UDC 004.77 https://doi.org/10.20535/2786-8729.4.2024.303467

METHOD OF INCREASING DATA TRANSMISSION STABILITY IN SOFTWARE DEFINED NETWORK CONSIDERING METRICS OF QUALITY OF SERVICE

Oleksii Cherevatenko
National Technical University of Ukraine
“Igor Sikorsky Kyiv Polytechnic Institute”, Kyiv, Ukraine
ORCID: http://orcid.org/0000-0001-9686-0555

Yurii Kulakov
National Technical University of Ukraine
“Igor Sikorsky Kyiv Polytechnic Institute”, Kyiv, Ukraine
ORCID: http://orcid.org/0000-0002-8981-5649

This article presents a description of the method of increasing data transmission stability using the dynamic reconfiguration of a software-defined network (SDN) considering the parameters of the quality of service of communication channels and the reliability of nodes. It is proposed to consider four quality of service metrics for path construction: the number of hops between vertices, the available bandwidth, the time delay, and the percentage of lost packets.

Keywords: SDN, multipath routing, virtual network, dynamic reconfiguration, topology

1. Introduction

The practice of using computer networks in recent decades shows a tendency towards a rapid increase in the number of users. This leads to increased risks for the functioning of the network infrastructure like overloading of communication channels, a drop in the connection speed and, as a result, the failure of separate nodes or the entire network as a whole. In addition, the presence of the human factor in network configuration and the need for a separate configuration for each node in the system significantly impairs the flexibility and scalability of traditional computer networks. The technology of software defined networking (SDN) is designed to solve this problem. The implementation of such networks is relevant for a number of reasons, as they embody an innovative approach to network management. Thus, SDN allows network administrators to centrally manage and configure network devices through a special controller. Due to the centralized management of traffic and resources, it is possible to avoid excessive use of bandwidth and optimize paths between nodes. Compared to traditional networks, SDN networks scale much more effectively, and their management is simple and understandable because of the use of software interfaces between the controller and network devices. A centralized approach to management opens up opportunities to modify existing routing algorithms to speed up data transmission and achieve lower reception losses. Since computing capabilities and centralization of management now allow more effective approaches to be applied, the problem of increasing routing efficiency can be solved comprehensively today. Therefore, the construction of a software-defined network with an optimized routing algorithm becomes an extremely actual problem to solve.

2. Literature review and problem statement

The construction of an SDN network with a controller that interacts with all switches through a specialized tree is described in the article [1]. It solves the task of finding the minimum tree by weight, where the weight is defined as the sum of the weights of all switches, and the tree itself is built by the proposed heuristic algorithm. This algorithm minimizes not only the weight of the tree, but also the average distance between the controller and the switch. However, this approach is only a concept of the idea of distance metric calculations based on the number of transitions. It is expected

© The Author(s) 2024. Published by Igor Sikorsky Kyiv Polytechnic Institute.
This is an Open Access article distributed under the terms of the license CC BY 4.0 (https://creativecommons.org/licenses/by/4.0/), which permits re-use, distribution, and reproduction in any medium, provided the original work is properly cited.

that it will encounter unaccounted delays and channel overload after being implemented in a real network.

A number of publications study the problem of channel load balancing in SDN networks. For example, in the article [2], a combination of weight planning strategy and dynamic route switching is proposed to solve the congestion problem, which forms a new load balancing algorithm. It uses information hashing, which helps to reduce latency and also provides a certain degree of security. Nevertheless, the algorithm has only been tested on one fairly simple topology, and its effectiveness has not been proven for high-dimensional networks with a complex topological graph.

An original approach to solve the problem of considering channel congestion in an SDN network was proposed in the article [3]. When a problem area appears and it is necessary to bypass it, the theory of decision-making under conditions of uncertainty is used, since the nature of traffic transmission is considered when choosing the best bypass option. Using this approach, the loss of inelastic traffic would be reduced. The article considers a method of reducing the probability of dynamic reconfiguration by predicting the loading of channels in the transport network. The SDN controller generates disjoint paths using the backwave algorithm, and then the estimates of the generated paths are calculated based on fuzzy logic inference. But although the algorithm developed in these studies improves the quality of service in software-defined networks, its practical implementation may lead to problems with scaling and uniform loading of channels. That is because the use of fuzzy logic with an increase in the number of hops carries the risks for the route that is built between hosts being too hypothetical.

A number of recent publications deal with the search for new solutions to the problem of finding the shortest paths. Thus, in [4], the idea of mathematical prioritization of flows into nodes with multiple destinations is proposed. By prioritizing routes to each destination node, the proposed algorithm allows traffic to reach destinations faster. However, although thread prioritization can be useful, this approach can lead to an uneven distribution of resources. That means that some data streams will receive fewer resources than needed, which can reduce overall network performance.

A similar solution is proposed in [5]. The algorithm developed in the study is designed to find the optimal shortest path in each network that satisfies certain constraints under the control of the ONOS controller. But in this article, the author did not analyze the scalability and fault tolerance of their algorithm. They did not consider the necessity and possibility of convergence based on dynamic changes in the network, such as changes in the traffic structure or node failure.

There are publications that suggest routing process by distance vector in a software-defined network. For example, in [6], multipath routing is applied according to the route determination criterion. When determining the main path to the final host, routes from intermediate links to it are also calculated, which contributes to a more even distribution of the load. Then the route information is saved and is not recalculated for previously formed paths. This approach is quite promising and fully implements the concept of SDN. But still, it does not provide a benefit in case of dynamic changes in the network configuration. It often requires reformatting of the settings on the part of the controller.

Some of the studies criticize the routing by building shortest paths. In [7], a new energy-efficient routing algorithm is proposed. It utilizes the Naïve Bayes algorithm to dynamically distribute traffic evenly across relay nodes based on their energy levels and proximity to the sink node. Authors convince that extensive experiments with the use of their algorithm demonstrate its superior performance over the widely known Dijkstra's algorithm. The problem is, the authors’ solution is too limited in use and there is no analysis of the resource utilization. Because of that, it is hard to understand whether the solution can really reduce the expenses while building an SDN architecture.

Considering the existing problems, there is a need to create a comprehensive solution for practical implementation in existing infrastructures, where the advantages of virtual networks and SDN technology would be used. The solution must consider the quality of service (QoS) parameters of the virtual channels to make data transmission more balanced and the load of the vertices to avoid problematic areas. To be widely implemented in real SDN networks, the solution, unlike most of the existing, should be hardware-efficient.

3. The aim and methods of the study

The aim of the research is to increase data transmission stability in a virtual software-defined network in case of failures of nodes and increased loads on communication channels.

The first objective of the research is to create a method for finding a set of disjoint optimal paths that would consider various metrics of the quality of service of communication channels. The second research objective is to develop a dynamic reconfiguration mechanism for SDN switch flow tables to be able to reroute traffic in case of individual node failure.

The object of the article's research is the process of building a virtual computer network based on SDN technology using a modified algorithm for finding a set of disjoint optimal paths. We also propose an improved dynamic reconfiguration mechanism in the flow tables of SDN switches.

The subject of the article's research is appropriate methods of building a virtual computer network based on SDN technology, which, due to dynamic reconfiguration, allows to increase the speed and quality of data transmission to the addressee.

The research methods of the article are the search and analysis of theoretical material about virtual computer networks, routing algorithms, principles of operation of SDN networks and controllers. In the research process, the methods of comparative analysis, statistical analysis, metrics of computational complexity, metrics of speed and amount of data transfer, software tools for modeling were used.

4. Methods of creating the virtual SDN network

4.1. Method of creating a link weight matrix based on links QoS metrics

The topology of any computer network can be represented as a directed graph *G(V, E)*, where *V* means all the nodes of the network, and *E* means all the connections between them. Each connection $e_{(i,j)} \epsilon E$ has its own properties such as available bandwidth λ_e, delay μ_e, and packet loss ρ_e.

When a routing request is made, the network status and metrics are updated to improve the accuracy of available resource data and determine a path that satisfies quality of service parameters [8]. The task of searching for the shortest path is to find a route *P* from the source node *s* to the destination node *d* that satisfies different network parameters. It allows achieving lower time costs for communication between nodes than alternative routes at the same time. For the path *P*, we define its construction function.

The main QoS metrics, which are considered when building routes, are as follows.

The hops metric indicates the number of connections between nodes *s* and *d*. In this case, the cost of each hop (and, accordingly, the weight of each edge of the graph of the topological picture) is equal to 1 as in formula (1).

$$h(P) = \sum_{u,v \in P} e, \tag{1}$$

where *h(P)* is the number of hops in the path.

The bandwidth metric denotes the minimum bandwidth between nodes *s* and *d*, and the cost metric of each channel is set inversely proportional to the bandwidth of that channel as in formula (2).

$$\lambda(P) = min_{\forall e \in P} \left\{\frac{1}{\lambda_e}\right\}, \tag{2}$$

where *λ(P)* is the minimum of the available bandwidths for links on the path.

The path delay metric is the sum of all the link delays that occur during the transmission of data as in formula (3).

$$\mu(P) = \sum_{e \epsilon P} e_\mu, \tag{3}$$

where *μ(P)* is the total path delay.

The packet loss metric indicates the percentage of lost packets compared to the total number of sent packets (respectively, a zero value of the metric means no packet loss) as in formula (4).

$$\rho(P) = 1 - \prod_{e \epsilon P}(1 - e_{\rho}), \tag{4}$$

where $\rho(P)$ is the percentage of lost packets relative to the number of sent packets.

In existing solutions, when performing network routing tasks, only individual quality of service metrics is always considered, but not all together. This is due to the load on the low-performance hardware in the routing devices, which the developers of these devices usually do not calculate for heavy computing loads. For example, the RIP protocol considers only the number of hops between nodes when building its routing tree. The OSPF protocol performs a search for the shortest paths by the number of transitions and considering the status of the communication channel (only available bandwidth). This feature limits the possibilities of building the effective communication in the network [9].

The delay function is additive. This means that the value of the total delay is always equal to the sum of the delays on the individual channels regardless of their order in the path. The bandwidth function is concave and has a negative second derivative. The packet loss function is multiplicative. This means that the value of the function for the product of the parameters is equal to the product of the function values for each of these parameters. On the other hand, it can be transformed into an additive function by taking the logarithm of the ratio. In this case it is possible to ensure network QoS [10].

To evaluate the performance of the path, we consider throughput as a performance measure that characterizes the successful delivery of data $E_{s,t}$ over the communication channel. In this case, it is a path where $E_{s,t}$ is the received d_{rx} data from the s_{tx} source, excluding retransmitted r_{data} data during the entire observation time T_t as in formulas (5), (6) [11].

$$t_h = \frac{E_{s,t}}{T_t}, \tag{5}$$

where, respectively:

$$E_{s,t} = \frac{s_{tx} - (d_{rx} - r_{data})}{s_{tx}}. \tag{6}$$

The shortest path algorithm must determine the route with the lowest link cost, defined by a value inversely proportional to the available bandwidth of each link. Dijkstra's algorithm is usually used for this type of search. Since four different metrics were chosen for the formation of routes, the options for finding the shortest path using Dijkstra's algorithm should be formed by working it out for each of them separately. Then they are combined into a single output matrix. It is necessary to consider the parameters of the path with the least number of hops, the cost of bandwidth, delay and packet loss [12].

A path of minimum hops denotes the shortest path with the least number of hops between a source node and an end node as in formula (7).

$$H = min\ (h(P)). \tag{7}$$

The minimum bandwidth cost path represents the least available bandwidth among the available links as in formula (8).

$$B = min(\lambda(P)). \tag{8}$$

The minimum delay path represents the smallest of the total delays of the available routes as in formula (9).

$$D = min(\mu(P)). \tag{9}$$

The path with minimum packet loss represents the lowest percentage of packets lost when transmitting data using the available routes as in formula (10).

$$L = min(\rho(P)). \tag{10}$$

By changing the minimum forwarding cost for the metric which is used to evaluate the construction of the shortest route, it is possible to dynamically choose a path for data between the source node and the destination node.

The priority of considering the QoS metrics of network's communication channels will be determined by the network user himself. Based on the requirements for administration, it is proposed to determine the priorities of the QoS parameters by using whole numbers which we call prioritization coefficients. For example, the consideration of bandwidth availability is more important to network data exchange than the number of required hops between switches. Then the prioritization coefficient for the B_{ij} matrix will be set higher than the H_{ij} matrix factor by a certain amount.

The possible division of the network into segments involves calculating the weights of the edges of the subnet graph separately in each segment. To exchange data between devices within the same segment, no additional actions, other than those already described, are required. In case data must pass from one segment to another through some certain nodes, the segments should be considered a single entity. Thus, for x segments, their matrices K_{ij} will be combined through summation. But the prioritization coefficients in each segment can be determined separately and considered at the stage of summation of conditional matrices of the segment [13].

We have a general mathematical representation of the formation of the weight matrix K_{ij}. It is formed by the optimized Dijkstra algorithm for finding the most optimal path between network devices as in formula (11).

$$K_{ij} = \sum_{x \epsilon N^*} (a \cdot H_{ij} + b \cdot B_{ij} + c \cdot D_{ij} + d \cdot L_{ij})_x, \tag{11}$$

where a is the hops prioritization coefficient; b – bandwidth prioritization coefficient; c – delay prioritization coefficient; d – packet loss prioritization coefficient; x is the number of the segment (subnet).

4.2. Method of prioritizing the paths by the nodes' reliability

To improve the quality of traffic transmission, the controller must collect information not only about the state of connections between nodes. In practice, switches and routers have their own hardware or software flaws that prevent them from forwarding packets between their ports in a completely error-free manner. The SDN controller has the ability to dynamically collect and update node statistics and store them in its memory. To do this, it analyzes the number of packets passing through the switch. Then it compares the volume of traffic received by the input port and that sent through the output port. This ratio is proposed to be called the reliability coefficient of the node as in formula (12).

$$p_i = \frac{e_{out}}{e_{in}}, \tag{12}$$

where e_{out} is the number of packets sent from the output port during a period of time; e_{in} is the number of packets received by the input port during the same time.

Let us consider some route which should be analyzed to calculate the probability of packet loss due to the unreliability of nodes. This time we would not consider the quality of service metrics of the links. Based on the ratio of the number of packets successfully sent from the input port to the output port of each switch to the total number of packets sent from the first node, it is possible to estimate the reliability of each node $p_i \epsilon (0, 1]$.

To calculate the probability of packet loss $P(A_i)$ for a route from a set of disjoint paths, it is suggested to use the formula (13) for the product of independent events [14]:

$$P(A_j) = 1 - \prod_{i=1}^{n} (p_i), \tag{13}$$

where $P(A_j)$ is the probability of losing a packet when it is transmitted along path j, p_i is the node reliability coefficient.

The reliability of nodes is not a metric of the quality of service of links, therefore it is not summed up in the general conditional weight matrix according to formula (11). It is proposed to be used not as a criterion, but as a limitation. With certain network settings it can be used in path selection scenarios if the administrator needs it. For example, some rule can be specified, according to which one of the alternative routes, which has a higher cost in terms of link QoS metrics, but a lower reliability of the nodes, can be considered the main route. Another option of setting the rule is to exclude a route with too low reliability of nodes from the list of alternative disjoint routes. In this case this route will be ignored by the controller for traffic routing.

5. Results of method implementation for a virtual SDN network graph

5.1. Calculating and building the paths

There are 10 nodes in the experimental graph. It is assumed that node 1 is the starting point, and a path must be built to node 10 (Fig. 1).

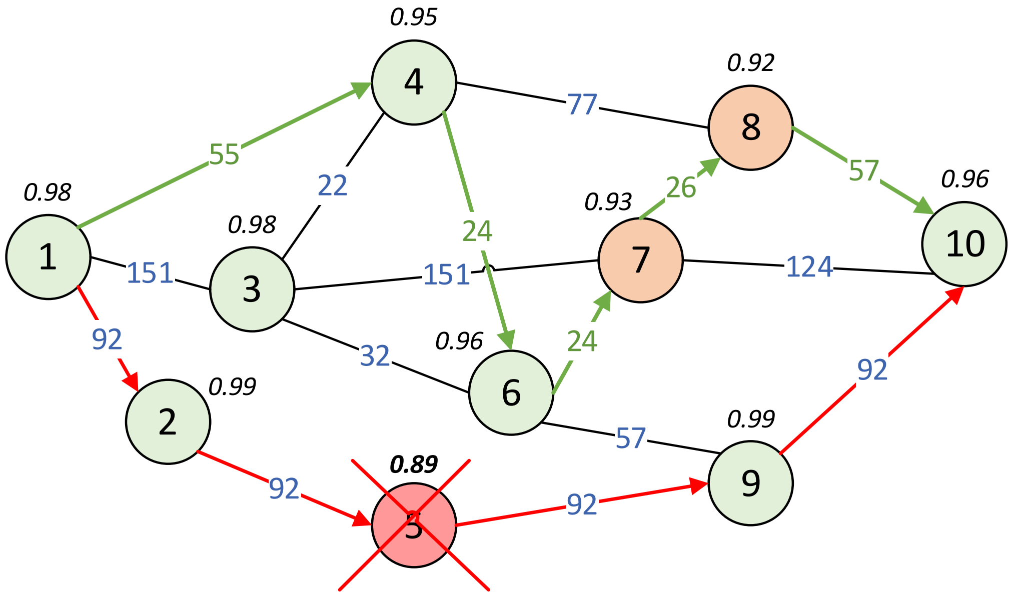

Fig. 1. Network graph after construction and prioritization of communication channels

During its calculations, the controller needs to form a set of disjoint routes from the source vertex to the addressee or conclude that it is impossible to establish a connection between them [15, 16].

Based on the requirements for prioritization coefficients and network state metrics at the time of traffic generation, the SDN controller should perform the calculation of the generalized conditional matrix based on formula (11) as follows in formula (14):

$$K_{ij} = \sum(5H + 2B + 10D + 5L\,), \tag{14}$$

where H, B, D and L are the corresponding weight matrices of the edges of the graph.

Based on the obtained matrix, according to the transition graph, two optimal communication channels can be formed: 1→4→6→7→8→10 and 1→2→5→9→10.

When using the restriction on the reliability of individual nodes p_i=*0.9* for the same problem, the first route will remain unalternative. When combining the probabilities of packet loss as independent events, this route is generally less reliable. But the second alternative is the route passes through vertex 5, which has unsatisfactory reliability according to the requirements, because $p_5 < p_i$. Because of that, the unreliable vertex will be removed from the list of vertices available for data transmission. The second route only ceases to be suitable for data transmission.

So, the route 1 is determined as the only optimal route for data transmission based on the process of dynamic network reconfiguration, which is shown in fig. 1.

5.2. Dynamic reconfiguration of the network after cutting off the nodes

With the centralized approach capabilities provided by the SDN controller, network reconfiguration can be performed. This will make data transmission to be continued seamlessly for the network state graph depicted in Fig. 1.

After selection of the optimal data transfer path, route 1→4→6→7→8→10 is the only route that is appropriate for transferring data according to the requirements. We suppose that vertices 4 and 6 have been removed from the physical topology and can no longer transport data. At the same time, the network continued to function during their removal, and according to the requirements, it should perform data transfer between vertices 1 and 10 without interruptions for the convenience of users.

The full algorithm of dynamic reconfiguration after the reconstruction of communication channels has begun in the SDN network, is shown in Fig. 2.

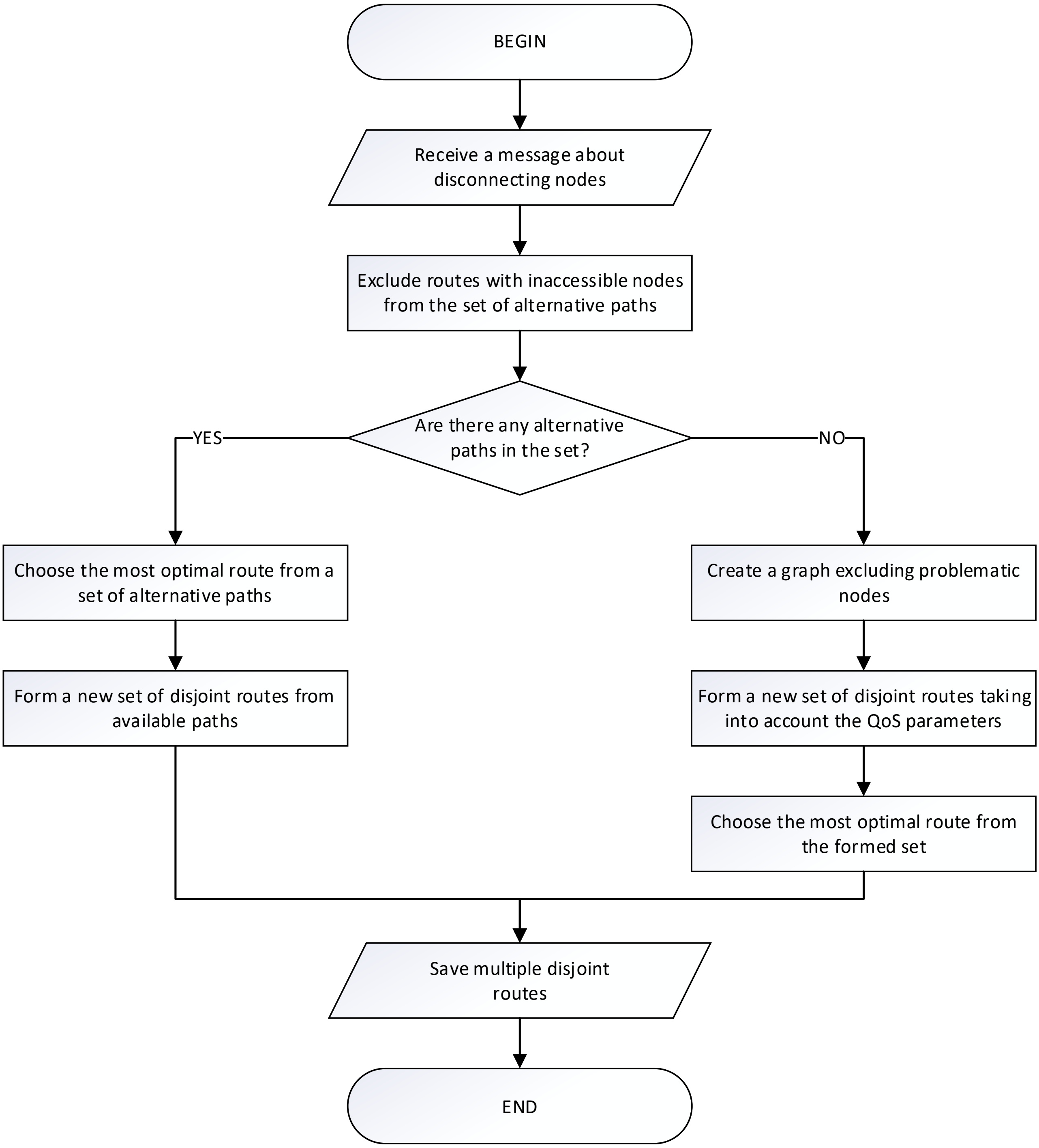
BEGIN
Receive a message about disconnecting nodes
Exclude routes with inaccessible nodes from the set of alternative paths
Are there any alternative paths in the set?
YES
NO
Choose the most optimal route from a set of alternative paths
Form a new set of disjoint routes from available paths
Create a graph excluding problematic nodes
Form a new set of disjoint routes taking into account the QoS parameters
Choose the most optimal route from the formed set
Save multiple disjoint routes
END

Fig. 2. Block diagram of SDN network dynamic reconfiguration algorithm

As soon as network nodes stop transmitting traffic, the SDN controller receives a notification, and then begins the process of reconfiguring the network to redirect the traffic [17]. The first thing it checks is the possibility of choosing an alternative path from the set of paths that were built earlier. The first from the list of the most optimal paths will be selected for traffic transmission. If there are no alternative paths, then the process of searching for new detour routes takes place. From a technical point of view, it is the same process of building communication channels as proposed earlier, but for a graph that includes information about previously defined network features. Vertices 4 and 6, as well as all connections to them on the graph, are missing due to the unavailability of these nodes for data transmission. In addition, the graph does not show vertex 5, which was removed during the construction of the original communication channels due to its non-compliance with reliability requirements.

In order to avoid a delay in the construction of a new packet transmission route, all QoS parameters that were calculated earlier are considered the same during reconfiguration. Thus, the graph with the weights of the edges, on the basis of which the costs of the routes are determined, is ready for analysis by the controller. The only technical problem that it needs to solve in order to build a new communication channel is to choose the least expensive route between the source vertex and the destination. This can be done through a one-time use of Dijkstra's algorithm. After a stable connection is established and data transmission resumes, the controller will recalculate the QoS parameters of links and the reliability of nodes. After that, it will carry out a planned reconfiguration of the network topology and, if necessary, change the optimal communication channels.

With the help of the proposed method of building communication channels and dynamic reconfiguration, the SDN controller is able to ensure stable, fast and reliable data transmission in the computer network. Due to automation, the time spent on building and prioritizing communication channels between vertices is much less than for traditional networks.

The topological graph of the network, on the basis of which the controller carries out emergency reconfiguration after some nodes have disconnected is shown in Fig. 3.

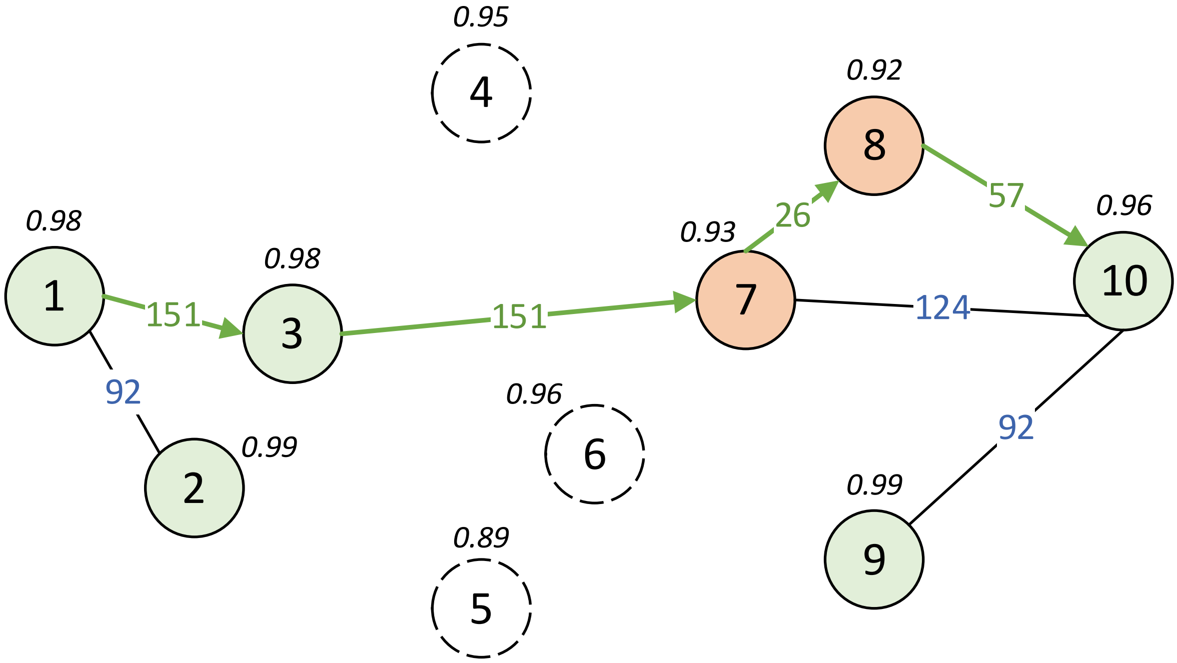

Fig. 3. Construction of a workaround after some part of the nodes were disconnected

Instead of the route 1→4→6→7, which turned out to be unavailable after disconnecting the nodes, the controller has the opportunity to choose the valuable route 1→3→7. The traffic to vertex 10 would be sent through the working segment 7→8→10 instead of direct connection 7→10.

6. Analysis of the obtained results

Implementation of the proposed methods of forming the communication channels and transmitting traffic in a software-defined network consists of the following. At first, it is the calculating QoS metrics for each of the links between devices in the network and then creating a set of disjoint paths using the modified Dijkstra algorithm. Then using the generalized weight matrix of QoS metrics of communication channels, there is a need to choose the most optimal path among the set of alternative disjoint paths. Using node reliability coefficients as constraints for choosing the optimal path between vertices, the main route from a set of alternatives can be chosen.

Finally, if in the process of traffic transmission there is a failure in the intermediate links of the network, then monitoring data of the SDN controller are used for reconfiguration and bypassing the problem area. This allows the controller to quickly choose an alternative path for sending traffic and build a new path based on already known data about communication channels in the network.

Due to the centralized approach to the traffic transmission based on SDN technology, the central controller has the current data about the topology, the state of nodes and the loading of communication channels in its memory. Therefore, it is able to perform the effective calculation of the most optimal paths between nodes when the topological graph of the network was changed. Unlike traditional networks where the service information is stored in each device, the SDN controller, having all the necessary data about the nodes, can effectively regulate the data transmission.

The expected result of the implementation of the proposed method on the SDN controller is an increase in the speed and reliability of traffic transmission. It would also lead to a uniform loading of communication channels according to the parameters of their quality of service. Unlike alternative methods of data transmission in SDN networks, the proposed method provides path calculations that require moderate amount of resources and simpler approach to be implemented in existing SDN controllers’ software solutions.

Further research in this direction will include: checking the capabilities of the controller for routing in the network when scaling the topology and transmitting a large amount of traffic through network devices; comparison of the performance of the controller based on the proposed algorithm for different topologies.

7. Conclusions

After the completion of the study, both defined tasks were accomplished. Based on a combined QoS matrix a virtual network can be created. It was described how the SDN controller will consider various network parameters in order to speed up and stabilize data transmission between hosts. In order to reduce the amount of lost data and increase the efficiency of communication channels, a method of building optimal communication channels in SDN networks was proposed. It considers quality of service parameters and node reliability coefficients. The network administrator is asked to set the prioritization coefficients for the QoS metrics and the rule for limiting the reliability of the nodes. After that the controller selects the main route and sets the hierarchy of backup routes. The controller uses this information to organize the reconfiguration of the network in case of failure of a section of the network and to bypass it by modifying previously formed routes.

The second task of the research was accomplished by describing the method of dynamic reconfiguration for the SDN network. It has been demonstrated that excluding problematic vertices from the network graph can reduce the time complexity of building the communication channels. In addition, the presence of disjoint routes allows the controller to avoid problem areas in data transmission in advance. This reduces the probability of transmission interruptions and reduces the time it takes to update network data.

References

[1] Z. Yang and K. L. Yeung, “Minimum weight controller tree design in SDN,” Computer Networks, vol. 165, p. 106949, Dec. 2019, doi: https://doi.org/10.1016/j.comnet.2019.106949.

[2] Evans Osei Kofi and E. Ahene, "Enhanced network load balancing technique for efficient performance in software defined network," PLOS ONE, vol. 18, no. 4, pp. e0284176–e0284176, Apr. 2023, doi: https://doi.org/10.1371/journal.pone.0284176.

[3] Yurii Kulakov, A. Kohan, and Yuliia Hrabovenko, "Multipath Routing in Intelligent Transport Networks," Lecture notes on data engineering and communications technologies, pp. 81–90, Jan. 2022, doi: https://doi.org/10.1007/978-3-031-04809-8_7.

[4] M. K. Sepehrifar, A. Fanian, and M. Sepehrifar, "Shortest Path Computation in a Network with Multiple Destinations," Arabian Journal for Science and Engineering, vol. 45, no. 4, pp. 3223–3231, Jan. 2020, doi: https://doi.org/10.1007/s13369-020-04340-w.

[5] S. Ray, "A Constraint Based K-Shortest Path Searching Algorithm for Software Defined Networking," Vixra.org, 2020. https://vixra.org/abs/2002.0186 (accessed Jun. 19, 2024).

[6] Y. O. Kulakov and V. Y. Shchur, "A method of balancing traffic in SDN networks based on a modified distance vector routing protocol," Problems of Informatization and Management, vol. 2, no. 74, pp. 62–67, Jun. 2023, doi: https://doi.org/10.18372/2073-4751.74.17883.

[7] Amine Tcherak, Samia Loucif, and Mohamed Ould Khaoua, "On Efficient Routing for SDN-Based Wireless Sensor Networks," Dec. 2023, doi: https://doi.org/10.1109/acit58888.2023.10453672.

[8] D. I. G. Amalarethinam and P. Mercy, "An Analysis on Quality of Service (QoS) Based Routing In Internet of Things (IoT)," International Journal of Advanced Science and Technology, vol. 29, no. 5s, pp. 488–496, Apr. 2020, Accessed: May 02, 2024. [Online]. Available: http://sersc.org/journals/index.php/IJAST/article/view/7457.

[9] M. A. Aulia, A. A. Sukmandhani, and J. Ohliati, "RIP and OSPF Routing Protocol Analysis on Defined Network Software," IEEE Xplore, Aug. 01, 2022. https://ieeexplore.ieee.org/abstract/document/9888355.

[10] J. E. Gonzalez-Trejo et al., "A Novel Strategy for Computing Routing Paths for Software-Defined Networks Based on MOCell Optimization," Applied Sciences, vol. 12, no. 22, p. 11590, Nov. 2022, doi: https://doi.org/10.3390/app122211590.

[11] M. T. Naing, T. T. Khaing, and A. H. Maw, "Evaluation of TCP and UDP Traffic over Software-Defined Networking," IEEE Xplore, Nov. 01, 2019. https://ieeexplore.ieee.org/abstract/document/8921086.

[12] P. P. Ray, "A survey on cognitive packet networks: Taxonomy, state-of-the-art, recurrent neural networks, and QoS metrics," Journal of King Saud University - Computer and Information Sciences, Jun. 2021, doi: https://doi.org/10.1016/j.jksuci.2021.05.017.

[13] F. Bannour, S. Souihi, and A. Mellouk, "Adaptive distributed SDN controllers: Application to Content-Centric Delivery Networks," Future Generation Computer Systems, vol. 113, pp. 78–93, Dec. 2020, doi: https://doi.org/10.1016/j.future.2020.05.032.

[14] M.-H. Zhou and N. Gao, "Research on Optimal Path based on Dijkstra Algorithms," Jan. 2019, doi: https://doi.org/10.2991/icmeit-19.2019.141.

[15] Y. Papageorgiou, Merkouris Karaliopoulos, and Iordanis Koutsopoulos, "Joint Controller Placement and TDMA Link Scheduling in SDN-enabled Tactical MANETs," MILCOM 2022 - 2022 IEEE Military Communications Conference (MILCOM), Nov. 2022, doi: https://doi.org/10.1109/milcom55135.2022.10017553.

[16] O. David, P. Thornley, and M. Bagheri, "Software Defined Networking (SDN) for Campus Networks, WAN, and Datacenter," Jul. 2023, doi: https://doi.org/10.1109/smartnets58706.2023.10215722.

[17] M. Hammad, Nabil Hewahi, and Wael Elmedany, "Enhancing Network Intrusion Recovery in SDN with machine learning: an innovative approach," Arab journal of basic and applied sciences, vol. 30, no. 1, pp. 561–572, Sep. 2023, doi: https://doi.org/10.1080/25765299.2023.2261219.

УДК 004.77 https://doi.org/10.20535/2786-8729.4.2024.303467

МЕТОД ПІДВИЩЕННЯ СТАБІЛЬНОСТІ ПЕРЕДАЧІ ДАНИХ У ПРОГРАМНО КОНФІГУРОВАНІЙ МЕРЕЖІ З УРАХУВАННЯМ МЕТРИК ЯКОСТІ ОБСЛУГОВУВАННЯ

Олексій Череватенко
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського», Київ, Україна
ORCID: http://orcid.org/0000-0001-9686-0555

Юрій Кулаков
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського», Київ, Україна
ORCID: http://orcid.org/0000-0002-8981-5649

У даній статті представлений опис методу підвищення стабільності передачі даних у програмно конфігурованій мережі (SDN) шляхом її динамічної реконфігурації з урахуванням параметрів якості обслуговування (QoS) каналів зв’язку та надійності вузлів.

Існуючі алгоритми не забезпечують комплексне вирішення задачі визначення найоптимальнішого шляху, який би забезпечив з'єднання від вихідної точки до адресата. Ними враховуються лише окремі параметри мережі, в залежності від яких витрати часу на зв’язок між вузлами є меншими, ніж у випадку з іншими маршрутами. В умовах зростання кількості користувачів комп’ютерних мереж та кількості вузлових комутаторів, існує актуальна проблема підвищення стабільності передачі даних у розгалужених комп’ютерних мережах. Це пропонується вирішити через урахування параметрів якості обслуговування каналів зв’язку при побудові маршрутів.

Метою дослідження є підвищення стійкості передачі даних у віртуальній програмно-конфігурованій мережі в умовах збоїв вузлів і збільшення навантаження на канали зв’язку.

Першим завданням дослідження є створення методу пошуку множини непересічних оптимальних шляхів, що враховувала би різні метрики якості обслуговування каналів зв’язку. Другим завданням дослідження є розробка механізму динамічної реконфігурації для таблиць потоків комутаторів SDN, щоб мати можливість перенаправляти трафік у разі збою окремих вузлів.

В ході дослідження перше завдання було виконане математичним обґрунтуванням побудови узагальненої вагової матриці переходів між вузлами графа мережі. При створенні запиту на визначення маршруту для трафіку у мережі її статус та показники оновлюються з метою збільшення точності даних про доступні ресурси та встановлення оптимального шляху, що відповідає критеріям якості обслуговування. Пропонується враховувати для побудови шляхів чотири метрики якості обслуговування: кількість переходів між вершинами, доступну смугу пропускання, часову затримку та відсоток втрачених пакетів. За допомогою узагальненої вагової матриці переходів, контролер SDN обирає найоптимальніший шлях серед набору альтернативних непересічних шляхів. Потім використовуючи коефіцієнти надійності вузлів як обмеження для вибору оптимального шляху між вершинами, контролер обирає основний маршрут з множини альтернатив згідно встановленій адміністратором вимозі до надійності вузлів, а інші маршрути вважає резервними.

Друге завдання було вирішене пропозицією методу динамічної реконфігурації мережі з використанням відомих шляхів. Якщо у процесі передачі трафіку виникає збій у проміжних ланках мережі або каналах зв’язку між ними, то для реконфігурації та обходу проблемної ділянки використовуються моніторингові дані SDN контролера, що дозволяють оперативно обрати альтернативний шлях для надсилання трафіку і побудувати новий шлях на базі вже відомих даних про канали зв’язку в мережі.

Отримані результати теоретичних досліджень свідчать про правильність обраних рішень та пропозицій для поставлених у статті задач.

Ключові слова: SDN, багатошляхова маршрутизація, віртуальна мережа, динамічна реконфігурація, топологія