UDC 004.77
DOI: 10.15587/2706-5448.2022.262787
Article type «Reports on Research Projects»

**Dmytro Korenko,
Oleksii Cherevatenko,
Volodymyr Rusinov,
Yurii Kulakov**

CREATION OF THE METHOD OF MULTIPATH ROUTING USING KNOWN PATHS IN SOFTWARE-DEFINED NETWORKS

The object of the research is the creation of a multipath routing algorithm for software-defined networks (SDN) including known paths, subject of the research is network parameters, designed according to a certain topology and using the developed algorithm. One of the most problematic aspects of contemporary computer networks (including applied SDN networks) is overloading. This results in harder control and limiting traffic and amount of users. Most routing algorithms that are used today have a rather large time complexity.

In the course of the study, the following methods were used: study of known routing solutions for SDN networks and results of their application, method of path metric computation on the given topology by the amount of «hops» (transitions between network nodes), optimization of procedure for finding the path using SDN technology capabilities. These methods were united and integrated into the development of the overall routing algorithm, which is proposed in this article.

The proposed multipath routing algorithm allows for the improvement of the process of traffic construction in the SDN network. This was achieved by decreasing the time complexity of the routing algorithm through the usage of previously known paths in the topology without the need to construct new ones. Involvement in the modification of the algorithm of forming distance vectors facilitated timely network reconfiguration in case its state changed. Using a centralized SDN controller made it possible to increase the stability of the network and save all configuration data in one place. The above factors make it possible to deploy an SDN network on an Edge architecture.

Obtained results of the application of the multipath routing algorithm allow to consider it effective when compared with previously proposed algorithms, based on obtained results from a practical network model, where the proposed multipath routing algorithm is used. This is because the research task was formed correctly, and the solution for this task gave correct results. Results of using the described algorithm are demonstrated, and an analysis of the obtained results is conducted, which makes it possible to confirm the accuracy of scientific research.

Keywords: *software-defined networking, multipath routing, known routes, distance vector.*

Received date: 27.06.2022
Accepted date: 09.08.2022
Published date: 15.08.2022

How to cite
Korenko, D., Cherevatenko, O., Rusinov, V., Kulakov, Y. (2022). Creation of the method of multipath routing using known paths in software-defined networks. Technology Audit and Production Reserves, 4 (2 (66)), 19–24. doi: http://doi.org/10.15587/2706-5448.2022.262787

1. Introduction

Until today, computer networks have grown big with a massive amount of data and users in them. Because of this, the process of management of the network becomes harder so as building private virtual networks (VPNs), network data centers, distributed clusters, and using cloud technologies. Edge architecture has been suggested as a solution to distribute the workload between devices, however, edge usually consists of smaller, less powerful devices, therefore a way to distribute tasks on the network efficiently, without resorting to specialized hardware [1]. Software-defined networking (SDN) is now being widely used to solve these problems. It allows to increase the efficiency of network equipment, reduce operating costs, and increase network manageability and security. Using SDN it is possible to manage the network on the software level and by that, it is possible to increase the control of transmitting data in the network [2].

Taking the growing popularity of the SDN technology into account, there is a problem of developing methods for constructing traffic especially for SDN architecture principles. In the general case, traffic engineering consists in distributing flows along selected paths. In this case, the choice of a path or the formation of a plurality of paths is determined by the given parameters of the quality of service (QoS) of the transmitted information and load balancing in the network [3].

One of the most promising ways to construct traffic in large-scale computer networks is the use of multipath routing. In this article let's describe the method of using this principle by reusing known paths to save resources while

ISSN 2664-9969

transferring data in the network with focus on QoS, load balancing and avoiding of overloads. The last is provided by multi-path routing itself, as it can use different routes to decrease the load of the network [4].

Because most of the known algorithms have significant time complexity, the best practice is to use a wave algorithm for generating the maximum set of non-intersecting paths between two graph vertices, which differs from the known algorithms by minimal time complexity [5, 6].

Software-defined network has network management which is carried out at the software level using virtual switches and a central SDN controller. Centralized management of network devices allows to expand the functionality of the traffic construction procedure. Under centralized management, the SDN controller updates the routing information for the SDN switches by updating their routing tables to choose the best route based on the specified path metric.

Compared to distributed traffic engineering methods, the centralized method eliminates the need to exchange overhead information between network switches [7].

The use of a centralized multipath routing algorithm can improve network performance by 10–15 % by reducing the volume of service packets. This reduces energy consumption by approximately 41 % and increases the maximum utilization of communication channels by 60 % compared to distributed methods of traffic engineering and its balancing [8].

The use of the wave routing algorithm allows to simultaneously form the optimal set of paths not only between the initial and final nodes, but also from all intermediate nodes to the final node. This allows for effective balancing of traffic in computer networks [9, 10].

So, the problem is actual because of the increasing size of different networks and the number of their users. As traditional routing methods get more complicated, a new way of implementing routing information between multiple nodes is needed. SDN support is also needed as this technology spreads rapidly and has promising practical application.

Thus, *the object of research* is the creation of a multipath routing algorithm for software-defined networks (SDN) including known paths, subject of the research is network parameters, designed according to a certain topology and using the developed algorithm.

The purpose of this research is to develop a method of multi routing in the network, which would provide better network parameters such as throughput and number of users, than previous methods that were already created, and reduce the time complexity of the algorithm.

2. Research methodology

The centralized way of constructing traffic allows to optimize the process of forming routes by using previously formed paths.

When constructing traffic, a set of minimal non-intersecting routes is used. This reduces the likelihood of reforming paths when the network topology or data link parameters change.

For example, for the topology shown in Fig. 1, there are two sets of minimal disjoint paths between vertices v_1 and v_{12}.

$$P_1=\{L_1(1,13),L_2(1,13),L_3(1,13)\},$$
$$P_2=\{L_1(1,13),L_2(1,13),L_4(1,13)\}. \quad (1)$$

$$L_1(1,13)=(v_1\to v_4\to v_7\to v_{10}\to v_{12}\to v_{13}),$$
$$L_2(1,13)=(v_1\to v_3\to v_7\to v_9\to v_{13}),$$
$$L_3(1,13)=(v_1\to v_2\to v_6\to v_9\to v_{13}),$$
$$L_4(1,13)=(v_1\to v_2\to v_5\to v_8\to v_{11}\to v_{13}). \quad (2)$$

As a path metric, let's use the number of transitions (hops) between network nodes. This metric is most often used in mobile computer networks.

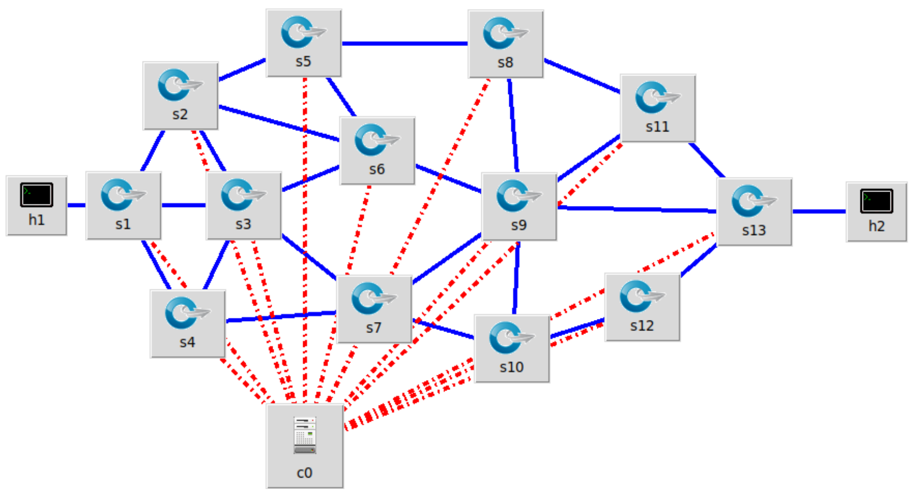

Fig 1. The network's topology

When using a multipath routing algorithm by the distance vector for the path from vertex v_1 to vertex v_{13}, a table of vectors of paths from the initial vertex v_s to the final vertex v_d towards the adjacent vertex v_a is formed (Table 1). The metric M_i is equal to the number of edges of the path l_i.

The presence of a path $L_i(s,d)$ between the vertices v_s and v_d determines the tables of distance vectors from all intermediate nodes v_j of this path to the final node v_d. For a path $L_1(1,13)=(v_1\to v_4\to v_7\to v_{10}\to v_{12}\to v_{13})$ the following path vector tables are formed (Tables 2–5).

Table 1

Vertex v_1 distance vectors

$R_i(s,d)$	v_s	v_d	v_a	M_i
$R_1(1,13)$	v_1	v_{13}	v_4	5
$R_2(1,13)$	v_1	v_{13}	v_3	4
$R_3(1,13)$	v_1	v_{13}	v_2	4
$R_4(1,13)$	v_1	v_{13}	v_2	5

Table 2

Vertex v_4 distance vectors

$R_i(s,d)$	v_s	v_d	v_a	M_i
$R_1(4,13)$	v_4	v_{13}	v_7	4

Table 3

Vertex v_7 distance vectors (first route)

$R_i(s,d)$	v_s	v_d	v_a	M_i
$R_1(7,13)$	v_7	v_{13}	v_{10}	3

Table 4

Vertex v_{10} distance vectors

$R_i(s,d)$	v_s	v_d	v_a	M_i
$R_1(10,13)$	v_{10}	v_{13}	v_{12}	2

Table 5

Vertex v_{12} distance vectors

$R_i(s,d)$	v_s	v_d	v_a	M_i
$R_1(12,13)$	v_{12}	v_{13}	v_{13}	1

For the path $L_2(1,13)=(v_1\to v_3\to v_7\to v_9\to v_{13})$ the following path vector tables are formed (Tables 6–8).

Table 6

Vertex v_3 distance vectors

$R_i(s,d)$	v_s	v_d	v_a	M_i
$R_2(3,13)$	v_3	v_{13}	v_7	3

Table 7

Vertex v_7 distance vectors (second route)

$R_i(s,d)$	v_s	v_d	v_a	M_i
$R_2(7,13)$	v_7	v_{13}	v_9	2

Table 8

Vertex v_9 distance vectors (first route)

$R_i(s,d)$	v_s	v_d	v_a	M_i
$R_2(9,13)$	v_9	v_{13}	v_{13}	1

For the path $l_3=(v_1\to v_2\to v_6\to v_9\to v_{13})$ the following path vector tables are formed (Tables 9–11).

Table 9

Vertex v_2 distance vectors (first route)

$R_i(s,d)$	v_s	v_d	v_a	M_i
$R_3(2,13)$	v_2	v_{13}	v_6	3

Table 10

Vertex v_6 distance vectors

$R_i(s,d)$	v_s	v_d	v_a	M_i
$R_3(6,13)$	v_6	v_{13}	v_9	2

Table 11

Vertex v_9 distance vectors (second route)

$R_i(s,d)$	v_s	v_d	v_a	M_i
$R_3(9,13)$	v_9	v_{13}	v_{13}	1

For a path $l_4=(v_1\to v_2\to v_5\to v_8\to v_{11}\to v_{13})$ the following path vector tables are formed (Tables 12–15).

SDN technology allows to optimize the procedure for forming new routes using already formed paths or their sections. For this purpose, in this paper it is proposed to form a matrix of vectors of optimal paths $A=\|a_{i,j}\|$, where $a_{i,j}$ is the nearest node v_a to node v_i in the direction of node v_j with the minimum path metric. In the Table 16 adjacent vertices of all vertices of the optimal path $L_3(1,13)=(v_1\to v_2\to v_6\to v_9\to v_{13})$ are showed. The result of path determination using the developed algorithm for the specified topology can be seen in Fig. 2.

Table 12

Vertex v_2 distance vectors (second route)

$R_i(s,d)$	v_s	v_d	v_a	M_i
$R_4(2,13)$	v_2	v_{13}	v_5	4

Table 13

Vertex v_5 distance vectors

$R_i(s,d)$	v_s	v_d	v_a	M_i
$R_4(5,13)$	v_5	v_{13}	v_8	3

Table 14

Vertex v_8 distance vectors

$R_i(s,d)$	v_s	v_d	v_a	M_i
$R_4(8,13)$	v_8	v_{13}	v_{11}	2

Table 15

Vertex v_{11} distance vectors

$R_i(s,d)$	v_s	v_d	v_a	M_i
$R_4(11,13)$	v_{11}	v_{13}	v_{13}	1

Table 16

Matrix of vectors of optimal paths

i \ j	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9	v_{10}	v_{11}	v_{12}	v_{13}
v_1	–	v_2	×	×	×	v_2	×	×	v_2	×	×	×	v_2
v_2	×	–	×	×	×	v_6	×	×	v_6	×	×	×	v_6
v_3	×	×	–	×	×	×	×	×	×	×	×	×	×
v_4	×	×	×	–	×	×	×	×	×	×	×	×	×
v_5	×	×	×	×	–	×	×	×	×	×	×	×	×
v_6	×	×	×	×	×	–	×	×	×	×	×	×	×
v_7	×	×	×	×	×	×	–	×	v_9	×	×	×	v_9
v_8	×	×	×	×	×	×	×	–	×	×	×	×	×
v_9	×	×	×	×	×	×	×	×	–	×	×	×	v_{13}
v_{10}	×	×	×	×	×	×	×	×	×	–	×	×	×
v_{11}	×	×	×	×	×	×	×	×	×	×	–	×	×
v_{12}	×	×	×	×	×	×	×	×	×	×	×	–	×
v_{13}	×	×	×	×	×	×	×	×	×	×	×	×	–

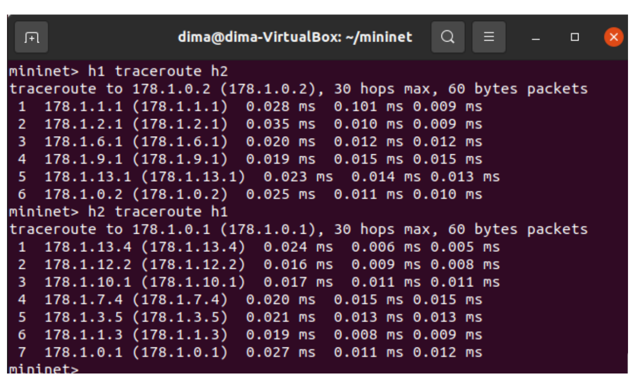

Fig. 2. Derivation of the optimal route for packet transmission from v_1 to v_{13}

The values of the optimal path vectors are formed on the basis of the vertex distance vectors (Tables 1–15). Based on Table 16, the search and formation of optimal routes is carried out.

3. Research results and discussion

A block diagram of traffic construction is demonstrated in the Fig. 3.

It is also possible to use this method to distribute the workload over the network using edge architecture. One such example includes neural enabling AI on the premise. Such network might include multiple devices, having been assigned different roles, for example, sensors, that record and sift the data, and a more powerful computer to train the neural network. Let's analyze the creation of distance vectors $R_1(v_1, v_{13}, v_a)$; $R_2(v_3, v_{13}, v_a)$ and $R_3(v_3, v_9, v_a)$ for a graph from Fig. 1.

Let's suppose there is the sensor from which it is necessary to send the data onto node that is performing machine learning tasks. For the path $L_1(1,13)=(v_1\to v_2\to v_6\to v_9\to v_{13})$ v_a corresponds to the value $d_{1,13}\in D=\|d_{i,j}\|$ and equals $v_a=v_2$. The path vector is formed $R_1(v_1, v_{13})$ and its metric is in Table 17.

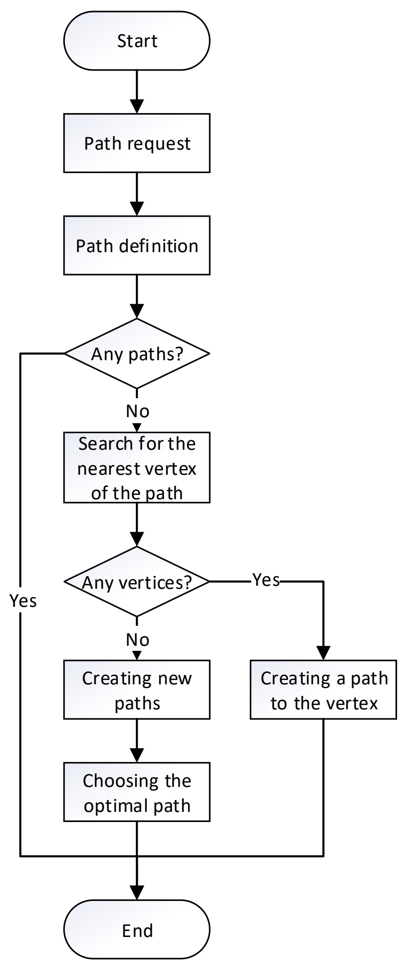

Fig. 3. The algorithm of creating the paths

Table 17

Vertex v_1 chosen distance vector

$R_i(s,d)$	v_s	v_d	v_a	M_i
$R_1(1,13)$	v_1	v_{13}	v_2	4

Let's see how the vector $R_i(v_3, v_{13}, v_a)$ paths $L_i(v_3, v_{13})$ can be created from the vertex v_3 to v_{13}.

For the path $L_i(v_3, v_{13})$ in the table $D=\|d_{i,j}\|$ there is no value for $d_{3,13}$.

Stage 1. When $i=1$ a set of vertices $V_1=\{v_3\}$. Based on $E=\|e_{i,j}\|$ let's define the set of vertices $V_2=\{v_1, v_2, v_4, v_6, v_7\}$ which are adjacent with a vertex $v_3\in v_1$. Create paths from $v_3\in v_i$ to the set of vertices $V_2=\{v_1, v_2, v_4, v_6, v_7\}$.

The path $L_1(v_3, v_1)=(v_3\to v_1)$. Distance vector from v_3 to v_1 equals $R_1(v_3, v_1, v_1)$. The path's metric $L_1(v_3, v_1)$ equals $M_1(v_3, v_1)=1$.

The path $L_2(v_3, v_6)=(v_3\to v_6)$. Distance vector from v_3 to v_6 equals $R_2(v_3, v_6, v_6)$. The path's metric $L_2(v_3, v_6)$ equals $M_2(v_3, v_6)=1$.

From the table $D=\|d_{i,j}\|$ for the vertex v_6 let's define the vector $R_1(v_6, v_{13}, v_9)$ of the path $L_1(v_6, v_{13})=(v_6\to v_9\to v_{13})$. The path's vector $L_1(v_6, v_{13})$ equals $M_1(v_6, v_{13})=2$.

A path is created $L_2(v_3, v_{13})=L_2(v_3, v_6)+L_1(v_6, v_{13})$ with vector $R_2(v_3, v_{13}, v_6)$.

The path $L_2(v_3, v_{13})=(v_3\to v_6\to v_9\to v_{13})$ with metric $M_2(v_3, v_{13})=3$.

Time complexity of path formation $L_2(v_3,v_{13})$ decreases for 2/3 in comparison with that for the whole path.

The path $L_3(v_3, v_2)=(v_3\to v_2)$. Distance vector from v_3 to v_2 equals $R_3(v_3, v_2, v_2)$. The path's metric $L_3(v_3, v_2)$ equals $M_3(v_3, v_2)=1$.

The path $L_4(v_3, v_4)=(v_3\to v_4)$. Distance vector from v_3 to v_4 equals $R_4(v_3, v_4, v_4)$. The path's metric $L_4(v_3, v_4)$ equals $M_4(v_3, v_4)=1$.

The path $L_5(v_3, v_7)=(v_3\to v_7)$. Distance vector from v_3 to v_7 equals $R_5(v_3, v_7, v_7)$. The path's metric $L_5(v_3, v_7)$ equals $M_5(v_3, v_7)=1$.

Thus, after the first iteration, these paths are formed:

$$L_1(v_3, v_1)=(v_4\to v_1),\ M_1(v_4, v_1)=1;$$

$$L_2(v_3, v_{13})=(v_3\to v_6\to v_9\to v_{13}),\ M_2(v_3, v_{13})=3;$$

$$L_3(v_3, v_{13})=(v_3\to v_2\to v_6\to v_9\to v_{13}),\ M_3(v_3, v_{13})=4.$$

$$L_4(v_3, v_4)=(v_3\to v_4),\ M_4(v_3, v_4)=1.$$

$$L_5(v_3, v_7)=(v_3\to v_7),\ M_5(v_3, v_7)=1.$$

Vectors' definitions $R_1(v_3, v_1, v_1)$, $R_2(v_3, v_{13}, v_6)$, $R_2(v_3, v_{13}, v_2)$, $R_2(v_3, v_4, v_4)$ and $R_3(v_3, v_7, v_7)$ written in the table $E=\|e_{i,j}\|$.

Stage 2. From the table $E=\|e_{i,j}\|$ let's define the vertex v_5 which is adjacent with the end vertex v_2 of the path $L_2(v_3, v_2)=(v_3\to v_2)$. A path is being created $L_2(v_3, v_5)=$ $=(v_3\to v_2\to v_5)$ with metric $M_2(v_3, v_5)=2$. Distance vector from v_3 to v_5 equals $R_2(v_3, v_5, v_2)$.

From the table $E=\|e_{i,j}\|$ let's define the vertex v_{10} which is adjacent with the end vertex v_7 of the path $L_5(v_3, v_7)=$ $=(v_3\to v_7)$. A path is being created $L_5(v_3, v_{10})=(v_3\to v_7\to v_{10})$ with metric $M_5(v_3, v_{10})=2$. Distance vector from v_3 to v_{10} equals $R_5(v_3, v_{10}, v_7)$.

From the table $E=\|e_{i,j}\|$ let's define the vertex v_2 which is adjacent to the vertex v_1 of the path $L_1(v_3, v_1)=(v_3\to v_1)$. The vertex v_2 is included in the path $L_3(v_3, v_{13})=$ $=(v_3\to v_2\to v_6\to v_9\to v_{13})$. That is why the path from v_1 to v_2 does not extend.

From the table $E=\|e_{i,j}\|$ let's define the vertex v_7 which is adjacent to the vertex v_4 of the path $L_4(v_3, v_4)=(v_3\to v_4)$. The vertex v_7 is included in $L_5(v_3, v_7)=(v_3\to v_7)$. Because of that the path from v_4 to v_7 does not extend.

Thus, after the second iteration, these paths are formed:

$L_2(v_3,\ v_5)=(v_3\to v_2\to v_5),\ M_2(v_3,\ v_5)=2;$

$L_5(v_3,\ v_{10})=(v_3\to v_7\to v_{10}),\ M_5(v_3,\ v_{10})=2.$

Vectors' definitions $R_2(v_3,\ v_5,\ v_2)$, $R_5(v_3,\ v_{10},\ v_7)$ written in the table $E=\|e_{i,j}\|$.

Stage 3. From the table $E=\|e_{i,j}\|$ let's define vertices v_6 and v_8 which are adjacent to the vertex v_5 of the path $L_2(v_3,\ v_5)=(v_3\to v_2\to v_5)$. The path is formed $L_2(v_3,\ v_6)=$ $=(v_3\to v_2\to v_5\to v_6)$ with metric $M_2(v_3,\ v_6)=3$ and the path $L_6(v_3,\ v_8)=(v_3\to v_2\to v_5\to v_8)$ with metric $M_6(v_3,\ v_8)=3$. Distance vector from v_3 to v_6 equals $R_2(v_3,\ v_6,\ v_2)$. Distance vector from v_3 to v_8 equals $R_6(v_3,\ v_8,\ v_2)$.

From the table $E=\|e_{i,j}\|$ let's define vertices v_9 and v_{12} which are adjacent to the end vertex v_{10} of the path $L_5(v_3,\ v_{10})=(v_3\to v_7\to v_{10})$. The vertex v_9 is included to the path $L_2(v_3,\ v_{13})=(v_3\to v_6\to v_9\to v_{13})$. That is why the path from v_{10} to v_9 does not extend.

The path is formed $L_5(v_3,\ v_{12})=(v_3\to v_7\to v_{10}\to v_{12})$ with metric $M_5(v_3,\ v_{12})=3$. Distance vector from v_3 to v_{12} equals $R_5(v_3,\ v_{12},\ v_7)$.

Thus, after the third iteration, these paths are formed:

$L_2(v_3,\ v_6)=(v_3\to v_2\to v_5\to v_6),\ M_2(v_3,\ v_6)=3;$

$L_5(v_3,\ v_{12})=(v_3\to v_7\to v_{10}\to v_{12}),\ M_5(v_3,\ v_{12})=3;$

$L_6(v_3,\ v_8)=(v_3\to v_2\to v_5\to v_8),\ M_6(v_3,\ v_8)=3.$

Vectors' definitions $R_2(v_3,\ v_6,\ v_2)$, $R_5(v_4,\ v_{12},\ v_7)$ and $R_6(v_3,\ v_8,\ v_2)$ written in the table $E=\|e_{i,j}\|$.

Stage 4. From the table $E=\|e_{i,j}\|$ let's define the vertex v_9 which is adjacent to the end vertex v_6 of the path $L_2(v_3,\ v_6)=(v_3\to v_2\to v_5\to v_6)$. The vertex v_9 is included in the path $L_2(v_3,\ v_{13})=(v_3\to v_6\to v_9\to v_{13})$. That is why the path v_6 to v_9 does not extend.

From the table $E=\|e_{i,j}\|$ let's define the vertex v_{13} which is adjacent to the end vertex v_{12} of the path $L_5(v_3,\ v_{12})=$ $=(v_3\to v_7\to v_{10}\to v_{12})$. The path is formed $L_5(v_3,\ v_{13})=$ $=(v_3\to v_7\to v_{10}\to v_{12}\to v_{13})$ with metric $M_1(v_3,\ v_{13})=4$. Distance vector from v_3 to v_{13} equals $R_5(v_3,\ v_{13},\ v_7)$.

From the table $E=\|e_{i,j}\|$ let's define vertices v_9 and v_{11} which are adjacent to the end vertex v_8 of the path $L_6(v_3,\ v_8)=(v_3\to v_2\to v_5\to v_8)$. The vertex v_9 is included in the path $L_2(v_3,\ v_{13})=(v_3\to v_6\to v_9\to v_{13})$. That is why the path from v_8 to v_9 does not extend.

The path is formed $L_6(v_3,\ v_{11})=(v_3\to v_2\to v_5\to v_8\to v_{11})$ with metric $M_6(v_3,\ v_{11})=4$. Distance vector from v_3 to v_{11} equals $R_6(v_3,\ v_{11},\ v_2)$.

Thus, after the fourth iteration, these paths are formed:

$L_5(v_3,\ v_{13})=(v_3\to v_7\to v_{10}\to v_{12}\to v_{13}),\ M_3(v_3,\ v_{13})=4;$

$L_6(v_3,\ v_{11})=(v_3\to v_2\to v_5\to v_8\to v_{11}),\ M_5(v_4,\ v_9)=4.$

Vectors' definitions $R_5(v_3,\ v_{13},\ v_7)$ and $R_6(v_3,\ v_{11},\ v_2)$ written in the table $E=\|e_{i,j}\|$.

Stage 5. From the table $E=\|e_{i,j}\|$ let's define the vertex v_{13} which is adjacent to the end vertex v_{11} of the path $L_6(v_3,\ v_{11})$. The path is formed $L_6(v_3,\ v_{13})=(v_3\to v_2\to v_5\to v_8\to v_{11}\to v_{13})$ with metric $M_6(v_3,\ v_{13})=5$. Distance vector from v_3 to v_{13} equals $R_6(v_3,\ v_{13},\ v_2)$.

Thus, after the fifth iteration, these paths are formed:

$L_5(v_3,\ v_{13})=(v_3\to v_7\to v_{10}\to v_{12}\to v_{13}),\ M_5(v_3,\ v_{13})=4;$

$L_6(v_3,v_{13})=(v_3\to v_2\to v_5\to v_8\to v_{11}\to v_{13}), M_6(v_3,v_{13})=5.$

Vectors' definitions $R_5(v_3,\ v_{13})$, $R_6(v_3,\ v_{13})$ written in the table $E=\|e_{i,j}\|$.

As a result, the table of vertices vectors is formed as in Table 18.

Table 18

Vertex v_4 distance vectors overall

$R_i(s,d)$	v_s	v_d	v_a	M_i
$R_2(3,13)$	v_3	v_{13}	V_6	3
$R_3(3,13)$	v_3	v_{13}	V_2	4
$R_5(3,13)$	v_3	v_{13}	V_7	4
$R_6(3,13)$	v_3	v_{13}	V_2	5

Values of the vectors of generated routes with a minimum metric are written in Table 19. This allows to determine the optimal paths from the beginning and all intermediate vertices to the final vertex without generating routes, which implies the time of creating traffic.

Table 19

Matrix of vectors of optimal paths (after the stages)

i \ j	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9	v_{10}	v_{11}	v_{12}	v_{13}
v_1	–	v_2	×	×	×	v_2	×	×	v_2	×	×	×	v_2
v_2	×	–	×	×	×	v_6	×	×	v_6	×	×	×	v_6
v_3	×	v_2	–	×	×	v_6	×	×	v_6	×	×	×	v_6
v_4	×	×	×	–	×	×	×	×	×	×	×	×	×
v_5	×	×	×	×	–	×	×	×	×	×	×	×	×
v_6	×	×	×	×	×	–	×	×	×	×	×	×	×
v_7	×	×	×	×	×	×	–	×	v_9	×	×	×	v_9
v_8	×	×	×	×	×	×	×	–	×	×	×	×	×
v_9	×	×	×	×	×	×	×	×	–	×	×	×	v_{13}
v_{10}	×	×	×	×	×	×	×	×	×	–	×	×	×
v_{11}	×	×	×	×	×	×	×	×	×	×	–	×	×
v_{12}	×	×	×	×	×	×	×	×	×	×	×	–	×
v_{13}	×	×	×	×	×	×	×	×	×	×	×	×	–

In case of network reconfiguration or optimal path channel overload SDN controller makes changes to the optimal path vector table. Path reconfiguration is performed. For example, in case of overload on the path $L_2(v_3,\ v_{13})=(v_3\to v_6\to v_9\to v_{13})$ $a_{3,6}=v_3$ is removed from optimal path vector table (Table 19). New path is chosen $L_3(v_3,\ v_{13})=(v_3\to v_2\to v_6\to v_9\to v_{13})$ with path vector $R_3(3,13)=(v_3,\ v_{13},\ v_2,\ M_3)$. Thus, value $a_{3,2}=v_2$ is added to the vector table.

The existence of multiple paths permits dynamic path reconfiguration. For example, during the process of data transfer through path $L_2(v_3,\ v_{13})=(v_3\to v_6\to v_9\to v_{13})$ part of the path $v_3\to v_6$ turns out to be overloaded. SDN controller removes $a_{3,6}$ from optimal path vector table and adds values $a_{3,7}=v_7$ and $a_{7,9}=v_9$. These values are formed using path vector $R_2(3,13)=(v_3, v_{13}, v_6, M_2=3)$ and $R_5(6,13)=(v_6, v_{13}, v_9, M_3=2)$. As a result path $L_2(v_3, v_{13})=$ $=(v_3\to v_6\to v_9\to v_{13})$ is dynamically reconfigured to $L_2(v_3, v_{13})=$ $=(v_3\to v_7\to v_9\to v_{13})$.

The main strength of this research is the new modified method that we propose. We were able to achieve greater network parameters in our solution such as traffic transferring and speed. Practical tests confirmed that this algorithm can be used on Edge architecture in various tasks benefitting from a drop in power consumption and aforementioned fault-tolerance.

Weaknesses of this method are that it only provides the way to transfer traffic but we do not take into account different types of data and potential nodes. These things are for our future research. Also, we tried our method only for a model, not for some real topology, and we also plan to do this in the future.

Additional opportunities of the solution arise form those of SDN – better speed, easier management, bigger scalability. Also, SDN is becoming more ubiquitous due to its high performance per watt ratio for Edge architecture and its high fault-tolerance in upscaled production environments.

The options proposed in this work are fully theoretical. A practical study and justification needed before a decision for possible implementation can be made. Therefore, the risks are obvious and related to the process of the practical use and implementation of the method.

4. Conclusions

In this article we did a study of already known routing solutions for SDN networks and the results of their applied use, a method of calculating path metrics on the topology by the number of «hops» (transitions between network nodes), optimization of the route search procedure using SDN technological capabilities. Based on that, our own modified method of traffic construction is presented that allows usage of known paths to lower the time complexity of the routing procedure. Formation of routing information is performed centrally in the SDN controller.

Using the modified method of distance vector formation using a multipath wave algorithm allows for a prompt rerouting procedure to be performed, depending on changes in the network status.

The main advantage of this method is that it enables sophisticated networking tasks on edge architecture, where the number of connected clients is changing. Also, this SDN is less demanding than using specialized hardware, therefore improving both speed and cost for the edge network. Energy consumption for centralized management is also up to 41 % lower than decentralized routing methods, with lower (up to 47 % decrease) power consumption numbers observed during method testing.

Additionally, to balance the network load, it is necessary to perform the dynamic rerouting of paths based on the analysis of load of individual channels. For this task, channel utilization needs to be used as a metric.

Conflict of interest

The authors declare that they have no conflict of interest in relation to this research, whether financial, personal, authorship or otherwise, that could affect the research and its results presented in this paper.

References

1. Merenda, M., Porcaro, C., Iero, D. (2020). Edge machine learning for ai-enabled iot devices: A review. *Sensors, 20 (9),* 2533. doi: http://doi.org/10.3390/s20092533
2. Akbaş, M. (2016). A Preliminary Survey on the Security of Software-Defined Networks. *Proceedings of International Conference on Advanced Technology & Sciences (ICAT'16),* 468–473.
3. Raza Shah, S. A., Bae, S., Jaikar, A., Noh, S.-Y. (2016). An adaptive load monitoring solution for logically centralized SDN controller. *18th Asia-Pacific Network Operations and Management Symposium (APNOMS).* doi: http://doi.org/10.1109/apnoms.2016.7737207
4. Lemeshko, A. V., Evseeva, O. Yu., Garkusha, S. V. (2014). Research on Tensor Model of Multipath Routing in Telecommunication Network with Support of Service Quality by Greate Number of Indices. *Telecommunications and Radio Engineering, 73 (15),* 1339–1360. doi: http://doi.org/10.1615/telecomradeng.v73.i15.30
5. Abich, G., Reis, R., Ost, L. (2021). The impact of precision bitwidth on the soft error reliability of the MobileNet network. *2021 IEEE 12th Latin America Symposium on Circuits and System (LASCAS),* 1–4. doi: http://doi.org/10.1109/lascas51355.2021.9667153
6. Kulakov, Y., Kohan, A., Kopychko, S., Cherevatenko, R.; Hu, Z., Petoukhov, S., Dychka, I., He, M. (Eds.) (2021). Load Balancing in Software Defined Networks Using Multipath Routing. *Advances in Computer Science for Engineering and Education III. ICCSEEA 2020. Advances in Intelligent Systems and Computing. Vol. 1247.* Cham: Springer, 384–395. doi: http://doi.org/10.1007/978-3-030-55506-1_35
7. Lemeshko, A. V., Evseeva, O. Yu., Garkusha, S. V. (2014). Research on Tensor Model of Multipath Routing in Telecommunication Network with Support of Service Quality by Greate Number of Indices. *Telecommunications and Radio Engineering, 73 (15),* 1339–1360. doi: http://doi.org/10.1615/telecomradeng.v73.i15.30
8. Kreutz, D., Ramos, F. M. V., Verissimo, P. J. E., Rothenberg, C. E., Azodolmolky, S., Uhlig, S. (2015). Software-Defined Networking: A Comprehensive Survey. *Proceedings of the IEEE, 103 (1),* 14–76. doi: http://doi.org/10.1109/jproc.2014.2371999
9. Chemeritskiy, E., Smeliansky, R. (2014). On QoS Management in SDN by Multipath Routing. *Science and Technology Conference (Modern Networking Technologies) (MoNeTeC).* doi: http://doi.org/10.1109/monetec.2014.6995581
10. Agrawal, R., Imieliński, T., Swami, A. (1993). Mining Association Rules Between Sets of Items in Large Databases. *ACM SIGMOD Record, 22 (2),* 207–216. doi: http://doi.org/10.1145/170036.170072

Dmytro Korenko, *Postgraduate Student, Department of Computer Engineering, National Technical University of Ukraine «Igor Sikorsky Kyiv Polytechnic Institute», Kyiv, Ukraine, ORCID: https://orcid.org/0000-0003-0463-189X*

✉***Oleksii Cherevatenko,*** *Postgraduate Student, Department of Computer Engineering, National Technical University of Ukraine «Igor Sikorsky Kyiv Polytechnic Institute», Kyiv, Ukraine, e-mail: chereva@ukr.net, ORCID: https://orcid.org/0000-0001-9686-0555*

Volodymyr Rusinov, *Postgraduate Student, Department of Computer Engineering, National Technical University of Ukraine «Igor Sikorsky Kyiv Polytechnic Institute», Kyiv, Ukraine, ORCID: https://orcid.org/0000-0002-4362-0248*

Yurii Kulakov, *Doctor of Technical Sciences, Professor, Department of Computer Engineering, National Technical University of Ukraine «Igor Sikorsky Kyiv Polytechnic Institute», Kyiv, Ukraine, ORCID: https://orcid.org/0000-0002-8981-5649*

✉*Corresponding author*