DOI: https://doi.org/10.15276/hait.08.2025.16
UDC 004.05

On the computational complexity of cascade GL-models for 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
Alexei M. Romankevich[1)]
ORCID: https://orcid.org/0000-0001-5634-8469; romankev@scs.kpi.ua. Scopus Author ID: 6602114176
Yehor O. Nikishyn[1)]
ORCID: https://orcid.org/0009-0008-9772-0261; yehor.nikishyn@gmail.com
[1)] National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute", 37, Beresteiskyi Ave. Kyiv, 03056, Ukraine

ABSTRACT

The article addresses the problem of evaluating the computational complexity of basic cascade GL-models used for modeling the behavior of fault-tolerant multiprocessor systems in the flow of failures. The purpose of this work is to reduce the complexity of such models by optimal selection of their parameters. It has been demonstrated that a single system usually corresponds to an entire family of cascade GL-models, differing in cascade depth and parameters, with each having its own computational complexity. To simplify the process of modeling the system behavior under a flow of failures, it is advisable to choose the cascade GL-model configuration with the lowest complexity. However, additional constraints on the model, such as cascade depth limitations, must also be considered. This work applies an empirical-analytical research method. An analysis of computational complexity for cascade GL-models was conducted using specially developed software, which automated model construction for various combinations of parameters. Subsequently, a comparative analysis of the complexity of their edge function expressions was performed to identify dependencies on parameter values. Experimental studies were carried out for fault-tolerant multiprocessor systems with varying numbers of processors and different maximum allowable failure multiplicities (but not exceeding half of the total number of processors in the system). It was shown that cascade GL-models typically have significantly lower computational complexity compared to standard basic GL-models, especially for systems with a small maximum number of allowed failures. However, in cases where the allowed number of failures equals or exceeds half of the processor count, standard models may become less complex. Based on the conducted analysis, practical recommendations for selecting the parameters of cascade GL-models were formulated for the first time. In particular, the lowest complexity is achieved when the fault tolerance coefficient of the auxiliary model is minimized at each cascade level; however, this leads to a maximal cascade depth. If cascade depth is limited, the lowest complexity is achieved by evenly or nearly evenly distributing the fault-tolerance coefficients among auxiliary models. If an even distribution is impossible, it is advisable to place higher-value coefficients at deeper cascade levels. Experimental results demonstrate that the application of the proposed recommendations can significantly reduce the overall complexity of edge function expressions in the cascade GL-model compared to the basic GL-model, with the effectiveness of the approach increasing as the system size grows.

Keywords: Cascade GL-models; fault-tolerant multiprocessor systems; computational complexity; basic systems; parameters adjustment; reliability assessment

***For citation*: Romankevich, V. A., Morozov, K. V., Romankevich, A. M., Nikishyn Y. O. "On the computational complexity of cascade GL-models for fault-tolerant multiprocessor systems". *Herald of Advanced Information Technology*. 2025; Vol.8 No.2: 245–258. DOI: https://doi.org/10.15276/hait.08.2025.16**

INTRODUCTION

Nowadays, automated and fully automatic systems are being increasingly implemented. This frees up human resources from routine tasks and allows them to focus on solving more complex problems. The use of such systems helps minimize the human factors influence, consequently improving the quality of management by reducing decision-making time, increasing the volume of information processed per unit of time, and eliminating the impact of operator fatigue and emotional exhaustion. Additionally, the application of automated systems makes operation feasible under conditions and environments where human work is impossible or extremely dangerous. Control systems (CS) are used to manage such facilities [1], [2].

Special attention should be given to so-called critical application systems or safety-related systems [3], [4], [5]. Failures of these systems may lead to significant material losses, cause environmental damage, or pose hazards to human health and life. Given the complexity of control algorithms and increased computational resource requirements, fault-tolerant multiprocessor systems (FTMS) are often used as control systems for these objects [6], [7]. Such systems can maintain operability even

if individual processor components fail, significantly increasing their reliability.

One of the critical stages in the FTMS development process is assessing their reliability parameters. This is particularly relevant for complex control systems characterized by many components and numerous potential failure combinations, where this task is typically non-trivial and resource-intensive [8], [9], [10], [11]. Thus, there is a necessity for effective methods to evaluate reliability parameters in such systems.

It is worth noting that FTMS are classified as basic and non-basic. Basic systems remain resistant to failures of any subset of processors as long as the number of these failures does not exceed a predefined threshold. Such systems are traditionally denoted as *K(m, n)*, where *n* is the total number of processors, and *m* is the maximum allowable number of processor failures at which the system maintains operability. Non-basic systems, on the other hand, have more complex structures and may remain stable only for specific combinations of failures while being unstable for other combinations of failures of the same multiplicity. This significantly complicates the task of reliability analysis for non-basic FTMS and requires special modeling approaches.

LITERATURE REVIEW

Approaches to evaluating the reliability of fault-tolerant multiprocessor systems are conventionally divided into two main groups [8], [12], [13], [14].

The first group – analytical methods – is based on constructing exact mathematical expressions for computing reliability parameters [15], [16], [17], [18]. The primary advantages of this approach are the simplicity of calculations and high accuracy of the obtained results. However, its limitations include restricted applicability, as typically each new type of FTMS requires developing a separate calculation method [12], [13], [17]. Therefore, this approach is most suitable for systems whose structure can be described by a relatively small number of parameters. Such systems include basic structures like *k*-out-of-*n* systems [15], as well as certain non-basic structures such as *k*-to-*l*-out-of-*n* [19], consecutive *k*-out-of-*n* [20], [21], [22], [23], [24], [25], [26], [27], consecutive *k*-within-*m*-out-of-*n* [28], [29], consecutive-k_r-out-of-n_r [30], consecutive *k*-out-of-*r*-from-*n* [31], [32], [33], (*n*,*f*,*k*) systems [34], [35], [36], <*n*,*f*,*k*> systems [34], [35], consecutive-(*k*, *l*)-out-of-*n* [37], [38], (*r*, *s*)-out-of-(*m*, *n*) [39], [40], and others.

The second group comprises methods based on statistical modeling of system behavior in a flow of failures [41], [42]. The advantages of these methods include flexibility, allowing the approach to be adapted for various types of systems. However, their drawback is the dependency of reliability evaluation accuracy on the number of conducted experiments, which in practice leads to the necessity of performing numerous tests involving considerable time and resource expenditures. Consequently, reducing the complexity of these experiments directly influences the efficiency of resource utilization and enhances the accuracy of reliability estimations.

CASCADE GL-MODELS AND THEIR PROPERTIES

GL-models can be used for modeling the behavior of FTMS under the flow of failures [43], [44]. A GL-model is represented as an undirected graph, where each edge corresponds to a Boolean edge function defined on the system's state vectors – Boolean vectors in which each element represents the state of a particular processor within the system (1 for operational, 0 for failed). If the value of a Boolean function for a particular edge equals 0, the corresponding edge is removed from the graph, affecting its connectivity. The connectivity of the resulting graph directly corresponds to the state of the system in the flow of failures: a connected graph indicates an operational system, and vice versa.

Basic GL-models are denoted as *K(m, n)*, where *n* is the total number of processors in the system, and *m* is its fault-tolerance degree – the maximum number of processor failures that the system can withstand while maintaining its functionality.

The fault-tolerance coefficient of a basic GL-model *K(m, n)* is the value of the fault-tolerance degree of the corresponding FTMS, specifically the value *m*.

In such models, it is assumed that the system remains operational if the number of failures does not exceed *m*. Upon the occurrence of *m+1* failure, the system transitions to a failed state.

GL-models can be constructed based on cyclic graphs, as proposed in [44]. The advantage of such models is the simplified connectivity evaluation procedure: the graph remains connected if and only if it loses no more than one (arbitrary) edge. The number of edges *N* in a *K(m, n)* model built according to [44] equals $N = n - m + 1$. In vectors with $l \geq m + 1$ zeros, corresponding to l processor failures, the graph loses more than one edge, thus

ISSN 2663-0176 (Print)
ISSN 2663-7731 (Online)

losing its connectivity, interpreted as the system becoming inoperative.

Fig. 1 illustrates a GL-model corresponding to a system consisting of 11 processors, maintaining functionality with up to 5 arbitrary processor failures. Such a system's model, constructed according to [44], is denoted as $K(5, 11)$, and includes Boolean edge functions $f_1, f_2, \ldots, f_7$.

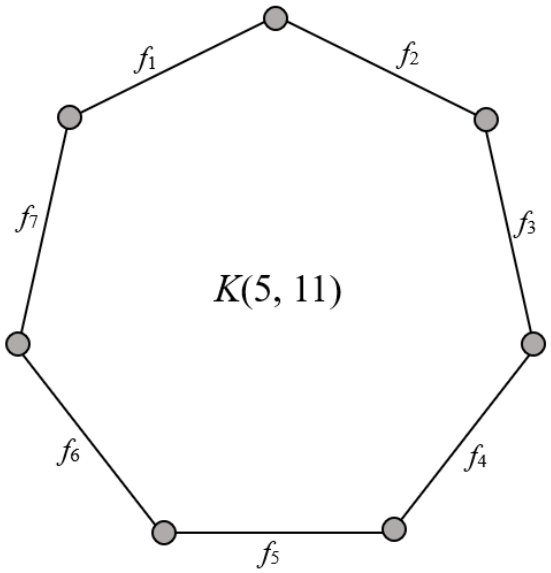

Fig. 1. **Model *K*(5, 11)**
***Source*: compiled by author**

Cascade GL-models [45] expand the traditional paradigm of modeling FTMS behavior in a flow of failures by sequentially combining multiple basic single-level GL-models constructed according to [44]. The output values (results of edge function calculations) of one model serve as the input vector for the next model. This process can be repeated an arbitrary number of times, forming a multi-level structure denoted as $K([m_1, m_2, \ldots, m_T], n)$, where m_i represents the fault-tolerance coefficient of the i-th sub-model, and T is the cascade depth.

Each sub-model is an independent basic GL-model, and the entire cascade model remains a basic model with a combined fault-tolerance degreeμ, calculated as follows:

$$\mu = \sum_{i=1}^{T} m_i - T + 1.$$

This formula reflects the method of aggregating the fault-tolerance coefficients of sub-models. Consequently, the cascade model behaves as a single basic model $K(\mu, n)$ with the same number of edges, namely$N = n - \mu + 1$.

Despite its multi-level structure, the cascade GL-model does not increase overall structural complexity: its final graph has a cyclic topology and demonstrates the same number of edges and edge-loss characteristics as a traditional single-level basic GL-model with an equivalent fault-tolerance coefficient. Although intermediate sub-models possess internal graphs, these are exclusively used to compute composite edge functions and do not affect the final graph structure. Utilizing this approach efficiently subdivides the construction and calculation of logical expressions of model edge functions into simpler sequential subtasks. This modularity has several advantages: it simplifies model construction by allowing the combination of several smaller sub-models instead of creating a single large model containing complex Boolean expressions as edge functions; it reduces computational complexity through reusing intermediate results and avoiding redundant operations.

For example, consider a multiprocessor system with $n = 11$ processors that remains operational with no more than $m = 5$ arbitrary component failures. The basic GL-model for this system is denoted as $K(5, 11)$ and has the graphical structure depicted in Fig. 1, where graph edges correspond to Boolean functions $f_1, f_2, \ldots, f_7$ calculated according to [44].

Applying the cascade approach, this same system can be represented by seven alternative cascade configurations: $K^1([2, 4], 11)$, $K^2([3, 3], 11)$, $K^3([4, 2], 11)$, $K^4([2, 2, 3], 11)$, $K^5([2, 3, 2], 11)$, $K^6([3, 2, 2], 11)$, $K^7([2, 2, 2, 2], 11)$. To clearly illustrate the construction of cascade GL-models, let us examine the model $K^5([2, 3, 2], 11)$ in greater detail. By sequentially applying method [44] to construct each level of this model, we obtain the final graph shown in Fig. 2, which contains the same number of edges ($N = 7$) as the basic model $K(5, 11)$. It is important to note that intermediate sub-models are used only to calculate edge function values of the cascade model; their internal graph structures are not taken into account. Thus, the cascade approach enables forming multiple alternative configurations for a single system, providing reduced computational complexity and increased modeling flexibility compared to traditional basic GL-models.

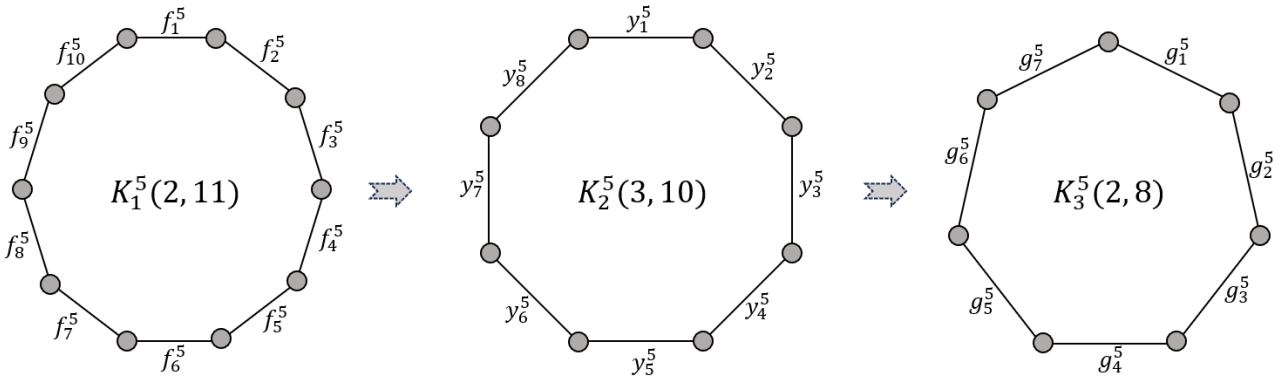

Fig. 2. **Cascade GL-model $K^5([2,3,2],11)$**
***Source*: compiled by the authors**

PROBLEM STATEMENT

As previously noted, when solving the task of reliability assessment for fault-tolerant multiprocessor systems, it is crucial to ensure the reduction of computational complexity of the corresponding GL-model. For large systems, the use of basic GL-models can be inefficient due to the large number of logical operations required. A promising alternative is the application of cascade GL-models.The overall complexity of a cascade model significantly depends on its configuration - specifically, the number of cascade levels and the distribution of fault-tolerance coefficients among sub-models. It is important to note that the number of possible cascade model configurations grows exponentially with an increase in the fault-tolerance degree of the system. This fact is confirmed by numerous experiments conducted within the scope of this study. Consequently, there arises the problem of selecting optimal cascade parameters to achieve the highest efficiency.

PURPOSE AND OBLECTIVE OF THE RESEARCH

The purpose of this research is to reduce the computational complexity of cascade GL-models for fault-tolerant multiprocessor systems by formulating recommendations for selecting cascade parameters. Such recommendations can be derived by identifying patterns in forming optimal configurations of cascade GL-models through a series of statistical experiments and a detailed analysis of obtained data, taking into account constraints on cascade depth and the distribution of fault-tolerance coefficients among cascade levels.

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

1) develop a software tool capable of automatically generating cascade GL-models for specified parameters (fault-tolerance level and size of the multiprocessor system), allowing determination of their computational complexity based on the number of Boolean operations;

2) conduct a series of statistical experiments to evaluate the computational complexity of various configurations of cascade models and to identify patterns in their behavior depending on selected parameters;

3) perform a comparative analysis of the obtained results for cascade and basic single-level GL-models, identifying the advantages and disadvantages of applying the cascade approach.

EXPEREMENTS AND ANALYSIS

In this research, the experimental part was conducted using a specially developed software tool that enables automated generation of cascade GL-models according to specified parameters. This tool calculates the number of Boolean operations (conjunctions and disjunctions) in the expressions of their edge functions and verifies the connectivity of the resulting model for an arbitrary system state vector. Experiments were carried out on multiprocessor systems containing between 6 and 30 components, with the allowable number of failures varying from 2 to 15, provided that the maximum number of failures did not exceed 50% of the system size (for example, for a 10-processor system, a maximum of 5 failures was permitted).

The obtained experimental data were analyzed using descriptive statistical methods and presented in graphical form, allowing for data structuring and identification of the main trends in computational complexity as a function of cascade parameters.

Let us evaluate the change in computational complexity of cascade GL-models of type $K(m, 30)$ to gain a general understanding and identify relevant

ISSN 2663-0176 (Print)
ISSN 2663-7731 (Online)

trends within the range $m \in [3,15]$. The case $m = 2$ is not representative in this evaluation, as it is the same to the basic GL-model. And the case $m = 15$ corresponds exactly to the 50% failure threshold. Considering the specific nature of cascade models, namely the existence of multiple possible combinations for cascade formation, the arithmetic mean of computational complexity across all permissible combinations was calculated for models with the same cascade depth.

The analysis of the graphs presented in Fig. 3 demonstrates that the computational complexity of the basic GL-model construction approach (where $T = 1$) is the highest compared to cascade models. Even at the minimal cascade depth ($T = 2$), a complexity reduction is observed compared to the single-level model. For example, for a system with 30 processors and an allowable failure multiplicity of $m = 14$, the basic GL-model $K(14, 30)$ has a complexity of 6511 Boolean operations. In contrast, the cascade model with depth $T = 2$, denoted as $K^{12}([13, 2], 30)$ – even though it exhibits the worst performance among all possible configurations – demonstrates a computational complexity of 6451 operations, corresponding to a 0.9% reduction in complexity.

Thus, even in the worst cases, cascade models demonstrate advantages over basic single-level models. At the same time, to ensure that the applied averaging does not distort the overall conclusion and that the trend of decreasing computational complexity with increasing cascade depth is preserved, it is advisable to analyze the minimum complexity values for each cascade depth.

Fig. 4 shows that the behavior of the models remains consistent with the previously identified trend. Further analysis of the dependence of computational complexity on parameter T indicates a significant reduction in the number of logical operations as cascade depth increases.

For the same system of 30 processors, cascade models with depths from 2 to 13 were constructed. The lowest complexity was obtained for the model $K^{4095}([2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,], 30)$, where the number of Boolean operations is 1157, corresponding to a 82.2% reduction in complexity compared to the basic approach.

The obtained results indicate that, under conditions where the number of failures in the system does not exceed 50% of its total size, the cascade approach allows decomposing complex Boolean expressions into a series of simpler subtasks, significantly reducing the overall computational complexity of the model. This makes cascade GL-models more suitable for use in large-scale systems, where traditional single-level models lose efficiency due to an excessive number of logical operations.

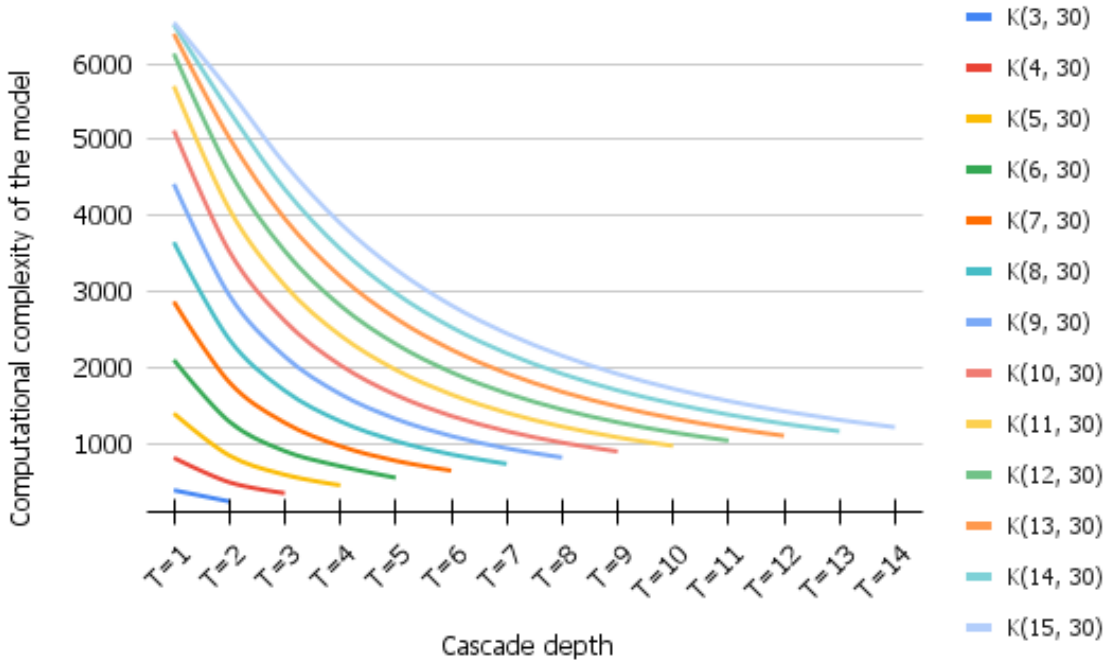

Fig. 3. **Average model complexity depending on cascade depth**
***Source*: compiled by the authors**

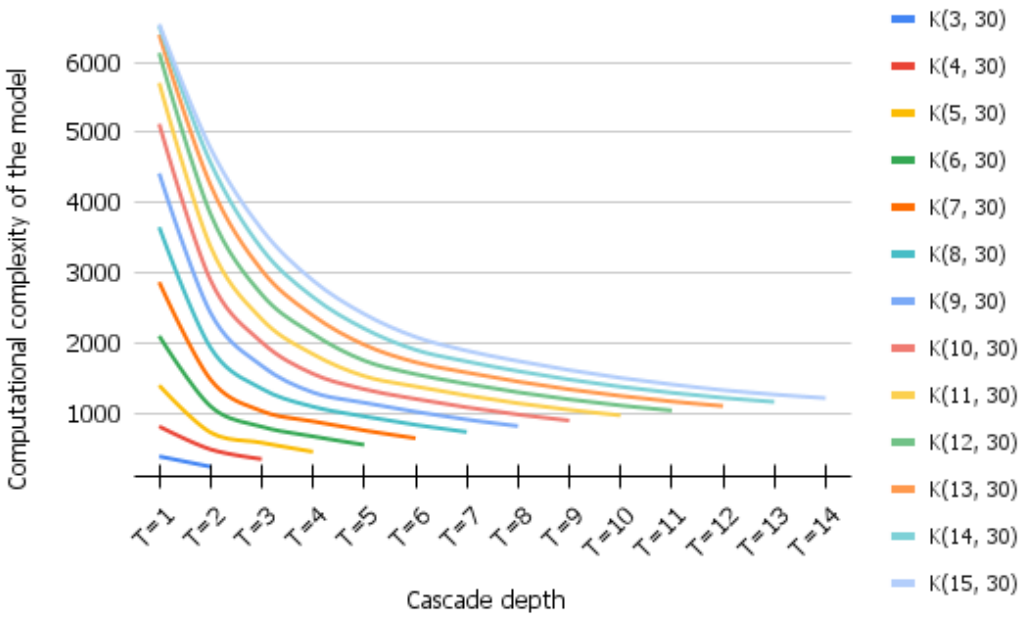

Fig. 4. **Minimum model complexity depending on cascade depth**
***Source*: compiled by the authors**

However, there are cases where the cascade approach may yield slightly worse results compared to the basic approach. This occurs when the system's fault-tolerance degree equals or exceeds 50% of its size. For example, for the model $K(15, 30)$, the computational complexity is 6542 Boolean operations, whereas for cascade models $K^{12}([13, 3], 30)$ and $K^{13}([14, 2], 30)$, complexities of 6556 and 6565 operations were obtained, respectively. As the difference between the fault-tolerance degree and the system size decreases, the number of such cases increases.

Considering these results, it can be hypothesized that the efficiency of cascade GL-models is determined not only by the cascade depth but also by the distribution of fault-tolerance coefficients at each level. Such a distribution potentially affects the complexity of subtasks at individual cascade levels and, consequently, the overall computational efficiency of the model.

According to the cascade GL-model construction algorithm [45], the specific values of the fault-tolerance coefficients for each sub-model significantly influence the number of Boolean operations required to implement that cascade level. Increasing the cascade depth is accompanied by a rapid growth in the number of possible coefficient distribution combinations, underscoring the importance of a detailed investigation into the impact of these distributions on overall model complexity.

To illustrate this aspect, a detailed analysis of the model of type $K(8, 23)$ was conducted, examining all permissible cascade configurations. To confirm the general trend, models $K(8, 17)$, $K(8, 20)$, $K(8, 26)$, and $K(8, 29)$ were also analyzed. The obtained results are presented in Figs. 5 - 9. The basic GL-model $K(8, 23)$, with a complexity of 1685 Boolean operations, serves as a benchmark for comparison with cascade implementations.

Further efficiency analysis is carried out in two main directions: the first concerns the impact of the fault-tolerance coefficient values at the initial cascade levels, and the second addresses the dependence of computational complexity on the uniformity of their distribution across the entire cascade depth.

For the analysis in the first direction, models with cascade depths $T = 2$, 3, and 4 were selected, since at shallow depths it is possible to form configurations in which one cascade level has a significantly higher fault-tolerance coefficient than the others. The initial stage considers cascade models with depth $T = 2$; the results are shown in Fig. 5.

The analysis of the graph allows for highlighting several important observations. In particular, a significant increase in computational complexity is observed for the models $K^4([5, 4], 23)$, $K^5([6, 3], 23)$, and $K^6([7, 2], 23)$ compared to the previous configurations $K^1([2, 7], 23)$, $K^2([3, 6], 23)$, and $K^3([4, 5], 23)$. Analyzing the fault-tolerance coefficients according to algorithm [45], one can hypothesize that a larger fault-tolerance coefficient

ISSN 2663-0176 (Print)
ISSN 2663-7731 (Online)

at the first cascade level leads to an increase in the number of required intermediate calculations. Conversely, if the first cascade level is characterized by smaller coefficients, the initial partitioning of the set into subsets becomes simpler, reducing the number of required Boolean operations. Thus, the more complex computations are transferred to the second cascade level, which operates on fewer states, thereby reducing overall computational complexity. Accordingly, lower coefficient values at the initial levels contribute to reducing the computational complexity of cascade GL-models.

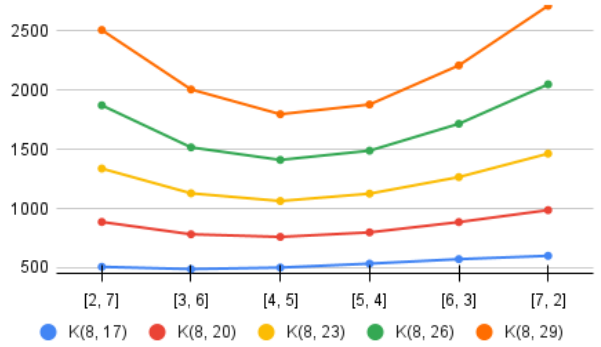

Fig. 5. **Computational complexity of cascade GL-models for $T = 2$**
***Source*: compiled by the authors**

The transition to models with greater depths ($T = 3$ and $T = 4$) confirms the identified pattern. The graphs for the corresponding configurations are shown in Fig. 6 and Fig. 7. As seen in Fig. 6, models with high fault-tolerance coefficients at the initial cascade levels demonstrate greater computational complexity. Specifically, configurations $K^{11}([2, 6, 2], 23)$, $K^{20}([5, 3, 2], 23)$, $K^{21}([6, 2, 2], 23)$, and others exhibit pronounced peaks, indicating an increase in the number of logical operations in cases of uneven distribution of fault-tolerance coefficients.

Similar behavior is observed in Fig. 7, confirming the negative impact of large fault-tolerance coefficients at the initial cascade levels. At the same time, the same graphs reveal configurations that demonstrate the opposite trend – significantly lower computational complexity.

Thus, among the configurations with depth $T = 4$, the lowest complexity values are demonstrated by five models: $K^{27}([2, 3, 3, 3], 23)$ with 678 Boolean operations; $K^{33}([3, 2, 3, 3], 23)$ with 685 operations; $K^{35}([3, 3, 2, 3], 23)$ with 692; $K^{36}([3, 3, 3, 2], 23)$ with 699; and $K^{23}([2, 2, 3, 4], 23)$ with 716 operations. A common feature of four of these configurations is the principle of distributing the fault-tolerance coefficients: the values are distributed evenly or nearly evenly across cascade levels. Such a structure contributes to reducing the computational load at individual levels and, accordingly, the total number of Boolean operations in the model.

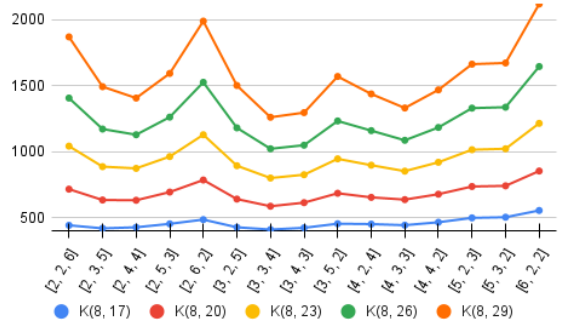

Fig. 6. **Computational complexity of cascade GL-models for $T = 3$**
***Source*: compiled by the authors**

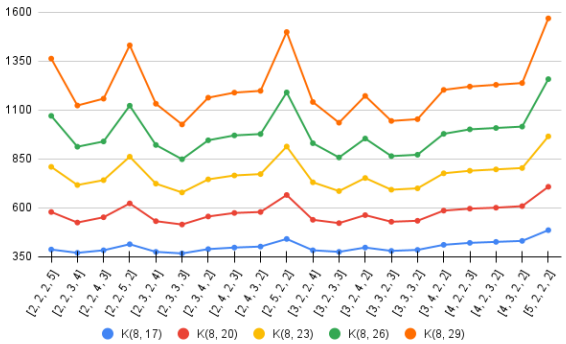

Fig. 7. **Computational complexity of cascade GL-models for $T = 4$**
***Source*: compiled by the authors**

In contrast, models with pronounced coefficient imbalances, such as $K^{41}([5, 2, 2, 2], 23)$, $K^{25}([2, 2, 5, 2], 23)$, and others, concentrate the majority of intermediate computations at a single cascade level. This leads to an increase in the number of intermediate operations at one level of the cascade and, consequently, to a rise in the overall computational complexity of the model.

Thus, the results of the first part of the analysis showed that large fault-tolerance coefficient values at the initial levels can noticeably increase the model's computational complexity. At the same time, it is already apparent at this stage that the overall nature of the coefficient distribution across all cascade levels is equally important. This formed the basis for the second analysis direction – examining the impact of uniformity in fault-tolerance coefficient distribution on computational complexity.

With a further increase in the number of cascade levels, the coefficients at each level gradually decrease, naturally leading to an

approximation of uniform weight distribution. The analysis of the data in Fig. 8 and Fig. 9, which illustrate cascade models with depths of 5 and 6 levels, allows for the identification of configurations in which the smallest fault-tolerance coefficients are concentrated at the initial levels. Combined with an almost uniform weight distribution, this provides a significant reduction in the overall computational complexity of the model.

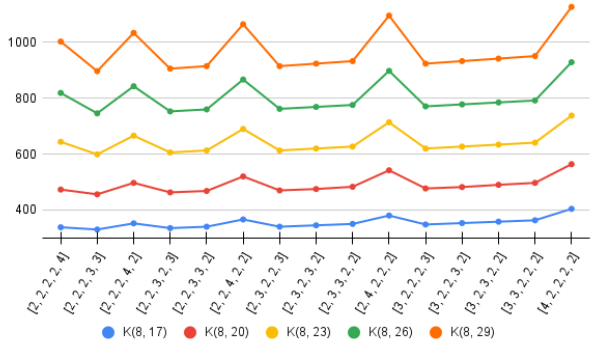

Fig. 8. **Computational complexity of cascade GL-models for $T = 5$**
***Source*: compiled by the authors**

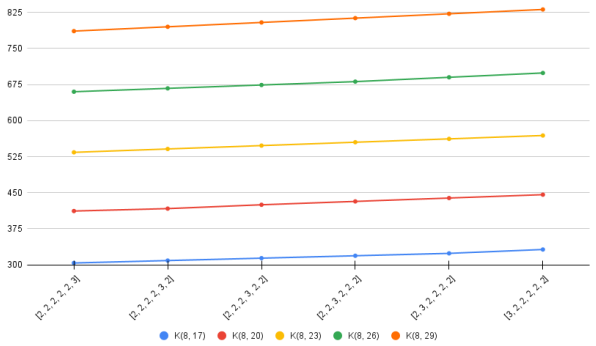

Fig. 9. **Computational complexity of cascade GL-models for $T = 6$**
***Source*: compiled by the authors**

For example, the model $K^{43}([2, 2, 2, 3, 3], 23)$ demonstrates significantly lower complexity compared to the configuration $K^{56}([4, 2, 2, 2, 2], 23)$, which is unbalanced and has a higher coefficient at the first level. The latter is the least efficient among models with depth $T = 5$ and has approximately 20% higher computational complexity. This dependence is visually illustrated in Fig. 9, which presents configurations with a single coefficient equal to 3 while the remaining coefficients are set to 2. As shown in the graph, moving the larger coefficient closer to the start of the cascade leads to an increase in overall complexity.

The lowest complexity among all models examined for the system $K(8, 23)$ is exhibited by the configuration $K^{63}([2, 2, 2, 2, 2, 2, 2], 23)$, which is characterized by a uniform distribution of fault-tolerance coefficients (all cascade levels have the same minimum coefficient of 2). The total computational complexity of this model is 483 Boolean operations, which is 3.5 times less compared to the basic GL-model with complexity of 1685 operations. Such a significant complexity gain is explained by the fact that, with uniform and minimal coefficient values at each cascade level, the set of processor states is sequentially split into small subsets, substantially reducing the number of intermediate computations.

RECOMMENDATIONS FOR CONSTRUCTING CASCADE GL-MODELS

Based on the conducted analysis of computational complexity for cascade GL-models of fault-tolerant multiprocessor systems, a series of practical recommendations can be formulated to reduce the overall computational complexity.

The lowest complexity is exhibited by configurations in which the fault-tolerance coefficients at each cascade level are set at their minimal values ($m_i = 2$ for $i = \overline{1,T}$, where $T = m - 1$). These configurations have the general form $K([2, 2, \ldots, 2], n)$.

However, using such configurations necessitates creating a cascade with significant depth, which leads to a substantial increase in the total number of model levels. At the same time, developers may impose constraints on the maximum allowable cascade depth.

In these cases, it is advisable to apply a uniform or near-uniform distribution of fault-tolerance coefficients among the individual cascade levels. If a uniform distribution is unattainable, it is recommended to arrange the coefficients such that their values gradually increase towards deeper cascade levels. This approach allows simpler computations at initial cascade levels, and more complex computations at deeper levels, where the input vector size is smaller. Such a distribution ensures balanced computational workload and reduces the total number of Boolean operations compared to a single-level basic GL-model.

The recommendations provided help avoid excessive structural complexity due to moderate cascade depth, ensure high computational speed and efficiency, and render cascade GL-models suitable for practical application in real-world tasks of evaluating reliability parameters for fault-tolerant multiprocessor systems.

ISSN 2663-0176 (Print)
ISSN 2663-7731 (Online)

ALGORITHM FOR CONSTRUCTING CASCADE GL-MODELS

Based on the previously formulated recommendations for reducing computational complexity, the algorithm for constructing a cascade GL-model [45] is defined as follows.

Consider a system of size *n* with a required fault-tolerance level *m*, and a cascade GL-model constrained to a fixed depth *T*. Under these conditions, the construction of a cascade GL-model of the form $K([m_1, m_2, \ldots, m_T], n)$ begins with determining the fault-tolerance coefficients for each level of the cascade.

This requires solving the problem of partitioning the integer *m* into a sequence $\{m_i\}_{i=1}^{T}$ subject to the following constraints:

– the total sum of the coefficients must satisfy $\sum_{i=1}^{T} m_i = m + T - 1$;

– each coefficient must satisfy $m_i \geq 2$;

– the sequence of coefficients should be as balanced as possible;

– higher values should be assigned to the deeper levels of the cascade.

The computation of the cascade coefficients will consist of the following steps.

Step 1. Compute the total required sum of coefficients:

$$S = \sum_{i=1}^{T} m_i = m + T - 1.$$

Step 2. Determine the base value $q = \left\lfloor \frac{S}{T} \right\rfloor$ and the remainder $r = S \bmod T$.

Step 3. Generate the sequence of coefficients:

$$m_i = \begin{cases} q, & 1 \leq i \leq T - r \\ q + 1, & T - r < i \leq T \end{cases}.$$

The resulting sequence satisfies all specified constraints, including the total sum and the desired ordering of larger values toward the end.

The construction of the cascade GL-model is then carried out iteratively as follows:

Step 4. Set the initial cascade level index: $i := 1$.

Step 5. Set the initial model size $n_i := n$.

Step 6. Define the input vector *v*, representing the state of the system components.

Step 7. Construct the base model $K(m_i, n_i)$ using *v* as the input vector.

Step 8. Replace *v* with the edge function values computed from the model.

Step 9. Update the model size for the next cascade level using: $n_{i+1} := n_i - m_i + 1$.

Step 10. Increment the cascade level: $i := i + 1$.

Step 11. If $i \leq T$, repeat steps *7-10*.

This algorithm produces a cascade GL-model that incorporates the recommended coefficient allocation strategy, minimizes computational complexity, and respects predefined structural constraints such as cascade depth.

CONCLUSIONS

The article investigates the dependence of computational complexity of cascade GL-models for describing the behavior of fault-tolerant multiprocessor systems under the flow of failures, focusing on their configuration parameters. The analysis was conducted taking into account cascade depth and the distribution of fault-tolerance coefficients among cascade levels. For this purpose, a series of statistical experiments was performed over a wide range of parameters using a specially developed software tool.

The study encompasses systems ranging in size from 6 to 30 components, with allowable failure multiplicities (fault-tolerance coefficients) ranging from 2 to 15, but not exceeding 50% of the system size. It has been demonstrated that cascade models are capable of significantly reducing the number of Boolean operations compared to basic single-level approaches, especially under conditions of a small allowable number of failures.

The configurations showing the lowest complexity are characterized by a uniform or near-uniform distribution of fault-tolerance coefficients, as well as by gradually increasing coefficient values towards deeper cascade levels. However, in cases where the number of allowable failures is at least half of the total number of components, cascade models may become less efficient compared to basic single-level models.

Based on the obtained results, practical recommendations were formulated regarding the selection of cascade configurations that minimize computational complexity, considering system size, allowable failure multiplicity, and cascade depth constraints. Importantly, experimental results demonstrate that applying the proposed recommendations can reduce the total complexity of edge function expressions in cascade models by up to 83%, compared to the worst-case scenario that may arise if the model is constructed without following any guidelines. In such cases, arbitrary or

suboptimal parameter selection can lead to unnecessarily high complexity.

Further research should be aimed at refining these recommendations and developing methods for the automated selection of cascade model parameters in modeling tasks involving the behavior of fault-tolerant multiprocessor systems under a flow of failures.

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.* 2021; 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. Publ. Nauka i Tekhnika*. 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/record/display.uri?eid=2-s2.0-85104838273&origin=resultslist.

6. Romankevich, A., Feseniuk, A., Romankevich, V. & Sapsai, T. “About a faut-tolerant multiprocessor control system in the pre-dangerous state”. *2018 IEEE 9th International Conference on Dependable Systems, Services and Technologies (DESSERT).* 2018. p. 207–211, https://www.scopus.com/record/display.uri?eid=2-s2.0-85050654902&origin=resultslist.
DOI: https://doi.org/10.1109/DESSERT.2018.8409129

7. Abbaspour, A., Mokhtari, S., Sargolzaei, A. & Yen, K. K. “A survey on active fault-tolerant control systems”. *Electronics*. 2020; 9 (9): 1513. DOI: https://doi.org/10.3390/electronics9091513.

8. Kuo, W. & Zuo, M. “Optimal Reliability Modeling: Principles and Applications”. *John Wiley & Sons*. 2003.

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. Romankevich, V. A., Morozov, K. V., Feseniuk A. P., Romankevich, A. M. & Zacharioudakis, L. “On evaluation of reliability increase in fault-tolerant multiprocessor systems”. *Applied Aspects of Information Technology*. 2024; 7 (1): 81–95. DOI: https://doi.org/10.15276/aait.07.2024.7.

11. Huang, L. & Qiang, X. “On modeling the lifetime reliability of homogeneous manycore systems”. *14th IEEE Pacific Rim International Symposium on Dependable Computing*. 2008. p. 87–94, https://www.scopus.com/record/display.uri?eid=2-s2.0-60349106919&origin=resultslist.
DOI: https://doi.org/10.1109/PRDC.2008.23.

12. 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/record/display.uri?eid=2-s2.0-84928741612&origin=resultslist.
DOI: https://doi.org/10.1016/j.ress.2015.04.005.

13. Zimmermann, A. “Reliability modeling and evaluation of dynamic systems with stochastic Petri nets (Tutorial)”. *VALUETOOLS 2013. 7th International Conference on Performance Evaluation Methodologies and Tools*. 2014: 324264. DOI: https://doi.org/10.4108/icst.valuetools.2013.254370.

14. Xing, L. “Reliability modeling and analysis of complex hierarchical systems”. *International Journal of Reliability, Quality and Safety Engineering*. 2005; 12 (6): 477–492, https://www.scopus.com/record/display.uri?eid=2-s2.0-29144462463&origin=resultslist.
DOI: https://doi.org/10.1142/S0218539305001963.

ISSN 2663-0176 (Print)
ISSN 2663-7731 (Online)

15. Barlow, R. E. & Heidtmann K. D. “Computing k-out-of-n System Reliability”. *IEEE Transactions on Reliability*. 1984; R-33 (4): 322–323, https://www.scopus.com/record/display.uri?eid=2-s2.0-0021506074&origin=resultslist. DOI: https://doi.org/10.1109/TR.1984.5221843.

16. Gao, S., Wang, J. & Chen, Q. “Reliability evaluation for a circular Con/k/n:F system with a novel differential repair policy”. *IEEE Transactions on Reliability*. 2025, https://www.scopus.com/pages/publications/85215432234. DOI: https://doi.org/10.1109/TR.2024.3524329.

17. Huang, Y., Lin, L., Xu, L. & Hsieh, S.-Y. “Probabilistic reliability via Subsystem structures of arrangement graph networks”. *IEEE Transactions on Reliability*. 2024; 73 (1): 279–289, https://www.scopus.com/record/display.uri?eid=2-s2.0-85168268383&origin=resultslist.
DOI: https://doi.org/10.1109/TR.2023.3301629.

18. Rushdi, A. M. A. “Utilization of symmetric switching functions in the symbolic reliability analysis of multi-state k-out-of-n systems”. *International Journal of Mathematical, Engineering and Management Sciences*. 2019; 4 (2): 306–326, https://www.scopus.com/pages/publications/85061839460. DOI: https://doi.org/10.33889/IJMEMS.2019.4.2-025.

19. Mo, Y., Xing, L. & Dugan, J. B. “Performability analysis of k-to-l-out-of-n computing systems using binary decision diagrams”. *IEEE Transactions on Dependable and Secure Computing*. 2018; 15 (1): 126–137. DOI: https://doi.org/10.1109/TDSC.2015.2504092.

20. Gökdere, G., Gürcan, M. & Kılıç, M. “A new method for computing the reliability of consecutive k-out-of-n:F systems”. *Open Physics*. 2016; 14 (1): 166–170, https://www.scopus.com/record/display.uri?eid=2-s2.0-84973539438&origin=resultslist.
DOI: https://doi.org/10.1515/phys-2016-0015.

21. 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/record/display.uri?eid=2-s2.0-85163222383&origin=resultslist.
DOI: https://doi.org/10.1002/net.22168.

22. Gökdere, G. & Gürcan, M. “Dynamic reliability evaluation of linear consecutive k-out-of-n0:F system with multi-state components”. *ITM Web of Conferences*. 2018; 22: 01057. DOI: https://doi.org/10.1051/itmconf/20182201057.

23. 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, https://www.scopus.com/record/display.uri?eid=2-s2.0-85100736521&origin=resultslist.
DOI: https://doi.org/10.1016/j.ress.2021.107532.

24. 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*. 2022; 224: 108549, https://www.scopus.com/record/display.uri?eid=2-s2.0-85142023916&origin=resultslist. DOI: https://doi.org/10.1177/1748006X221130540.

25. Yi, H., Cui, L. & Gao, H. “Reliabilities of some multistate consecutive k systems”. *IEEE Transactions on Reliability*. 2020; 69 (2): 414–429, https://www.scopus.com/record/display.uri?eid=2-s2.0-85074664125&origin=resultslist. DOI: https://doi.org/10.1109/TR.2019.2897726.

26. 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.

27. Belaloui, S. & Bennour, B. “Reliability of linear and circular consecutive-k-out-of-n systems with shock model”. *Afrika Statistika*. 2015; 10 (1): 795–805. DOI: http://doi.org/10.16929/as/2015.795.70.

28. Zhu, X., Mahmoud, B. & Mohamed, R. “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. DOI: https://doi.org/10.1109/TR.2015.2484079.

29. 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/record/display.uri?eid=2-s2.0-84937631892&origin=resultslist.
DOI: https://doi.org/10.1016/j.apm.2014.12.042.

30. 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/record/display.uri?eid=2-s2.0-84973885922&origin=resultslist.
DOI: https://doi.org/10.1109/TR.2016.2570545.

31. 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/record/display.uri?eid=2-s2.0-85069848232&origin=resultslist. DOI: https://doi.org/10.1142/S0218539320500035.

32. Wu, C., Zhao, X., Wang, S. & Song, Y. “Reliability analysis of consecutive-k-out-of-r-from-n subsystems: F balanced systems with load sharing”. *Reliability Engineering and System Safety*. 2022; 228: 108776, https://www.scopus.com/pages/publications/85137169658.
DOI: https://doi.org/10.1016/j.ress.2022.108776.

33. Amirian, Y. & Khodadadi, A. “Exact reliability for a multi-state consecutive linear (circular) k-out-of-r-from-n:F system”. *International Journal of Reliability, Quality and Safety Engineering*. 2021; 28 ()2: 2150014, https://www.scopus.com/pages/publications/85092244347.
DOI: https://doi.org/10.1142/S0218539321500145.

34. Kamalja, K. K. & Shinde, R. L. “On the reliability of (n, f, k) and <n, f, k> systems”. *Communications in Statistics – Theory and Methods*. 2014; 43 (8): 1649–1665, https://www.scopus.com/record/display.uri?eid=2-s2.0-84898856152&origin=resultslist.
DOI: https://doi.org/10.1080/03610926.2012.673674.

35. Cui, L. R., Wang, M. Q. & Jiang, W. X. “Reliability analysis of A combination of (n,f,k) and <n,f,k> Systems”. *Reliability Engineering & System Safety*. 2024; 249: 110191. DOI: https://doi.org/10.1016/j.ress.2024.110191.

36. Triantafyllou, I. & Koutras, M. “Reliability properties of (n,f,k) systems”. *IEEE Transactions on Reliability*. 2014; 63 (1): 357–366, https://www.scopus.com/record/display.uri?eid=2-s2.0-84896314614&origin=resultslist. DOI: https://doi.org/10.1109/TR.2014.2299495.

37. 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/record/display.uri?eid=2-s2.0-85129557917&origin=resultslist.
DOI: https://doi.org/10.1016/j.ress.2022.108549.

38. 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/record/display.uri?eid=2-s2.0-85019236438&origin=resultslist.
DOI: https://doi.org/10.1016/j.ress.2017.05.023.

39. 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*. 2020; 236 (2): 329–338. DOI: https://doi.org/10.1177/1748006X19893533.

40. 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/record/display.uri?eid=2-s2.0-85148634528&origin=resultslist. DOI: https://doi.org/10.1007/s11009-023-09984-3.

41. 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/record/display.uri?eid=2-s2.0-85047465084&origin=resultslist. DOI: https://doi.org/10.1007/978-3-319-91008-6_19.

42. Silva, G. S. M. & Droguett, E. L. “Quantum fault trees and minimal cut sets identification”. *Reliability Engineering and System Safety*. 2025; 262: 111147, https://www.scopus.com/pages/publications/105003875128.
DOI: https://doi.org/10.1016/j.ress.2025.111147.

ISSN 2663-0176 (Print)
ISSN 2663-7731 (Online)

43. Romankevich, A. M., Romankevich, V. A., Kononova, A. A. & Rabah Al Shbul "On some features of GL-models K(2, n)" (In Russian). *Visnyk NTUU "KPI" – Informatics, Operation and Computer Science*. 2004; 41: 85–92.

44. Romankevich, V. A., Potapova, E. R., Bakhtari Kh. & Nazarenko, V. V. "GL-model of behavior of fault-tolerant multiprocessor systems with a minimal number of lost edges" (In Russian). *Visnyk NTUU "KPI" – Informatics, Operation and Computer Science*. 2006; 45: 93–100.

45. Romankevitch, A. M., Morozov, K. V., Mykytenko, S. S. & Kovalenko, O. P. "On the cascade GL-model and its properties". *Applied Aspects of Information Technology*. 2022; 5 (3): 256–271. DOI: https://doi.org/10.15276/aait.05.2022.18.

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 20.04.2025
Received after revision 12.06.2025
Accepted 19.06.2025

DOI: https://doi.org/10.15276/hait.08.2025.16
УДК 004.05

Про обчислювальну складність каскадних GL-моделей 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
Романкевич Олексій Михайлович[1)]
ORCID: https://orcid.org/0000-0001-5634-8469; romankev@scs.kpi.ua. Scopus Author ID: 6602114176
Нікішин Єгор Олексійович[1)]
ORCID: https://orcid.org/0009-0008-9772-0261 ; yehor.nikishyn@gmail.com
[1)] Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського», пр. Берестейський, 37. Київ, 03056, Україна

АНОТАЦІЯ

Роботу присвячено проблемі оцінки обчислювальної складності базових каскадних GL-моделей поведінки відмовостійких багатопроцесорних систем у потоці відмов. Метою роботи є зменшення складності таких моделей шляхом вибору їх параметрів. Показано, що одній системі зазвичай може відповідати ціле сімейство каскадних GL-моделей, які відрізняються глибиною та параметрами каскаду, причому кожна з них має власну обчислювальну складність. З метою спрощення процесу моделювання поведінки системи у потоці відмов доцільно обирати таку конфігурацію каскадної GL-моделі, яка має найменшу складність. Водночас необхідно враховувати можливі додаткові обмеження на модель (наприклад, обмеження на глибину каскаду). У роботі застосовано емпірико-аналітичний метод дослідження. Здійснено аналіз обчислювальної складності каскадних GL-моделей: за допомогою спеціально розробленого програмного забезпечення проведено автоматизовану побудову моделей для різних комбінацій параметрів, після чого виконано порівняння складності виразів їхніх реберних функцій з метою виявлення залежностей від значень параметрів. Експериментальні дослідження проведено для відмовостійких багатопроцесорних систем із різною кількістю процесорів і різною максимально допустимою кратністю відмов (але не більшою за половину загальної кількості процесорів у системі). Показано, що каскадні GL-моделі зазвичай мають суттєво нижчу обчислювальну складність порівняно зі звичайними базовими GL-моделями, особливо для систем із невеликою максимально допустимою кратністю відмов. Водночас у випадках, коли ця кратність дорівнює або перевищує половину кількості процесорів, звичайні моделі можуть виявитися менш складними. На основі проведеного аналізу вперше сформульовано практичні рекомендації щодо вибору параметрів каскадної GL-моделі. Зокрема, найменшої складності вдається досягти, коли на кожному рівні каскаду коефіцієнт відмовостійкості допоміжної моделі є мінімальним можливим, проте в цьому випадку глибина каскаду стає максимальною. Якщо ж глибина каскаду обмежена, найменша складність досягається за умов рівномірного або близького до рівномірного розподілу коефіцієнтів відмовостійкості допоміжних моделей (якщо рівномірного розподілу досягти неможливо, доцільно розміщувати коефіцієнти з більшими значеннями на останніх рівнях каскаду). Результати проведених експериментів

демонструють, що застосування запропонованих рекомендацій дозволяє суттєво знизити загальну складність виразів реберних функцій каскадної GL-моделі порівняно з базовою GL-моделлю, причому ефективність підходу зростає зі збільшенням розмірів системи.

Ключові слова: каскадні GL-моделі; відмовостійкі багатопроцесорні системи; обчислювальна складність; базові системи; вибір параметрів; оцінка надійності

ABOUT THE 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, Beresteiskyi 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 та спеціалізованих комп’ютерних систем, Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського», пр. Берестейський, 37, Київ, 03056, Україна

Kostiantyn V. Morozov – Candidate of Engineering Sciences, Assistant at System Programming and Specialized Computer System Department. National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”, 37, Beresteiskyi 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 та спеціалізованих комп’ютерних систем, Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського», пр. Берестейський, 37, Київ, 03056, Україна

Alexei M. Romankevich – Doctor of Engineering Sciences, Professor, Professor at System Programming and Specialized Computer System Department. National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”, 37, Beresteiskyi Ave. Kyiv, 03056, Ukraine
ORCID: https://orcid.org/0000-0001-5634-8469, romankev@scs.kpi.ua, Scopus Author ID: 6602114176
***Research field*:** Fault-tolerant multiprocessor systems reliability estimation, GL-models, Self-diagnosable systems, Diagnosis of multiprocessor systems, Multi-valued logic.

Романкевич Олексій Михайлович – доктор технічних наук, професор, професор кафедри Системного програмування та спеціалізованих комп’ютерних систем. Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського», пр. Берестейський, 37, Київ, 03056, Україна

Yehor O. Nikishyn – student, System Programming and Specialized Computer System Department. National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”, 37, Beresteiskyi Ave. Kyiv, 03056, Ukraine
ORCID: https://orcid.org/0009-0008-9772-0261; y@gmail.com
***Research field*:** GL-models, Fault-tolerant multiprocessor systems reliability estimation, Discrete mathematics.

Нікішин Єгор Олексійович – студент, кафедра Системного програмування та спеціалізованих комп’ютерних систем, Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського», пр. Берестейський, 37, Київ, 03056, Україна