I. J. Engineering and Manufacturing, 2026, 3, 254-269
Published Online on June 8, 2026 by MECS Press (http://www.mecs-press.org/)
DOI: 10.5815/ijem.2026.03.15

A Generalized Method for Constructing Graph-Logical Models of Non-Basic Fault-Tolerant Multiprocessor Systems via Model Combination

Vitaliy Romankevich
Department of System Programming and Specialized Computer Systems. National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute", 37, Peremogy Ave. Kyiv, 03056, Ukraine
E-mail: zavkaf@scs.kpi.ua
ORCID iD: https://orcid.org/0000-0003-4696-5935

Kostiantyn Morozov*
Department of System Programming and Specialized Computer Systems. National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute", 37, Peremogy Ave. Kyiv, 03056, Ukraine
E-mail: mcng@ukr.net
ORCID iD: https://orcid.org/0000-0003-0978-6292
*Corresponding author

Alexei Romankevich
Department of System Programming and Specialized Computer Systems. National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute", 37, Peremogy Ave. Kyiv, 03056, Ukraine
E-mail: romankev@scs.kpi.ua
ORCID iD: https://orcid.org/0000-0001-5634-8469

Petro Malezhyk
Department of System Programming and Specialized Computer Systems. National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute", 37, Peremogy Ave. Kyiv, 03056, Ukraine
E-mail: malezhyk.petro@lll.kpi.ua
ORCID iD: https://orcid.org/0000-0001-6816-988X

Lefteris Zacharioudakis
Neapolis University Pafos, 2, Danais Ave., Pafos, 8042, Cyprus
E-mail: l.zacharioudakis@nup.ac.cy
ORCID iD: https://orcid.org/0000-0002-9658-3073

Received: 28 August, 2025; Revised: 25 February, 2026; Accepted: 28 March, 2026; Published: 08 June, 2026

Abstract: The paper proposes a generalized method for constructing graph-logical models of the failure behavior of fault-tolerant multiprocessor systems. Such models are used for evaluating system reliability by means of statistical experiments based on simulation of failure behavior. The method is applicable to non-basic systems whose failure behavior cannot be characterized solely by the number of failed components and therefore requires more flexible modeling approaches. The proposed method is based on combining auxiliary models corresponding to different operating conditions of the system into a single model that correctly represents the overall failure behavior. In contrast to existing approaches, it imposes no restrictions on the graph structures of the auxiliary models and does not depend on the specific procedures used for their construction. The key idea of the approach is to integrate such models under a set of mutually exclusive logical conditions, each of which determines the applicability of a particular auxiliary model for a given system state. A set of model transformations is introduced, and it is shown that these transformations preserve model equivalence, that is, correspondence to the same failure behavior of the system. It is demonstrated that these transformations are sufficient to transform auxiliary models to graphs with identical structures, which is a necessary condition for their combination within the proposed framework. Several illustrative examples of the application of the method are provided. The correctness of the constructed models is validated through analysis of representative system states and through exhaustive evaluation over all possible states. The results for the considered example indicate that the proposed method can construct models that accurately represent system failure behavior while yielding more compact

This work is open access and licensed under the Creative Commons CC BY 4.0 License.

graph structures compared to the existing approach. At the same time, comparable logical complexity is maintained. The evaluation is limited to a representative example and exhaustive analysis of system states, and further validation on a broader class of systems remains a direction for future work.

Index Terms: GL-models, fault-tolerant multiprocessor systems, non-basic systems, reliability evaluation, failure behavior, equivalent transformations

1. Introduction

The automation of various processes has become increasingly widespread in recent decades. On the one hand, it reduces the need for humans to perform routine tasks; on the other hand, it minimizes the influence of human error on task performance. One of the key components of automatic and automated systems is the control system (CS) [1]-[5], which receives input data from various sensors and generates control signals for actuators based on a predefined control program [6]-[8]. It is evident that a failure of the CS may lead to incorrect operation of the entire system. Therefore, ensuring the reliable operation of the CS is an important task in the design and development of modern automated systems.

In certain cases, systems are required to meet enhanced reliability requirements. This is particularly relevant for so-called critical application systems (CAS), whose failure may lead to significant material losses, pose a threat to human health or life, or compromise national security [9]-[14]. The control tasks associated with such systems are often complex and require high performance from their control systems. As a result, the CS of CAS may be implemented using fault-tolerant multiprocessor systems (FTMS), which consist of multiple processors and are capable of maintaining correct operation even in the presence of individual processor failures [15]-[20]. Such an architecture makes it possible to achieve both high reliability and high performance [21]-[25].

At a certain stage in the design of a fault-tolerant multiprocessor system, the developer faces the task of evaluating the reliability of the designed system [26]-[30], which is often a nontrivial problem [31]-[36]. Reliability evaluation methods for FTMS can generally be divided into two categories. The first category involves constructing analytical expressions for calculating reliability metrics [37]-[42]. These methods typically provide high accuracy but lack generality, since a separate calculation method often has to be developed for each specific type of system [43]-[52]. The second category is based on statistical experiments using models that describe FTMS failure behavior [53], [54]. In this case, the accuracy of reliability estimates generally depends on the number of experiments performed. The main advantage of these methods lies in their universality: it is sufficient to construct a model that describes the failure behavior of the system. However, the construction of such models for complex systems remains a challenging task.

The proposed method contributes to the modeling of complex system failure behavior under uncertainty, particularly in the context of reliability evaluation of fault-tolerant multiprocessor systems.

2. Background

Graph-logical models (GL-models) [54], which combine graph structures with Boolean functions, provide a convenient representation of system failure behavior. A GL-model is based on an undirected graph in which each edge is associated with a Boolean edge function. The arguments of these functions are elements of the system state vector $\mathbf{v}$ – a Boolean vector in which each element corresponds to the state of a particular processor in the system. An element takes the value 1 if the corresponding processor is operational and 0 if it has failed [55]-[57]. If an edge function evaluates to 0, the corresponding edge is removed from the graph. The connectivity of the resulting graph reflects the operational state of the system: a connected graph corresponds to a functioning system, whereas a disconnected graph indicates a system failure. Thus, the GL-model makes it possible to determine the system state under a given failure configuration by analyzing the connectivity of the resulting graph.

A single FTMS may correspond to multiple distinct GL-models constructed using different methods [54], [58]-[63]. These models may differ in their graph structures and/or in the expressions of their edge functions while still representing identical system failure behavior. Existing methods typically enable the construction of models for so-called basic systems, i.e., systems that remain operational under any failure scenario in which the multiplicity of failures does not exceed a specified threshold value. A basic FTMS consisting of n processors and tolerant to at most m failures and its corresponding GL-model are both usually denoted by $K(m, n)$.

The basic systems described above are also known as k-out-of-n systems [64]-[66]. It can be shown that a k-out-of-n:G system, which consists of n components and remains operational as long as at least k of them are operational [67], is equivalent to the system $K(n - k, n)$. This equivalence follows directly from the definition of failure multiplicity. Similarly, a k-out-of-n:F system, which consists of n components and fails when at least k of them fail [68], is equivalent to the system $K(k - 1, n)$. Thus, a direct correspondence is established between the classical k-out-of-n reliability models and the $K(m, n)$ representation used in this work.

GL-models of basic FTMS can be constructed in various ways. For example, one of the known approaches [63]

makes it possible to construct models of systems of the form $K(2, n)$ based on a cycle graph with n edges. The edge functions of such a model are given by:

$$f_i(\mathbf{v}) = x_i \vee \bigwedge_{j=1}^{\left\lfloor \frac{n-1}{2} \right\rfloor} x_{(i+j) \bmod n}, \tag{1}$$

where $\mathbf{v} = \langle x_1, x_2, \ldots, x_n \rangle$ is the Boolean state vector of the system, and x_i is the element corresponding to the state of the i-th processor, $i = 1, 2, \ldots, n$.

Another known approach [63] allows constructing GL-models of systems of the form $K(m, n)$ for any values $1 \le m \le n$. A model constructed according to this approach is based on a cycle graph with $n - m + 1$ edges. For state vectors containing $m + 1$ zeros, such models always lose the minimum number of edges required to break the connectivity of the cycle graph, namely, two edges. Therefore, such models are referred to as MLE-models (minimum lost edges).

The edge functions of such models are constructed according to the following recursive scheme.

For the case $n > 1$, the input state vector is divided into two parts of sizes n_1 and n_2, where $n_1 + n_2 = n$. In general, the values of n_1 and n_2 can be chosen arbitrarily; however, in practice, it is reasonable to split the vector into equal or nearly equal parts. The set of edge functions is then constructed as the union of the following three sets:

1. The set of edge functions of an auxiliary MLE-model $K_1(m, n_1)$, constructed for the first part of the state vector, in the case $m \le n_1$; otherwise, this set is empty.
2. The set of edge functions of the form

$$f(\mathbf{v}) = \bigwedge_{f^{(1)}(\mathbf{v}) \in \Phi_1(m-i, n_1)} f^{(1)}(\mathbf{v}) \vee \bigwedge_{f^{(2)}(\mathbf{v}) \in \Phi_2(i, n_2)} f^{(2)}(\mathbf{v}),$$

   for $i = a, a + 1, \ldots, b$, where $a = \max(1, m - n_1)$, $b = \min(n_2, m - 1)$; $\Phi_1(m - i, n_1)$ is the set of edge functions of the auxiliary MLE-model $K_1(m - i, n_1)$, and $\Phi_2(i, n_2)$ is the set of edge functions of the auxiliary MLE-model $K_2(i, n_2)$. The bounds on i follow from the feasibility conditions of the corresponding auxiliary models, namely $1 \le m - I \le n_1$ and $1 \le i \le n_2$.
3. The set of edge functions of an auxiliary MLE-model $K_2(m, n_2)$, constructed for the second part of the state vector, in the case $m \le n_2$; otherwise, this set is empty.

For the case $n = 1$, the model $K(1, 1)$ contains a single edge function $f(\mathbf{v}) = x_i$, where x_i is the only component of the corresponding subvector. This case serves as the base of the recursion.

It can be shown that an MLE-model of the form $K(1, n)$ contains n edge functions

$$f_i(\mathbf{v}) = x_i, i = 1,2, \ldots, n,$$

while an MLE-model of the form $K(n, n)$ contains a single edge function

$$f(\mathbf{v}) = x_1 \vee x_2 \vee \cdots \vee x_n.$$

These models can be conveniently used as additional stopping conditions of the recursion, simplifying the construction procedure.

In practice, fault-tolerant control systems are often non-basic. Such systems may remain operational under some failure configurations of a given multiplicity, while failing under other configurations of the same multiplicity. In other words, their operability depends not only on the failure multiplicity but also on the structure of failures.

Examples of such systems include consecutive-k-out-of-n systems [38], [40]-[43], [62], consecutive-k-within-m-out-of-n [45], [46], consecutive-k-out-of-r-from-n [47], m-consecutive-k-out-of-n [39], [48], [49], m-consecutive-k,l-out-of-n [51], (n, f, k) [50], [62], and $\langle n, f, k \rangle$ [52], as well as more complex systems with interacting subsystems [58]. The analysis and modeling of such systems are significantly more complex compared to basic systems.

GL-models of non-basic systems are generally obtained by modifying basic models. Such modifications may involve changes in the graph structure, the expressions of the edge functions, or both. As a result of these modifications, the behavior of the resulting model differs from that of the corresponding basic model for a certain set of so-called blocked system state vectors [59]-[61].

In [60], a method for constructing GL-models of non-basic FTMS by combining several auxiliary basic models was proposed. It is assumed that the system operates under a set of conditions specified by corresponding logical expressions. Under each condition, the system behaves as a certain basic system, i.e., it remains tolerant to failures whose multiplicity does not exceed a given threshold value, which may differ for different conditions. However, the method proposed in [60]

imposes a structural restriction on the auxiliary GL-models: their graphs must be cycle graphs, which significantly limits the class of systems whose failure behavior can be represented using this approach.

In [61], this restriction was removed by proposing a method that allows combining GL-models with arbitrary graph structures. In this approach, the graphs of the corresponding models are combined into a single graph through vertex merging. Specifically, this is achieved by successively selecting one vertex from each of the graphs being combined and merging them into a single vertex in the resulting graph. Although this approach eliminates the structural restriction of [60], the resulting graphs may become rather large, which leads to GL-models with increased structural complexity. Therefore, the problem of constructing GL-models with a compact graph structure remains an open problem.

The objective of this work is to develop a generalized method for constructing GL-models of non-basic FTMS that combines the advantages of the approaches proposed in [60] and [61] while avoiding their limitations, without necessarily increasing logical complexity, as well as to formalize the underlying principles of model combination and transformation. The proposed method removes the requirement that auxiliary GL-models must be constructed on cycle graphs and simultaneously provides greater flexibility in combining them. In particular, it allows merging multiple vertices and edges when integrating the corresponding graphs, rather than restricting the process to a single vertex from each graph.

The scientific novelty of the proposed approach lies not in introducing a new problem domain, but in a generalized method for combining auxiliary GL-models with arbitrary graph structures, enabling flexible graph transformations and integration under mutually exclusive logical conditions. As a result, the constructed GL-model correctly reproduces the failure behavior of the FTMS under all considered conditions while providing a more compact and structurally efficient representation and maintaining comparable logical complexity. In contrast to existing approaches, the proposed method generalizes the model combination procedure itself, rather than focusing on specific classes of systems or graph structures.

3. Method for Constructing GL-Models of Non-Basic Systems

Let the failure behavior of the FTMS correspond to GL-models $M_1, M_2,\ldots, M_k$ under a set of conditions $C_1, C_2,\ldots, C_k$. Furthermore, each condition $C_1, C_2,\ldots, C_k$ corresponds to a logical expression $c_1(\mathbf{v}), c_2(\mathbf{v}),\ldots, c_k(\mathbf{v})$, where $\mathbf{v}$ is the Boolean state vector of the system. It should be noted that exactly one of the conditions $C_1, C_2,\ldots, C_k$ holds for any given system state; that is, for any vector $\mathbf{v}$, only one of the logical expressions $c_1(\mathbf{v}), c_2(\mathbf{v}),\ldots, c_k(\mathbf{v})$ is equal to 1, while all others are equal to 0. If this is not the case, the conditions and the corresponding expressions should be modified accordingly, for instance as described in [60].

First, consider the case in which all auxiliary models $M_1, M_2,\ldots, M_k$ are based on the same graph G. Let the edge functions of the models be denoted as f_j^i, where i is the model index and j is the edge function index. It is assumed that edge functions with the same index correspond to the same edges of graph G (i.e., a consistent edge indexing is assumed). The following proposition defines the edge functions of the resulting GL-model.

Proposition 1. The failure behavior of the considered FTMS corresponds to a GL-model constructed on graph G, whose edge functions f_j are expressed as follows:

$$f_j(\mathbf{v}) = \bigvee_{i=1}^{k} f_j^i(\mathbf{v}) c_i(\mathbf{v}). \tag{2}$$

Proof. Consider an arbitrary system state vector $\mathbf{v}$. As stated above, exactly one of the expressions $c_1(\mathbf{v}), c_2(\mathbf{v}),\ldots, c_k(\mathbf{v})$ is equal to 1. Let this expression have the index $t_\mathbf{v}$, i.e., condition $C_{t_\mathbf{v}}$ holds for vector $\mathbf{v}$, and the failure behavior of the FTMS corresponds to model $M_{t_\mathbf{v}}$. Note that

$$c_i(\mathbf{v}) = \begin{cases} 1, \text{if } i = t_\mathbf{v} \\ 0, \text{if } i \neq t_\mathbf{v} \end{cases},$$

and therefore

$$f_j^i(\mathbf{v}) c_i(\mathbf{v}) = \begin{cases} f_j^i(\mathbf{v}), \text{if } i = t_\mathbf{v} \\ 0, \text{if } i \neq t_\mathbf{v} \end{cases}.$$

Thus, equation (2) reduces to

$$f_j(\mathbf{v}) = f_j^{t_\mathbf{v}}(\mathbf{v}).$$

Hence, the values of the edge functions of the constructed GL-model at vector $\mathbf{v}$ coincide with those of the corresponding edge functions of model $M_{t_\mathbf{v}}$, and the graphs of the models also coincide. This holds for any vector $\mathbf{v}$; therefore, the constructed GL-model correctly represents the failure behavior of the considered FTMS. ■

In the general case, the auxiliary models $M_1, M_2,\ldots, M_k$ may be based on different graphs $G_1, G_2,\ldots, G_k$. To solve the

problem of constructing a GL-model for such an FTMS, it is sufficient to transform the auxiliary models so that their graphs become identical. Afterward, formula (2) can be applied to obtain the edge functions of the constructed model. Such transformations can be performed as described below.

4. Transformations of Auxiliary GL-Models

As previously mentioned, the failure behavior of an FTMS can be represented by different GL-models. Models that correspond to the same system will be referred to as *equivalent* models. In other words, equivalent models produce identical connectivity results for any given system state vector: their graphs either remain connected simultaneously or become disconnected simultaneously. Transformations of GL-models that preserve their equivalence will be called *equivalent transformations*.

The following transformations of GL-models are considered, and it is shown that each of them preserves model equivalence.

Proposition 2. If an additional edge with an edge function of the form $f(\mathbf{v}) = 0$ is added between any pair of vertices in a GL-model (Fig. 1a), the model remains equivalent to the original GL-model.

Proof. Indeed, since all edges whose edge functions evaluate to zero are excluded from the model during the connectivity evaluation, the additional edge does not affect the connectivity of the graph in any way. Therefore, for any system state vector, the original and the modified models produce identical connectivity results.

Proposition 3. If a vertex of a GL-model is split into two vertices connected by an edge with an edge function of the form $f(\mathbf{v}) = 1$, and each of the edges originally incident to the original vertex is assigned to one of the two new vertices (e.g., as shown in Fig. 1b), the resulting model remains equivalent to the original GL-model.

Proof. Let vertex β be split, resulting in two new vertices β_1 and β_2. If, for a given system state vector, there exists a path from some vertex σ to β in the original model, then a corresponding path exists from σ to either β_1 or β_2 in the modified model, i.e., they belong to the same connected component. Conversely, if no such path exists in the original model, then neither β_1 nor β_2 is reachable from σ in the modified model. Furthermore, since β_1 and β_2 are connected by an edge with the edge function $f(\mathbf{v}) = 1$, this edge is never removed during the connectivity evaluation. Therefore, β_1 and β_2 will always belong to the same connected component. Consequently, for any system state vector, the graph of the modified model is connected if and only if the graph of the original model is connected.

Proposition 4. If an additional vertex is added to a GL-model and connected to any vertex of the model with an edge whose edge function is $f(\mathbf{v}) = 1$ (see Fig. 1c), the resulting model remains equivalent to the original GL-model.

Proposition 5. If any edge in a GL-model associated with an edge function $f(\mathbf{v})$ is split into two edges by inserting an intermediate vertex, and the two resulting edges are assigned edge functions $f_1(\mathbf{v}) = f(\mathbf{v})$ and $f_2(\mathbf{v}) = 1$, respectively (see Fig. 1d), the model remains equivalent to the original GL-model.

Propositions 4 and 5 follow directly from Proposition 3 and therefore are stated without detailed proofs.

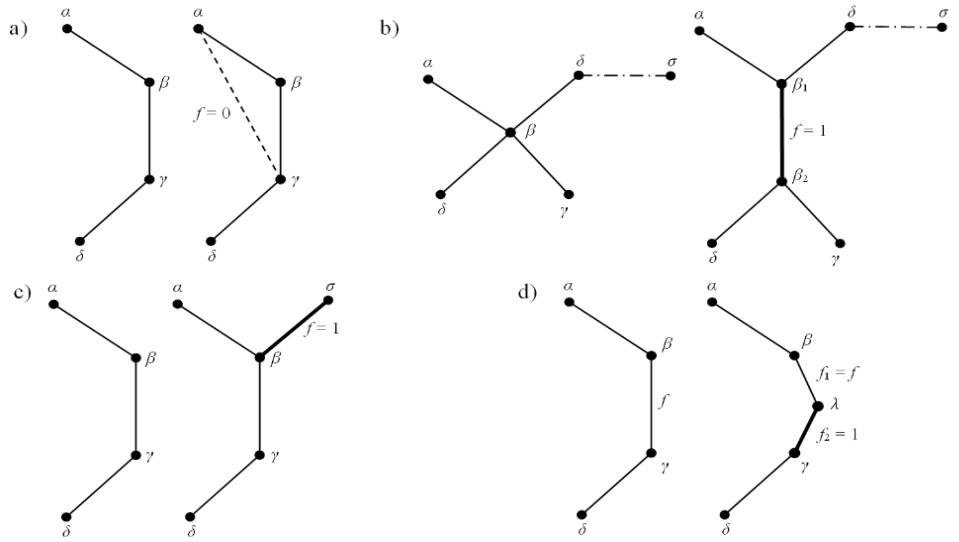


Fig. 1. Examples of equivalent transformations: (a) adding an edge $\alpha\gamma$ with edge function $f = 0$; (b) splitting a vertex β into two vertices β_1 and β_2 connected by an edge $\beta_1\beta_2$ with edge function $f = 1$; (c) adding a vertex σ and connecting it to vertex β by an edge $\beta\sigma$ with edge function $f = 1$; (d) splitting an edge $\beta\gamma$ with edge function f by introducing a vertex λ, resulting in edges $\beta\lambda$ and $\lambda\gamma$ with edge functions $f_1 = f$ and $f_2 = 1$, respectively

Proposition 6. An arbitrary set of GL-models can be transformed by means of equivalent transformations so that all resulting models are based on identical graphs.

Proof. It is shown that the equivalent transformations of GL-models – namely, "*edge addition*" (*Proposition 2*) and

"*vertex addition*" (*Proposition 4*) – are sufficient to transform any two models so that their underlying graphs become identical.

Let us consider two GL-models, M_1 and M_2, based on graphs G_1 and G_2, respectively. Select an arbitrary pair of vertices $\alpha_1 \in G_1$ and $\alpha_2 \in G_2$. Let $s_{\alpha_2}(G_2)$ denote a spanning tree of graph G_2 rooted at α_2. By applying the "*vertex addition*" transformation, we can construct an analogous tree in G_1 starting from α_1. This construction reproduces the structure of the spanning tree $s_{\alpha_2}(G_2)$ in graph G_1, rooted at α_1. Next, by applying the "*edge addition*" transformation, we add the remaining edges of graph G_2.

After that, consider $s_{\alpha_1}(G_1)$ – a spanning tree of graph G_1 rooted at α_1. Using the same approach, we apply the "*vertex addition*" transformation to augment the graph G_2 of model M_2 by constructing the tree $s_{\alpha_1}(G_1)$ starting from α_2, and then complete the transformation by adding the remaining edges of G_1. The resulting graphs of the transformed models are therefore structurally identical (as illustrated in Fig. 2).

These transformations can be applied sequentially to any number of GL-models, ensuring that all resulting models are based on an identical graph.

It is important to note that the above-described algorithm was introduced primarily to demonstrate the possibility of bringing auxiliary GL-models to a common graph (i.e., to prove *Proposition 6*), rather than as a practical procedure for modifying the models. Although it may also be applied when necessary, the algorithm is not always optimal and may lead to a significant increase in graph complexity.

In practice, however, GL-models can often be modified using alternative sequences of equivalent transformations, resulting in a smaller increase in graph complexity. These transformations form the basis of the formal algorithm for constructing GL-models of non-basic FTMS presented in the following section.

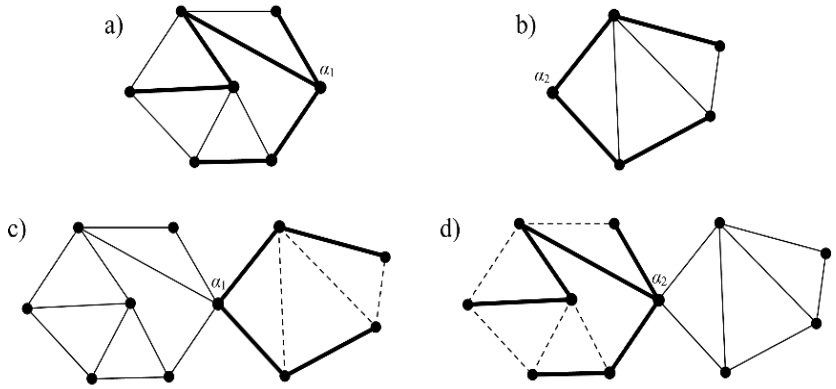


Fig. 2. Graph G_1 of model M_1 and its spanning tree $s_{\alpha_1}(G_1)$, shown in bold (a); graph G_2 of model M_2 and its spanning tree $s_{\alpha_2}(G_2)$, shown in bold (b); graphs of the transformed models M_1 (c) and M_2 (d)

5. Algorithm of the Proposed Method

We formulate an algorithm for constructing a GL-model of a fault-tolerant multiprocessor system whose failure behavior, under a set of conditions C_1, C_2,…, C_k, corresponds to given GL-models M_1, M_2,…, M_k. The algorithm consists of the following steps:

1. Transform the conditions C_1, C_2, …, C_k, if necessary, so that exactly one of them holds for any given system state.
2. Construct the logical expressions $c_1(\mathbf{v})$, $c_2(\mathbf{v})$,…, $c_k(\mathbf{v})$ corresponding to the conditions C_1, C_2,…, C_k.
3. Using the equivalent transformations described in this article, transform the models $M_1, M_2, \dots, M_k$ into models $M_1', M_2', \dots, M_k'$ such that all transformed models are based on the same graph G.
4. Construct the edge functions of the resulting GL-model according to formula (2), using the edge functions of the auxiliary models $M_1', M_2', \dots, M_k'$.

The resulting GL-model, defined on graph G with the edge functions obtained in Step 4, represents the failure behavior of the system.

6. Examples

To illustrate the proposed method, we consider a representative example of a fault-tolerant multiprocessor system consisting of 8 processors. The system is resilient to any failures whose multiplicity does not exceed 2, as well as to 22 failure configurations of multiplicity 3 corresponding to the following set of system state vectors: B = {00111011,

00111101, 01010111, 01011101, 01011110, 01100111, 01101110, 01111010, 01111100, 10011011, 10011101, 10101011, 10110110, 10111010, 10111100, 11000111, 11001011, 11001101, 11001110, 11011001, 11011100, 11100110}.

First, we construct the basic MLE-model $K_1(2, 8)$ [63], which is based on a cycle graph with 8 – 2 + 1 = 7 edges. Next, we modify this model by adding two additional internal edges, as shown in Fig. 3a. The resulting modified model, denoted as K_1', has the following edge functions:

$f_1^1 = x_1 \vee x_2;$
$f_2^1 = x_1x_2 \vee x_3x_4;$
$f_3^1 = x_3 \vee x_4;$
$f_4^1 = x_1x_2x_3x_4 \vee x_5x_6x_7x_8;$
$f_5^1 = x_5 \vee x_6;$
$f_6^1 = x_5x_6 \vee x_7x_8;$
$f_7^1 = x_7 \vee x_8;$
$f_8^1 = x_1x_3x_6x_7;$
$f_9^1 = x_2x_6x_7.$

As a result of this modification, the model behavior differs from that of the basic model $K(2, 8)$ for the following eight system state vectors containing three zeros: B_1 = {01010111, 01011110, 01100111, 01101110, 10110110, 11000111, 11001110, 11100110}.

Next, we construct a GL-model $K_2(2, 8)$ of a different type [63], defined on a cycle graph with 8 edges, whose edge functions are computed according to (1). We then modify this model by adding two additional internal edges, as shown in Fig. 3b. The resulting modified model, denoted as K_2', has the following edge functions:

$f_1^2 = x_1 \vee x_2x_3x_4;$
$f_2^2 = x_2 \vee x_3x_4x_5;$
$f_3^2 = x_3 \vee x_4x_5x_6;$
$f_4^2 = x_4 \vee x_5x_6x_7;$
$f_5^2 = x_5 \vee x_6x_7x_8;$
$f_6^2 = x_6 \vee x_7x_8x_1;$
$f_7^2 = x_7 \vee x_8x_1x_2;$
$f_8^2 = x_8 \vee x_1x_2x_3;$
$f_9^2 = x_4x_5;$
$f_{10}^2 = x_1x_5x_8.$

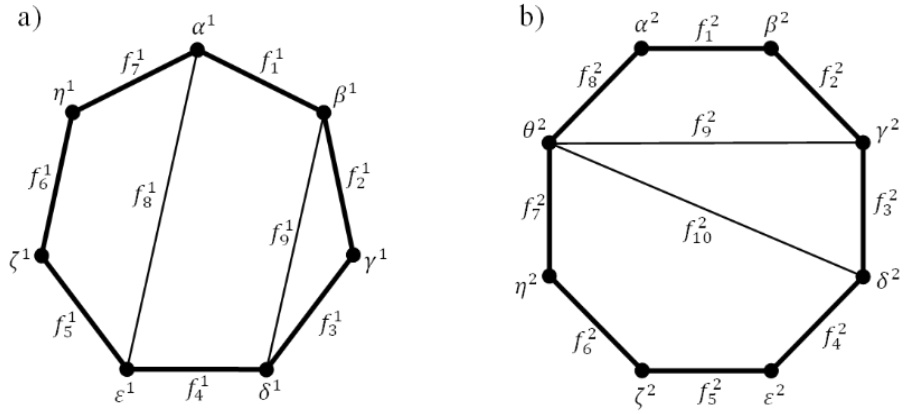


Fig. 3. Auxiliary GL-models K_1' (a) and K_2' (b); edges of the corresponding cycle graphs of the basic GL-models are shown in bold

As a result of this modification, the following fourteen system state vectors containing three zeros become blocked: B_2 = {00111011, 00111101, 01011101, 01111010, 01111100, 10011011, 10011101, 10101011, 10111010, 10111100, 11001011, 11001101, 11011001, 11011100}.

It can be observed that $B = B_1 \cup B_2$. Moreover, for all vectors in the set B_1, the 6th and 7th elements are equal to 1, whereas for no vector in B_2 are these two elements simultaneously equal to 1. Thus, when the 6th and 7th processors are operational (corresponding to the logical expression $c(\mathbf{v}) = x_6x_7$), the failure behavior of the system corresponds to the model K_1'; in all other cases, it corresponds to the model K_2'.

We transform the GL-models K_1' and K_2' so that they are based on identical graphs. To this end, in the model K_1', according to *Proposition 3*, we split the vertex α^1 into two vertices α_1^1 and α_2^1 connected by an edge $\alpha_1^1\alpha_2^1$ with the edge

function $f_{10}^1(\mathbf{v}) = 1$. In addition, according to *Proposition 2*, we add two edges $\alpha_2^1\gamma^1$ and $\alpha_2^1\delta^1$ with edge functions $f_{11}^1(\mathbf{v}) = 0$ and $f_{12}^1(\mathbf{v}) = 0$, respectively, thereby obtaining the model K_1'' (Fig. 4a). Similarly, in the model K_2', according to *Proposition 2*, we add two edges $\beta^2\delta^2$ and $\alpha^2\varepsilon^2$ with edge functions $f_{11}^2(\mathbf{v}) = 0$ and $f_{12}^2(\mathbf{v}) = 0$, resulting in the model K_2'' (Fig. 4b). Finally, using equation (2), we construct the edge functions of the resulting GL-model K_{12} (Fig. 4c) for the considered system, taking K_1'' and K_2'' as auxiliary models and using the logical expressions

$c_1(\mathbf{v}) = c(\mathbf{v}) = x_6x_7;$
$c_2(\mathbf{v}) = \bar{c}(\mathbf{v}) = \overline{x_6x_7} = \bar{x}_6 \vee \bar{x}_7.$

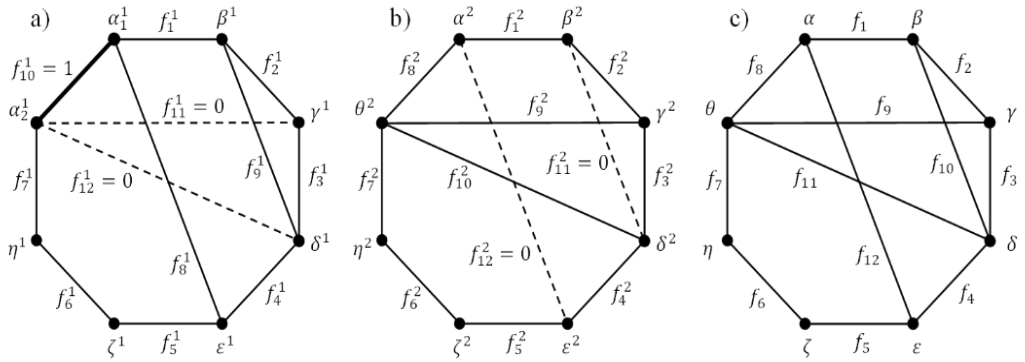


Fig. 4. GL-models K_1'' (a) and K_2'' (b), obtained by transforming models K_1' and K_2', respectively; combined model K_{12} (c) obtained by merging them

The corresponding edge function expressions are given as follows:

$f_1 = f_1^1c \vee f_1^2\bar{c} = (x_1 \vee x_2)x_6x_7 \vee (x_1 \vee x_2x_3x_4)(\bar{x}_6 \vee \bar{x}_7) = x_1 \vee x_2(x_3x_4 \vee x_6x_7);$
$f_2 = f_2^1c \vee f_2^2\bar{c} = (x_1x_2 \vee x_3x_4)x_6x_7 \vee (x_2 \vee x_3x_4x_5)(\bar{x}_6 \vee \bar{x}_7) = x_2(x_1 \vee \bar{x}_6 \vee \bar{x}_7) \vee x_3x_4(x_5 \vee x_6x_7);$
$f_3 = f_3^1c \vee f_3^2\bar{c} = (x_3 \vee x_4)x_6x_7 \vee (x_3 \vee x_4x_5x_6)(\bar{x}_6 \vee \bar{x}_7) = x_3 \vee x_4x_6(x_5 \vee x_7);$
$f_4 = f_4^1c \vee f_4^2\bar{c} = (x_1x_2x_3x_4 \vee x_5x_6x_7x_8)x_6x_7 \vee (x_4 \vee x_5x_6x_7)(\bar{x}_6 \vee \bar{x}_7) = x_4(x_1x_2x_3 \vee \bar{x}_6 \vee \bar{x}_7) \vee x_5x_6x_7x_8;$
$f_5 = f_5^1c \vee f_5^2\bar{c} = (x_5 \vee x_6)x_6x_7 \vee (x_5 \vee x_6x_7x_8)(\bar{x}_6 \vee \bar{x}_7) = x_5 \vee x_6x_7;$
$f_6 = f_6^1c \vee f_6^2\bar{c} = (x_5x_6 \vee x_7x_8)x_6x_7 \vee (x_6 \vee x_7x_8x_1)(\bar{x}_6 \vee \bar{x}_7) = x_1x_7x_8 \vee x_6(x_5 \vee \bar{x}_7 \vee x_8);$
$f_7 = f_7^1c \vee f_7^2\bar{c} = (x_7 \vee x_8)x_6x_7 \vee (x_7 \vee x_8x_1x_2)(\bar{x}_6 \vee \bar{x}_7) = x_7 \vee x_8(x_1x_2 \vee x_6x_7);$
$f_8 = f_{10}^1c \vee f_8^2\bar{c} = c \vee f_8^2\bar{c} = c \vee f_8^2 = x_1x_2x_3 \vee x_6x_7 \vee x_8;$
$f_9 = f_{11}^1c \vee f_9^2\bar{c} = f_9^2\bar{c} = x_4x_5(\bar{x}_6 \vee \bar{x}_7);$
$f_{10} = f_9^1c \vee f_{11}^2\bar{c} = f_9^1c = (x_2x_6x_7)(x_6x_7) = x_2x_6x_7\ ;$
$f_{11} = f_{12}^1c \vee f_{10}^2\bar{c} = f_{10}^2\bar{c} = x_1x_5x_8(\bar{x}_6 \vee \bar{x}_7);$
$f_{12} = f_8^1c \vee f_{12}^2\bar{c} = f_8^1c = (x_1x_3x_6x_7)(x_6x_7) = x_1x_3x_6x_7.$

Other transformations of the models K_1' and K_2' are also possible. For example, in model K_1', we split the vertex α^1 into two vertices α_1^1 and α_2^1 connected by an edge $\alpha_1^1\alpha_2^1$ with an edge function $f_{10}^1(\mathbf{v}) = 1$, and add a single additional edge $\alpha_1^1\delta^1$ with an edge function $f_{11}^1(\mathbf{v}) = 0$, resulting in model K_1''' (Fig. 5a). Similarly, in model K_2', we add a single additional edge $\alpha^2\gamma^2$ with an edge function $f_{11}^2(\mathbf{v}) = 0$, resulting in model K_2''' (Fig. 5b). The edge functions of the resulting GL-model K_{12}' (Fig. 5c), constructed according to equation (2), are given as follows:

$f_1' = f_2^1c \vee f_1^2\bar{c} = (x_1x_2 \vee x_3x_4)x_6x_7 \vee (x_1 \vee x_2x_3x_4)(\bar{x}_6 \vee \bar{x}_7) = x_1(x_2 \vee \bar{x}_6 \vee \bar{x}_7) \vee x_3x_4(x_2 \vee x_6x_7);$
$f_2' = f_3^1c \vee f_2^2\bar{c} = (x_3 \vee x_4)x_6x_7 \vee (x_2 \vee x_3x_4x_5)(\bar{x}_6 \vee \bar{x}_7) = x_2(\bar{x}_6 \vee \bar{x}_7) \vee x_3x_4x_5 \vee x_6x_7(x_3 \vee x_4);$
$f_3' = f_4^1c \vee f_3^2\bar{c} = (x_1x_2x_3x_4 \vee x_5x_6x_7x_8)x_6x_7 \vee (x_3 \vee x_4x_5x_6)(\bar{x}_6 \vee \bar{x}_7) =$
$= x_3(x_1x_2x_4 \vee \bar{x}_6 \vee \bar{x}_7) \vee x_5x_6(x_4\bar{x}_7 \vee x_7x_8);$
$f_4' = f_5^1c \vee f_4^2\bar{c} = (x_5 \vee x_6)x_6x_7 \vee (x_4 \vee x_5x_6x_7)(\bar{x}_6 \vee \bar{x}_7) = x_4 \vee x_6x_7;$
$f_5' = f_6^1c \vee f_5^2\bar{c} = (x_5x_6 \vee x_7x_8)x_6x_7 \vee (x_5 \vee x_6x_7x_8)(\bar{x}_6 \vee \bar{x}_7) = x_5 \vee x_6x_7x_8;$
$f_6' = f_7^1c \vee f_6^2\bar{c} = (x_7 \vee x_8)x_6x_7 \vee (x_6 \vee x_7x_8x_1)(\bar{x}_6 \vee \bar{x}_7) = x_1x_7x_8 \vee x_6;$
$f_7' = f_{10}^1c \vee f_7^2\bar{c} = c \vee f_7^2\bar{c} = c \vee f_7^2 = x_6x_7 \vee x_7 \vee x_8x_1x_2 = x_1x_2x_8 \vee x_7;$
$f_8' = f_1^1c \vee f_8^2\bar{c} = (x_1 \vee x_2)x_6x_7 \vee (x_8 \vee x_1x_2x_3)(\bar{x}_6 \vee \bar{x}_7) = x_1x_2x_3 \vee x_6x_7(x_1 \vee x_2) \vee x_8(\bar{x}_6 \vee \bar{x}_7);$
$f_9' = f_9^1c \vee f_{11}^2\bar{c} = f_9^1c = (x_2x_6x_7)(x_6x_7) = x_2x_6x_7;$
$f_{10}' = f_{11}^1c \vee f_9^2\bar{c} = f_9^2\bar{c} = x_4x_5(\bar{x}_6 \vee \bar{x}_7);$
$f_{11}' = f_8^1c \vee f_{10}^2\bar{c} = (x_1x_3x_6x_7)x_6x_7 \vee (x_1x_5x_8)(\bar{x}_6 \vee \bar{x}_7) = x_1\big(x_3x_6x_7 \vee x_5x_8(\bar{x}_6 \vee \bar{x}_7)\big).$

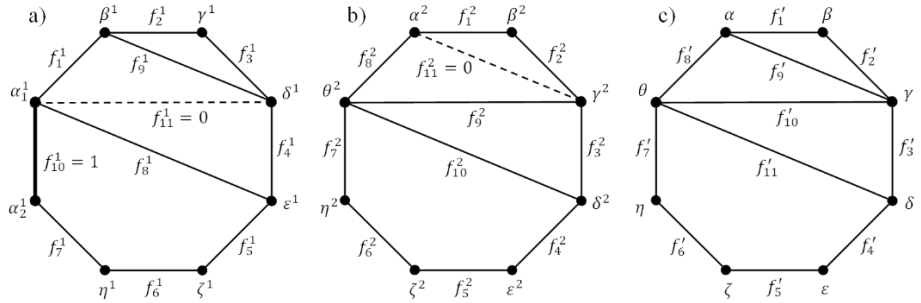


Fig. 5. GL-models K_1''' (a) and K_2''' (b), obtained by transforming models K_1' and K_2'; combined model K_{12}' (c) obtained by merging them

It is worth noting that the model K_{12} is based on a graph with 12 edges, whereas the model K_{12}' is based on a graph with only 11 edges. Nevertheless, both models represent identical failure behavior of the same FTMS. This demonstrates that structurally different GL-models may be equivalent in terms of failure behavior.

Experimental results show that the set of blocked vectors for both the K_{12} and K_{12}' models coincides with the set B.

Consider the vector **v** = 10101111. This vector corresponds to the failure of two processors. The system remains operational in this case. For this vector, in the K_{12} model, the functions f_2, f_9, f_{10}, and f_{11} evaluate to zero, and the corresponding edges are removed from the graph. The graph remains connected (Fig. 6a), as the removed edges do not partition the graph into multiple connected components, which corresponds to the operational state of the system. In the K_{12}' model, the functions f_1', f_9', and f_{10}' evaluate to zero, and the graph also remains connected (Fig. 7a).

Next, consider the vector **v** = 11100110. This vector corresponds to the failure of three processors but belongs to the set B (specifically, subset B_1), and therefore the system remains operational. In the K_{12} model, the functions f_4, f_6, f_9, and f_{11} evaluate to zero, and the graph remains connected (Fig. 6b). In the K_{12}' model, the functions f_3', f_5', and f_{10}' evaluate to zero, and the graph also remains connected (Fig. 7b).

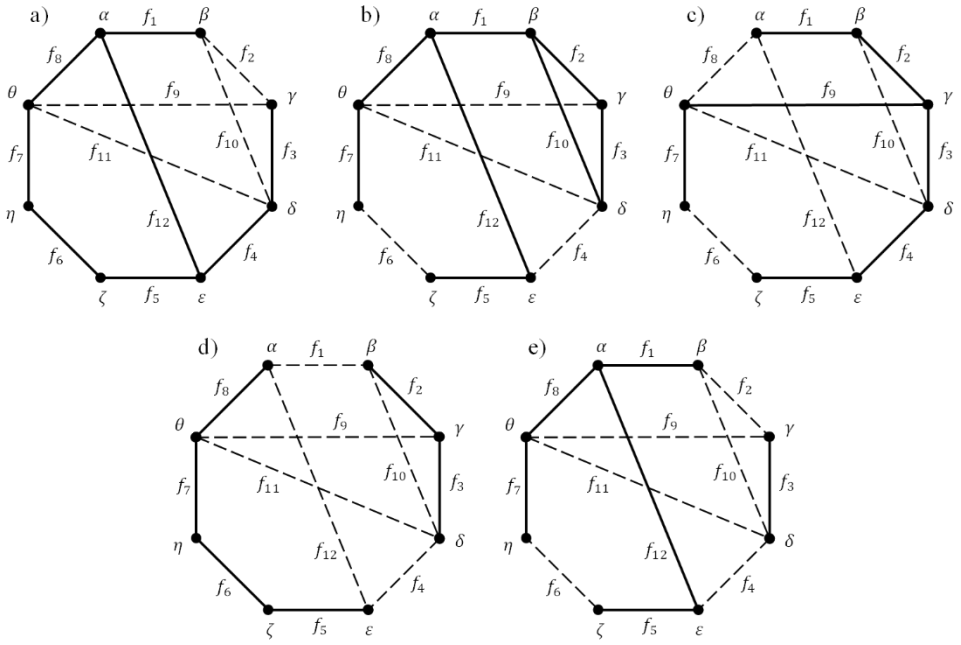


Fig. 6. Graphs of the GL-model K_{12} for vectors 10101111 (a), 11100110 (b), 01111010 (c), 00110111 (d), and 10100110 (e)

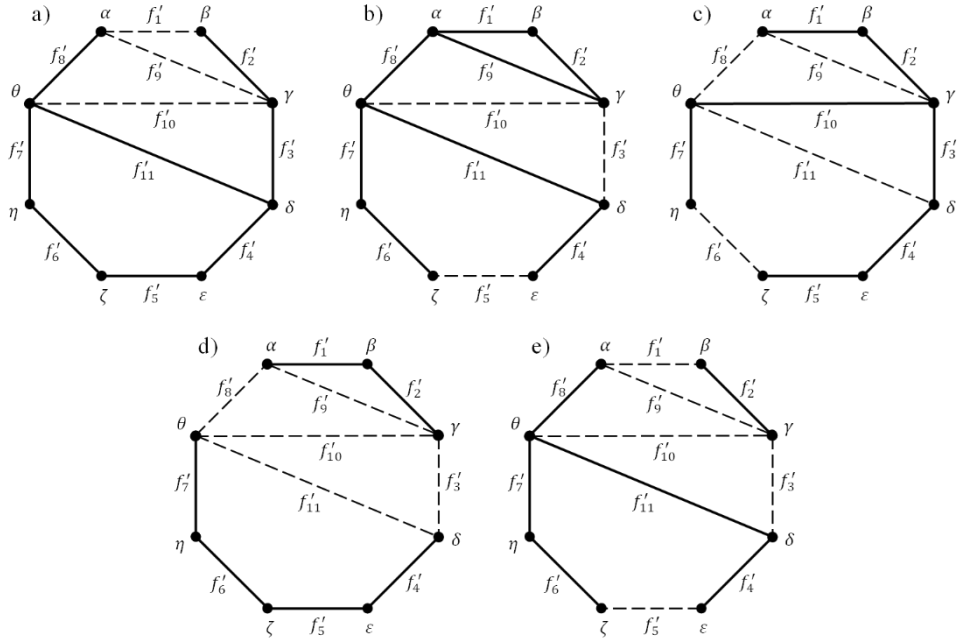


Fig. 7. Graphs of the GL-model K'_{12} for vectors 10101111 (a), 11100110 (b), 01111010 (c), 00110111 (d), and 10100110 (e)

Now consider the vector **v** = 01111010. This vector also corresponds to the failure of three processors and belongs to the set *B* (specifically, subset B_2), and thus the system remains operational. In the K_{12} model, the functions f_6, f_8, f_{10}, f_{11}, and f_{12} evaluate to zero, and the graph remains connected (Fig. 6c). In the K'_{12} model, the functions f'_6, f'_8, f'_9, and f'_{11} evaluate to zero, and the graph also remains connected (Fig. 7c).

Consider the vector **v** = 00110111. This vector corresponds to the failure of three processors but does not belong to the set *B*, and therefore the system fails in this case. In the K_{12} model, the functions f_1, f_4, f_9, f_{10}, f_{11}, and f_{12} evaluate to zero, and the graph becomes disconnected (Fig. 6d), as the removal of the corresponding edges partitions the graph into multiple connected components, which corresponds to system failure. In the K'_{12} model, the functions f'_3, f'_8, f'_9, f'_{10}, and f'_{11} evaluate to zero, and the graph also becomes disconnected (Fig. 7d).

Finally, consider the vector **v** = 10100110. This vector corresponds to the failure of four processors, and the system fails in this case. In the K_{12} model, the functions f_2, f_4, f_6, f_9, f_{10}, and f_{11} evaluate to zero, and the graph becomes disconnected (Fig. 6e). In the K'_{12} model, the functions f'_1, f'_3, f'_5, f'_9, and f'_{10} evaluate to zero, and the graph also becomes disconnected (Fig. 7e).

This agreement holds for all vectors $\mathbf{v} \in \{0,1\}^8$, as confirmed by exhaustive evaluation, thereby establishing the equivalence of the models.

In [61], another method for constructing GL-models for non-basic FTMS was proposed. In particular, for the system considered in this example (*Example 1* in [61]), two variants of the GL-model were constructed, denoted as *M* and M'.

Model M is based on the graph shown in Fig. 8a and includes the following edge functions:

$$\tilde{f}_1^1 = f_1^1 \vee \bar{c}_1 = x_1 \vee x_2 \vee \bar{x}_6 \vee \bar{x}_7;$$
$$\tilde{f}_2^1 = f_2^1 \vee \bar{c}_1 = x_1x_2 \vee x_3x_4 \vee \bar{x}_6 \vee \bar{x}_7;$$
$$\tilde{f}_3^1 = f_3^1 \vee \bar{c}_1 = x_3 \vee x_4 \vee \bar{x}_6 \vee \bar{x}_7;$$
$$\tilde{f}_4^1 = f_4^1 \vee \bar{c}_1 = x_1x_2x_3x_4 \vee x_5x_6x_7x_8 \vee \bar{x}_6 \vee \bar{x}_7 = x_1x_2x_3x_4 \vee x_5x_8 \vee \bar{x}_6 \vee \bar{x}_7;$$
$$\tilde{f}_5^1 = f_5^1 \vee \bar{c}_1 = x_5 \vee x_6 \vee \bar{x}_6 \vee \bar{x}_7 = 1;$$
$$\tilde{f}_6^1 = f_6^1 \vee \bar{c}_1 = x_5x_6 \vee x_7x_8 \vee \bar{x}_6 \vee \bar{x}_7 = x_5 \vee x_8 \vee \bar{x}_6 \vee \bar{x}_7;$$
$$\tilde{f}_7^1 = f_7^1 \vee \bar{c}_1 = x_7 \vee x_8 \vee \bar{x}_6 \vee \bar{x}_7 = 1;$$
$$\tilde{f}_8^1 = f_8^1 \vee \bar{c}_1 = x_1x_3x_6x_7 \vee \bar{x}_6 \vee \bar{x}_7 = x_1x_3 \vee \bar{x}_6 \vee \bar{x}_7;$$
$$\tilde{f}_9^1 = f_9^1 \vee \bar{c}_1 = x_2x_6x_7 \vee \bar{x}_6 \vee \bar{x}_7 = x_2 \vee \bar{x}_6 \vee \bar{x}_7.$$

In addition, model M includes the following edge functions:

$$\tilde{f}_1^2 = f_1^2 \vee \bar{c}_2 = x_1 \vee x_2x_3x_4 \vee x_6x_7;$$
$$\tilde{f}_2^2 = f_2^2 \vee \bar{c}_2 = x_2 \vee x_3x_4x_5 \vee x_6x_7;$$
$$\tilde{f}_3^2 = f_3^2 \vee \bar{c}_2 = x_3 \vee x_4x_5x_6 \vee x_6x_7 = x_3 \vee x_6(x_4x_5 \vee x_7);$$
$$\tilde{f}_4^2 = f_4^2 \vee \bar{c}_2 = x_4 \vee x_5x_6x_7 \vee x_6x_7 = x_4 \vee x_6x_7;$$

$\tilde{f}_5^2 = f_5^2 \vee \bar{c}_2 = x_5 \vee x_6x_7x_8 \vee x_6x_7 = x_5 \vee x_6x_7;$
$\tilde{f}_6^2 = f_6^2 \vee \bar{c}_2 = x_6 \vee x_7x_8x_1 \vee x_6x_7 = x_6 \vee x_7x_8x_1;$
$\tilde{f}_7^2 = f_7^2 \vee \bar{c}_2 = x_7 \vee x_8x_1x_2 \vee x_6x_7 = x_7 \vee x_8x_1x_2;$
$\tilde{f}_8^2 = f_8^2 \vee \bar{c}_2 = x_8 \vee x_1x_2x_3 \vee x_6x_7;$
$\tilde{f}_9^2 = f_9^2 \vee \bar{c}_2 = x_4x_5 \vee x_6x_7;$
$\tilde{f}_{10}^2 = f_{10}^2 \vee \bar{c}_2 = x_1x_5x_8 \vee x_6x_7.$

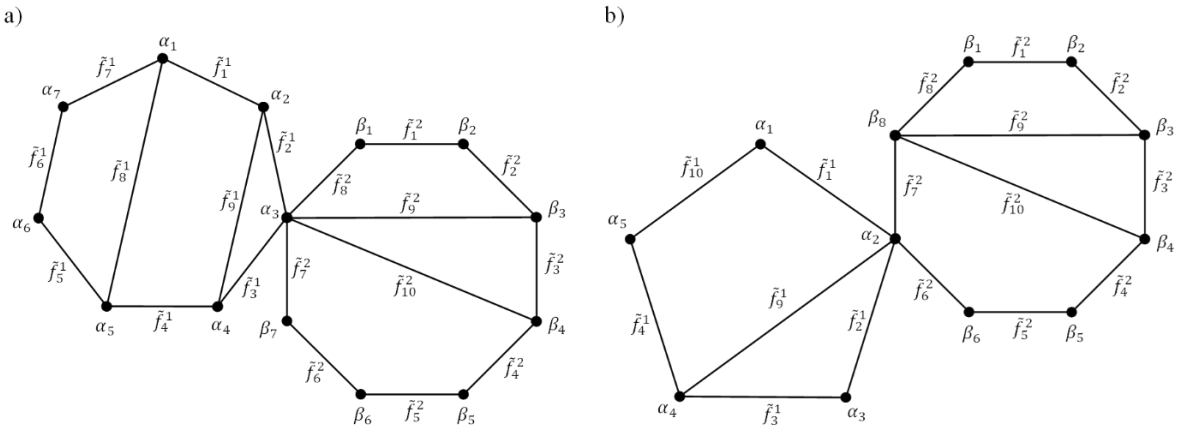


Fig. 8. GL-models M and M' proposed in [61]

Model M' is based on the graph shown in Fig. 8b and contains a subset of the edge functions of model M, whose expressions coincide with the corresponding expressions in M. Specifically, some edge functions of model M are removed, and an additional edge function is introduced:

$$\tilde{f}_{10}^1 = \tilde{f}_6^1 \vee \tilde{f}_8^1 = x_5 \vee x_8 \vee \bar{x}_6 \vee \bar{x}_7 \vee x_1x_3 \vee \bar{x}_6 \vee \bar{x}_7 = x_1x_3 \vee x_5 \vee x_8 \vee \bar{x}_6 \vee \bar{x}_7.$$

Table 1 presents a comparison of the complexity of GL-models constructed for the FTMS considered in the example using the proposed method and the method described in [61]. As can be seen from the table, the total complexity of the edge function expressions (in terms of the number of logical operations) for the models constructed using the proposed method is comparable, and in some cases even lower than that of the models obtained using the method described in [61]. At the same time, the graphs of the proposed models are structurally simpler, as they contain fewer vertices and edges. This indicates that the proposed method may reduce the structural complexity of GL-models, as observed in the presented example, while maintaining comparable logical complexity.

7. Discussion

The presented results indicate that the proposed method is capable of constructing GL-models that accurately reproduce the failure behavior of the considered system while maintaining a relatively compact graph structure. At the same time, the evaluation is based on a representative example and exhaustive analysis of system states, and therefore primarily reflects the performance of the method under the considered conditions.

Table 1. Comparison of GL-model complexity for the proposed method and an existing approach, in terms of the number of vertices, edges (i.e., edge functions), and total logical operations in the corresponding expressions

Model	Vertices	Edges	Disjunctions	Conjunctions	Inversions	Binary operations	Total operations
K_{12}	8	12	21	35	9	56	65
K'_{12}	8	11	23	37	13	60	73
M from [61]	14	19	33	29	14	62	76
M' from [61]	12	16	32	29	12	61	73

The comparison with the approach in [61] indicates that the proposed method may reduce structural complexity while preserving comparable logical complexity. However, this conclusion is drawn for the specific system considered, and its generalization to a broader class of systems requires further investigation.

It is also important to note that the presented figures and examples are primarily illustrative and provide qualitative evidence of the method's behavior rather than a comprehensive validation. A more extensive evaluation, including a

wider range of system configurations and additional complexity metrics, may provide deeper insight into the advantages and limitations of the approach.

8. Conclusion

This paper proposes a generalized method for constructing GL-models of non-basic fault-tolerant multiprocessor systems based on the combination of auxiliary GL-models corresponding to different operating conditions. The proposed method contributes to the development of compact and flexible reliability models for complex fault-tolerant systems. Unlike existing approaches, it imposes no restrictions on the graph structures of auxiliary models and enables their integration within a unified modeling framework.

The theoretical contribution of the paper consists in the formalization of a generalized framework for model combination and the introduction of equivalence-preserving transformations sufficient to transform an arbitrary set of models into a common graph structure. The contribution is primarily methodological, extending existing approaches to the construction of GL-models and generalizing their applicability.

The applicability and effectiveness of the proposed method are demonstrated through a representative example of a non-basic system and should be further evaluated on a broader class of systems. The correctness of the constructed GL-models is validated through analysis of representative system states and through exhaustive enumeration of all possible states (i.e., all 2^8 system states for the considered example). The results show that identical failure behavior can be represented by structurally different GL-models.

A comparative analysis with an existing approach demonstrates the potential for improved structural efficiency of the resulting GL-models, in terms of the number of vertices and edges. At the same time, comparable or, in some cases, lower logical complexity of the edge function expressions is maintained. The comparison in this work is limited to a representative method from the literature, and further benchmarking against a wider range of approaches remains a direction for future research.

The study assumes that auxiliary GL-models are given and does not address their construction from a predefined set of blocked vectors. The generalizability of the proposed approach to a broader class of systems requires further investigation. Future work may focus on this problem, as well as on developing methods for the optimal transformation and combination of auxiliary models aimed at minimizing both structural and logical complexity of the resulting GL-models.

All the Declarations and Statements

Author Contribution Statement

V. Romankevich – Conceptualization and methodology: Developed the main idea of the proposed approach, defined the overall research direction and high-level framework, and structured the methodology, outlining the core principles of the model combination approach.
K. Morozov – Formal analysis and validation: Performed the formalization of the proposed method, developed and proved the theoretical propositions, and ensured the mathematical correctness of the framework.
A. Romankevich – Verification and theoretical support: Contributed to the validation of the proposed approach, performed theoretical analysis and consistency checks, and supported the interpretation of the obtained results.
P. Malezhyk – Experimental support: Assisted in the preparation and verification of the experimental setup, supported the analysis of representative examples, and contributed to the validation of the obtained results.
L. Zacharioudakis – Case studies and examples: Contributed to the development and analysis of representative examples, supported the evaluation of the proposed method in practical scenarios, and assisted in refining the presentation of results.

All authors have read and agreed to the published version of the manuscript.

Conflict of Interest Statement

The Authors declared no conflicts of interest.

Funding Declaration

This research received no external funding.

Data Availability Statement

The data that support the findings of this study are from the corresponding author upon reasonable request.

Ethical Declarations

This study did not involve human participants or animals.

Acknowledgements
None.

Declaration of Generative AI in Scholarly Writing
During the preparation of the manuscript, the authors carefully reviewed the content and take full responsibility for the accuracy and integrity of the manuscript.

Abbreviations
The following abbreviations are used in this manuscript:

CS – Control System
CAS – Critical Application System
FTMS – Fault-Tolerant Multiprocessor Systems
GL-model – Graph-Logical Model
MLE-model – "Minimum Lost Edges" type GL-model

References

[1] Yalcinkaya, E., Maffei, A., & Onori, M. (2017). Application of Attribute Based Access Control Model for Industrial Control Systems. International Journal of Computer Network and Information Security, 2, 12-21. DOI: 10.5815/ijcnis.2017.02.02.
[2] Labane, C., Mekkaoui, M., Zemalache, M. K., & Meddahi, Y. (2024). Model Predictive Control for Nonlinear Inverted Pendulum Mobile Robot with Disturbance. International Journal of Computer Network and Information Security, 16(4), 1919-1929.
[3] Gokhale, S., Dalvi, A., & Siddavatam, I. (2020). Industrial Control Systems Honeypot: a Formal Analysis of Conpot. International Journal of Computer Network and Information Security, 6, 44-56. DOI: 10.5815/ijcnis.2020.06.04.
[4] Nazarova, O. S., Osadchyy, V. V., & Rudim, B. Y. (2023). Computer Simulation of the Microprocessor Liquid Level Automatic Control System. Applied Aspects of Information Technology, 6(2), 163-174. DOI: 10.15276/aait.06.2023.12.
[5] Khilenko, V. V., Denysiuk, S. P., Bielokha, H. S., Stepanov, O. V., & Kang, L.-W. (2026). Optimizing the Control of a Transactive Energy System with Periodically Changing Architecture Based on Internet of Energy. Herald of Advanced Information Technology, 9(1), 85-99. DOI: 10.15276/hait.09.2026.07.
[6] Kotov, D. O. (2022). A Generalized Model of an Adaptive Information-Control System of a Car with Multi-Sensor Channels of Information Interaction. Applied Aspects of Information Technology, 5(1), 25-34. DOI: 10.15276/aait.05.2022.2.
[7] Khudov, H., Kivshar, O., Khizhnyak, I., Taran, I., & Solomonenko, Y. (2026). Method for Predicting the Flight Path of Long-Range Unmanned Aerial Systems Based on the Elite Ants Algorithm. Advanced Information Systems, 10(2), 52-59. DOI: 10.20998/2522-9052.2026.2.06.
[8] Barabash, O., & Kyrianov, A. (2023). Development of Control Laws of Unmanned Aerial Vehicles for Performing Group Flight at the Straight-Line Horizontal Flight Stage. Advanced Information Systems, 7(4), 13-20. DOI: 10.20998/2522-9052.2023.4.02.
[9] Nair, P. P., Sarkar, A., & Biswas, S. (2021). Fault-Tolerant Real-Time Fair Scheduling on Multiprocessor Systems with Cold-Standby. IEEE Transactions on Dependable and Secure Computing, 18(4), 1718-1732. DOI: 10.1109/TDSC.2019.2934098.
[10] Pathan, R. M. (2017). Real-time scheduling algorithm for safety-critical systems on faulty multicore environments. Real-Time Systems, 53(1), 45-81. DOI: 10.1007/s11241-016-9258-z.
[11] Drozd, O. V., Rucinski, A., Zashcholkin, K. V., Drozd, M. O., & Sulima, Y. Y. (2021). Improving FPGA Components of Critical Systems Based on Natural Version Redundancy. Applied Aspects of Information Technology, 4(2), 168-177. DOI: 10.15276/aait.02.2021.4.
[12] Ibrahimov, B., Hasanov, A., & Hashimov, E. (2024). Research and Analysis of Efficiency Indicators of Critical Infrastructures in the Communication System. Advanced Information Systems, 8(2), 58-64. DOI: 10.20998/2522-9052.2024.2.07.
[13] Mohlabeng, M., & Vallabhapurapu, S. (2025). Securing Critical Infrastructure: Design, Incidents, and Protection Strategies Railway Cable. International Journal of Computer Network and Information Security, 17(2), 383-399.
[14] Shavarskyi, M. A., & Kryvechuk, Y. P. (2026). Anomaly Detection on Time Series Data for Autonomous Robot Operation. Herald of Advanced Information Technology, 9(1), 46-59. DOI: 10.15276/hait.09.2026.04.
[15] Arfat, Y., & Eassa, F. E. (2016). A Survey on Fault Tolerant Multi Agent System. International Journal of Information Technology and Computer Science, 9, 39-48. DOI: 10.5815/ijitcs.2016.09.06.
[16] Belarbi, M. (2015). Formal and Informal Modeling of Fault Tolerant Noc Architectures. International Journal of Intelligent Systems and Applications, 12, 32-42. DOI: 10.5815/ijisa.2015.12.03.
[17] Abbaspour, A., Mokhtari, S., Sargolzaei, A., & Yen, K. K. (2020). A Survey on Active Fault-Tolerant Control Systems. Electronics, 9(9), 1513. DOI: 10.3390/electronics9091513.
[18] Al-Allaf, A. F., & Farej, Z. K. (2023). Simulation-Based Fault-Tolerant Multiprocessors System. Telkomnika (Telecommunication Computing Electronics and Control), 21(2), 354-363. DOI: 10.12928/TELKOMNIKA.v21i2.24253.
[19] Chen, L., Feng, H., & Wu, J. (2025). A Path-Based Topology-Agnostic Fault Diagnosis Strategy for Multiprocessor Systems. IEEE Transactions on Computers, 74(6), 1886-1896. DOI: 10.1109/TC.2025.3543701.

[20] Rahdari, D., Rahmani, A. M., Aboutaleby, N., & Karambasti, A. S. (2015). A Distributed Fault Tolerance Global Coordinator Election Algorithm in Unreliable High Traffic Distributed Systems. International Journal of Information Technology and Computer Science, 7(3), 1-11. DOI: 10.5815/ijitcs.2015.03.01.
[21] Nazemi, E., Talebi, T., & Elyasi, H. (2015). Self-Healing Mechanism for Reliable Architecture with Focus on Failure Detection. International Journal of Intelligent Engineering and Electronic Business, 3, 32-38. DOI: 10.5815/ijieeb.2015.03.05.
[22] Arasteh, B. (2017). Enhancing Software Reliability against Soft-Error using Minimum Redundancy on Critical Data. International Journal of Computer Network and Information Security, 9(5), 21-30. DOI: 10.5815/ijcnis.2017.05.03.
[23] Sinha, B., Singh, A. K., & Saini, P. (2019). A Failure Detector for Crash Recovery Systems in Cloud. International Journal of Information Technology and Computer Science, 7, 9-16. DOI: 10.5815/ijitcs.2019.07.02.
[24] Raskin, L., Ivanchikhin, Y., Sukhomlyn, L., Sviatkin, I., & Korsun, R. (2022). Evaluation Model of the Recovery Processes of Non-Markovian Systems, Considering the Elements Unreliability Under Arbitrary Distribution Laws. Advanced Information Systems, 6(3), 28–35. DOI: 10.20998/2522-9052.2022.3.04.
[25] Huang, Y., Lin, L., Xu, L., & Hsieh, S.-Y. (2024). Probabilistic Reliability via Subsystem Structures of Arrangement Graph Networks. IEEE Transactions on Reliability, 73(1), 279-289. DOI: 10.1109/TR.2023.3301629.
[26] Myrnenko, M. D., & Shevel, V. V. (2025). Technological Platforms, Tools, and Standardization in the Unified Information Space for Automated Unmanned Aerial Vehicle Design. Applied Aspects of Information Technology, 8(3), 316-333. DOI: 10.15276/aait.08.2025.21.
[27] Wang, X., Li, S., Liu, F., & Fan, X. (2012). Reliability Analysis of Combat Architecture Model Based on Complex Network. International Journal of Engineering and Manufacturing, 2(2), 15-22. DOI: 10.5815/ijem.2012.02.03.
[28] Thomas, M., & Babak, B. (2017). Reliability Evaluation Metrics for Internet of Things Car Tracking System: a Review. International Journal of Information Technology and Computer Science, 9(2), 1-10. DOI: 10.5815/ijitcs.2017.02.01.
[29] Liu, X., Fan, W., He, J., Han, Z., & Chi, C.-H. (2025). Reliability Assessment of Multiprocessor System Based on Exchanged Crossed Cube Networks. Concurrency and Computation: Practice and Experience, 37(2), e8325. DOI: 10.1002/cpe.8325.
[30] Krasnobayev, V., Koshman, S., Nikolsky, S., & Kovalchuk, D. (2022). Mathematical Model of Computer System Reliability in Residual Classes. Advanced Information Systems, 6(4), 19–24. DOI: 10.20998/2522-9052.2022.4.03.
[31] Kuo, W., & Zuo, M. J. (2002). Optimal Reliability Modeling: Principles and Applications. Hoboken, NJ: John Wiley & Sons.
[32] Ahmed, W., & Wu, Y. W. (2013). A survey on reliability in distributed systems. Journal of Computer and System Sciences, 79(8), 1243-1255. DOI: 10.1016/j.jcss.2013.02.006.
[33] Wason, R., Soni, A. K., & Rafiq, M. Q. (2015). Estimating Software Reliability by Monitoring Software Execution through Opcode. International Journal of Information Technology and Computer Science, 7(9), 23-30. DOI: 10.5815/ijitcs.2015.09.04.
[34] Raskin, L., Sviatkin, I., Ivanchikhin, Y., & Korsun, R. (2022). Semi-Markov Reliability Models. Advanced Information Systems, 6(1), 70–74. DOI: 10.20998/2522-9052.2022.1.12.
[35] Kaswan, K. S., Choudhary, S., & Sharma, K. (2015). Software Reliability Modeling using Soft Computing Techniques: Critical Review. International Journal of Information Technology and Computer Science, 7(9), 90-101. DOI: 10.5815/ijitcs.2015.07.10.
[36] Kumari, P., & Jain, N. (2024). Evaluating the Impact of Scheduling Algorithms on Performance and Reliability in Delay-Tolerant Networks. International Journal of Computer Network and Information Security, 16(4), 1019-1025.
[37] Suo, B., Cheng, Y. S., Zeng, C., & Li, J. (2012). Calculation of Failure Probability of Series and Parallel Systems for Imprecise Probability. International Journal of Engineering and Manufacturing, 2(2), 79-85. DOI: 10.5815/ijem.2012.02.12.
[38] Jianu, M., Daus, L., Dragoi, V. F., & Beiu, V. (2023). Reliability Polynomials of Consecutive-k-out-of-n:F Systems Have Unbounded Roots. Networks, 82(3), 222-228. DOI: 10.1002/net.22168.
[39] Triantafyllou, I. (2020). m-Consecutive-k-out-of-n: F Structures with a Single Change Point. Mathematics, 8(12), 2203. DOI: 10.3390/math8122203.
[40] Gökdere, G., Gürcan, M., & Kılıç, M. (2016). A New Method for Computing the Reliability of Consecutive k-out-of-n:F Systems. Open Physics, 14(1), 166-170. DOI: 10.1515/phys-2016-0015.
[41] Yin, J., Balakrishnan, N., & Cui, L. (2022). 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, 224, 108549. DOI: 10.1177/1748006X221130540.
[42] Yin, J., & Cui, L. (2021). Reliability for Consecutive-k-out-of-n:F Systems with Shared Components Between Adjacent Subsystems. Reliability Engineering & System Safety, 210, 107532. DOI: 10.1016/j.ress.2021.107532.
[43] Chopra, G., & Ram, M. (2022). Linear Consecutive-k-out-of-n:G System Reliability Analysis. Journal of Reliability and Statistical Studies, 15(2), 669-692. DOI: 10.13052/jrss0974-8024.15211.
[44] Yi, H., Cui, L., & Gao, H. (2020). Reliabilities of Some Multistate Consecutive-k Systems. IEEE Transactions on Reliability, 69(2), 414-429. DOI: 10.1109/TR.2019.2897726.
[45] Eryilmaz, S., & Kan, C. (2011). Dynamic Reliability Evaluation of Consecutive-k-within-m-out-of-n:F System. Communications in Statistics: Simulation and Computation, 40(1), 58-71. DOI: 10.1080/03610918.2010.530366.
[46] Torrado, N. (2015). Tail Behaviour of Consecutive 2-within-m-out-of-n Systems with Nonidentical Components. Applied Mathematical Modelling, 39(15), 4586-4592. DOI: 10.1016/j.apm.2014.12.042.
[47] Amirian, Y., Khodadadi, A., & Chatrabgoun, O. (2020). 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, 27(1), 2050003. DOI: 10.1142/S0218539320500035.
[48] Nashwan, I. (2021). Reliability and Failure Probability Functions of the m-Consecutive-k-out-of-n:F Linear and Circular Systems. Baghdad Science Journal, 18(2), 430. DOI: 10.21123/bsj.2021.18.2.0430.
[49] Triantafyllou, I. S. (2023). Combined m-Consecutive-k-out-of-n:F and Consecutive kc-out-of-n:F Structures with Cold Standby Redundancy. Mathematics, 11(12), 1-13. DOI: 10.3390/math11122597.
[50] Triantafyllou, I., & Koutras, M. (2014). Reliability Properties of (n, f, k) Systems. IEEE Transactions on Reliability, 63(1), 357-366. DOI: 10.1109/TR.2014.2299495.

[51] Cui, L., Lin, C., & Du, S. (2015). m-Consecutive-k,l-out-of-n Systems. IEEE Transactions on Reliability, 64(1), 386-393. DOI: 10.1109/TR.2014.2337091.
[52] He, S., Wang, C., Wu, Z., & Guan, Q. (2026). Reliability Analysis for Linear Multi-State (n, f, k) and <n, f, k> Systems. Reliability Engineering and System Safety, 272, 112245. DOI: 10.1016/j.ress.2026.112245.
[53] Lee, W. S., Grosh, D. L., Tillman, F. A., & Lie, C. H. (1985). Fault Tree Analysis, Methods and Applications: a Review. IEEE Transactions on Reliability, 34(3), 194-203. DOI: 10.1109/TR.1985.5222114.
[54] Romankevich, A., Feseniuk, A., Maidaniuk, I., & Romankevich, V. (2019). Fault-Tolerant Multiprocessor Systems Reliability Estimation using Statistical Experiments with GL-Models. Advances in Intelligent Systems and Computing, 754, 186-193. DOI: 10.1007/978-3-319-91008-6_19.
[55] Romankevich, A. M., Morozov, K. V., & Romankevich, V. A. (2022). On Determining the State of the Processors of a Multiprocessor System During its Testing. CEUR Workshop Proceedings, 3156, 1-12.
[56] Romankevich, A. M., Morozov, K. V., & Romankevich, V. A. (2022). Generalization of the Formal Method for Determining the State of Processors of a Multiprocessor System under Testing. Lecture Notes in Data Engineering and Communications Technologies, 134, 363-375. DOI: 10.1007/978-3-031-04812-8_31.
[57] Romankevich, V. A., Morozov, K. V., Shuryha, O. V., Romankevich, A. M., & Zacharioudakis, L. (2025). Enhancement of the Formal Method of Determining Processor States of a Multiprocessor System under Testing. Lecture Notes in Data Engineering and Communications Technologies, 242, 142-155. DOI: 10.1007/978-3-031-84228-3_12.
[58] Romankevich, A., Morozov, K., Romankevich, V., Halytskyi, D., & Zacharioudakis, E. (2023). Improved GL-Model of Behavior of Complex Multiprocessor Systems in Failure Flow. Lecture Notes in Data Engineering and Communications Technologies, 181, 236-245. DOI: 10.1007/978-3-031-36118-0_21.
[59] Romankevich, V. A., Morozov, K. V., Romankevich, A. M., Morozova, A. V., & Zacharioudakis, L. (2024). On the Modification of GL-Models by Adding Edges to a Cyclic Graph. Herald of Advanced Information Technology, 7(2), 185-198. DOI: 10.15276/hait.07.2024.13.
[60] Romankevich, V. A., Morozov, K. V., Romankevich, A. M., Morozova, A. V., & Zacharioudakis, L. (2024). 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, 7(2), 175-188. DOI: 10.15276/aait.07.2024.13.
[61] Romankevich, V. A., Morozov, K. V., Romankevich, A. M., Halytsky, D. V., & Zacharioudakis, L. (2026). A Method for Constructing Non-Basic GL-Models by Combining Models with Arbitrary Graph Structures. Herald of Advanced Information Technology, 9(1), 30-45. DOI: 10.15276/hait.09.2026.03.
[62] Romankevich, V. A., Yermolenko, I. A., Morozov, K. V., & Romankevich, A. M. (2024). Graph-Logical Models for (n, f, k) and Consecutive-k-out-of-n Systems. Herald of Advanced Information Technology, 7(3), 296-308. DOI: 10.15276/hait.07.2024.21.
[63] Romankevich, V. A., Morozov, K. V., Romankevich, A. M., Halytsky, D. V., & Zhurba, A. V. (2025). GL-Models for Analyzing Multiprocessor Systems Tolerant to One and Two Failures. Applied Aspects of Information Technology, 8(1), 113-128. DOI: 10.15276/aait.08.2025.9.
[64] Zajicek, J., & Kamenicky, J. (2015). K-out-of-N Systems: Design, Operation and Preventive Maintenance. Safety and Reliability of Complex Engineered Systems – Proceedings of the 25th European Safety and Reliability Conference, ESREL 2015, 2323-2327. DOI: 10.1201/b19094-304.
[65] Wu, B., Tan, Z., & Che, A. (2026). Resilience Analysis of k-out-of-n Systems. Naval Research Logistics, 73(3), 352-369. DOI: 10.1002/nav.70017.
[66] Dembińska, A., & Nikolov, N. I. (2026). Reliability Properties of k-out-of-n Systems with Several Cold Standby Units. Journal of Computational and Applied Mathematics, 477, 117203. DOI: 10.1016/j.cam.2025.117203.
[67] Mannai, N., & Jarray, M. (2026). Optimal Design of k-out-of-n:G System under Load Redistribution and Stochastic Mission Duration. Reliability Engineering and System Safety, 269, 112029. DOI: 10.1016/j.ress.2025.112029.
[68] Wang, X., Zhao, X., Zhao, X., Chen, X., & Ning, R. (2024). Reliability Assessment for a Generalized k-out-of-n: F System under a Mixed Shock Model with Multiple Sources. Computers and Industrial Engineering, 196, 110459. DOI: 10.1016/j.cie.2024.110459.

Authors' Profiles



Vitaliy 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". Research field: Fault-tolerant multiprocessor systems reliability estimation; GL-models; Self-diagnosable systems; Diagnosis of multiprocessor systems; Discrete mathematics.



Kostiantyn Morozov – PhD (Eng), Associate Professor of System Programming and Specialized Computer System Department. National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute".
Research field: GL-models; Fault-tolerant multiprocessor systems reliability estimation; Self-diagnosable multiprocessor systems.



Alexei Romankevich – Doctor of Engineering Sciences, Professor, Professor of System Programming and Specialized Computer System Department. National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute".
Research field: Fault-tolerant multiprocessor systems reliability estimation; GL-models; Self-diagnosable systems; Diagnosis of multiprocessor systems; Multi-valued logic.



Petro Malezhyk – Candidate of Physical and Mathematical Sciences, Doctor of Pedagogical Sciences, Professor, Professor of System Programming and Specialized Computer System Department. National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute".
Research field: Computer systems architecture and design; Systems programming; Information technology.



Lefteris Zacharioudakis – PhD (Eng), Assistant professor, Neapolis University Pafos.
Research field: Principles of CyberSecurity, Operating systems, Information Security, Cryptography, Penetration Testing.

How to cite this paper

Vitaliy Romankevich, Kostiantyn Morozov, Alexei Romankevich, Petro Malezhyk, Lefteris Zacharioudakis, "A Generalized Method for Constructing Graph-Logical Models of Non-Basic Fault-Tolerant Multiprocessor Systems via Model Combination", International Journal of Engineering and Manufacturing (IJEM), Vol.16, No.3, pp.254-269, 2026. DOI:10.5815/ijem.2026.03.15