DOI: https://doi.org/10.15276/aait.05.2022.18
UDC 004.05

On the cascade GL-model and its properties

Alexei M. Romankevitch[1)]
ORCID: https://orcid.org/0000-0001-5634-8469; romankev@scs.kpi.ua. Scopus Author ID: 6602114176
Kostiantyn V. Morozov[1)]
ORCID: https://orcid.org/0000-0003-0978-6292, mcng@ukr.net. Scopus Author ID: 57222509251
Serhii S. Mykytenko[1)]
ORCID: https://orcid.org/0000-0001-8842-004X; sergei.mykytenko@gmail.com
Olena P. Kovalenko[1)]
ORCID: https://orcid.org/0000-0002-2548-2347; kop100298@gmail.com
[1)] National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute", 37, Peremogy Av. Kyiv, 03056, Ukraine

ABSTRACT

The article proposes a new direction for the further development of GL-models – models on the basis of which performs the calculation of the reliability parameters of fault-tolerant multiprocessor systems. Such models reflect the reaction of the system to the appearance of failures of arbitrary multiplicity. The essence of the new direction is the construction of a model by composition of several basic GL-models in such a way that the values of the edge functions of one model form the input vector of the next one. This article shows that the model obtained in this way, which is proposed to be called cascade model, will also be basic and, in general case, can consist of an arbitrary number of submodels. This article gives a formula that allows one to determine the value of the degree of fault tolerance of the cascade model, depending on the values of the levels of fault tolerance of its component submodels. This article shows that the graphs of both the cascade and regular models are cyclic and have the same number of edges. At the same time, despite the fact that the intermediate submodels also have graphs, their presence does not increase the complexity of the model as a whole, since only the expressions of the edge functions are used in them. This article contains examples that confirm the correctness of the theoretically obtained results, and it also shows that the cascade model, at least in some cases, has lower computational complexity (the total number of logical operations in the expressions of edge functions) compared to the basic model. It was found that although the cascade model is basic, the sets of edges it loses and the regular basic GL-model on some input vectors may differ. In certain cases, several alternative cascade models can be built, which will differ in their parameters, but will have the same resulting value of the degree of fault tolerance. Given an example, where the properties of such alternative cascade models are compared. It was found that such models differ both in computational complexity and, in some cases, in the sets of edges they lose on certain input vectors. The possibility of modifying the cascade model was shown by changing the expressions of the edge functions of its component submodels, both individually and several simultaneously. At the same time, it is possible to block vectors with an increased multiplicity of zeros. A number of tasks for future research were formulated.

Keywords: Cascade GL-models; fault-tolerant multiprocessor systems; modification of GL-models; calculation of reliability parameters

***For citation*: 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; Vol. 5 No. 3. 256–271. DOI: https://doi.org/10.15276/aait.05.2022.18**

INTRODUCTION

In the modern world, constantly increasing amounts of various devices and systems are becoming automated or completely automatic [1]. Management of their operation is partially or completely based on a special system, the so-called control system (CS) [2]. It receives signals from various sensors, processes them and generates control signals according to a certain algorithm.

At the same time, in some cases, especially for complex systems, the computational complexity of the tasks performed by the CS can be very high. In addition, an important property is the reliability of the control system [3], because its failure can lead, at best, the controlled object's operation to a halt, and sometimes to an accident.

This is especially important for so-called critical application systems (CAS). [4,5], [6], the refusal of which can lead to significant material losses, threaten the life or health of people, the security of the state, cause significant damage to the environment, etc. (for example, power plants, military equipment [7], space vehicles [8, 9], complex production processes, aviation [10], railway, and recently some types of personal transport [11] etc.).

Both of the above-mentioned problems can be solved by using the so-called fault-tolerant multiprocessor systems (FTMS) as CS [12, 13].

Such systems often contain a large number of processors and can continue full operation even if some of them fail.

ANALYSIS OF THE LITERATURE DATA AND STATEMENT OF THE PROBLEM

One of the tasks faced by the FTMS developer is the assessment of its reliability parameters, for example, such as the probability of failure-free operation during the specified time. These values are needed to evaluate the properties of the system at the development stage, for example, in order to identify its most vulnerable places, and to confirm its compliance with the specified criteria at the implementation stage [14]. There are two main approaches to solving this problem [15, 16], [17, 18].

The first one is based on the construction of analytical expressions that accurately or approximately allows the determination of system reliability parameters [19, 20], [21, 22], [23, 24], [25]. One of the disadvantages of this approach is its limited application: each system usually requires the development of its own approach and a set of formulas [26, 27], [28, 29], [30, 31, [32].

The second approach boils down to determining system reliability parameters by conducting statistical experiments with models of system behavior in the flow of failures [33, 34]. One of the disadvantages of this approach is that the parameters are determined only with a certain accuracy, which usually depends on the number of experiments conducted with the models. Thus, to increase the accuracy of the estimate, it is desirable to increase the number of experiments, which can be achieved, in particular, by reducing the complexity of the model.

PURPOSE AND OBJECTIVES OF THE RESEARCH

As models of FTMS behavior in the flow of failures can be used the so-called graph-logic models (GL-models) [34, 35]. This approach is universal, that is, it allows building a model for any system.

The basis of the GL-model is an undirected graph, each edge of which corresponds to a boolean edge function. The arguments of the edge functions are the elements of the Boolean vector of the system state, each element of which corresponds to a certain processor of the system and takes the value 1 if this processor is healthy, or 0 if it has failed.

If the edge function takes a zero value, the edge corresponding to it is removed from the graph. The connectivity of the model graph, in turn, corresponds to the performance of the system. Thus, to evaluate the behavior of the system in the flow of failures, it is enough to calculate the values of all edge functions of the model, and then determine the connectivity of the resulting graph.

The authors suggest that the developer can use controlled generators of pseudorandom vectors to use them as input vectors for models. After that, by using the methods of mathematical statistics, the values of the probability of fault-free operation of the system are being estimated with a certain accuracy.

Among FTMS, it is worth noting those that are resistant precisely to failures of a certain multiplicity, that is, they remain operational until no more than a certain number of any processors fail. Such systems, as well as GL-models corresponding to them, are called basic and denoted $K(m, n)$, where n is the number of processors in the system, and m is the maximum allowable number of failures during which the system will remain operational. All other systems and their corresponding models are called non-basic.

There are a number of different methods for building GL-models [36-38], among them it is worth noting [37]. Basic model $K(m, n)$, constructed by this method is based on a cyclic graph with $n - m + 1$ edge and loses exactly two edges on the vectors with $m + 1$ zero. Using a cyclic graph allows to make procedure of checking the connectivity of the graph trivial.

The creation of methods for building new types of GL-models, as well as modifications of existing models, remain relevant, in particular, with the aim of simplifying the process of building models, reducing the computational complexity of the models, building models of non-basic systems, etc.

CASCADE MODELS AND THEIR PROPERTIES

GL-model $K(m, n)$, built according to [37] in the column will contain exactly $N = n - m + 1$ edges. To construct the expressions of its edge functions, the input vector is divided into 2 parts. In the general case, such a division can be arbitrary, but in practice it is advisable to divide it into two equal or almost equal parts.

A set of the model edge functions $K(m, n)$ built in accordance with [37] will look like that:

$$\begin{array}{l} K_1(m,n_1) \\ \kappa_1(m-1,n_1)\vee\kappa_2(1,n_2) \\ \kappa_1(m-2,n_1)\vee\kappa_2(2,n_2) \\ \vdots \\ \kappa_1(m-i,n_1)\vee\kappa_2(i,n_2) \\ \vdots \\ \kappa_1(2,n_1)\vee\kappa_2(m-2,n_2) \\ \kappa_1(1,n_1)\vee\kappa_2(m-1,n_2) \\ K_2(m,n_2) \end{array}, \quad (1)$$

where $K_1(m, n_1)$ and $K_2(m, n_2)$ – edge functions of similar models constructed, if possible, for the corresponding parts of the input vector, and $\kappa_1(i, n_1)$ with $\kappa_2(j, n_2)$ are conjunctions of expressions of edge functions of the models $K_1(i, n_1)$ and $K_2(j, n_2)$, constructed for the corresponding parts of the input vector.

Thus, the construction process is recursive and ends with the construction of trivial models $K(1, 1)$, each of which contains exactly one edge function of the type $f = x_k$, where x_k is a certain element of the input vector.

In addition, in [39] it was proved that on any vector with l zeros she will lose exactly L edges, where

$$L=\begin{cases}0, \text{ when } l<m \\ l-m+1, \text{ when } l\geq m\end{cases}. \quad (2)$$

Note: the loss of an edge by the GL-model occurs due to the fact that the corresponding edge function takes a zero value. Therefore, on vectors with l zeros in the *K(m, n)* model, exactly L child functions will take a value equal to zero. Let's combine the values of the edge functions of this model into a vector, denoted as v.

Let's build model $K'(M, N)$ in accordance with [37], which as input accepts vector v. In accordance with [39] on vectors with L zeros it losses exactly λ edges, where:

$$\lambda=\begin{cases}0, \text{ when } L<M \\ L-M+1, \text{ when } L\geq M\end{cases}. \quad (3)$$

In accordance with (2) we have: $L < M \sim$ $\sim l - m + 1 < M \sim l < M + m - 1$.

Similarly and $L \geq M \sim l \geq M + m - 1$.

In addition,
$L - M + 1 == l - (M + m - 1) + 1$.
Thus, we can rewrite (3) as

$$\lambda=\begin{cases}0, \text{ when } l<M+m-1 \\ L-(M+m-1)+1, \text{ when } l\geq M+m-1\end{cases}. \quad (4)$$

Let's denote $\mu = M + m - 1$. In that case (4) will take the form:

(1)
$$\lambda=\begin{cases}0, \text{ when } l<\mu \\ L-\mu+1, \text{ when } l\geq\mu\end{cases}. \quad (5)$$

Let's note that, as shown in [37] model $K'(M, N)$ will have exactly ν edges, where

$$\begin{aligned}\nu &= N - M + 1 = (n-m+1) - M + 1 = \\ &= n - (M+m-1) + 1 = n - \mu + 1\end{aligned}. \quad (6)$$

Let's also consider the model $\Lambda(\mu, n)$, built according to [37]. It is easy to spot that it will also have ν edges, and on vectors with l zeros it will also lose exactly λ edges (in accordance with (5)). Thus, properties of the models K' and Λ in terms of the number of edges, as well as the number of edges that are lost on vectors with a certain multiplicity of zeros, coincide.

The procedure described above, namely, using the values of the edge functions of one model as the input vector of the next one, can be repeated an arbitrary number of times. Thus, a model can be obtained, which we will call a *cascade* model. Let's denote the value of the parameters of the degree of fault tolerance of each of the models of the *cascade* model as $m_1, m_2, \ldots, m_T$, where T is the number of those models. Let's denote this model as $K([m_1, m_2, \ldots, m_T], n)$. Number of T models in a cascade model will be called its *depth*.

One can see that the resulting model will have properties (in terms of the number of edges, as well as the number of edges that are lost on vectors with a certain multiplicity of zeros) similar to the properties of the model $\Lambda(\mu, n)$,
where

$$\mu=\sum_{i=1}^{T} m_i - T + 1. \quad (7)$$

EXAMPLE

For this example, let's build a model of a basic multiprocessor system, which contains 8 processors and is resistant to the failure of any 3 of them. A model of such a system, built according to [37], let's denote it as $K(3, 8)$, will contain the next edge functions:

$$\begin{aligned} &f_1 = x_1 \vee x_2 \vee x_3 x_4 \\ &f_2 = x_1 x_2 \vee x_3 \vee x_4 \\ &f_3 = (x_1 \vee x_2)(x_1 x_2 \vee x_3 x_4)(x_3 \vee x_4) \vee \\ &\vee x_5 x_6 x_7 x_8 \\ &f_4 = x_1 x_2 x_3 x_4 \vee \\ &\vee (x_5 \vee x_6)(x_5 x_6 \vee x_7 x_8)(x_7 \vee x_8) \\ &f_5 = x_5 \vee x_6 \vee x_7 x_8 \\ &f_6 = x_5 x_6 \vee x_7 \vee x_8 \end{aligned} \quad (8)$$

Note that x_1, x_2, …, x_8 here denote the elements of the input vector of the system state.

This model will correspond to the cyclic graph presented on Fig. 1.

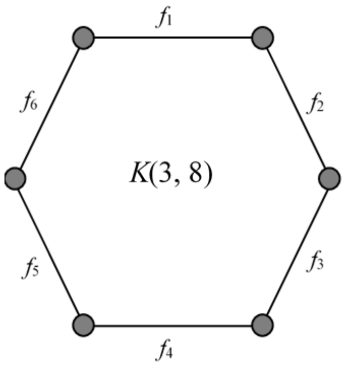


Fig. 1. **Model *K* (3, 8)**
***Source:* compiled by the authors**

Let's also build a cascade model $K'([2, 2], 8)$ for this system. To do this, we will first build a model $K_1(2, 8)$ in accordance with [37], that will have the next edge functions:

$$\begin{aligned} &y_1 = x_1 \vee x_2 \\ &y_2 = x_1 x_2 \vee x_3 x_4 \\ &y_3 = x_3 \vee x_4 \\ &y_4 = x_1 x_2 x_3 x_4 \vee x_5 x_6 x_7 x_8 \\ &y_5 = x_5 \vee x_6 \\ &y_6 = x_5 x_6 \vee x_7 x_8 \\ &y_7 = x_7 \vee x_8 \end{aligned} \quad (9)$$

Next, using the obtained values of the edge functions as an input vector $\langle y_1, y_2, \ldots, y_7 \rangle$, let us build model $K_2(2, 7)$ in accordance with [37], that will have the next edge functions:

$$\begin{aligned} &\varphi_1 = y_1 \vee y_2 \\ &\varphi_2 = y_1 y_2 \vee y_3 y_4 \\ &\varphi_3 = y_3 \vee y_4 \\ &\varphi_4 = y_1 y_2 y_3 y_4 \vee y_5 y_6 y_7 \\ &\varphi_5 = y_5 \vee y_6 \\ &\varphi_6 = y_5 y_6 \vee y_7 \end{aligned} \quad (10)$$

The resulting cascade model is presented on Fig. 2.

Experimental data confirm the adequacy of both models: both of them do not lose a single edge on vectors with less than two zeros, and on vectors with more zeros they lose exactly $l - 2$ edges each, where l is number of zeros in the input vector.

However, it is worth noting that the combinations of edges that each model loses on some input vectors may differ. Thus, on vector <10101100> model $K(3, 8)$ will lose edges, corresponding to edge functions f_3 and f_4. At the same time, an intermediate model $K_1(2, 8)$ will lose edges corresponding to edge functions y_2, y_4 and y_7. Accordingly, the model $K_2(2, 7)$, and, therefore, the model $K'([2, 2], 8)$, will lose edges corresponding

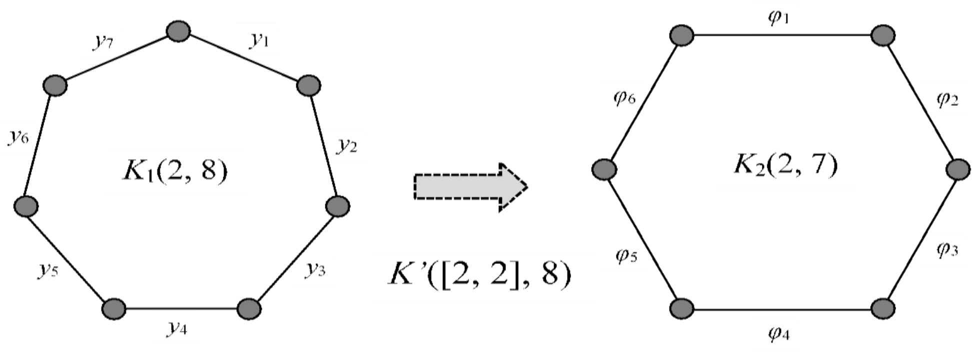


Fig. 2. C**ascade model *K'*([2, 2], 8)**
***Source:* compiled by the authors**

to edge functions φ_2 and φ_4 (Fig. 3a). On the other side, on vector <11001100> the model $K(3, 8)$ will also lose edges, that correspond to edge functions f_3 and f_4. Intermediate model $K_1(2, 8)$ in turn, it will lose the edges corresponding to the edge functions y_3, y_4 and y_7. And the model $K_2(2, 7)$ and, accordingly, the model $K'([2, 2], 8)$, will lose edges that correspond to edge functions φ_3 and φ_4 (Fig. 3b). This means that for the considered example, the same behavior of the models in terms of sets of edges that are lost on different input vectors cannot be achieved by permuting these edges.

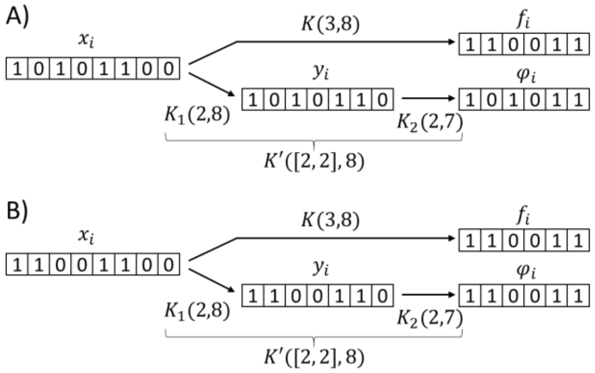


Fig. 3. **Behavior of the models K (3, 8) and K'([2, 2], 8) on different input vectors: a – K (3, 8); b – K'([2, 2], 8)**
Source: **compiled by the authors**

We also compare the complexity of the considered models, namely the number of operations that must be performed to calculate their edge functions.

So, the expressions of the edge functions of the model $K(3, 8)$ contain 16 disjunctions and 18 conjunctions. Expressions of edge functions of the model $K_1(2, 8)$ contain 7 disjunctions and 10 conjunctions, and expressions of edge functions of the model $K_2(2, 7)$ contain 6 disjunctions and 8 conjunctions. Thus, in general, to calculate the edge functions of the cascade model K' ([2, 2], 8) it is necessary to perform 13 disjunctions and 18 conjunctions, which is 3 operations less compared to the usual model.

The numbers of logical operations used in the edge functions of each of the models are given in Table 1.

It is also worth noting that the graphs of both models are the same – in both cases they are cyclic graphs with 6 vertices. In this case, the graph of the intermediate model can be ignored, since it is not actually used, and only the expressions of edge functions are taken from it.

Table 1. **The number of logical operations in the models from *Example 1***

Model	The number of conjunctions	The number of disjunctions	Number of operations
***K*(3,8)**	**18**	**16**	**34**
K_1(2,8)	10	7	17
K_2(2,7)	8	6	14
***K'*([2,2],8)**	**18**	**13**	**31**

Source: **compiled by the authors**

VARIABILITY OF CASCADE MODELS

It is worth noting that the model $K(1, n)$, built in accordance with [37] will have trivial expressions of edge functions of the form $f_j = x_j$. Thus, the submodels of the cascade model with values $m_i = 1$ are trivial and can simply be removed from it without changing its behavior. Therefore, when analyzing cascading GL-models, it makes sense to consider only those of them that have values m_i of each of the submodels is at least 2.

In accordance with (7), it is easy to notice that only for case $\mu = 3$ (as in above-mentioned example) it is possible to build only one version of the cascade model, which will consist of two sub-models, in which $m_1 = m_2 = 2$. Otherwise if μ is bigger, a few different variants of such a model can be built. At the same time, their behavior and complexity may also differ among themselves.

For example, consider a system that contains 10 processors and is resistant to the failure of no more than 4 of them. For such a system will correspond GL-model $K(4, 10)$. In accordance with [37] such a model will have 7 edge functions:

$$
\begin{aligned}
&f_1 = x_1 \vee x_2 \vee x_3 \vee x_4 x_5 \\
&f_2 = (x_1 \vee x_2)(x_1 x_2 \vee x_3) \vee x_4 \vee x_5 \\
&f_3 = (x_1 \vee x_2 \vee x_3) \wedge \\
&\wedge \left((x_1 \vee x_2)(x_1 x_2 \vee x_3) \vee x_4 x_5\right) \wedge \\
&\wedge (x_1 x_2 x_3 \vee x_4 \vee x_5) \vee x_6 x_7 x_8 x_9 x_{10} \\
&f_4 = (x_1 \vee x_2)(x_1 x_2 \vee x_3)(x_1 x_2 x_3 \vee x_4 x_5) \wedge \\
&\wedge (x_4 \vee x_5) \vee (x_6 \vee x_7)(x_6 x_7 \vee x_8) \wedge \\
&\wedge (x_6 x_7 x_8 \vee x_9 x_{10})(x_9 \vee x_{10}) \\
&f_5 = x_1 x_2 x_3 x_4 x_5 \vee (x_6 \vee x_7 \vee x_8) \wedge \\
&\wedge \left((x_6 \vee x_7)(x_6 x_7 \vee x_8) \vee x_9 x_{10}\right) \wedge \\
&\wedge (x_6 x_7 x_8 \vee x_9 \vee x_{10}) \\
&f_6 = x_6 \vee x_7 \vee x_8 \vee x_9 x_{10} \\
&f_7 = (x_6 \vee x_7)(x_6 x_7 \vee x_8) \vee x_9 \vee x_{10}
\end{aligned}
\quad . \qquad (11)
$$

Similarly to the previous example, x_1, x_2, …, x_{10} denote the elements of the system state vector.

This model will correspond to the cyclic graph presented on Fig. 4.

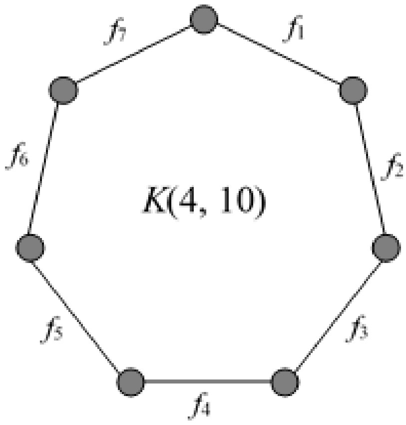


Fig. 4. **Model *K* (4, 10)**
Source: **compiled by the authors**

Note that a total of 39 disjunctions and 42 conjunctions are used in the child function expressions of the built model, in other words, a total of 81 logical operations.

It is possible to also build several different cascade models, each of which will correspond to the system under consideration, namely: K^1 ([2, 3], 10), K^2 ([3, 2], 10), K^3 ([2, 2, 2], 10).

Let's build model K^1([2, 3], 10). To do this, let's first build a model $K_1^1(2,10)$, that in accordance with [37] will have the next edge functions:

$$\begin{aligned}
y_1^1 &= x_1 \vee x_2 \\
y_2^1 &= x_1 x_2 \vee x_3 \\
y_3^1 &= x_1 x_2 x_3 \vee x_4 x_5 \\
y_4^1 &= x_4 \vee x_5 \\
y_5^1 &= x_1 x_2 x_3 x_4 x_5 \vee x_6 x_7 x_8 x_9 x_{10} \\
y_6^1 &= x_6 \vee x_7 \\
y_7^1 &= x_6 x_7 \vee x_8 \\
y_8^1 &= x_6 x_7 x_8 \vee x_9 x_{10} \\
y_9^1 &= x_9 \vee x_{10}
\end{aligned} . \tag{12}$$

After that, using the obtained values of edge functions as an input vector $\langle y_1^1, y_2^1, \ldots, y_9^1 \rangle$, let's build model $K_2^1(3,9)$. In accordance with [37] it will have the next edge functions:

$$\begin{aligned}
&\varphi_1^1 = y_1^1 \vee y_2^1 \vee y_3^1 \\
&\varphi_2^1 = \left(y_1^1 \vee y_2^1\right)\left(y_1^1 y_2^1 \vee y_3^1\right) \vee y_4^1 y_5^1 \\
&\varphi_3^1 = y_1^1 y_2^1 y_3^1 \vee y_4^1 \vee y_5^1 \\
&\varphi_4^1 = \left(y_1^1 \vee y_2^1\right)\left(y_1^1 y_2^1 \vee y_3^1\right)\left(y_1^1 y_2^1 y_3^1 \vee y_4^1 y_5^1\right) \wedge \\
&\wedge \left(y_4^1 \vee y_5^1\right) \vee y_6^1 y_7^1 y_8^1 y_9^1 \\
&\varphi_5^1 = y_1^1 y_2^1 y_3^1 y_4^1 y_5^1 \vee \left(y_6^1 \vee y_7^1\right) \wedge \\
&\wedge \left(y_6^1 y_7^1 \vee y_8^1 y_9^1\right)\left(y_8^1 \vee y_9^1\right) \\
&\varphi_6^1 = y_6^1 \vee y_7^1 \vee y_8^1 y_9^1 \\
&\varphi_7^1 = y_6^1 y_7^1 \vee y_8^1 \vee y_9^1
\end{aligned} . \tag{13}$$

The resulting cascade model is shown on Fig. 5.

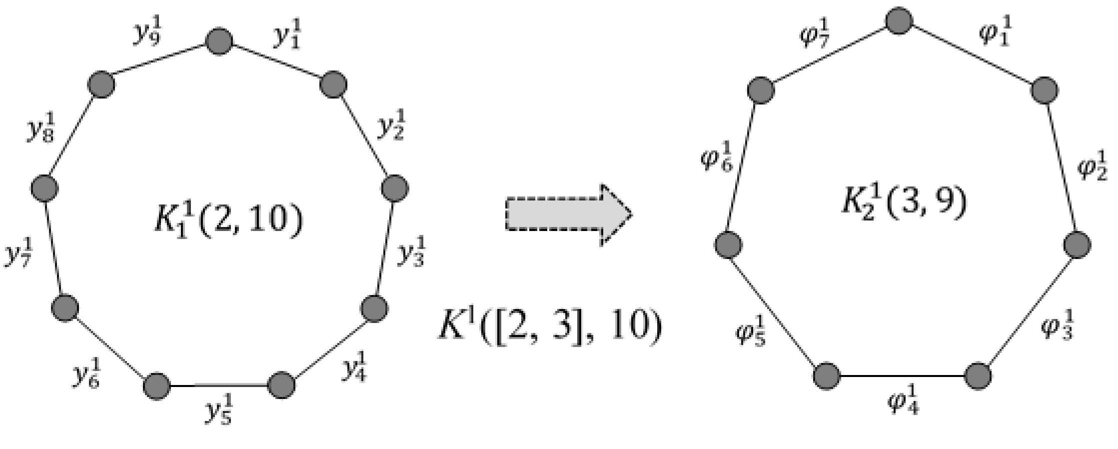


Fig. 5. **Cascade model K^1 ([2, 3], 10)**
Source: **compiled by the authors**

Let's count the number of logical operations used in the edge functions of this model. Edge functions of the models $K_1^1(2,10)$ contain 9 disjunction and 16 conjunctions, thus, a total of 25 logical operations. At the same time, edge functions of the model $K_2^1(3,9)$ include 20 disjunctions and 25 conjunctions, in other words 45 logical operations. Thus, in edge functions of the model $K^1([2, 3], 10)$ there are 29 disjunctions and 41 conjunctions, in other words 70 logical operations.

Now, let's build a cascade model $K^2([3, 2], 10)$. As a first step let's build model $K_1^2(3,10)$, that in accordance with [37] will have next edge functions:

$$
\begin{aligned}
y_1^2 &= x_1 \vee x_2 \vee x_3 \\
y_2^2 &= (x_1 \vee x_2)(x_1x_2 \vee x_3) \vee x_4x_5 \\
y_3^2 &= x_1x_2x_3 \vee x_4 \vee x_5 \\
y_4^2 &= (x_1 \vee x_2)(x_1x_2 \vee x_3)(x_1x_2x_3 \vee x_4x_5) \wedge \\
&\wedge (x_4 \vee x_5) \vee x_6x_7x_8x_9x_{10} \\
y_5^2 &= x_1x_2x_3x_4x_5 \vee (x_6 \vee x_7)(x_6x_7 \vee x_8) \wedge \\
&\wedge (x_6x_7x_8 \vee x_9x_{10})(x_9 \vee x_{10}) \\
y_6^2 &= x_6 \vee x_7 \vee x_8 \\
y_7^2 &= (x_6 \vee x_7)(x_6x_7 \vee x_8) \vee x_9x_{10} \\
y_8^2 &= x_6x_7x_8 \vee x_9 \vee x_{10}
\end{aligned} \quad . \tag{14}
$$

As the next step, in accordance with [37] let's build model $K_2^2(2,8)$, while using as input vectors received on the previous step values of the edge functions $\langle y_1^2, y_2^2, \ldots, y_8^2 \rangle$. Let's note that expressions of edge functions of the model that's being built, with precision to renaming of input and output variables will correspond to expressions (9).

The resulting expressions will take form of:

$$
\begin{aligned}
\varphi_1^2 &= y_1^2 \vee y_2^2 \\
\varphi_2^2 &= y_1^2y_2^2 \vee y_3^2y_4^2 \\
\varphi_3^2 &= y_3^2 \vee y_4^2 \\
\varphi_4^2 &= y_1^2y_2^2y_3^2y_4^2 \vee y_5^2y_6^2y_7^2y_8^2 \\
\varphi_5^2 &= y_5^2 \vee y_6^2 \\
\varphi_6^2 &= y_5^2y_6^2 \vee y_7^2y_8^2 \\
\varphi_7^2 &= y_7^2 \vee y_8^2
\end{aligned} \quad . \tag{15}
$$

The build cascade model is presented on Fig. 6.

Let's also count the number of logical operations used in the expressions of edge functions of the model. Edge functions of the model $K_1^2(3,10)$ contain 24 disjunctions and 32 conjunctions, in other words, a total of 56 logical operations. Edge functions of the model $K_2^2(2,8)$ include 7 disjunctions and 10 conjunctions, in other words, a total of 17 logical operations. Thus, edge functions of the model $K^2([3, 2], 10)$ contain 31 disjunctions and 42 conjunctions, in other words, 73 logical operations.

As a final step, let's build model $K^3([2, 2, 2], 10)$. For this at first we build model $K_1^3(2,10)$.

In accordance with [37] it will contain edge functions, expressions of which, as one can notice, will correspond to (12), namely:

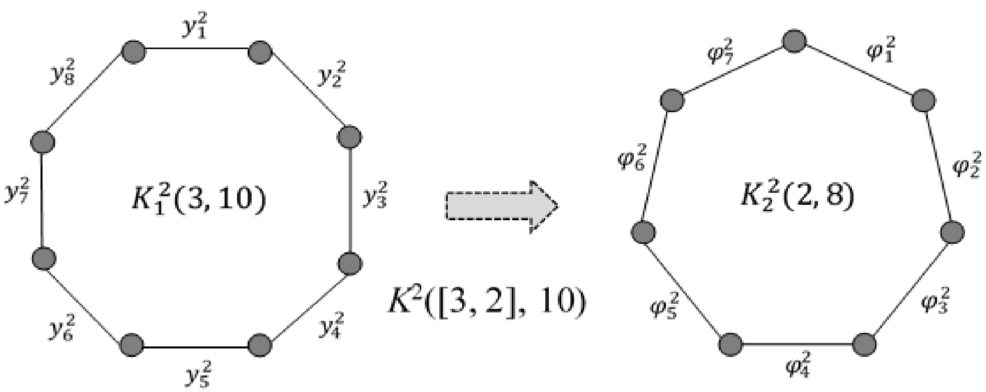


Fig. 6. **Cascade model K^2 ([3, 2], 10)**
Source: **compiled by the authors**

$$\begin{aligned}
y_1^3 &= x_1 \vee x_2 \\
y_2^3 &= x_1x_2 \vee x_3 \\
y_3^3 &= x_1x_2x_3 \vee x_4x_5 \\
y_4^3 &= x_4 \vee x_5 \\
y_5^3 &= x_1x_2x_3x_4x_5 \vee x_6x_7x_8x_9x_{10} \\
y_6^3 &= x_6 \vee x_7 \\
y_7^3 &= x_6x_7 \vee x_8 \\
y_8^3 &= x_6x_7x_8 \vee x_9x_{10} \\
y_9^3 &= x_9 \vee x_{10}
\end{aligned} \quad (16)$$

Next let's build model $K_2^3(2,9)$, for which let's formulate input vector from received on the previous step values of edge functions $\langle y_1^3, y_2^3, \ldots, y_9^3 \rangle$. In accordance with [37]:

$$\begin{aligned}
z_1^3 &= y_1^3 \vee y_2^3 \\
z_2^3 &= y_1^3y_2^3 \vee y_3^3 \\
z_3^3 &= y_1^3y_2^3y_3^3 \vee y_4^3y_5^3 \\
z_4^3 &= y_4^3 \vee y_5^3 \\
z_5^3 &= y_1^3y_2^3y_3^3y_4^3y_5^3 \vee y_6^3y_7^3y_8^3y_9^3 \\
z_6^3 &= y_6^3 \vee y_7^3 \\
z_7^3 &= y_6^3y_7^3 \vee y_8^3y_9^3 \\
z_8^3 &= y_8^3 \vee y_9^3
\end{aligned} \quad (17)$$

Finally, let's build model $K_3^3(2,8)$, input vector for which is formulated from values of edge functions of the previous model, in other words $\langle z_1^3, z_2^3, \ldots, z_8^3 \rangle$. Let's formulate expressions of edge functions of this model in accordance with [37]. One can notice that they correspond to expressions (15) with precision to change the names of the input variables. Namely:

$$\begin{aligned}
\varphi_1^3 &= z_1^3 \vee z_2^3 \\
\varphi_2^3 &= z_1^3z_2^3 \vee z_3^3z_4^3 \\
\varphi_3^3 &= z_3^3 \vee z_4^3 \\
\varphi_4^3 &= z_1^3z_2^3z_3^3z_4^3 \vee z_5^3z_6^3z_7^3z_8^3 \\
\varphi_5^3 &= z_5^3 \vee z_6^3 \\
\varphi_6^3 &= z_5^3z_6^3 \vee z_7^3z_8^3 \\
\varphi_7^3 &= z_7^3 \vee z_8^3
\end{aligned} \quad (18)$$

The resulting cascade model is shown on Fig. 7.

Let's calculate the number of logical operations, used in expression of edge functions, and for this model. Edge functions of the model $K_1^3(2,10)$ have 9 disjunction and 16 conjunctions, in other words, 25 logical operations. Edge functions of the model $K_2^3(2,9)$ have 8 disjunction and 13 conjunctions, in other words, a total of 21 logical operations. Edge functions of the model $K_3^3(2,8)$ includes 7 disjunction and 10 conjunctions, in other words, a total of 17 logical operations. Thus, edge functions of the model $K^3([2, 2, 2], 10)$ have 24 disjunctions and 39 conjunctions, in other words, 63 logical operations.

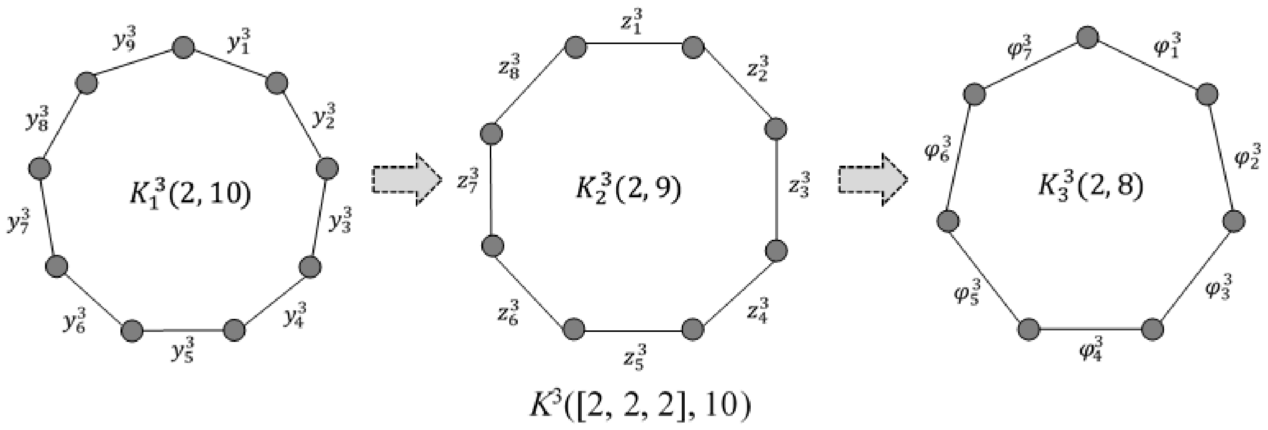


Fig. 7. **Cascade model K^2([3, 2], 10)**
***Source:* compiled by the authors**

Number of logical operations that are being used in edge functions of each model is shown in the Table 2.

Similarly to the previous example, the graphs of all models are the same – in all cases they are cyclic graphs with 7 nodes. Graphs of intermediate models and their number can be ignored, since, as mentioned before, they are not actually used, and only the expressions of their edge functions are taken from them.

From Table 1 and Table 2 one can see that cascade models in all of the considered cases required the use of a smaller number of logical operations compared to conventional basic models. Whether this is true for all cases requires further research. It can also be noticed that in some cases the cascade model contains a smaller number of both disjunctions and conjunctions compared to the usual one, and in others only the number of disjunctions is smaller, and the number of conjunctions remains the same. The study of this fact can also become the topic of future works. Further work may concern the search for parameters of the optimal partition of the cascade of the model, in other words, parameter values m_i, which make it possible to obtain a model that allows the use of the smallest number of logical operations.

Table 2. **Number of logical operations in models from *Example 2***

Model	Number of conjunctions	Number of disjunctions	Number of operations
K(4,10)	**42**	**39**	**81**
$K_1^1(2,10)$	16	9	25
$K_2^1(3,9)$	25	20	45
K^1([2,3],10)	**41**	**29**	**70**
$K_1^2(3,10)$	32	24	56
$K_2^2(2,8)$	10	7	17
K^2([3,2],10)	**42**	**31**	**73**
$K_1^3(2,10)$	16	9	25
$K_2^3(2,9)$	13	8	21
$K_3^3(2,8)$	10	7	17
K^3([2,2,2],10)	**39**	**24**	**63**

Source: **compiled by the authors**

Experimental data confirms the adequacy of all four models from *Example 2*: all of them do not lose a single edge on vectors with less than three zeros and on vectors with more zeros they lose exactly l - 3 edges, where l is number of zeros in the input vector.

However, as in the previous example, the combinations of edges that each model loses on some input vectors may differ.

Thus, on vector <1001011100> model K(4, 10) loses edges, that correspond to edge functions f_3 and f_4. At the same time, intermediate model $K_1^1(2,10)$ will lose edges, that correspond to edge functions y_2^1, y_3^1, y_5^1 and y_9^1. Therefore, model $K_2^1(3,9)$, and, thus, and model K^1([2, 3], 10), will lose edges, that correspond to edge functions φ_2^1 and φ_4^1 (Fig. 8a). On the other hand, on vector <1100011100> model K(4, 10) will also lose edges, that correspond to edge functions f_3 and f_4. At the same time, intermediate model $K_1^1(2,10)$ will lose edges, that correspond to edge functions y_3^1, y_4^1, y_5^1 and y_9^1, thus model $K_2^1(3,9)$ and, therefore, model K^1([2, 3], 10), will lose edges, that correspond to functions φ_3^1 and φ_4^1 (Fig. 8b).

On vector <1110001001> model K(4, 10) will lose edges, that correspond to edge functions f_4 and f_5. At the same time, intermediate model $K_1^2(3,10)$ will lose edges, that correspond to edge functions y_4^2, y_5^2 and y_7^2. Models $K_2^2(2,8)$ and K^2([3, 2], 10) will lose edges, that correspond to edge functions φ_4^2 and φ_6^2 (Fig. 9a). However, on vector <1011000011> model K(4, 10) will also lose edges, that correspond to edge functions f_4 and f_5. On the other hand, intermediate model $K_1^2(3,10)$ will lose edges, that correspond to edge functions y_4^2, y_5^2 and y_6^2. Meanwhile model $K_2^2(2,8)$ and model K^2([3, 2], 10) will lose edges, that correspond to edge functions φ_4^2 and φ_5^2 (Fig. 9b).

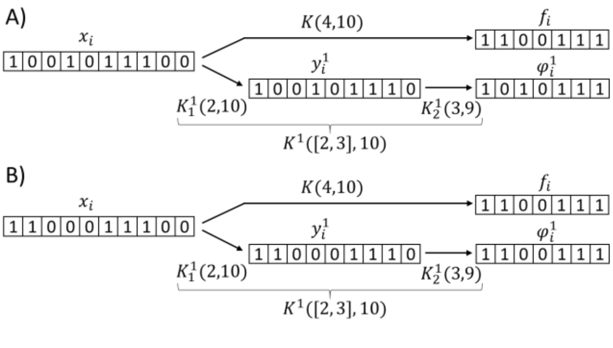


Fig. 8. **Behaviour of the models K(4, 10) and K^1([2, 3], 10) for different input vectors: a – K(4, 10); b – K^1([2, 3], 10)**
Source: **compiled by the authors**

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

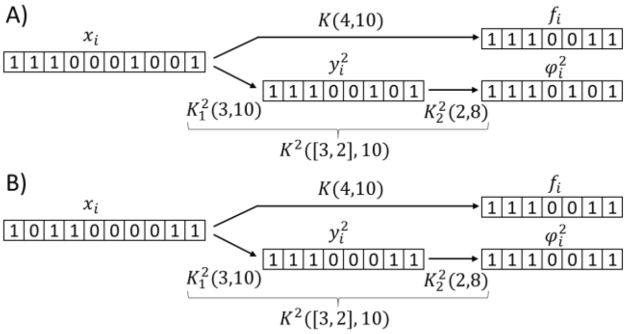


Fig. 9. **Behaviour of the models *K*(4, 10) and *K*²([3, 2], 10) on different input vectors: a – *K*(4, 10); b – *K*²([3, 2], 10)**
***Source:* compiled by the authors**

Similarly, on the vector <1110000101> model $K(4, 10)$ will also lose the edges corresponding to the edge functions f_4 and f_5. Meanwhile intermediate model $K_1^3(2,10)$ will lose edges, that correspond to edge functions y_4^3, y_5^3, y_6^3 and y_8^3, while model $K_2^3(2,9)$ will lose edges, that correspond to the functions z_4^3, z_5^3 and z_7^3. Therefore, model $K_3^3(2,8)$ and $K^3([2, 2, 2], 10)$ will lose edges, that correspond to edge functions φ_4^3 and φ_6^3 (Fig. 10a). Yet for input vector <1010100011> model $K(4, 10)$ will also lose edges, that correspond to edge functions f_4 and f_5. Meanwhile intermediate model $K_1^3(2,10)$ on that vector will lose edges, that correspond to functions y_3^3, y_5^3, y_6^3 and y_7^3. Model $K_2^3(2,9)$ accordingly will lose edges, that correspond to functions z_3^3, z_5^3 and z_6^3. Meanwhile model $K_3^3(2,8)$ and $K^3([2, 2, 2], 10)$ will lose edges, that correspond to edge functions φ_4^3 and φ_5^3 (Fig. 10b).

Similarly to the previous example, the same behavior of models in terms of sets of edges that are lost on different input vectors cannot be achieved by permuting these edges.

Let's also analyze the behavior of cascade models K^1, K^2 and K^3 on different input vectors. On vector <1111010000> intermediate model $K_1^3(2,10)$ will lose edges, that correspond to functions y_5^1, y_7^1, y_8^1 and y_9^1. In turn, model $K_2^1(3,9)$ and $K^1([2, 3], 10)$ will lose edges, that correspond to edge functions φ_5^1 and φ_7^1. On the other hand, intermediate model $K_1^2(3,10)$ will lose edges, that correspond to edge functions y_5^2, y_7^2 and y_8^2, meanwhile model $K_2^2(2,8)$ and $K^2([3, 2], 10)$ will lose edges, that correspond to edge functions φ_6^2 and φ_7^2 (Fig. 11a). Yet on vector <1111100000> intermediate model $K_1^3(2,10)$ will lose edges, that correspond to functions y_6^1, y_7^1, y_8^1 and y_9^1.

Accordingly, model $K_2^1(3,9)$ and $K^1([2, 3], 10)$ will lose edges, that correspond to edge functions φ_6^1 and φ_7^1. On the same vector, intermediate model $K_1^2(3,10)$ will lose edges, that correspond to edge functions y_6^2, y_7^2 and y_8^2. Accordingly, model $K_2^2(2,8)$ and K^2 ([3, 2], 10), similarly to the previous example, will lose edges, that correspond to edge functions φ_6^2 and φ_7^2 (Fig. 11b).

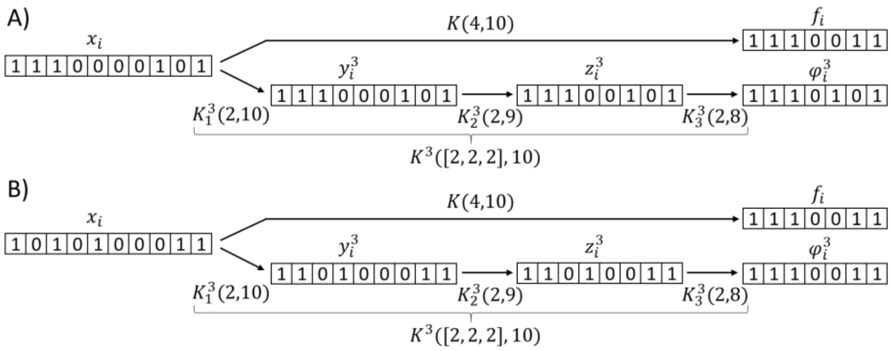


Fig. 10. **Behaviour of the models *K*(4, 10) and *K*³([2, 2, 2], 10) on different input vectors: a – *K*(4, 10); b – *K*³([2, 2, 2], 10)**
***Source:* compiled by the authors**

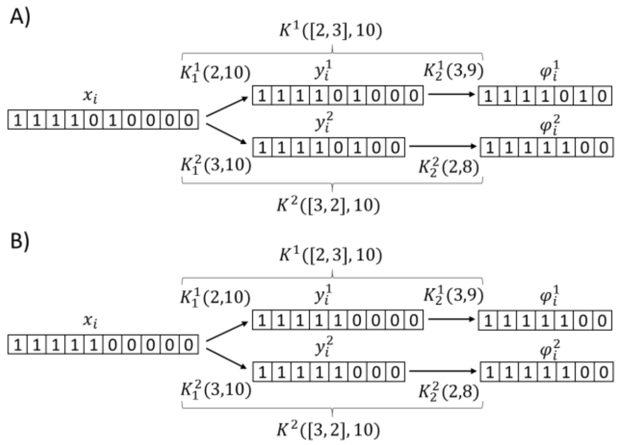


Fig. 11. **Behaviour of the models K^1([2, 3], 10) and K^2([3, 2], 10) on different input vectors: a – K^1([2, 3], 10); b – K^2([3, 2], 10)**
Source: **compiled by the authors**

On vector <1110010001> intermediate model $K_1^3(2,10)$ will lose edges, that correspond to edge functions y_4^1, y_5^1, y_7^1 and y_8^1. Therefore, model $K_2^1(3,9)$, and, thus, model K^1([2, 3], 10) will lose edges, that correspond to edge functions φ_4^1 and φ_5^1. On the other hand, intermediate model $K_1^3(2,10)$ on that vector will lose edges, that correspond to functions y_4^3, y_5^3, y_7^3 and y_8^3, meanwhile model $K_2^3(2,9)$ will lose edges, that correspond to functions z_4^3, z_5^3 and z_7^3. Accordingly, model $K_3^3(2,8)$ and K^3([2, 2, 2], 10) will lose edges, that correspond to edge functions φ_4^3 and φ_6^3 (Fig. 12a). However, on vector <1010100011> intermediate model $K_1^3(2,10)$ will lose edges, that correspond to edge functions y_3^1, y_5^1, y_6^1 and y_7^1. Therefore, model $K_2^1(3,9)$ and thus model K^1([2, 3], 10), similarly to the previous example, will lose edges, that correspond to edge functions φ_4^1 and φ_5^1. At the same time, intermediate model $K_1^3(2,10)$ on that vector will lose edges, that correspond to functions y_3^3, y_5^3, y_6^3 and y_7^3, model $K_2^3(2,9)$ will lose edges, that correspond to functions z_3^3, z_5^3 and z_6^3, meanwhile model $K_3^3(2,8)$ and K^3([2, 2, 2], 10) will lose edges, that correspond to edge functions φ_4^3 and φ_5^3 (Fig. 12b).

Note that for the considered pairs of cascade models, the same behavior in terms of sets of edges lost on different input vectors cannot be achieved by permuting these edges either.

At the same time, experimental data shows that the model K^2([3, 2], 10) and K^3([2, 2, 2], 10) behave identically on each of the input vectors, in other words, $\varphi_i^2 \equiv \varphi_i^3$.

The formation of criteria that will allow determining in which cases the behavior of models (classical/cascade or several different cascades) will be identical, and in which will differ, for which edges and on which input vectors, also requires further research.

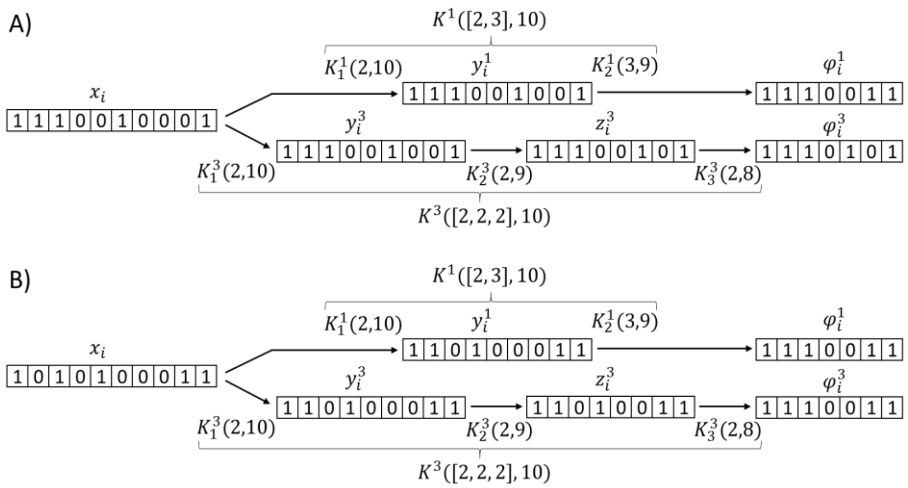


Fig. 12. **Behaviour of the models K^1([2, 3], 10) and K^3([2, 2, 2], 10) on different input vectors: a – K^1([2, 3], 10); b – K^3([2, 2, 2], 10)**
Source: **compiled by the authors**

MODIFICATION OF THE CASCADE MODELS

Let us note that submodels of the cascade model can be modified, in particular, by the methods described in [40]. It is expected that as a result of such a modification, some non-basic GL-model will be obtained, the behavior of which will differ from the original one on some set of so-called blocked input vectors. That is, on these vectors, the modified model shows the operational state of the system, unlike the original model.

As a result of modification of the basic model $K(m, n)$, in the way proposed in [40], the set of blocked vectors will contain vectors exactly with $m+1$ zeroes. Using as an example the cascade model *K'*([2, 2], 8), considered above, let's examine the effect of some cases of modification of its component submodels in the manner described in [40] on a set of blocked vectors.

Thus, one of the variants of such a modification may consist of replacing the expression of the edge function y_4 of the model $K_1(2, 8)$ in (9) with expression:

$$y'_4 = (x_1 \vee x_2)(x_1x_2 \vee x_3x_4)(x_3 \vee x_4) \vee \\ \vee x_5x_6x_7x_8 . \quad (19)$$

As a result, 16 vectors with 4 zeros will be blocked: B_1 = {<01110001>, <01110010>, <01110100>, <01111000>, <10110001>, <10110010>, <10110100>, <10111000>, <11010001>, <11010010>, <11010100>, <11011000>, <11100001>, <11100010>, <11100100>, <11101000>}.

Another option for modification may be to change the expression of the function φ_4 of the model $K_2(2, 7)$ in (10) with expression:

$$\varphi'_4 = (y_1 \vee y_2)(y_1y_2 \vee y_3y_4)(y_3 \vee y_4) \vee \\ \vee y_5y_6y_7 . \quad (20)$$

As a result of this modification, exactly the same vectors will be blocked as in the previous case.

We can also perform both of the above-mentioned modifications of submodels at once $K_1(2, 8)$ and $K_2(2, 7)$. The set of blocked vectors will be the same as in the previous cases, in other words, B_1.

If the modification consists of changing the expression of the function y_2 of the model $K_1(2, 8)$ in (9) with expression:

$$y'_2 = x_1 \vee x_2 \vee x_3x_4 , \quad (21)$$

the set of blocked vectors will contain 32 vectors with 4 zeros: B_2 = {<01000111>, <01001011>, <01001101>, <01001110>, <01010011>, <01010101>, <01010110>, <01011001>, <01011010>, <01011100>, <01100011>, <01100101>, <01100110>, <01101001>, <01101010>, <01101100>, <10000111>, <10001011>, <10001101>, <10001110>, <10010011>, <10010101>, <10010110>, <10011001>, <10011010>, <10011100>, <10100011>, <10100101>, <10100110>, <10101001>, <10101010>, <10101100>}, which is obviously different from set B_1, obtained as a result of previous modifications.

By adding to this modification a modification of the model $K_2(2, 7)$, considered above, we will get a set of blocked vectors, which will include 48 vectors with 4 zeros blocked by applying each of them separately, in other words, $B_1 \cup B_2$. However, 16 more vectors with an increased multiplicity of zeros, in this case with 5 zeros, will also be blocked: {<01010001>, <01010010>, <01010100>, <01011000>, <01100001>, <01100010>, <01100100>, <01101000>, <10010001>, <10010010>, <10010100>, <10011000>, <10100001>, <10100010>, <10100100>, <10101000>}.

An assessment of exactly what the set of blocked vectors will be in the case of modification of certain submodels of the cascade model, as well as whether it will include vectors with an increased multiplicity of zeros, requires additional research.

CONCLUSIONS

The article discusses a new direction of GL-model development, which is distinguished by the formation of one model through the cascading application of several basic GL-models. It shows that the model obtained in this way, which is proposed to be called *cascade*, will also be basic and will have properties similar to those of the usual basic GL-model, namely: the total number of edges and the number of edges it loses on vectors of a certain multiplicity. Preservation of the above-mentioned properties allows building a cascade model of arbitrary depth.

Was given examples and performed comparison of conventional and cascade GL-models. Was shown that despite the conservation of properties, the specific sets of edges lost by each of the models on certain state vectors of the system may differ. In addition, a comparison of the number of logical operations required to calculate the edge functions of each of the models was performed. Was shown that, at least in some cases, the use of cascade models reduces the time of performing one statistical experiment with the model, which leads to an increase in the accuracy of the system reliability calculation.

Based on the obtained experimental data, a number of questions for future research were formulated. Namely: does the calculation of edge functions cascade model always require the execution of a smaller number of logical operations compared to the classical model; in which cases it is possible to achieve a reduction in the number of both conjunctions and disjunctions, and in which - only disjunctions; what are the optimal cascade model parameters; in which cases the behavior of the models in terms of the set of lost edges on different vectors is identical, and in which cases it differs, for exactly which edges, on which input vectors, etc.

It was also shown that the cascade model can be modified by changing its component sub-models, both individually and several at the same time. At the same time, vectors with an increased multiplicity of zeros can also be blocked. Evaluation of the set of vectors blocked as a result of cascade model modification requires additional research.

REFERENCES

1. Kharchenko, V. S. "Safety of critical infrastructures: Mathematical and engineering methods of analysis and support" (In Russian). Ministry of Education and Science of Ukraine. National Aerospace University "Kharkiv Aviation Institute". 2011. 641. ISBN 978-966-662-228-3.

2. Azarskov, V. N. & Strelnikov, V. P. "Reliability of control systems and automation: Tutorial" (In Russian). Kyiv: Ukraine. National Aviation University. 2004.164. ISBN: 966-598-173-0.

3. Bebeshko, E. V., Illiashenko, O. A. & Kharchenko, V. S. "Multi-stage analysis of reliability and safety of information and control systems" (in Russian). *Radio-Electronic and Computer Systems* 2010; 7(48): 283–288.

4. Yaremchuck, S. A. "Problems of a Priori evaluation of reliability indicators of guaranteed critical information systems" (in Russian). *Radio-Electronic and Computer Systems.* 2013; 5(64): 414–421.

5. 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.

6. Drozd, O. V., Rucinski, A., Zashcholkin, K. V., Drozd, M. O. & Sulima, Yu. Yu. "Improving FPGA components of critical systems based on natural version redundancy". *Applied Aspects of Information Technology*. 2021; 4 (2): 168–177. DOI: https://doi.org/10.15276/aait.02.2021.4.

7. Yizhong, F., Yuting, L., Tuo, H., et al. "Design of missile incremental adaptive fault tolerant control system" (in Chinese). *Journal of Beijing University of Aeronautics and Astronautics*. 2022; 48 (5): 920–928. DOI: https://doi.org/10.13700/j.bh.1001-5965.2021.0454.

8. Hu, Q., Niu, G. & Wang, C. "Spacecraft attitude fault-tolerant control based on iterative learning observer and control allocation". Control *Allocation for Spacecraft under Actuator Faults.* 2018; 75: 245–253. DOI: https://doi.org/10.1016/j.ast.2017.12.031.

9. Joshi, H. & Sinha, N. K. "Adaptive fault tolerant control design for stratospheric airship with actuator faults". IFAC-PapersOnLine. 2022; 55 (1): 819825. DOI: https://doi.org/10.1016/j.ifacol.2022.04.134.

10. Sun, R., Han, Y. & Wang, Y. "Design of generalized fault diagnosis observer and active adaptive fault tolerant controller for aircraft control system". *Mathematical Biosciences and Engineering*. 2022; 19 (6): 5591–5609. DOI: https://doi.org/10.3934/mbe.2022262.

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

12. Odarushchenko, O. N., Zhivilo, S. V., Kharchenko, V. S. & Odarushchenko, E. B. "Analysis of architectures of fault-tolerant servers to evaluate their reliability" (in Russian). *Radio-Electronic and Computer Systems*. 2012; 7(59): 60–68.

13. 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.

14. Verlan, A. F., Polozhaenko, S. A., Prokofieva, L. L. & Shylov, V. P. “Diagnosis of substances, which characterize the properties of independent observation and control” (in Ukrainian). *Applied Aspects of Information Technology*. 2018; 1 (1): 33–47. DOI: https://doi.org/10.15276/aait.01.2018.2.

15. Kuo, W. & Zuo M. J. “Optimal reliability modeling: principles and applications”. *John Willey & Sons*. 2003: 560.

16. 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. DOI: https://doi.org/10.1109/PRDC.2008.23.

17. Dumitrescu, M. “Fault tolerant control multiprocessor systems modelling using advanced stochastic petri nets”. *Procedia Technology* 2016; 22: 623–628. DOI: https://doi.org/10.1016/j.protcy.2016.01.129.

18. 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*. 324264. 2014. DOI: https://doi.org/10.4108/icst.valuetools.2013.254370.

19. 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. DOI: https://doi.org/10.1016/j.ress.2015.04.005.

20. Locks, M. O. “Comments on: Improved method of inclusion-exclusion applied to k-out-of-n systems”. *IEEE Transactions on Reliability*. 1984; R-33: 321–322. DOI: https://doi.org/10.1109/TR.1984.5221841.

21. Rosenthal, A. “Note on `closed form solutions for delta-star and star-delta conversion of reliability networks”. *IEEE Transactions on Reliability*. R-27: 110–111. DOI: https://doi.org/10.1109/TR.1978.5220272.

22. Distefano, S. & Xing, L. “A new approach to modeling the system reliability: Dynamic reliability block diagrams”. *Proceedings – Annual Reliability and Maintainability Symposium*. 2006. p. 189–195. DOI: https://doi.org/10.1109/RAMS.2006.1677373.

23. Belfore, L. A. “An $O(n(\log_2(n))^2)$ algorithm for computing the reliability of k-out-of-n:G & k-to-l-out-of-n:G systems”. *IEEE Transactions on Reliability*. 1995; R-44 (1): 132–136. DOI: https://doi.org/10.1109/24.376535.

24. Rushdi, A. M. “Utilization of symmetric switching function in the computation of k-out-of-n system reliability”. *Microelectronics Reliability*. 1986; 26 (5): 973–987. DOI: https://doi.org/10.1016/0026-2714(86)90239-8.

25. Rushdi, A. M. “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: 306–326. DOI: https://doi.org/10.33889/IJMEMS.2019.4.2-025.

26. Fu, J. C. & Hu, B. “On reliability of a lLarge consecutive-k-out-of-n:F system with (k - 1)-step Markov dependence”. *IEEE Transactions on Reliability*. 1987; Vol. 36: 75–77. DOI: https://doi.org/10.1109/TR.1987.5222299.

27. Eryilmaz, S., Kan, C. & Akici, F. “Consecutive k-Within-m-Out-of-n:F system with exchangeable components”. *Naval Research Logistics*. 2009; 56 (6): 503–510. DOI: https://doi.org/10.1002/nav.20354.

28. Griffith, W. S. “On consecutive k-out-of-n failure systems and their generalizations”. *Reliability and Quality Control*. 1986; 20: 157–165.

29. Sfakianakis, M., Kounias, S. & Hillaris, A. “Reliability of a consecutive k-out-of-r-from-n:F system”. *IEEE Transactions on Reliability*, 1992; 41 (3): 442–447. DOI: https://doi.org/10.1109/24.159817.

30. Habib, A., Al-Seedy, R. O. & Radwan, T. “Reliability evaluation of multi-state consecutive k-out-of-r-from-n:G system”. *Applied Mathematical Modeling*. 2007; 31 (11): 2412–2423. DOI: https://doi.org/10.1016/j.apm.2006.09.006.

31. Chang, G. J., Cui, L. & Hwang, F. K. "Reliabilities for (n, f, k) systems". *Statistics & Probability Letters*.1999; 43: 237–242. DOI: https://doi.org/10.1016/S0167-7152(98)00263-6.

32. Wu, J. S. & Chen, R. J. "An algorithm for computing the reliability of a weighted-k-out-of-n system". *IEEE Transactions on Reliability* .1994; R-43: 327–328. DOI: https://doi.org/10.1109/24.295016.

33. Lee, W. S., Grosh, D. L., Tillman, F. A. & Lie, C. H. "Fault tree analysis, methods and applications: a review". *IEEE Transactions on Reliability*. 1985; 34 (3): 194–203. DOI: https://doi.org/10.1109/TR.1985.5222114.

34. Romankevich, V. A. "On one model of behavior of a fault-tolerant multiprocessor system" (In Russian). *Radioelectronics & Informatics*. 1999; 1: 75–76.

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

36. 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.

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

38. Romankevich, A. M., Morozov, K. V. & Romankevich, V. A. "Graph-Logic Models of hierarchical fault-tolerant multiprocessor systems". *International Journal of Computer Science and Network Security*. 2019; 19 (7): 151–156.

39. Maidaniuk, I. V., Morozov, K. V., Potapova, E. R. & Shuriga, A. V. "On one property of the GL-model with a minimum number of lost edges" (In Russian). *Naukovy Visnyk of Chernivtsi University. Series: Computer Systems and Components.* 2010; 1 (2): 31–34.

40. Romankevich, V. A., Morozov, K. V. & Feseniuk, A. P. "On one method of modification of edge functions of GL-models" (in Russian). *Radio-Electronic and Computer Systems*. 2014; 6: 95–99

Conflicts of Interest: the authors declare no conflict of interest

Received 12.07.2022
Received after revision 14.09.2022
Accepted 17.10.2022

DOI: https://doi.org/10.15276/aait.05.2022.18
УДК 004.05

Про каскадну GL-модель та її властивості

Романкевич Олексій Михайлович [1)]
ORCID: https://orcid.org/0000-0001-5634-8469; romankev@scs.kpi.ua. Scopus Author ID: 6602114176
Морозов Костянтин В'ячеславович[1)]
ORCID: https://orcid.org/0000-0003-0978-6292; mcng@ukr.net. Scopus Author ID: 57222509251
Микитенко Сергій Сергійович[1)]
ORCID: https://orcid.org/0000-0001-8842-004X; sergei.mykytenko@gmail.com
Коваленко Олена Павлівна[1)]
ORCID: https://orcid.org/0000-0002-2548-2347; kop100298@gmail.com
[1)] Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського», пр. Перемоги, 37. Київ, 03056, Україна

АНОТАЦІЯ

В статті пропонується новий напрямок подальшого розвитку GL-моделей – моделей, на базі яких виконується розрахунок надійнісних параметрів відмовостійких багатопроцесорних систем. Такі моделі віддзеркалюють реакцію системи на появу відмов довільної кратності. Суть нового напрямку – побудова моделі шляхом композиції декількох базових GL-моделей таким чином, що, значення реберних функцій однієї моделі формують вхідний вектор наступної.

Показано, що отримана таким чином модель, яку запропоновано називати каскадною, також буде базовою та в загальному випадку може складатися із довільної кількості підмоделей. Наведено формулу, що дозволяє визначати значення ступеня відмовостійкості каскадної моделі, в залежності від значень рівнів відмовостійкості її складових підмоделей. Показано, що графи як каскадної, так і звичайної моделей є циклічними та мають однакову кількість ребер. При цьому, не зважаючи на те, що проміжні підмоделі також мають графи, їх наявність не підвищує складності моделі в цілому, оскільки в них використовуються лише вирази реберних функцій. На прикладах підтверджено коректність теоретично отриманих результатів, а також показано, що каскадна модель, принаймні, в деяких випадках має меншу розрахункову складність (загальну кількість логічних операцій у виразах реберних функцій), в порівнянні зі звичайною. Виявлено, що, хоч каскадна модель є базовою, множини ребер, які втрачає вона та звичайна базова GL-модель на деяких вхідних векторах можуть відрізнятися. В певних випадках може бути побудовано декілька альтернативних каскадних моделей, що відрізнятимуться своїми параметрами, але матимуть однакове результуюче значення ступеня відмовостійкості. На прикладі виконано порівняння властивостей таких альтернативних каскадних моделей. Виявлено, що такі моделі відрізняються як за розрахунковою складністю, так і, в деяких випадках, за множинами ребер, які вони втрачають на певних вхідних векторах. Показано можливість модифікації каскадної моделі шляхом зміни виразів реберних функцій її складових підмоделей, як кожної окремо, так і декількох одночасно. При цьому можливим є блокування векторів із підвищеною кратністю нулів. Сформульовано ряд задач для майбутніх досліджень.

Ключові слова: каскадні GL-моделі; відмовостійкі багатопроцесорні системи; модифікація GL-моделей; розрахунок параметрів надійності

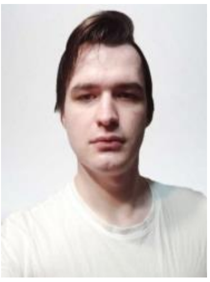
ABOUT THE AUTHORS

Alexei M. Romankevitch - Doctor of Engineering Sciences, Professor, Professor of System Programming and Special Computer System Department. National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute", 37, Peremogy Av. 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 спеціальних комп'ютерних систем. Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського», пр. Перемоги, 37. Київ, 03056, Україна

Kostiantyn V. Morozov - PhD, Assistant of System Programming and Special Computer System Department. National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute", 37, Peremogy Av. 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 спеціальних комп'ютерних систем. Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського», пр. Перемоги, 37. Київ, 03056, Україна



Serhii S. Mykytenko - Master (Comp. Eng), PhD student of System Programming and Special Computer System Department, National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute", 37, Peremogy Av. Kyiv, 03056, Ukraine
ORCID: https://orcid.org/0000-0001-8842-004X, sergei.mykytenko@gmail.com
Research field: GL-models; computer systems; computer networks; cellular networks; computer engineering; computer systems; system engineering

Микитенко Сергій Сергійович - магістер з комп'ютерної інженерії, аспірант кафедри Системного програмування та спеціальних комп'ютерних систем. Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського», пр. Перемоги, 37. Київ, 03056, Україна

Olena P. Kovalenko - PhD student of Department of System Programming and Specialized Computer Systems. National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute", 37, Peremogy Av. Kyiv, 03056, Ukraine
ORCID: https://orcid.org/0000-0002-2548-2347, kop100298@gmail.com
Research field: Reliability of multiprocessors systems: fault-tolerant multi-bus multiprocessor systems; optimal control; loss minimization

Коваленко Олена Павлівна - аспірант кафедри Системного програмування та спеціалізовах комп'ютерних систем. Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського», пр. Перемоги, 37. Київ, 03056, Україна