five edges (Fig. 2). It has the following edge functions:

$f_1^1 = x_1 \vee x_2;$
$f_2^1 = x_1 x_2 \vee x_3;$
$f_3^1 = x_1 x_2 x_3 \vee x_4 x_5 x_6;$
$f_4^1 = x_4 \vee x_5;$
$f_5^1 = x_4 x_5 \vee x_6.$

Model $K_2(1, 4)$ is constructed on a cycle graph with four vertices (β_1, β_2, β_3, β_4) and four edges (Fig. 2). Its edge functions are as follows:

$f_1^2 = x_7;$
$f_2^2 = x_8;$
$f_3^2 = x_9;$
$f_4^2 = x_{10}.$

Model $K_3(1, 5)$ is built on a cycle graph with five vertices (γ_1, γ_2, γ_3, γ_4, γ_5) and five edges (Fig. 2). The edge functions of this model have the following form:

$f_1^3 = x_4;$
$f_2^3 = x_5;$
$f_3^3 = x_6;$
$f_4^3 = x_7;$
$f_5^3 = x_8.$

Model $K_4(2, 5)$ is based on a cycle graph with four vertices (δ_1, δ_2, δ_3, δ_4) and four edges (Fig. 2). It has the following edge functions:

$f_1^4 = x_1 \vee x_3;$
$f_2^4 = x_1 x_3 \vee x_7;$
$f_3^4 = x_1 x_3 x_7 \vee x_8 x_9;$
$f_4^4 = x_8 \vee x_9.$

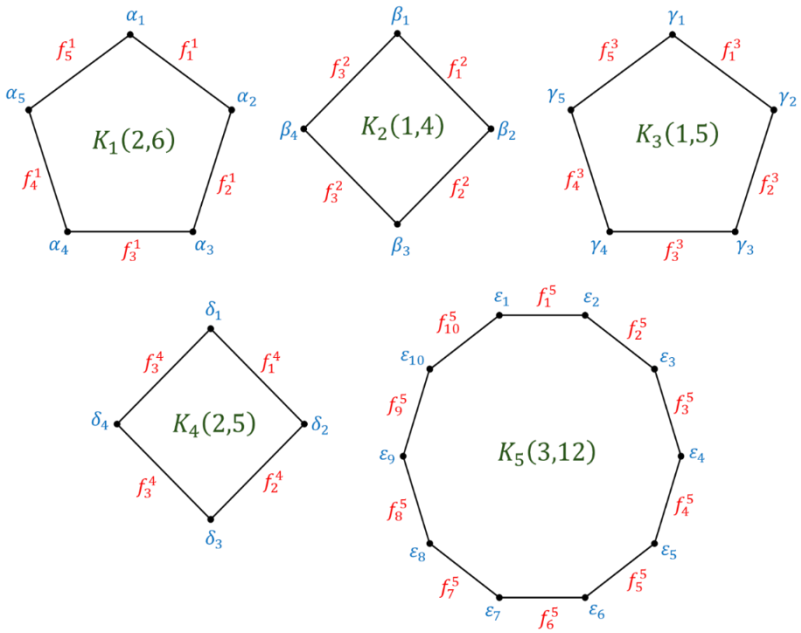

Fig. 2. **GL-models $K_1(2, 6)$, $K_2(1, 4)$, $K_3(1, 5)$, $K_4(2, 5)$ and $K_5(3, 12)$**
Source: **compiled by the authors**

Finally, model $K_5(3, 12)$ is based on a cycle graph with ten vertices (ε_1, ε_2, ε_3, ε_4, ε_5, ε_6, ε_7, ε_8, ε_9, ε_{10}) and ten edges (Fig. 2). Its edge functions are as follows:

$f_1^5 = x_1 \vee x_2 \vee x_3$;
$f_2^5 = (x_1 \vee x_2)(x_1x_2 \vee x_3) \vee x_4x_5x_6$;
$f_3^5 = x_1x_2x_3 \vee (x_4 \vee x_5)(x_4x_5 \vee x_6)$;
$f_4^5 = x_4 \vee x_5 \vee x_6$;
$f_5^5 = (x_1 \vee x_2)(x_1x_2 \vee x_3) \wedge$
$\wedge (x_1x_2x_3 \vee x_4x_5x_6)(x_4 \vee x_5)(x_4x_5 \vee x_6) \vee$
$\vee x_7x_8x_9x_{10}x_{11}x_{12}$;
$f_6^5 = x_1x_2x_3x_4x_5x_6 \vee (x_7 \vee x_8)(x_7x_8 \vee x_9) \wedge$
$\wedge (x_7x_8x_9 \vee x_{10}x_{11}x_{12})(x_{10} \vee x_{11}) \wedge$
$\wedge (x_{10}x_{11} \vee x_{12})$;
$f_7^5 = x_7 \vee x_8 \vee x_9$;
$f_8^5 = (x_7 \vee x_8)(x_7x_8 \vee x_9) \vee x_{10}x_{11}x_{12}$;
$f_9^5 = x_7x_8x_9 \vee (x_{10} \vee x_{11})(x_{10}x_{11} \vee x_{12})$;
$f_{10}^5 = x_{10} \vee x_{11} \vee x_{12}$.

Next, to construct the GL-model of the system under consideration, we perform a sequential merging of the previously constructed models. For example, we may first select model $K_1(2, 6)$, and then merge it with model $K_2(1, 4)$ through vertices α_2 and β_4 (the resulting merged vertex is denoted as ω_1). Then, the obtained model is merged with model $K_3(1, 5)$ through vertices β_2 and γ_5 (the corresponding vertex in the new model is denoted as ω_2). After that, we merge the resulting model with model $K_4(2, 5)$ through vertices α_4 and δ_1, denoting the merged vertex as ω_3. Finally, the obtained model is merged with model $K_5(3, 12)$ through vertices γ_4 and ε_2 (as in the previous cases, the corresponding vertex in the new model is denoted as ω_4). The GL-model obtained as a result of these successive mergers (Fig. 3) represents the behavior of the considered system under a failure flow, which was confirmed by the experiments performed with it.

According to the results of experiments (conducted for all possible Boolean vectors of length 12), the obtained GL-model represents the operable state of the system for all vectors containing no more than one zero (it is evident that under such vectors, none of the operability conditions of the considered FTMS can be violated), as well as for the following 51 vectors with two zeros: 111111111100, 111111111010, 111111110110, 111111101110, 111111011110, 111110111110, 111101111110, 111011111110, 110111111110, 101111111110, 011111111110, 111111111001, 111111110101, 111111101101, 111111011101, 111110111101, 111101111101, 111011111101, 110111111101, 101111111101, 011111111101, 111110111011, 111101111011, 111011111011, 110111111011, 101111111011, 011111111011, 111110110111, 111101110111, 111011110111, 110111110111, 101111110111, 011111110111, 110111101111, 101111101111, 011111101111, 110111011111, 101111011111, 011111011111, 110110111111, 101110111111, 011110111111, 110101111111, 101101111111, 011101111111, 110011111111, 101011111111, 011011111111, 100111111111,

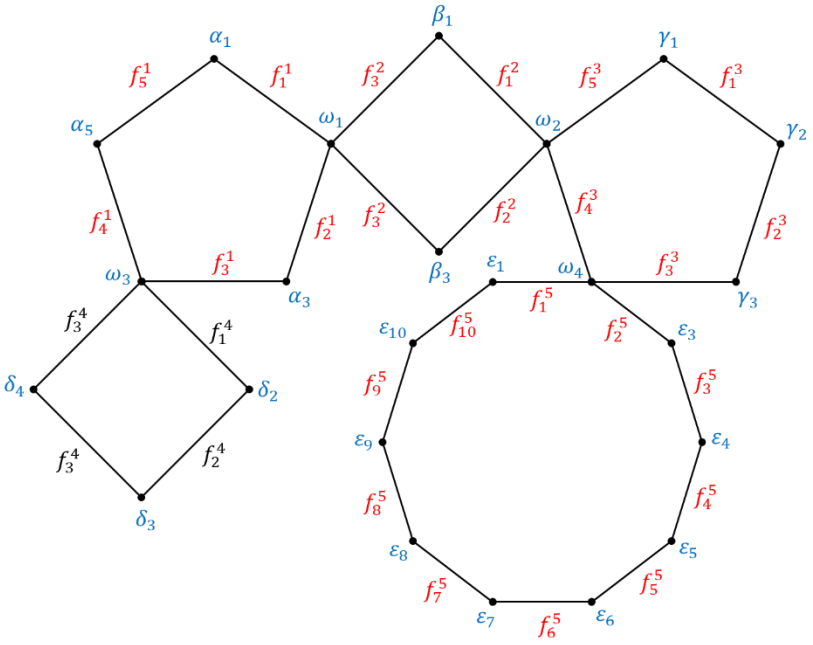

Fig. 3. **The GL-model of the FTMS constructed using the proposed method for *Example 1***
Source: **compiled by the authors**

ISSN 2617-4316 (Print)
ISSN 2663-7723 (Online)

010111111111, 001111111111; and the following 97 vectors with three zeros: 111111111000, 111111110100, 111111101100, 111111011100, 111110111100, 111101111100, 111011111100, 110111111100, 101111111100, 011111111100, 111110111010, 111101111010, 111011111010, 110111111010, 101111111010, 011111111010, 111110110110, 111101110110, 111011110110, 110111110110, 101111110110, 011111110110, 110111101110, 101111101110, 011111101110, 110111011110, 101111011110, 011111011110, 110110111110, 101110111110, 011110111110, 110101111110, 101101111110, 011101111110, 110011111110, 101011111110, 011011111110, 100111111110, 010111111110, 001111111110, 111110111001, 111101111001, 111011111001, 110111111001, 101111111001, 011111111001, 111110110101, 111101110101, 111011110101, 110111110101, 101111110101, 011111110101, 110111101101, 101111101101, 011111101101, 110111011101, 101111011101, 011111011101, 110110111101, 101110111101, 011110111101, 110101111101, 101101111101, 011101111101, 110011111101, 101011111101, 011011111101, 100111111101, 010111111101, 001111111101, 110110111011, 101110111011, 011110111011, 110101111011, 101101111011, 011101111011, 110011111011, 101011111011, 011011111011, 100111111011, 010111111011, 001111111011, 110110110111, 101110110111, 011110110111, 110101110111, 101101110111, 011101110111, 110011110111, 101011110111, 011011110111, 100111110111, 001111110111, 100111101111, 001111101111, 100111011111, 001111011111.

The model indicates an inoperable system state for all vectors containing four or more zeros (which directly follows from operability condition 5 of the system). A detailed analysis confirmed that the above-mentioned system state vectors with two and three zeros are indeed exactly those vectors that correspond to the simultaneous fulfillment of all five operability conditions of the FTMS considered in this example.

It should be noted that, as mentioned above, both the order of model merging and the choice of vertices through which their graphs are combined can be arbitrary. For example, one could first select model $K_5(3, 12)$, then merge it with model $K_4(2, 5)$ through vertices ε_9 and δ_2 (ω_1). Next, it can be merged with model $K_2(1, 4)$ through vertices ε_4 and β_4 (ω_2), and with model $K_1(2, 6)$ through vertices ε_7 and α_1 (ω_3). Finally, it can be merged with model $K_3(1, 5)$ through vertices ε_2 and γ_4 (ω_4). The GL-model obtained as a result of these transformations (Fig. 4), although differing from the previous one in the structure of its graph, demonstrates, as confirmed by experiments, behavior consistent with the previous model for identical input vectors.

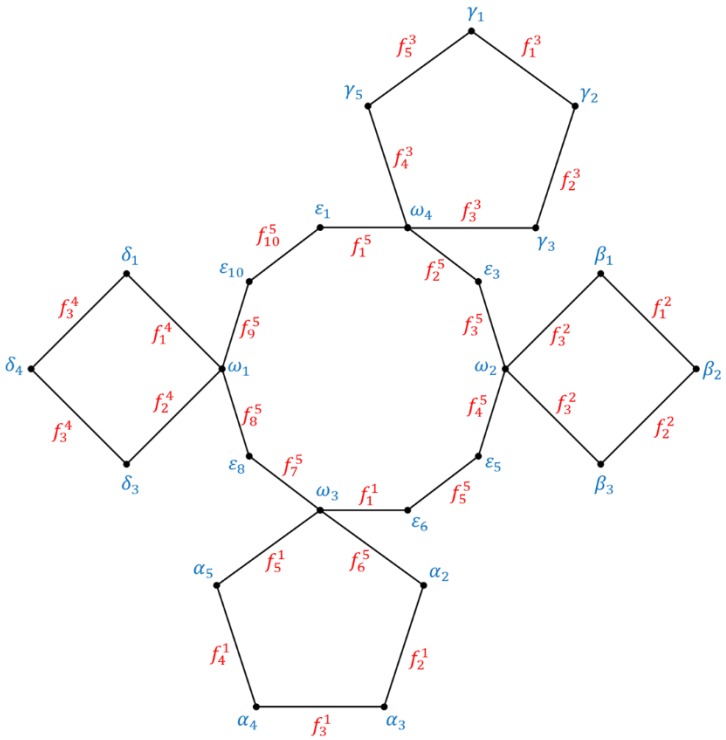

Fig. 4. **An alternative GL-model of the FTMS under consideration for *Example 1***
Source: **compiled by the authors**

It should also be noted that the proposed method does not impose any restrictions on how the GL-models for each condition are constructed.

Example 2. Let us construct a GL-model of an FTMS that also consists of 12 processors but has slightly different operability conditions (all of which must be satisfied simultaneously).

1. Among processors 1…6, there are no more than two faulty ones (similarly to the system in *Example 1*).

2. Among processors 7…10, there is at most one faulty processor, or, alternatively, in each of the pairs (7, 8) and (9, 10), there is at most one faulty processor (that is, up to two in total), provided that at least one of processors 7 or 9 is operational.

3. Among processors 4…8, there is at most one faulty processor, or, if processors 4 and 5 are operational, there may be up to two faulty ones.

4. Among processors 1, 3, 7, 8, and 9, there are no more than two faulty ones, or no more than three, provided that only one faulty processor is present among processors 1, 3, and 7.

5. In the entire system, there are no more than three faulty processors, and if among the faulty ones there are processors 1, 2, 5, or 10, then no more than two are allowed.

Let us construct the GL-models M_1, M_2, M_3, M_4, and M_5 for each of the above conditions. The first model, M_1, corresponds to a basic 2-failure-tolerant system. To construct it, we use the method described in [22]. It is based on a cycle graph with six vertices (α_1, α_2, α_3, α_4, α_5, α_6), six edges (Fig. 5), and the following edge functions:

$f_1^1 = x_1 \vee x_2x_3$;
$f_2^1 = x_2 \vee x_3x_4$;
$f_3^1 = x_3 \vee x_4x_5$;
$f_4^1 = x_4 \vee x_5x_6$;
$f_5^1 = x_5 \vee x_6x_1$;
$f_6^1 = x_6 \vee x_1x_2$.

The GL-model M_2 can be obtained from the basic model $K_2(1, 4)$, constructed in *Example 1*, by adding an additional edge $\beta_1\beta_3$ with the edge function $f_5^2 = x_7 \vee x_9$ [46]. Thus, the model will contain four vertices (β_1, β_2, β_3, β_4), five edges (Fig. 5), and the following edge functions:

$f_1^2 = x_7$;
$f_2^2 = x_8$;
$f_3^2 = x_9$;
$f_4^2 = x_{10}$;
$f_5^2 = x_7 \vee x_9$.

Model M_3 is constructed using the method described in [44], based on the expression $c = x_4x_5$, which corresponds to the simultaneous operability of processors 4 and 5. Accordingly, this model is based on a cycle graph with five vertices (γ_1, γ_2, γ_3, γ_4, γ_5), five edges (Fig. 5), and the following edge functions:

$f_1^3 = x_4 \vee cx_5x_6 = x_4 \vee x_4x_5x_5x_6 = x_4$;
$f_2^3 = x_5 \vee cx_6x_7 = x_5 \vee x_4x_5x_6x_7 = x_5$;
$f_3^3 = x_6 \vee cx_7x_8 = x_6 \vee x_4x_5x_7x_8$;
$f_4^3 = x_7 \vee cx_8x_4 = x_7 \vee x_4x_5x_8x_4 =$
$= x_7 \vee x_4x_5x_8$;
$f_5^3 = x_8 \vee cx_4x_5 = x_7 \vee x_4x_5x_4x_5 =$
$= x_7 \vee x_4x_5$.

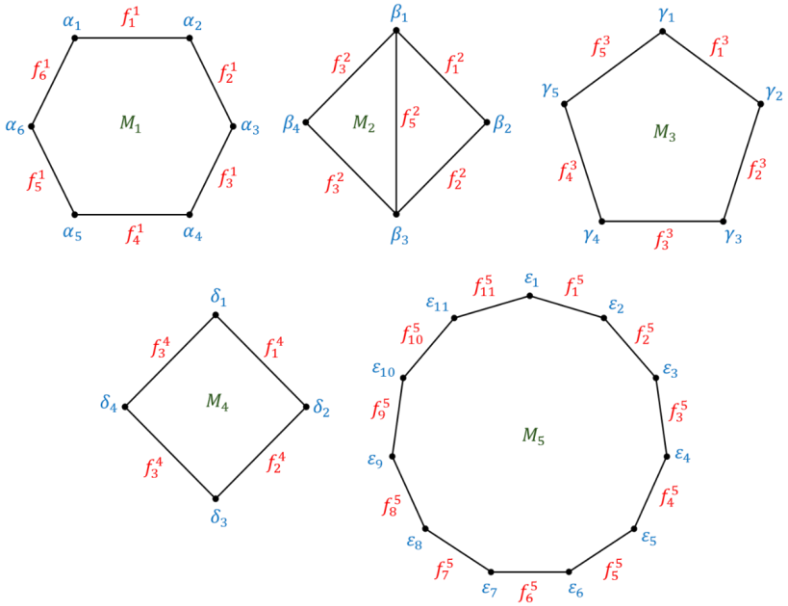

Fig. 5. **GL-models M_1, M_2, M_3, M_4 and M_5**
Source: **compiled by the authors**

ISSN 2617-4316 (Print)
ISSN 2663-7723 (Online)

Model M_4 is obtained by modifying the third edge function of model $K_4(2, 5)$ from *Example 1* in accordance with the method proposed in [45]. As a result, the GL-model is based on a cycle graph with four vertices (δ_1, δ_2, δ_3, δ_4), four edges, and the following edge functions:

$f_1^4 = x_1 \vee x_3;$
$f_2^4 = x_1x_3 \vee x_7;$
$f_3^4 = (x_1 \vee x_3)(x_1x_3 \vee x_7) \vee x_8x_9;$
$f_4^4 = x_8 \vee x_9.$

Model M_5 is constructed according to the method proposed in [24], using the expressions $c_1 = \bar{c}_2 = \bar{x}_1 \vee \bar{x}_2 \vee \bar{x}_5 \vee \bar{x}_{10}$ and $c_2 = \bar{c}_1 = x_1x_2x_5x_{10}$. Thus, the GL-model is based on a cycle graph with eleven vertices (ε_1, ε_2, ε_3, ε_4, ε_5, ε_6, ε_7, ε_8, ε_9, ε_{10}, ε_{11}) and eleven edges (Fig. 5), with the following edge functions:

$f_1^5 = c_1(x_1 \vee x_2) \vee \bar{c}_1(x_1 \vee x_2 \vee x_3) =$
$= x_1 \vee x_2;$
$f_2^5 = c_1(x_1x_2 \vee x_3) \vee$
$\vee \bar{c}_1\big((x_1 \vee x_2)(x_1x_2 \vee x_3) \vee x_4x_5x_6\big) =$
$= x_1x_2 \vee x_3;$
$f_3^5 = c_1(x_1x_2x_3 \vee x_4x_5x_6) \vee$
$\vee \bar{c}_1\big(x_1x_2x_3 \vee (x_4 \vee x_5)(x_4x_5 \vee x_6)\big) =$
$= x_1x_2x_3 \vee x_5\big(x_4x_6 \vee x_1x_2x_{10}(x_4 \vee x_6)\big);$
$f_4^5 = c_1(x_4 \vee x_5) \vee \bar{c}_1(x_4 \vee x_5 \vee x_6) =$
$= x_4 \vee x_5 \vee x_1x_2x_5x_6x_{10};$
$f_5^5 = c_1(x_4x_5 \vee x_6) \vee \bar{c}_1((x_1 \vee x_2) \wedge$
$\wedge (x_1x_2 \vee x_3)(x_1x_2x_3 \vee x_4x_5x_6)(x_4 \vee x_5) \wedge$
$\wedge (x_4x_5 \vee x_6) \vee x_7x_8x_9x_{10}x_{11}x_{12}) =$
$= (x_4x_5 \vee x_6) \wedge$
$\wedge (\bar{x}_1 \vee \bar{x}_2 \vee \bar{x}_5 \vee \bar{x}_{10} \vee x_3 \vee x_4x_6) \vee$
$\vee x_1x_2x_5x_7x_8x_9x_{10}x_{11}x_{12};$
$f_6^5 = c_1(x_1x_2x_3x_4x_5x_6 \vee x_7x_8x_9x_{10}x_{11}x_{12}) \vee$
$\vee \bar{c}_1(x_1x_2x_3x_4x_5x_6 \vee (x_7 \vee x_8)(x_7x_8 \vee x_9) \wedge$
$\wedge (x_7x_8x_9 \vee x_{10}x_{11}x_{12})(x_{10} \vee x_{11}) \wedge$
$\wedge (x_{10}x_{11} \vee x_{12})) =$
$= x_1x_2x_3x_4x_5x_6 \vee x_{10}((\bar{x}_1 \vee \bar{x}_2 \vee \bar{x}_5) \wedge$
$\wedge x_7x_8x_9x_{11}x_{12} \vee x_1x_2x_5(x_7 \vee x_8) \wedge$
$\wedge (x_7x_8 \vee x_9)(x_7x_8x_9 \vee x_{11}x_{12})(x_{11} \vee x_{12}));$
$f_7^5 = c_1(x_7 \vee x_8) \vee \bar{c}_1(x_7 \vee x_8 \vee x_9) =$
$= x_7 \vee x_8 \vee x_1x_2x_5x_9x_{10};$
$f_8^5 = c_1(x_7x_8 \vee x_9) \vee \bar{c}_1 \wedge$
$\wedge \big((x_7 \vee x_8)(x_7x_8 \vee x_9) \vee x_{10}x_{11}x_{12}\big) =$
$= (x_7x_8 \vee x_9)(\bar{x}_1 \vee \bar{x}_2 \vee \bar{x}_5 \vee \bar{x}_{10} \vee x_7 \vee x_8) \vee$
$\vee x_1x_2x_5x_{10}x_{11}x_{12};$
$f_9^5 = c_1(x_7x_8x_9 \vee x_{10}x_{11}x_{12}) \vee$
$\vee \bar{c}_1\big(x_7x_8x_9 \vee (x_{10} \vee x_{11})(x_{10}x_{11} \vee x_{12})\big) =$
$= x_7x_8x_9 \vee (\bar{x}_1 \vee \bar{x}_2 \vee \bar{x}_5)x_{10}x_{11}x_{12} \vee$
$\vee x_1x_2x_5x_{10}(x_{11} \vee x_{12});$
$f_{10}^5 = c_1(x_{10} \vee x_{11}) \vee \bar{c}_1(x_{10} \vee x_{11} \vee x_{12}) =$
$= x_{10} \vee x_{11};$
$f_{11}^5 = c_1(x_{10}x_{11} \vee x_{12}) \vee \bar{c}_1 =$
$= x_{10}(x_{11} \vee x_1x_2x_5) \vee x_{12}.$

To construct the GL-model of the FTMS, we perform a merging of the graphs of the auxiliary models developed above. For example, let us combine the graphs of models M_1, M_2, M_3, M_4, and M_5 by merging the vertices α_2 and β_3 (the resulting vertex denoted as ω_1), β_2 and γ_5 (ω_2), α_6 and δ_2 (ω_3), and γ_3 and ε_{11} (ω_4). The GL-model obtained as a result of this merging is shown in Fig. 6.

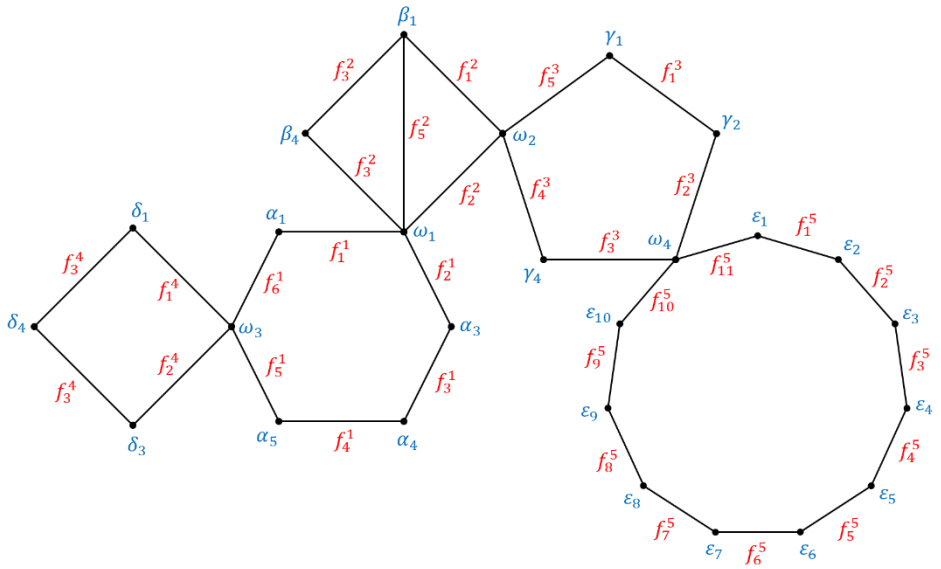

Fig. 6. **The GL-model of the FTMS constructed using the proposed method for *Example 2***
Source: **compiled by the authors**

Experimental verification has confirmed that, as in *Example 1*, the model accurately represents the operable state of the system for those and only those vectors that correspond to the simultaneous fulfillment of all the above conditions. The full set of such system state vectors is omitted here for brevity.

A comparison was also performed between the complexity of the GL-models constructed using the proposed method and that of the models obtained by known approaches, in particular by blocking system state vectors through the modification of the edge-function expressions of basic models using the corresponding zero and one constituents. As baseline models, we considered the models $K(1, 12)$, $K(2, 12)$, and $K(3, 12)$, constructed according to [22].

The complexity (number of logical operations) of the edge-function expressions for each model in *Examples 1* and *2* is presented in *Tables 1* and *2*, respectively. As the results show, the GL-models constructed using the method proposed in this work have significantly simpler edge-function expressions.

Table 1. **Number of logical operations in the edge-function expressions of the GL-models for *Example 1***

Model	Disj.	Conj.	Inv.	Binary ops.	Total ops.
Proposed	41	56	0	97	97
Modified $K(1, 12)$	245	2695	684	2940	3624
Modified $K(2, 12)$	273	1104	426	1377	1803
Modified $K(3, 12)$	1715	199	1279	1914	3193

Source: **compiled by the authors**

Table 2. **Number of logical operations in the edge-function expressions of the GL-models for *Example 2***

Model	Disj.	Conj.	Inv.	Binary ops.	Total ops.
Proposed	55	83	10	138	148
Modified $K(1, 12)$	120	1320	304	1440	1744
Modified $K(2, 12)$	153	384	186	537	723
Modified $K(3, 12)$	2320	254	1739	2574	4313

Source: **compiled by the authors**

DISCUSSION OF RESULTS

The GL-models constructed using the proposed method exhibit a rather complex structure, in particular, being based on graphs that do not belong to the class of cycle graphs. This somewhat complicates the procedure of assessing the model graph's connectivity compared to methods that generate models based on cycle graphs (e.g., [24], [44], [45]), since such an assessment can no longer be reduced to a simple count of removed edges. On the other hand, the complexity of their edge functions remains moderate, as the method directly employs the expressions of the edge functions from the auxiliary models. This feature distinguishes the proposed approach from, for instance, the method described in [24], where the edge functions typically become more complex because a single expression combines several functions from different models.

In general, it can be readily observed that the complexity of a GL-model obtained using the proposed method can be estimated as follows. The number of edges in the model graph:

$$e = \sum_{i=1}^{N} e_i,$$

where N is the number of auxiliary models, and e_i is the number of edges in the graph of the i-th auxiliary model (the edges of all auxiliary model graphs are preserved). The number of vertices in the model graph:

$$v = \sum_{i=1}^{N} v_i - N + 1,$$

where v_i is the number of vertices of the i-th auxiliary model (since $N - 1$ pairs of vertices are merged). The total complexity of the model's edge functions:

$$c = \sum_{i=1}^{N} \sum_{j=1}^{e_i} c_j^i,$$

where c_j^i denotes the complexity of the j-th edge function of the i-th model (all edge functions of the auxiliary models remain unchanged).

To assess the connectivity of the graph of a GL-model, depth-first search (DFS) or breadth-first search (BFS) algorithms may be applied. The computational complexity of these algorithms is known to be $O(e + v)$, where e is the number of edges in the graph and v is the number of vertices. Taking into account the above estimates for the numbers of edges and vertices in a GL-model constructed using the proposed method, it may be concluded that the complexity of connectivity

ISSN 2617-4316 (Print)
ISSN 2663-7723 (Online)

evaluation for its graph is comparable to that of the auxiliary GL-models (provided that analogous connectivity-check algorithms are used). Considering also the complexity of computing the model's edge-function expressions, it follows that the overall computational cost of using the resulting model is likewise comparable to the cost of using the auxiliary GL-models individually.

It should also be noted that since the graph of the GL-model is initially connected (i.e., prior to evaluating the edge functions), the inequality $e \geq v - 1$ holds (the number of edges cannot be smaller than that of a spanning tree, for which $e = v - 1$). Therefore, $v \leq e + 1$, and the complexity of the connectivity-check algorithm may be expressed as $O(e)$. On the other hand, evaluating the model still requires computing all e edge functions. Thus, even in comparison with the simple edge-counting procedure (which is applicable for cycle-graph-based models), using a connectivity-check algorithm for a graph that does not belong to the class of cycle graphs in the GL-model constructed by the proposed method does not lead to a significant increase in the overall computational complexity.

It is also worth noting that the proposed method does not impose any restrictions on the ways in which the auxiliary GL-models are constructed. For instance, when applying this method, it is possible to combine models built using different approaches simultaneously (e.g., models of basic systems constructed according to [22] and [23]). Moreover, the conditions do not necessarily have to correspond to basic systems (i.e., those defined by the failure of no more than a certain number of arbitrary processors), and the graphs of the auxiliary models are not required to be based on cycle graphs (as was, for example, required in [24]).

As a result of applying the proposed method to an FTMS, a single GL-model is obtained, which, if necessary, can be further modified using known techniques or employed as an auxiliary component for constructing a more complex model (e.g., in accordance with [25]). This would not be feasible in the case of analyzing the connectivity of separate GL-models corresponding to individual conditions.

It should be noted that, as demonstrated in *Example 1*, the merging of auxiliary model graphs can indeed be performed in various ways (depending on the selected pairs of vertices for merging). This makes it possible, in particular, to form an optimal structure of the resulting GL-model graph – for instance, to simplify computations or to improve the convenience of subsequent modifications. One possible optimization criterion in this context may be the minimization of the model graph's diameter.

CONCLUSIONS

This study proposes a method for constructing GL-models of non-basic fault-tolerant multiprocessor systems whose operability requires the simultaneous fulfillment of multiple conditions. Each of these conditions can be associated with a specific GL-model built using one of the known methods. Constructing models of such systems by existing techniques is nontrivial and often proves to be extremely complex and inefficient from a practical standpoint, since such FTMSs may significantly differ from basic ones. In particular, this may lead to highly complex edge-function expressions in the resulting GL-models. Even for relatively small systems considered in the examples, the overall complexity (i.e., the total number of logical operations) of the edge functions in the GL-models obtained using the proposed method was several times lower compared to the corresponding models constructed by conventional methods (through blocking of the respective system state vectors). For more complex systems, this difference may become even more significant.

The proposed method is based on combining the graphs of auxiliary GL-models constructed for each of the system's operability conditions. In this process, the edges of these model graphs along with their corresponding edge functions are preserved, while pairs of arbitrarily selected vertices are merged.

Examples are provided to demonstrate the application of the proposed method for constructing GL-models of fault-tolerant multiprocessor systems whose operability requires the simultaneous fulfillment of multiple specified conditions. Furthermore, experiments have been conducted to confirm that the constructed models adequately reflect the behavior of the corresponding systems under failure flow conditions.

Additionally, the paper presents complexity estimates for the GL-models obtained using the proposed method, including the number of edges and vertices in the graphs and the overall complexity of the edge function expressions, which are determined by the characteristics of the corresponding auxiliary GL-models.

REFERENCES

1. Kotov, D. O. “A generalized model of an adaptive information-control system of a car with multi-sensor channels of information interaction”. *Applied Aspects of Information Technology*. 2022; 5 (1): 25–34. DOI: https://doi.org/10.15276/aait.05.2022.2.

2. Nazarova, O. S., Osadchyy, V. V. & Rudim, B. Y. “Computer simulation of the microprocessor liquid level automatic control system”. *Applied Aspects of Information Technology*. 2023; 6 (2): 163–174. DOI: https://doi.org/10.15276/aait.06.2023.12.

3. Kovalev, I. S., Drozd, O. V., Rucinski, A., Drozd, M. O., Antoniuk, V. V. & Sulima Y. Y. “Development of Computer System Components in Critical Applications: Problems, Their Origins and Solutions”. *Herald of Advanced Information Technology*. 2020; 3 (4): 252–262. DOI: https://doi.org/10.15276/hait.04.2020.4.

4. Antoniuk, V. V., Drozd, M. O. & Drozd, O. B. “Power-oriented checkability and monitoring of the current consumption in FPGA projects of the critical applications”. *Applied Aspects of Information Technology*. 2019; 2 (2): 105–114. DOI: https://doi.org/10.15276/aait.02.2019.2.

5. Drozd, O., Ivanova, O., Zashcholkin, K., Romankevich, V. & Drozd, Yu. “Checkability Important for Fail-Safety of FPGA-based Components in Critical Systems”. *CEUR Workshop Proceedings*. 2021; 2853: 471–480, https://www.scopus.com/pages/publications/85104838273.

6. Abbaspour, A., Mokhtari, S., Sargolzaei, A. & Yen, K. K. “A survey on active fault-tolerant control systems”. *Electronics*. 2020; 9 (9): 1–23, https://www.scopus.com/pages/publications/85090902832. DOI: https://doi.org/10.3390/electronics9091513.

7. Joshi, H. & Sinha, N. K. “Adaptive fault tolerant control design for stratospheric airship with actuator faults”. *IFAC-PapersOnLine*. 2022; 55 (1): 819–825, https://www.scopus.com/pages/publications/85132157930.
DOI: https://doi.org/10.1016/j.ifacol.2022.04.134.

8. Nedeljkovic, J. N., Dosic, S. M. & Nikolic, G. S. “A Survey of Hardware Fault Tolerance Techniques”. *2023 58th International Scientific Conference on Information, Communication and Energy Systems and Technologies, ICEST 2023 – Proceedings*. 2023: 223–226, https://www.scopus.com/pages/publications/85167870342. DOI: https://doi.org/10.1109/ICEST58410.2023.10187275.

9. Billinton, R. & Allan, R. N. “Reliability Evaluation of Engineering Systems. Concepts and Techniques”. *Springer New York*. 1992. DOI: https://doi.org/10.1007/978-1-4899-0685-4.

10. Kuo, W. & Zuo, M. “Optimal Reliability Modeling”. *John Willey & Sons*. 2002.

11. Hu, B. & Seiler, P. “Pivotal decomposition for reliability analysis of fault tolerant control systems on unmanned aerial vehicles”. *Reliability Engineering & System Safety*. 2015; 140: 130–141, https://www.scopus.com/pages/publications/84928741612. DOI: https://doi.org/10.1016/j.ress.2015.04.005.

12. Lu, J., Yi, H., Li, X. & Balakrishnan, N. “Joint Reliability of Two Consecutive-(1, l) or (2, k)-out-of-(2, n): F Type Systems and Its Application in Smart Street Light Deployment”. *Methodology and Computing in Applied Probability*. 2023; 25 (1): 33, https://www.scopus.com/pages/publications/85148634528 .

13. Jianu, M., Daus, L., Dragoi, V. F. & Beiu, V. “Reliability polynomials of consecutive-k-out-of-n:F systems have unbounded roots”. *Networks*. 2023; 82 (3): 222–228, https://www.scopus.com/pages/publications/85163222383. DOI: https://doi.org/10.1002/net.22168.

14. Yin, J., Balakrishnan, N. & Cui, L. “Efficient reliability computation of consecutive- k-out-of-n: F systems with shared components”. *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability*. 2024; 238 (1): 122–135, https://www.scopus.com/pages/publications/85142023916. DOI: https://doi.org/10.1177/1748006X221130540.

15. Yin, J., Cui, L. & Balakrishnan N. “Reliability of consecutive-(k,l)-out-of-n: F systems with shared components under non-homogeneous Markov dependence”. *Reliability Engineering & System Safety*. 2022; 224: 108549, https://www.scopus.com/pages/publications/85129557917.
DOI: https://doi.org/10.1016/j.ress.2022.108549

16. Yi, H., Cui, L. & Gao, H. “Reliabilities of Some Multistate Consecutive-k Systems”. *IEEE Transactions on Reliability*. 2020; 69 (2): 414–429. DOI: https://doi.org/10.1109/TR.2019.2897726.

17. Dagal, I. "Probabilistic fault tree analysis and dynamic redundancy optimization for next-generation avionic flight control systems". *Reliability Engineering and System Safety*. 2026; 266: 111841, https://www.scopus.com/pages/publications/105020942559. DOI: https://doi.org/10.1016/j.ress.2025.111841.

18. Wu, J., Chen, C., Ma, Y., Xiu, Z., Cheng, Z., Pan, Y. & Song, S. "Integrated Fault Tree and Case Analysis for Equipment Conventional Fault IETM Diagnosis". *Sensors*. 2025; 25 (17): 5231, https://www.scopus.com/pages/publications/105015822431. DOI: https://doi.org/10.3390/s25175231.

19. Dong, H., Xie, H. & Xu, H. "Reliability evaluation of motor controllers based on FMEA and DFTA". *Hangkong Dongli Xuebao/Journal of Aerospace Power*. 2025; 40 (9): 20230656, https://www.scopus.com/pages/publications/105011205376. DOI: https://doi.org/10.13224/j.cnki.jasp.20230656.

20. Romankevich, A. M., Karachun, L. F. & Romankevich, V. A. "Graph-logical models for the analysis of complex fault-tolerant computing systems" (in Russian). *Elektronnoe Modelirovanie*. 2001; 23 (1): 102–111.

21. Romankevich, A., Feseniuk, A., Maidaniuk, I. & Romankevich, V. "Fault-tolerant multiprocessor systems reliability estimation using statistical experiments with GL-models". *Advances in Intelligent Systems and Computing*. 2019; 754: 186–193, https://www.scopus.com/pages/publications/85047465084. DOI: https://doi.org/10.1007/978-3-319-91008-6_19.

22. Romankevich, V. A., Potapova, E. R., Bakhtari, Hedayatollah & Nazarenko, V. V. "GL-model of the behavior of fault-tolerant multiprocessor systems with the minimal number of lost edges". *Visnyk NTUU "KPI". Informatyka, Upravlinnia ta Obchysliuvalna Tekhnika*. 2006; 45: 93–100.

23. Romankevich, A. M., Romankevich, V. A., Kononova, A. A. & Rabah Al Shbul. "On some features of GL-models K(2, n)". *Visnyk NTUU "KPI". Informatyka, Upravlinnia ta Obchysliuvalna Tekhnika*. 2004; 41: 85–92.

24. Romankevich, V. A., Morozov, K. V., Romankevich, A. M., Morozova, A. V. & Zacharioudakis, L. "On the method of building of non-basic GL-models which are formed on combination of edge functions of basic models". *Applied Aspects of Information Technology*. 2024; 7 (2): 175–188. DOI: https://doi.org/10.15276/aait.07.2024.13.

25. Romankevich, A. M., Morozov, K. V. & Romankevich, V. A. "Graph-Logic Models of Hierarchical Fault-Tolerant Multiprocessor Systems". *IJCSNS International Journal of Computer Science and Network Security*. 2019; 19 (7): 151–156.

26. Eryilmaz, S. & Kan, C. "The mean number of failed components in discrete time consecutive k-out-of-n: F system and its application to parameter estimation and optimal age-based preventive replacement". *Reliability Engineering and System* Safety. 2025; 263: 111229. DOI: https://doi.org/10.1016/j.ress.2025.111229. https://www.scopus.com/pages/publications/105005869605.

27. Yin, J. & Cui, L. "Reliability for consecutive-k-out-of-n:F systems with shared components between adjacent subsystems". *Reliability Engineering & System Safety*. 2021; 210: 107532. DOI: https://doi.org/10.1016/j.ress.2021.107532. https://www.scopus.com/pages/publications/85100736521.

28. Chopra, G. & Ram, M. "Linear Consecutive-k-out-of-n:G System Reliability Analysis". *Journal of Reliability and Statistical Studies*. 2022; 15 (2): 669–692. DOI: https://doi.org/10.13052/jrss0974-8024.15211.

29. Zhu, X., Boushaba, M. & Reghioua, M. "Reliability and Joint Reliability Importance in a Consecutive-k-Within-m-out-of-n:F System with Markov-Dependent Components". *IEEE Transactions on Reliability*. 2016; 65 (2): 802–815, https://www.scopus.com/pages/publications/84943426081. DOI: https://doi.org/10.1109/TR.2015.2484079.

30. Torrado, N. "Tail behaviour of consecutive 2-within-m-out-of-n systems with nonidentical components". *Applied Mathematical Modelling*. 2015; 39 (15): 4586–4592, https://www.scopus.com/pages/publications/84937631892. DOI: https://doi.org/10.1016/j.apm.2014.12.042.

31. Chachra, A., Ram, M. & Kumar, A. "A pythagorean fuzzy approach to consecutive k-out-of-r-from-n system reliability modelling". *International Journal of System Assurance Engineering and Management*. 2024, https://www.scopus.com/pages/publications/85200035598. DOI: https://doi.org/10.1007/s13198-024-02435-3.

32. Amirian, Y., Khodadadi, A. & Chatrabgoun, O. "Exact Reliability for a Consecutive Circular k-out-of-r-from-n:F System with Equal and Unequal Component Probabilities". *International Journal of Reliability, Quality and Safety Engineering*. 2020; 27 (1), https://www.scopus.com/pages/publications/85069848232. DOI: https://doi.org/10.1142/S0218539320500035.

33. Triantafyllou, I. "m-Consecutive-k-out-of-n: F Structures with a Single Change Point". *Mathematics*. 2020; 8 (12): 2203, https://www.scopus.com/pages/publications/85097534209. DOI: https://doi.org/10.3390/math8122203.

34. Nashwan, I. "Reliability and Failure Probability Functions of the m-Consecutive-k-out-of-n: F Linear and Circular Systems". *Baghdad Science Journal*. 2021; 18 (2): 430, https://www.scopus.com/pages/publications/85100161580. DOI: https://doi.org/10.21123/bsj.2021.18.2.0430.

35. Triantafyllou, I. S. "Combined m-Consecutive-k-Out-of-n: F and Consecutive kc-Out-of-n:F Structures with Cold Standby Redundancy". *Mathematics*. 2023; 11 (12): 1–13, https://www.scopus.com/pages/publications/85164204439. DOI: https://doi.org/10.3390/math11122597.

36. Cui, L., Wang, M. & Jiang, W. "Reliability analysis of A combination of (n,f,k) and <n,f,k> systems". *Reliability Engineering and System Safety*. 2024; 249: 110191, https://www.scopus.com/pages/publications/85193451324. DOI: https://doi.org/10.1016/j.ress.2024.110191.

37. Amrutkar, K.P. & Kamalja, K.K. "Efficient algorithm for reliability and importance measures of linear weighted-(n,f,k) and (n,f,k) systems". *Computers and Industrial Engineering*. 2017; 107: 85–99, https://www.scopus.com/pages/publications/85015035867. DOI: https://doi.org/10.1016/j.cie.2017.02.011.

38. Triantafyllou, I.S. "Reliability Study of <n, f, 2> Systems: A Generating Function Approach". *International Journal of Mathematical, Engineering and Management Sciences*. 2021; 6 (1): 44–65, https://www.scopus.com/pages/publications/85097568436.
DOI: https://doi.org/10.33889/IJMEMS.2021.6.1.005.

39. Zhu, X., Boushaba, M., Coit, D. W. & Benyahia, A. "Reliability and importance measures for m-consecutive-k,l-out-of-n system with non-homogeneous Markov-dependent components". *Reliability Engineering and System Safety, Elsevier*. 2017; 167 (C): 1–9, https://www.scopus.com/pages/publications/85019236438. DOI: https://doi.org/10.1016/j.ress.2017.05.023.

40. Özbey, F. "Reliability evaluation of m-consecutive- k, l-out-of-n: F system subjected to shocks". *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability*. 2022; 236 (6): 1135–1146, https://www.scopus.com/pages/publications/85116069008.
DOI: https://doi.org/10.1177/1748006X211048992

41. Nakamura, T., Yamamoto, H. & Akiba, T. "Reliability of a toroidal connected-(r,s)-out-of-(m,n):F lattice system". *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability*. 2022; 236 (2): 329–338, https://www.scopus.com/pages/publications/85077453315. DOI: https://doi.org/10.1177/1748006X19893533.

42. Nakamura, T., Homma, I. & Yamamoto, H. "Birnbaum importance-based simulated annealing algorithm for solving the component assignment problem of linear connected-(r, s)-out-of-(m, n):F lattice systems". *International Journal of System Assurance Engineering and Management*. 2024; 15 (4): 1407–1414, https://www.scopus.com/pages/publications/85144872424.
DOI: https://doi.org/10.1007/s13198-022-01848-2.

43. Lin, C., Cui L. R., Coit D. W. & Lv, M. "Reliability modeling on consecutive-kr-out-of-nr:F linear zigzag structure and circular polygon structure". *IEEE Transactions on Reliability*. 2016; 65 (3): 1509–1521, https://www.scopus.com/pages/publications/84973885922. DOI: https://doi.org/10.1109/TR.2016.2570545.

44. Romankevich, V. A., Morozov, K. V., Romankevich, A. M., Halytsky D. V. & Zhurba A. V. "GL-Models for Analyzing Multiprocessor Systems tolerant to one and two Failures". *Applied Aspects of Information Technology*. 2025; 8 (1): 113–128. DOI: https://doi.org/10.15276/aait.08.2025.9.

45. Romankevich, V. A., Morozov, K. V. & Feseniuk, A. P. "On a method for modifying edge functions of GL-models". *Radioelektronni i Komp'iuterni Systemy*. 2014; 6: 95–99.

46. Romankevich, A. M., Ivanov, V. V. & Romankevich, V. A. "Analysis of fault-tolerant multimodular systems with complex fault distribution based on cyclic GL-models". *Elektronnoe Modelirovanie*. 2004; 26 (5): 67–81.

Conflicts of Interest: The authors declare that they have no conflict of interest regarding this study, including financial, personal, authorship or other, which could influence the research and its results presented in this article

Received 13.10.2025
Received after revision 28.11.2025
Accepted 05.12.2025

DOI: https://doi.org/10.15276/aait.08.2025.30
УДК 004.05

Метод побудови GL-моделей поведінки в потоці відмов складних небазових відмовостійких багатопроцесорних систем

Романкевич Віталій Олексійович[1)]
ORCID: https://orcid.org/0000-0003-4696-5935; zavkaf@scs.kpi.ua. Scopus Author ID: 57193263058
Морозов Костянтин В'ячеславович[1)]
ORCID: https://orcid.org/0000-0003-0978-6292; mcng@ukr.net. Scopus Author ID: 57222509251
Галицький Даніїл Володимирович[1)]
ORCID: https://orcid.org/0009-0004-4421-3443; zipper135401@gmail.com; Scopus Author ID: 58553487600
Єрмоленко Ігор Андрійович[1)]
ORCID: https://orcid.org/0009-0008-5298-4888; yermolenkomail@gmail.com
Захаріудакіс Лефтеріс[2)]
ORCID: https://orcid.org/0000-0002-9658-3073; l.zacharioudakis@nup.ac.cy. Scopus Author ID: 57422876200
[1)] Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського»
пр. Перемоги, 37. Київ, 03056, Україна
[2)] Університет Неаполіс Пафос, Данаіс Авеню, 2. Пафос, 8042, Кіпр

АНОТАЦІЯ

У роботі запропоновано метод побудови GL-моделей поведінки складних небазових відмовостійких багатопроцесорних систем у потоці відмов. Метою дослідження є створення універсального підходу, який уможливлює формування цілісної GL-моделі для систем із кількома незалежними або слабко пов'язаними умовами роботоздатності. Такі моделі можуть бути використані, зокрема, для оцінки параметрів надійності зазначених систем із застосуванням статистичних методів моделювання. Розглядаються системи, роботоздатність яких визначається одночасним виконанням кількох відносно простих умов, для кожної з яких відомі способи побудови GL-моделей (наприклад, умова обмеження кількості відмов серед певної підмножини процесорів). До таких систем належать, зокрема, ієрархічні системи, що складаються з кількох підсистем із власним рівнем відмовостійкості, а також системи, які містять спеціалізовані процесори різних типів. Запропонований метод передбачає попереднє формування допоміжних GL-моделей для кожної з умов роботоздатності, після чого вони поєднуються в єдину модель шляхом послідовного об'єднання їхніх графів через вибрані вершини (об'єднані вершини формують одну, решта вершин і ребра копіюються). Послідовність об'єднання моделей і вибір відповідних вершин можуть задаватися довільно, що забезпечує гнучкість структури побудованої GL-моделі. Продемонстровано приклади застосування методу, у межах яких розглянуто різні варіанти вибору послідовності об'єднання допоміжних моделей і вершин з'єднання їхніх графів, а також використання різних методів для побудови цих моделей. Наукова новизна роботи полягає в узагальненні та формалізації процедури послідовного об'єднання GL-моделей, що дозволяє поєднувати моделі довільної структури та типу для формування єдиної моделі складної системи без втрати коректності її поведінки. Експериментальні результати підтверджують, що, незважаючи на різницю в структурі графів отриманих моделей, їхня поведінка на однакових вхідних векторах повністю збігається й адекватно відображає функціонування відмовостійкої багатопроцесорної системи в потоці відмов. Показано також, що метод не накладає обмежень на способи побудови GL-моделей для окремих умов: можуть поєднуватися моделі різних типів, умови не обов'язково відповідають базовим системам, а графи моделей можуть не бути циклічними. Крім того, подано оцінки складності GL-моделей, побудованих запропонованим способом, зокрема кількості вершин і ребер їхніх графів та загальної складності реберних функцій, залежно від характеристик відповідних допоміжних моделей. Практична цінність полягає в тому, що метод дозволяє автоматизувати побудову комплексних моделей для систем зі складними умовами роботоздатності та використовувати їх для ефективної оцінки надійності реальних багатопроцесорних систем.

Ключові слова: відмовостійкі багатопроцесорні системи; GL-моделі; небазові системи; системи керування; оцінка надійності; статистичні експерименти

ABOUT AUTHORS

Vitaliy A. Romankevich - Doctor of Engineering Sciences, Professor, Head of System Programming and Specialized Computer System Department. National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute", 37, Peremogy Ave. Kyiv, 03056, Ukraine
ORCID: https://orcid.org/0000-0003-4696-5935; zavkaf@scs.kpi.ua. Scopus Author ID: 57193263058
***Research field*:** Fault-tolerant multiprocessor systems reliability estimation; GL-models; Self-diagnosable systems; Diagnosis of multiprocessor systems; Discrete mathematics

Романкевич Віталій Олексійович - доктор технічних наук, професор, завідувач кафедри Системного програмування і спеціалізованих комп'ютерних систем. Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського», пр. Перемоги, 37. Київ, 03056, Україна

Kostiantyn V. Morozov - PhD, Assistant, System Programming and Specialized Computer System Department. National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute", 37, Peremogy Ave. Kyiv, 03056, Ukraine
ORCID: https://orcid.org/0000-0003-0978-6292, mcng@ukr.net. Scopus Author ID: 57222509251
***Research field*:** GL-models; Fault-tolerant multiprocessor systems reliability estimation; Self-diagnosable multiprocessor systems

Морозов Костянтин В'ячеславович – кандидат технічних наук, асистент кафедри Системного програмування і спеціалізованих комп'ютерних систем. Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського», пр. Перемоги, 37. Київ, 03056, Україна

Daniil V. Halytsky - postgraduate student, System Programming and Specialized Computer System Department, National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute", 37, Peremogy Ave. Kyiv, 03056, Ukraine
ORCID: https://orcid.org/0009-0004-4421-3443; zipper135401@gmail.com. Scopus Author ID: 58553487600
***Research field*:** GL-models building and processing optimization, Parallel computing and multi-core system optimization, Algorithm design for distributed and high-performance computing, Graph algorithms and their applications in complex systems, Computational efficiency in large-scale data processing.

Галицький Даніїл Володимирович - аспірант, кафедра Системного програмування і спеціалізованих комп'ютерних систем, Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського», пр. Перемоги, 37. Київ, 03056, Україна

Ihor A. Yermolenko - postgraduate student, System Programming and Specialized Computer System Department. National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute", 37, Peremogy Ave. Kyiv, 03056, Ukraine
ORCID: https://orcid.org/0009-0008-5298-4888; yermolenkomail@gmail.com
***Research field*:** GL-models; Fault-tolerant multiprocessor systems reliability estimation

Єрмоленко Ігор Андрійович - аспірант, кафедра Системного програмування і спеціалізованих комп'ютерних систем. Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського», пр. Перемоги, 37. Київ, 03056, Україна

Lefteris Zacharioudakis - PhD (Eng), Assistant professor, Neapolis University Pafos, 2, Danais Ave., Pafos, 8042, Cyprus
ORCID: https://orcid.org/0000-0002-9658-3073; l.zacharioudakis@nup.ac.cy. Scopus Author ID: 57422876200
***Research field*:** Principles of CyberSecurity, Operating systems, Information Security, Cryptography, Penetration Testing

Захаріудакіс Лефтеріс – кандидат технічних наук, доцент, Університет Неаполіс Пафос, Данаіс Авеню, 2, Пафос, 8042, Кіпр