I. J. Computer Network and Information Security, **2024, 1, 97-112**
Published Online on February 8, 2024 by MECS Press (http://www.mecs-press.org/)
DOI: 10.5815/ijcnis.2024.01.08

Fault Tolerance Exploration and SDN Implementation for de Bruijn Topology based on betweenness Coefficient

Artem Volokyta*
National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute"/ Department of computer engineering, Kyiv, 03056, Ukraine
E-mail: artem.volokita@kpi.ua
ORCID iD: https://orcid.org/0000-0001-9069-5544
*Corresponding author

Heorhii Loutskii
National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute"/ Department of computer engineering, Department of information systems and technologies, Kyiv, 03056, Ukraine
E-mail: georgijluckij80@gmail.com
ORCID iD: https://orcid.org/0000-0002-3155-8301

Oleksandr Honcharenko
National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute"/ Department of computer engineering, Department of information systems and technologies, Kyiv, 03056, Ukraine
E-mail: alexandr.ik97@ukr.net
ORCID iD: https://orcid.org/0000-0002-9086-6988

Oleksii Cherevatenko
National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute"/ Department of computer engineering, Department of information systems and technologies, Kyiv, 03056, Ukraine
E-mail: chereva@ukr.net
ORCID iD: https://orcid.org/0000-0001-9686-0555

Volodymyr Rusinov
National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute"/ Department of computer engineering, Department of information systems and technologies, Kyiv, 03056, Ukraine
E-mail: VRusinovIO51@office365.fiot.kpi.ua
ORCID iD: https://orcid.org/0000-0002-4362-0248

Yurii Kulakov
National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute"/ Department of computer engineering, Department of information systems and technologies, Kyiv, 03056, Ukraine
E-mail: ya.kulakov@gmail.com
ORCID iD: https://orcid.org/0000-0002-8981-5649

Serhii Tsybulia
The National Defence University of Ukraine/ Scientific and methodological center for the organization of scientific and technical activities, Kyiv, 03049, Ukraine
E-mail: kibtor@gmail.com
ORCID iD: https://orcid.org/0000-0003-0323-1771

Received: 01 September 2023; Revised: 27 October 2023; Accepted: 12 December 2023; Published: 08 February 2024

Abstract: This article considers the method of analyze potentially vulnerable places during development of topology for fault-tolerant systems based on using betweenness coefficient. Parameters of different topological organizations using De Bruijn code transformation are observed. This method, assessing the risk for possible faults, is proposed for other

topological organizations that are analyzed for their fault tolerance and to predict the consequences of simultaneous faults on more significant fragments of this topology.

Index Terms: De Bruijn Graph, betweenness, Fault Tolerance, SDN.

1. Introduction

1.1. Relevance of the Research Topic

As of today, there is no universal method for preventing failures, which is why extra attention is given to creating topological organizations with consideration for fault tolerance. While such methods do exist and are widely used, characteristics of such systems in the presence of failures of certain nodes in vulnerable segments are less frequently defined. The proposed method allows for the identification of nodes that have the most significant impact on the system's characteristics. These nodes are at greater risk, as attacks and failures involving them will lead to a more rapid degradation of the topology [1].

1.2. Problem Statement

During the prolonged operation of a high-performance computing system, certain nodes begin to fail. Failures in the system can lead to unpredictable consequences if the topology of the system has not been previously studied. On the other hand, fully exploring the system, considering all possible combinations of failures, is highly challenging and time-consuming, especially if the system employs state-of-the-art solutions as its underlying topology. Moreover, utilizing network protocols that dynamically investigate the system's topology can be quite resource-intensive.

1.3. Article Objective

The research aims to develop a method for enhancing the fault tolerance of topological organizations by utilizing the betweenness coefficient to identify the most susceptible nodes within the system. Furthermore, the study investigates the impact of failures of these identified nodes on the system's topological characteristics.

2. Review of Previous Studies

2.1. Analysis of Recent Research and Publications

Topology is one of the crucial components in designing a corporate distributed computer system, which will directly impact the future operation of the entire financial computer system of an organization. Therefore, the issue of fault tolerance directly affects the stability of all systems, including the financial resilience management system of the enterprise, aimed at improving its financial status.

Modern supercomputers are built based on various topologies such as Dragonfly, Jellyfish, and 6D-Torus. Dragonfly employs an approach that reduces the number of cables for connecting clusters by utilizing specialized adaptive routers to alleviate network congestion [2,3]. The Jellyfish topology aims to develop a network with high throughput capacity and horizontal scalability, based on the underlying topology of a random graph. This topology has proven to be more suitable for data centers than the fat-tree topology [4]. The fastest modern supercomputer, Fugaku, utilizes the 6D-Torus topology, which enables more parallel routes, a crucial factor for ensuring computational fault tolerance [5,6].

The classic approach to building supercomputers is based around massive parallelism and consists in using switched networks to connect nodes [7]. Technologies such as Gigabit Ethernet and InfiniBand are popular, allowing to achieve high parameters of bandwidth and fault tolerance, while ensuring strong connectivity between nodes. The complete opposite of this is the so-called opportunistic approach, which involves loosely coupled parallelism and is based on distributed computing technology. For interprocessor interaction, such systems use the Internet and do not have a topology in the usual sense, constantly changing in the process of connecting and disconnecting nodes. At the same time, there are intermediate approaches, such as high-performance systems in the cloud and quasi-opportunistic supercomputers, which, on the one hand, are based on classic network technologies, and on the other hand, provide strong connectivity and relative static topological organization.

Several publications have appeared in recent years documenting methods of traffic organization in networks based on Software-Defined Networking technology. Some of the latest solutions are described in [8] and [9]. These articles feature some analysis of SDN internal organization. All network organized into data centers which operate the routing and traffic. This positively affects the speed and efficiency of the network because of the using of specific traffic design and multi-path routing scheme.

These works also include several useful methods for their further modification. First is the method of traffic orchestrating, which considered better than those that existed before, because of simpler reconfiguration procedure and a centralized approach to create special load balance. The routing information is now being generated in the SDN central

controller in automatic mode which makes the whole network much more stable and faster. The authors developed a special routing algorithm and built it in the controller of the network so all the paths are being created from all network nodes to the end node which guarantees a hundred percent delivery. Another method they propose is basically a centralized organization of routing tables which allows a network to avoid rewriting them on each network node. Because of that, there are no problematic "double routing" issues in the network and no different routes during the single cycle of transmission.

2.2. Identification of Previously Unaddressed Aspects of the Overarching Problem

Topologies based on De Bruijn shifts have not been previously investigated for their fault tolerance. The proposed method is relatively novel and suggests a departure from the conventional approach of identifying a random node prone to failure. Instead, it models a system attack scenario, where intermediary nodes [1], responsible for channeling data flows between nodes in high-performance systems, are identified as the most vulnerable. By employing the Floyd-Warshall shortest path algorithm, these intermediary nodes can be pinpointed.

SDN networks were not built basing on De Brujin topologies. In recent studies, authors didn't create practical models of processing the network and didn't demonstrate where new proposed methods can be used. There is a lack of simulation results and there is an open question how the reconfiguration may affect the metric of the network.

3. Fault Tolerance Exploration based on betweenness Coefficient

The development of a topological organization requires an investigation into its fault tolerance. Fault tolerance considers the potential failure of one or more nodes, while the system as a whole remains operational, albeit with slightly compromised performance. Unlike networks designed at a local level, such as home, office, or university networks, high-performance computer systems have complex structures that warrant analytical examination prior to implementation. Designing a computer system that enables organizations to maintain their operational processes in a stable state, even in the presence of an elevated number of failures, is essential for ensuring the organization's long-term viability [10].

To study a topology, it is necessary to compute its metrics. The key metrics of topologies include the number N of nodes in the topology, the degree of the topology, the diameter of the topology, the average diameter of the topology, the topological traffic, cost, and the multiplicative parameter SD. The topology characteristics were calculated using the following formulas.

The diameter of a system (D) is the minimum distance between two farthest processors.

The average diameter is the average distance to any node and is described by the following formula:

$$\overline{D} = \frac{\sum_{i=1}^{n} \sum_{j=1}^{n-1} d_{ij}}{n \cdot (n-1)} \tag{1}$$

Degree (S) - a characteristic is defined as the maximum number of edges (connections) incident to a vertex (processor) in the topology graph of the system.

Cost is determined using the formula:

$$C = D \cdot n \cdot S \tag{2}$$

Topological traffic describes the efficiency of link utilization in the system and is determined using the formula:

$$T = \frac{2\overline{D}}{S} \tag{3}$$

The multiplicative parameter SD is determined using the formula:

$$SD = S \cdot D \tag{4}$$

To assess the fault tolerance of the investigated topology, we introduce the concept of betweenness coefficient. This coefficient indicates the number of shortest paths between certain pairs of nodes in the network. In theory, if a node or several nodes with high betweenness coefficients fail, the algorithm must recalculate new shortest paths, potentially resulting in higher distances. Nodes with the highest betweenness coefficients play a more crucial role in forming connections with other nodes in the system. The formula for calculating the betweenness coefficient is as follows [1]:

$$b_m = \sum_{i \neq j} \frac{B(i,m,j)}{B(i,j)} \tag{5}$$

where B(i,j) represents the total number of shortest paths between nodes i and j, and B(i,m,j) indicates the number of shortest paths between i and j that pass through node m. Given that the shortest routes may be unknown and a search algorithm is utilized within the network, the betweenness of a node can be expressed as the probability of this algorithm finding it. Hence, the concept of the dominant intermediary coefficient is introduced, calculated according to the formula

$$CPD = \frac{1}{n-1} \sum_i (B_{\max} - B_i) \tag{6}$$

where Bmax represents the maximum value of the dominant intermediary coefficient [1].

For pathfinding, a modified Floyd-Warshall algorithm is employed to retain path information between vertices. The Floyd-Warshall algorithm compares all possible paths in the graph between each pair of vertices. For a given graph G(V,E), we construct a matrix of distances and set the distance of all vertices to themselves as 0. We seek the distance from one vertex to another, while marking the nodes involved in finding the shortest path in a separate matrix. This yields two matrices as output: a distance matrix between vertices and an intermediary matrix. For the considered topologies in the second scaling step, intermediary nodes are marked in green [11].

Topology synthesis is carried out using De Bruijn shifts, but with a slightly different implementation. It was found that the topology obtained from De Bruijn shifts using redundant coding demonstrates superior characteristics and better fault tolerance. Utilizing this approach, a topological organization can be developed based on binary, ternary, and quaternary numeral systems along with additional coding.

The first variant is the binary redundant De Bruijn. When forming a new sequence, the most significant digit is evicted (disappears), and the least significant digit after the shift takes on values 0, 1, or T, which represents -1. Nodes obtained in this manner are linked to the node with the number that underwent the shift. As an example, consider the following table for generating a De Bruijn sequence for ternary numbers [12,13].

Figures 1-3 depict topological organizations for different numeral systems. For clarity, nodes of the 0-cluster are duplicated in the illustrations. This graphical duplication enhances the visualization of the inter-cluster principle. Additionally, it should be noted that the proposed graphs do not utilize concealed "quasi-quantum" connections [14]. Thus, the proposed graphs can be regarded as classical De Bruijn graphs in various numeral systems.

Table 1. Table of formation of de Bruijn sequences

Node number	**00**			**01**			**0T**		
Connections	00	01	0T	10	11	1T	T0	T1	TT
Node number	**10**			**11**			**1T**		
Connections	00	01	0T	10	11	1T	T0	T1	TT
Node number	**T0**			**T1**			**TT**		
Connections	00	01	0T	10	11	1T	T0	T1	TT

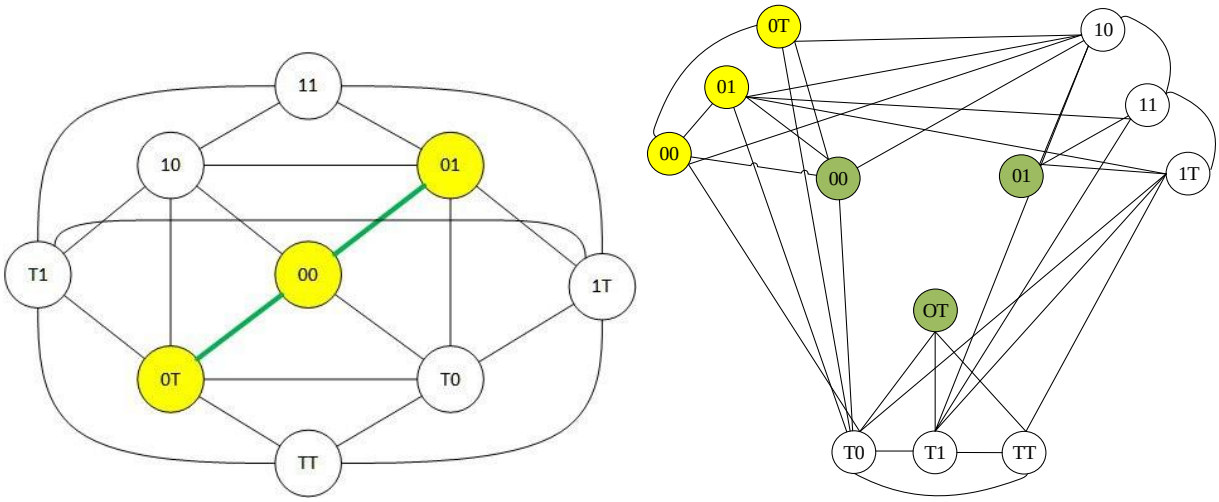

Fig.1. Cluster for the binary redundant system 01T. Topology A

Table 1 illustrates the establishment of connections between nodes in the topology. Connections are obtained through a leftward shift. The position of the least significant bit is occupied by a new bit with three values: 0, 1, T.

Figure 1 depicts a topology with 9 vertices. The nodes (00, 01, OT) highlighted in green are schematically duplicated, thus in the modeling, topology characteristics are calculated without considering the "green" nodes, only with the "yellow" nodes.

Table 2. Table of topological characteristics of topology A at the first 5 scaling steps

Number of nodes	**Diameter**	**Degree**	**Average diameter**	**Cost**	**Topological traffic**	**SD**
3	1	3	1	9	0.67	3
9	2	5	1.41667	90	0.5667	10
27	3	6	2.07692	486	0.6923	18
81	4	6	2.83333	1944	0.9444	24
243	5	6	3.6738	7290	1.2246	30
729	6	6	4.57224	26244	1.52408	36
2187	7	6	5.50623	91854	1.83541	42

As evident from Table 2, the topological traffic increases almost linearly, allowing for the design of topologies with an acceptable level of node degrees.

Table 3. Table of nodes with betweenness coefficient values

Node code	Node number	Number of the shortest ways through this node	mediation coefficient
0T	-1	10	0.2778
00	1	4	0.1111
01	0	10	0.2778
1T	1	10	0.2778
10	2	10	0.2778
11	3	4	0.1111
TT	-3	4	0.1111
T0	-2	10	0.2778
T1	-1	10	0.2778

In Table 3, the number of shortest paths passing through each node of the topology is presented, along with the corresponding betweenness coefficient for each node. Thus, we can identify nodes through which the highest number of routes traverse. Therefore, potentially, these nodes should be more resilient, possibly employing redundancy mechanisms.

The second topology variant employs a ternary numeral system and redundant coding. It is based on the following digits: 0, 1, 2, T, Z, where T holds a value of -1 and Z holds a value of -2. There are numerous shifts involved, so it makes sense to provide only a segment of the table to illustrate the principle.

Table 4. Fragment table of the formation of de Bruijn sequences

Node number	**00**					**01**					**0T**				
Connections	00	01	02	0T	0Z	10	11	12	1T	1Z	T0	T1	T2	TT	TZ
Node number	**10**					**11**					**1T**				
Connections	00	01	02	0T	0Z	10	11	12	1T	1Z	T0	T1	T2	TT	TZ
Node number	**T0**					**T1**					**TT**				
Connections	00	01	02	0T	0Z	10	11	12	1T	1Z	T0	T1	T2	TT	TZ

Table 4 illustrates the establishment of connections between nodes in the topology. Connections are obtained through a leftward shift. The position of the least significant bit is occupied by a new bit with five values: 0, 1, 2, T, Z.

Figure 2 depicts a topology with 25 vertices. The nodes (00, 01, 02, OT, 0Z) highlighted in green are schematically duplicated, thus in the modeling, topology characteristics are calculated without considering the 'green' nodes, only with the 'yellow' nodes.

As evident from Table 5, the topological traffic, similar to Topology A, exhibits linear growth, albeit starting from a lower value. Additionally, the degree of Topology B is 10, compared to Topology A's 6.

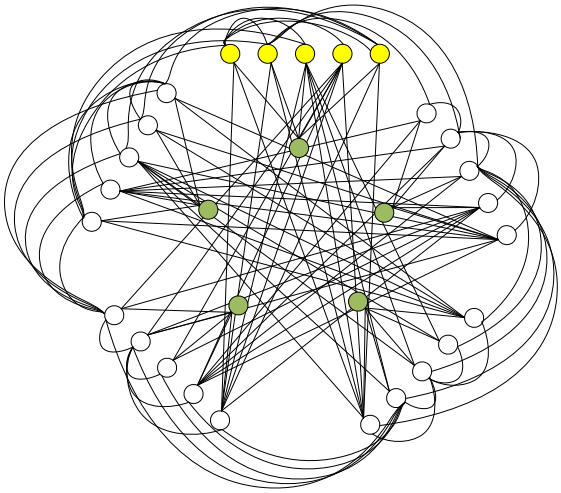

Fig.2. Cluster for the ternary redundant system 012TZ. Topology B.

Table 5. Table of topological characteristics for Topology B in the first 5 scaling steps

Number of nodes	**Diameter**	**Degree**	**Average diameter**	**Cost**	**Topological traffic**	**SD**
5	1	5	1.0	25	0.4	5
25	2	9	1.63333	450	0.36296	18
125	3	10	2.47097	3750	0.49419	30
625	4	10	3.37795	25000	0.67559	40
3125	5	10	4.33399	156250	0.8668	50

Table 6. Table of nodes with betweenness coefficient values

Node code	Node number	Number of the shortest ways through this node	mediation coefficient
0Z	-2	50	0.1667
0T	-1	50	0.1667
00	0	24	0.08
01	1	50	0.1667
02	2	50	0.1667
1Z	0	50	0.1667
1T	1	50	0.1667
10	3	50	0.1667
11	4	24	0.08
12	5	50	0.1667
2Z	4	50	0.1667
2T	5	50	0.1667
20	6	50	0.1667
21	7	50	0.1667
22	8	24	0.08
TZ	-5	50	0.1667
TT	-4	24	0.08
T0	-3	50	0.1667
T1	-2	50	0.1667
T2	-1	50	0.1667
ZZ	-8	24	0.08
ZT	-7	50	0.1667
Z0	-6	50	0.1667
Z1	-5	50	0.1667
Z2	-4	50	0.1667

In Table 6, similar to Table 3, the number of shortest paths passing through each node of the topology is presented, along with the corresponding betweenness coefficient for each node. It is evident that with the increase in the number of vertices in the topology, the number of shortest paths also increases. However, the overall count grows faster than a quadratic formula.

Let's consider the final topology variant based on the quaternary numeral system and de Bruijn shifts. During the numeral shift, the last position can hold the values 0, 1, 2, 3, T, Z, E. T holds a value of -1, Z holds a value of -2, and E holds a value of -3.

Table 7. Some node codes during the formation of de Bruijn sequences

Node number	**E0**							**ET**						
Connections	00	01	02	03	0T	0Z	0E	T0	T1	T2	T3	TT	TZ	TE
Node number	**00**							**0T**						
Connections	00	01	02	03	0T	0Z	0E	T0	T1	T2	T3	TT	TZ	TE
Node number	**20**							**2T**						
Connections	00	01	02	03	0T	0Z	0E	T0	T1	T2	T3	TT	TZ	TE

Table 7 illustrates a fragment of constructing de Bruijn sequences, indicating the connections that can be obtained for certain three-digit numbers.

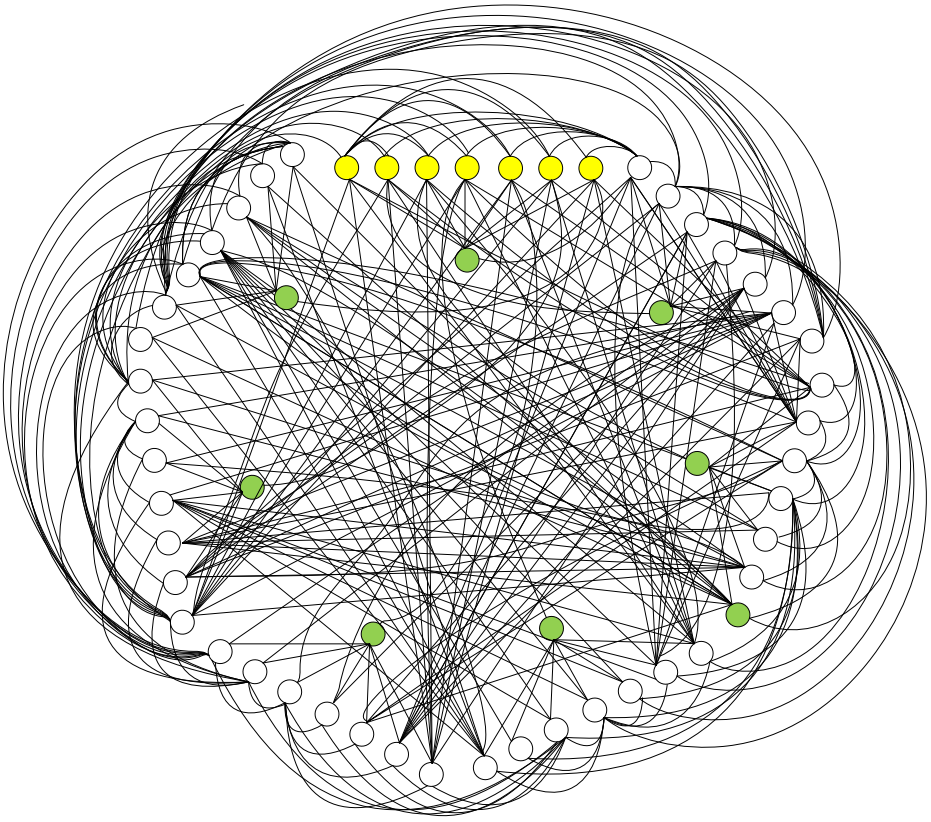

Fig.3. Cluster for the quaternary redundant system 0123TZE. Topology C

Figure 3 depicts a topology with 49 vertices. The nodes are highlighted in green (00, 01, OT, 02, OZ, 03, 0E), and are schematically duplicated, thus the characteristics of the topologies are computed in the simulation without considering the 'green' nodes, only the 'yellow' nodes are considered.

From Table 8, it can be observed that the last topology exhibits similar characteristics to the first two topologies, but with a higher degree compared to topology B by 4. When compared to Tables 2 and 5, the utilization of a larger numeral system enables the development of a topology with a greater number of vertices while maintaining the same diameter, albeit at the cost of sacrificing the degree.

Table 8. Table of topological characteristics of topology C for the first 5 scaling steps

Number of nodes	**Diameter**	**Degree**	**Average diameter**	**Cost**	**Topological traffic**	**SD**
7	1	7	1.00000	49	0.29	7
49	2	13	1.73214	1274	0.26648	26
343	3	14	2.63909	14406	0.37701	42
2401	4	14	3.59177	134456	0.51311	56
16807	5	14	4.57424	1176490	0.65346	70

Table 9. Table of nodes with mediation coefficient values

Node code	Node number	Number of the shortest ways through this node	mediation coefficient
0E	-3	122	0.1037
0Z	-2	122	0.1037
0T	-1	122	0.1037
00	0	60	0.0510
01	1	122	0.1037
02	2	122	0.1037
03	3	122	0.1037
1E	1	122	0.1037
1Z	2	122	0.1037
1T	3	122	0.1037
10	4	122	0.1037
11	5	60	0.0510
12	6	122	0.1037
13	7	122	0.1037
2E	5	122	0.1037
2Z	6	122	0.1037
2T	7	122	0.1037
20	8	122	0.1037
21	9	122	0.1037
22	10	60	0.0510
23	11	122	0.1037
3E	9	122	0.1037
3Z	10	122	0.1037
3T	11	122	0.1037
30	12	122	0.1037
31	13	122	0.1037
32	14	122	0.1037
33	15	60	0.0510
TE	-7	122	0.1037
TZ	-6	122	0.1037
TT	-5	60	0.0510
T0	-4	122	0.1037
T1	-3	122	0.1037
T2	-2	122	0.1037
T3	-1	122	0.1037
ZE	-11	122	0.1037
ZZ	-10	60	0.0510
ZT	-9	122	0.1037
Z0	-8	122	0.1037
Z1	-7	122	0.1037
Z2	-6	122	0.1037

Z3	-5	122	0.1037
EE	-15	60	0.0510
EZ	-14	122	0.1037
ET	-13	122	0.1037
E0	-12	122	0.1037
E1	-11	122	0.1037
E2	-10	122	0.1037
E3	-9	122	0.1037

The tendency highlighted in tables 3 and 6 persists in the case of the last topology as well. It can be observed that the use of redundant codes creates conditions where the number of alternative shortest routes is the same for the majority of nodes, which is crucial when designing a fault-tolerant topological organization.

4. Results

We perform node removal simulations, simulating node failures in the system. Initially, nodes with the highest mediation coefficient values are removed, and this process continues until 10% of all working nodes in the system have failed. The investigated topologies include those described earlier, utilizing binary, ternary, and quaternary redundant De Bruijn codes. The table presents the characteristic values for fault-free topologies, at points where significant changes occur, and with 10% node failures.

Table 10. Characteristics of Topology A (01T) as the number of mediating node failures increases

Number of failures	Diameter	Degree	Average diameter	Cost	Topological traffic	SD
0	7	6	5.50623	91854	1.83541	42
15	8	6	5.5189	104256	1.83963	48
21	9	6	5.52433	116964	1.84144	54
92	10	6	5.61364	125700	1.87121	60
210	11	6	5.80115	130482	1.93372	66
219	11	6	5.8164	129888	1.9388	66

Table 11. Characteristics of Topology B (ZT012A) as the number of mediating node failures increases

Number of failures	Diameter	Degree	Average diameter	Cost	Topological traffic	SD
0	5	10	4.33399	156250	0.8668	50
13	6	10	4.33707	186720	0.86741	60
98	7	10	4.36586	211890	0.87317	70
183	8	10	4.39902	235360	0.8798	80
313	8	10	4.48885	224960	0.89777	80

Table 12. Characteristics of Topology C (EZT0123) as the number of mediating node failures increases

Number of failures	Diameter	Degree	Average diameter	Cost	Topological traffic	SD
0	4	14	3.59178	134456	0.51311	56
18	5	14	3.59189	166810	0.5136	70
82	6	14	3.59521	194796	0.51591	84
176	7	14	3.61135	218050	0.52322	98
295	7	14	3.66253	211680	0.52925	98

Based on the obtained results, it can be observed that to significantly degrade the topological metrics, at least 7 nodes need to fail. Additionally, if the subsequent 8 nodes also fail, the metrics deteriorate even further. Therefore, it can be asserted that the identification of these nodes is crucial for maintaining the system's characteristics and performance.

In conclusion, the study underscores the importance of identifying and addressing critical nodes to ensure the resilience and efficiency of the system. The findings emphasize the need for proactive measures to monitor and manage potential node failures, thus safeguarding the overall stability and functionality of the network topology.

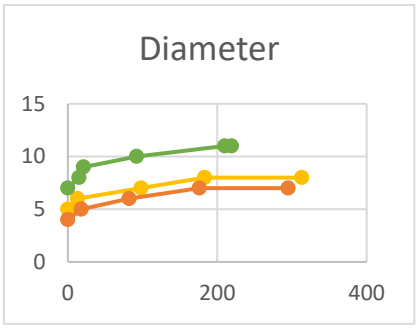

Fig.4. Dependency of diameter on the number of attacked nodes. Topologies A (green), B (yellow), C (orange)

As evident from the diagrams in the figure 4, the diameters of all topologies undergo significant changes when nodes with the highest intermediary coefficient values fail.

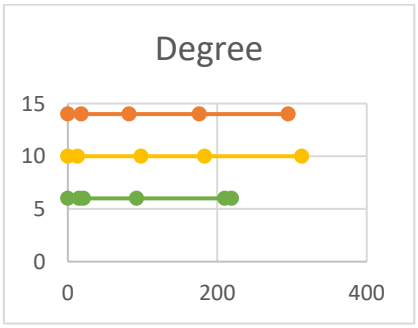

Fig.5. Dependency of degree on the number of attacked nodes. Topologies A (green), B (yellow), C (orange)

However, the potential failure of 10% of nodes does not affect the overall degree of the system in any way.

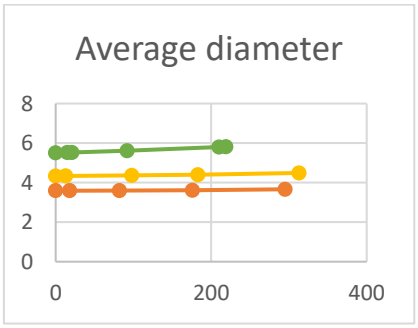

Fig.6. Dependency of the average diameter on the number of attacked nodes. Topologies A (green), B (yellow), C (orange)

The average diameter increases by 2% for the first topology, 5.3% for the second topology, and 3.6% for the third topology.

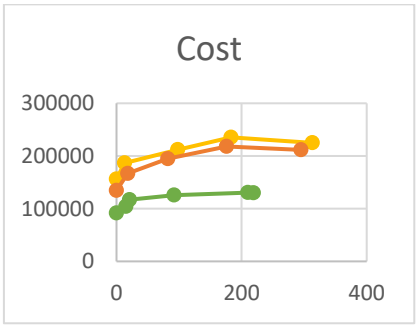

Fig.7. Dependency of cost on the number of attacked nodes. Topologies A (green), B (yellow), C (orange)

The cost depends on the diameter and the number of nodes, which is why the trends observed in the cost graphs are similar to the diameter graphs.

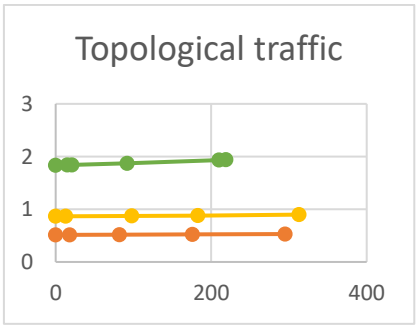

Fig.8. Dependency of topological traffic on the number of attacked nodes. Topologies A (green), B (yellow), C (orange)

The traffic for each topology increases with the failures of nodes with the highest betweenness centrality. For the first topology, this increase is 3%; for the second one, it is 5.3%; and for the third one, it is 3.5%

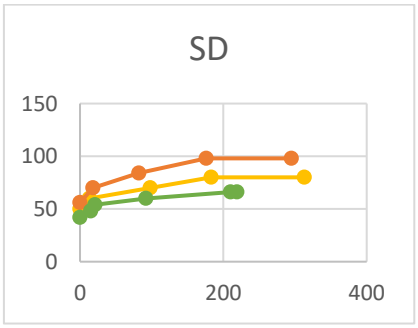

Fig.9. Dependency of the multiplicative parameter SD on the number of attacked nodes. Topologies A (green), B (yellow), C (orange)

The multiplicative parameter SD has the same growth as the diameter, since the degree remains constant throughout the studied area.

5. SDN Implementation

The theoretical assumptions made in the article already have some practical implementation within SDN technology. It separates the control plane from the data plane and implements each separately. The separation of the control plane and the data plane allows network switches to become simple forwarding devices, and the control logic to be implemented in a centralized controller. Thus, the flexibility of the network is increased due to the division of network management into parts. As traditional routing methods become more complex as networks scale, this approach to communicating information between multiple nodes becomes extremely efficient [15].

Applications for SDN networks are run directly on the controller. In particular, they are responsible for managing the flow table on network devices (data plane). Controllers often have their own set of common application modules, such as a forwarding switch, a router, a basic firewall, and a simple load balancer. Such features are called SDN applications and are often bundled with the controller [15].

The main functions of the controller include detection of end-user devices, detection of network devices, topology control of network devices, flow control.

Important for our study is that the controller maintains a view of the entire network and directly manages all SDN devices that make up the network infrastructure by providing APIs to them. The most common API used to communicate with the controller is the OpenFlow protocol. Thus, the controller actually knows the mediation coefficient of each vertex of the graph due to the number of flows passing through it. In addition, controllers implement common solutions for routing, forwarding, and load balancing. The controller automatically monitors the load on network nodes and balances traffic accordingly to keep connections working. This makes it possible to improve the versatility of the design and reduce the dependence of the reliability of the network infrastructure on the chosen type of topology.

Using a centralized multipath routing algorithm on the controller can increase network performance by 10-15% by reducing the volume of service packets. Compared to distributed methods of traffic management and balancing, it also increases the maximum utilization of communication channels by 60%. The use of the wave routing algorithm allows simultaneously forming an optimal set of paths not only between the initial and final nodes, but also from all intermediate nodes to the final node. This allows us to effectively balance traffic in a computer network, increasing reliability and

reducing dependence on topology connectivity [16].

We will conduct fault tolerance testing on a virtual network model based on the mininet utility and ONOS SDN controller. For testing, a topology with the following characteristics was created:

- containing hosts (user computers) connected to each other in the network with the address 10.0.0.0/8;
- containing vSwitches transmitting traffic based on the OpenFlow protocol;
- containing one SDN controller that controls all switches in the network;
- works on OpenFlow protocol version 1.3 for correct compatibility with ONOS controller software version [17].

We have created a mininet graphical scheme based on Figure 1 of this article (Fig. 10). The Iperf utility allows us to exchange traffic between vertices and obtain statistical results of the volume of this traffic. The virtual network model is idealized, so packet losses are minimal - this will allow us to assess the dependence of traffic flow on the topology.

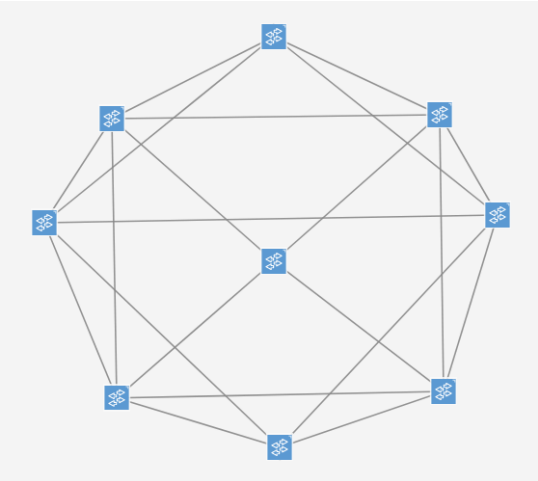

Fig.10. 01T topology presented on the ONOS controller

Let's find out the difference in bitrate between two nodes after switching off some random intermediate nodes (Table 13).

Table 13. Transfer and bitrate between two nodes for 01T topology in SDN network

Nodes	Captured on	Transfer, GB	Bitrate, GBits/sec
9	sender	55.0	48.0
	receiver		47.7
8	sender	53.9	46.3
	receiver		46.2
7	sender	55.0	47.2
	receiver		47.0
6	sender	53.1	45.6
	receiver		45.4
5	sender	54.9	47.1
	receiver		46.9

Based on the data obtained, we see that the network throughput has not changed much: the largest difference in bitrate was 5% (comparing the bitrate for the sender on 6 working nodes and on all 9). This fact is the result of several factors:

- the topology developed by the authors is sufficiently connected to guarantee the transmission speed and connectivity between the nodes even if some intermediate nodes fail to forward traffic;
- due to the way the iperf utility works, there is some statistical error, which can be seen from slightly different traffic volumes sent during one test session for each topology option;
- the load balancer built into the SDN controller reconfigures the network in a timely manner after the failure of each node, which allows us to keep the transmission speed almost at the same level [18].

Features of the controller also present for non-de Bruijn topologies. Mininet allows us to build a simple torus topology that is actively used in practice. Let's build 3x3, 6x6, and 9x9 torus (Fig. 11), check the traffic between two vertices, then disable a quarter of the links in each topology to simulate a mass shutdown of the nodes, and perform the recheck.

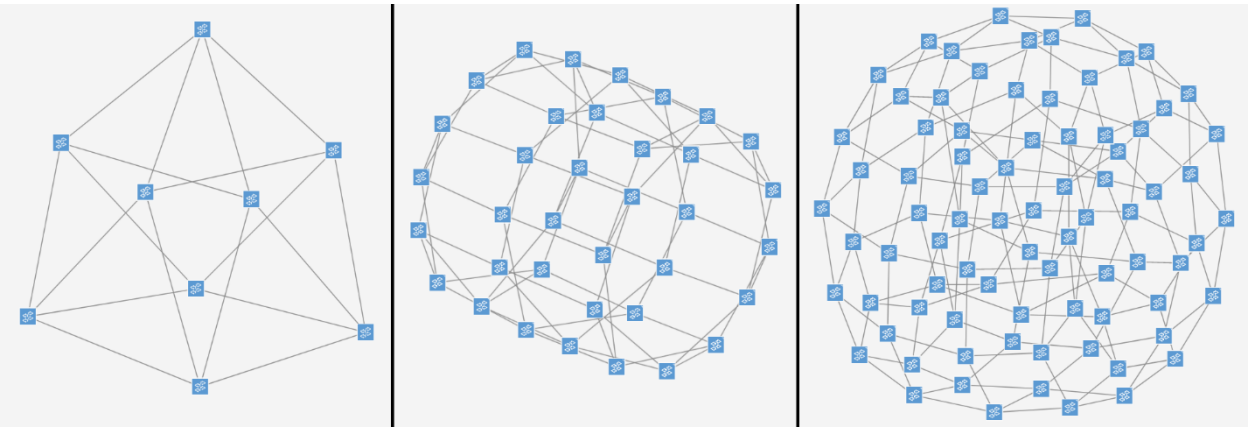

Fig.11. 3x3, 6x6 and 9x9 torus presented on the ONOS controller

In this case, we compare the results of traffic exchange between all vertices for fully operational topologies and after disconnection of 25% of links (Table 14).

Table 14. Transfer and bitrate between two nodes for torus topology in SDN network

Torus	Nodes	Links	Captured on	Transfer, GB	Bitrate, GBits/sec
3x3	9	18	sender	74.5	64.0
			receiver		63.7
		13	sender	74.4	63.9
			receiver		63.6
6x6	36	72	sender	72.0	61.8
			receiver		61.5
		54	sender	70.9	60.9
			receiver		60.7
9x9	81	162	sender	60.3	64.6
			receiver		64.5
		122	sender	68.5	63.1
			receiver		63.0

A similar trend was obtained for all topology sizes: even after disconnecting 25% of the links, the end-vertice bitrate drops very little (from 0.2% for a 9-vertex topology to 2.4% for 81), suggesting that the controller effectively routes and balances traffic. This is a practical demonstration that the use of the SDN controller to build the network topology will allow traffic exchange to be carried out so efficiently that even packet losses caused by the imperfection of real transmission protocols will not reduce the effectiveness of the network even should some nodes fail.

The test results demonstrate an obvious advantage of centralizing traffic management: eliminating the need to exchange service information between network link devices, which always leads to time costs in distributed traffic design methods. On the other hand, since all information important for management is stored in one place, there is a need to significantly increase the security of the controller in order to avoid unauthorized access to it [19].

6. Conclusions and Recommendations

The investigated topologies were designed with resilience in mind, which is why the experiments required a substantial number of failures in intermediary nodes to significantly degrade the characteristics. Despite attacking 10% of nodes with the highest intermediary coefficients in the system, the metrics of topological traffic and average diameter remained similar to their initial values. For example, topology 01T demonstrates growth of diameter (D) and characteristics of SD by +57.1%, cost (C) – by +41.4%, average diameter (AD) and topological traffic (T) – by +5.6%. Similarly, for the 012TZ topology, the deterioration of the characteristics is, respectively, D and SD +60%, AD and T

+3.6%, C +44%; for topology 0123TZE - D and SD +75%, AD and T +2%, C +57,4%. Substantial diameter growth leads to greater network delays and necessitates the development of programs considering higher latency. Thus, it can be stated that the proposed method for identifying intermediary nodes has proven effective in locating vulnerable nodes.

The proposed topology was tested using the SDN network model. Results show good effectiveness in using the SDN controller for the topology. The controller can dynamically reconfigure the routing to avoid bitrate losses so the real network can be built based on this stack. From the results of experiments, we can see that the network throughput has not changed much: the largest difference in bitrate was 5% (comparing the bitrate for the sender on 6 working nodes and on all 9). This means that SDN-based network architecture model can work effectively with De Brujin topology and provide much better fault tolerance than traditional network which cannot be dynamically reconfigured.

For further investigations, as a potential modification, analyzing shortest paths for establishing intermediary nodes in topologies with hidden connections could be considered. Another avenue for exploration could involve developing topological organizations that incorporate intermediary roles and employ additional nodes in these vulnerable sections. For SDN integration further studies may include checking the capabilities of the routing controller in the network in case of disconnection of the application intended for OpenFlow routing; stress test of the controller when scaling the topology and transmitting a large volume of traffic through network devices; comparison of the performance of the controller based on the routing algorithm using known routes for different topologies rather than De Brujin.

References

[1] A. Dodonov and D. Lande, "Modeling the survivability of network structures," in *CEUR Workshop Proceedings*, vol. 2859, pp. 1-10, 2021.

[2] M. Y. Teh, J. J. Wilke, K. Bergman, and S. Rumley, "Design space exploration of the dragonfly topology," in *High Performance Computing: ISC High Performance 2017 International Workshops, DRBSD, ExaComm, HCPM, HPC-IODC, IWOPH, IXPUG, P^ 3MA, VHPC, Visualization at Scale, WOPSSS*, J. J. Wilke, Ed., pp. 57-74, 2017.

[3] V. Rusinov, O. Honcharenko, A. Volokyta, H. Loutskii, O. Pustovit, and A. Kyrianov, "Methods of Topological Organization Synthesis Based on Tree and Dragonfly Combinations," in *Advances in Computer Science for Engineering and Education VI. ICCSEEA 2023*, Z. Hu, I. Dychka, and M. He, Eds., pp. 43-55, Springer, 2023. [Online]. Available: https://doi.org/10.1007/978-3-031-36118-0_43

[4] M. A. Nugroho and A. Rakhmatsyah, "Simulation of Jellyfish Topology Link Failure Handling Using Floyd Warshall and Johnson Algorithm in Software Defined Network Architecture," in *2021 9th International Conference on Information and Communication Technology (ICoICT)*, Aug. 2021, pp. 144-148.

[5] Y. Ajima, "High-dimensional Interconnect Technology for the K Computer and the Supercomputer Fugaku," *Fujitsu documentation*, 2020. [Online]. Available: https://www.fujitsu.com/global/about/resources/publications/technicalreview/topics/article005.html

[6] Y. Ajima et al., "The tofu interconnect d," in *2018 IEEE International Conference on Cluster Computing (CLUSTER)*, Sep. 2018, pp. 646-654. [Online]. Available: https://ieeexplore.ieee.org/document/8515069

[7] O. Godspower and V. I. E. Anireh, "EVOLUTION OF SUPERCOMPUTER ARCHITECTURE: A SURVEY," *International Journal of Computer Science and Mobile Applications*, vol. 10, no. 7, pp. 16-26, 2022.

[8] Y. Kulakov, A. Kohan, and S. Kopychko, "Traffic Orchestration in Data Center network based on software-defined networking technology," in *Advances in Computer Science for Engineering and Education II*, pp. 228–237, 2019. [Online]. Available: https://doi.org/10.1007/978-3-030-16621-2_21

[9] S. Hossen, "Traffic Engineering in Software Defined Network (SDN)," *East west university, Dhaka*, 2022. [Online]. Available: https://doi.org/10.13140/RG.2.2.34595.32807

[10] Z. Zhang et al., "Joint computation offloading and coin loaning for blockchain-empowered mobile-edge computing," *IEEE Internet of Things Journal*, vol. 6, no. 6, pp. 9934-9950, Jun. 2019.

[11] E. F. Rohman, R. Ritzkal, and Y. Afrianto, "Fail Path Analysis on Openflow Network Using Floyd-Warshall Algorithm," *Jurnal Mantik*, vol. 4, no. 3, pp. 1546-1550, 2020.

[12] O. G. Rehida, P. Rehida, A. Volokyta, H. Loutskii, and V. D. Thinh, "Routing Method Based on the Excess Code for Fault Tolerant Clusters with InfiniBand," in *Advances in Computer Science for Engineering and Education II. ICCSEEA 2019*, pp. 228-237, Springer, 2020. [Online]. Available: https://doi.org/10.1007/978-3-030-16621-2_31

[13] H. Loutskii, A. Volokyta, P. Rehida, O. Goncharenko, "Using excess code to design fault-tolerant topologies," *Technical sciences and technologies*, vol. 1, no. 15, pp. 134–144, 2019. [Online]. Available: https://doi.org/10.25140/2411-5363-2019-1(15)-134-144

[14] H. Loutskii et al., "Increasing the fault tolerance of distributed systems for the Hyper de Bruijn topology with excess code," in *2019 IEEE International Conference on Advanced Trends in Information Theory (ATIT)*, pp. 1-6, Kyiv, Ukraine, 2019. [Online]. Available: https://doi.org/10.1109/ATIT49449.2019.9030487

[15] A. Ahmad et al., "A Review on Software Defined Network (SDN) Based Network Security Enhancements," *International Journal of Computer Science and Mobile Applications*, vol. 7, pp. 1-8, 2021.

[16] A. M. A. Alnaser, Y. Kulakov, and D. Korenko, "Modified Method of Traffic Engineering in DCN with a Ramified Topology," *International Journal of Advanced Computer Science and Applications (IJACSA)*, vol. 12, no. 12, 2021. [Online]. Available: http://dx.doi.org/10.14569/IJACSA.2021.0121258

[17] D. Korenko, O. Cherevatenko, V. Rusinov, and Y. Kulakov, "Creation of the method of multipath routing using known paths in software-defined networks," *Technology Audit and Production Reserves*, vol. 4, no. 2(66), pp. 19–24, 2022. [Online]. Available: https://doi.org/10.15587/2706-5448.2022.262787

[18] O. Cherevatenko and Y. Kulakov, "Modeling of the multipath routing using known paths with ONOS SDN controller," *Problems of informatization and management*, vol. 2(74), pp. 55-61, 2023. [Online]. Available: https://doi.org/10.18372/2073-4751.74.17882

[19] M. S. Farooq, S. Riaz, and A. Alvi, "Security and Privacy Issues in Software-Defined Networking (SDN): A Systematic Literature Review," *Electronics*, vol. 12, no. 14, p. 3077, Jul. 2023. [Online]. Available: https://doi.org/10.3390/electronics12143077

Authors' Profiles

Associate professor Artem Volokyta, Department of Computer Engineering, National Technical University of Ukraine, "Igor Sikorsky Kyiv Polytechnic Institute", Ukraine.
(ORCID ID https://orcid.org/0000-0001-9069-5544) artem.volokita@kpi.ua

Major interests: High-performance computer systems and networks: theory, methods and means of hardware and software implementation; design of fault-tolerant distributed computing systems; network topological organization.

Professor Heorhii Loutskii, Department of Computer Engineering, National Technical University of Ukraine, "Igor Sikorsky Kyiv Polytechnic Institute", Ukraine
(ORCID ID https://orcid.org/0000-0002-3155-8301)

Major interests: High-performance computer systems and networks: theory, methods and means of hardware and software implementation; design of fault-tolerant distributed computing systems; network topological organization.

PHD Student Oleksandr Honcharenko, Department of Computer Engineering, National Technical University of Ukraine, "Igor Sikorsky Kyiv Polytechnic Institute", Ukraine
(ORCID ID https://orcid.org/0000-0002-9086-6988)

Major interests: High-performance computer systems and networks: theory, methods and means of hardware and software implementation; design of fault-tolerant distributed computing systems; network topological organization.

PHD Student Oleksii Cherevatenko, Department of Computer Engineering, National Technical University of Ukraine, "Igor Sikorsky Kyiv Polytechnic Institute", Ukraine
(ORCID ID https://orcid.org/0000-0001-9686-0555)

Major interests: Designing computer networks and organizing computing processes in them, developing and implementing network technologies, SDN technology, multipath routing algorithms, network topological organization, traffic and load balancing.

PHD Student Volodymyr Rusinov, Department of Computer Engineering, National Technical University of Ukraine, "Igor Sikorsky Kyiv Polytechnic Institute", Ukraine
(ORCID ID https://orcid.org/0000-0002-4362-0248)

Major interests: Computer engineering, cloud computing, neural networks, high-performance computer systems and networks, design of fault-tolerant distributed computing systems; network topological organization.

Professor Yurii Kulakov, Department of Computer Engineering, National Technical University of Ukraine, "Igor Sikorsky Kyiv Polytechnic Institute", Ukraine
(ORCID ID https://orcid.org/0000-0002-8981-5649)

Major interests: Designing computer networks and organizing computing processes in them, increasing the efficiency of problem-oriented distributed information processing systems, developing and implementing network technologies, developing methods and means of organizing the functioning of territorially distributed cluster and GRID systems.

Doctoral candidate Serhii Tsybulia, Scientific and methodological center for the organization of scientific and technical activities, The National Defence University of Ukraine, Ukraine.
(ORCID ID https://orcid.org/0000-0003-0323-1771)

Major interests: artificial intelligence, machine learning, artificial neural networks, convolutional network, computer vision, camouflage of objects, camouflage synthesis, texture mapping.

How to cite this paper: Artem Volokyta, Heorhii Loutskii, Oleksandr Honcharenko, Oleksii Cherevatenko, Volodymyr Rusinov, Yurii Kulakov, Serhii Tsybulia, "Fault Tolerance Exploration and SDN Implementation for de Bruijn Topology based on betweenness Coefficient", International Journal of Computer Network and Information Security(IJCNIS), Vol.16, No.1, pp.97-112, 2024. DOI:10.5815/ijcnis.2024.01.08