Multimedia Tools and Applications (2023) 82:33865–33888
https://doi.org/10.1007/s11042-023-14938-5

An efficient software defined network controller based routing adaptation for enhancing QoE of multimedia streaming service

Miran Taha [1]

Received: 30 September 2021 / Revised: 1 February 2023 / Accepted: 22 February 2023 /
Published online: 8 March 2023

Abstract
SDN (Software-Defined Networks) is a new network communication prototype. SDN can control the wide range of network activities and its responsibilities to select an optimum route for end-users. Recent studies are focusing on issues regarding routing congestion and delay of packets within SDNs. In this research work, an efficient and smart-based algorithm is proposed to change the directions of packets in SDN networks. The proposed model estimates the cost of the given paths in networks depending on five criteria; adaptive network packet size, accurate packet numbers, the overall required time interval, QoS (Quality of Service) link capacity (bandwidth), and the number of hops (shortest path). In this way, the optimal paths from sender to receiver can be easily determined. This mechanism allows the SDN controller to minimize the decision time that is needed for selecting the flows. According to the aforementioned criteria, a dataset has been created which contains information about routing delay. From the proposed model, three criteria which are packet size, number, and time have been used to find the optimal packet delay to be used later in the model to find the cost of each path. A benchmark comparison between state-of-the-art and the suggested algorithm reveals that the time consumption of selecting an optimal recovery path has a significant delay reduction which is estimated to be a few milliseconds. Consequently, it can reduce bottleneck routes and resource utilization. Experimental results indicate that the proposed algorithm has increased the QoE (Quality of Experience) of both objective and subjective video streaming.The model reduced the delay time of route selection up to 96.3% and this leads to end-user satisfaction.

Keywords Smart algorithm · Efficiency · SDN · Video streaming · QoE · QoS

✉ Miran Taha
miran.abdullah@univsul.edu.iq

[1] Department of Computer Science, College of Science, University of Sulaimani, Sulaymaniyah, Iraq

1 Introduction

SDN is a new paradigm in computer networking that separates the network's control and forwarding components, it makes computer networks more programmable and manageable from anywhere. A centralized SDN controller manages the switches and routers through a secured communication channel [27]. Correspondingly, any action deployed by the Flow switches will deliver the rule to the controller through the same secured channel. The SDN controller can identify flow paths and link failure [40]. Before the invention of SDN technology, high inaccuracy routing integration has been observed in the networks. In traditional networks to avoid network route failure, the network systems preserve redundant links to handle link failure. A new recovery path is determined by the network system and the system provides integrated network services [12]. In traditional networks, one of the most typical issues is the process of selecting the best network path and managing resources from the same set of accessible links [32]. SDN was offered as one of the solution strategies to address the previous issue. Two other mechanisms are also considered in routing protocols; in the reactive strategy, alternative routes are seek-on-demand which is time exceeding while in the proactive strategy, alternative paths are introduced which reduced the time consumption of delivering the packets [37]. Routing strategies are a collection of significant approaches for selecting flows in both fixed and dynamic routing networks such as Internet application service which is used by end-users [25]. The transmission of these applications is affected by QoS parameters including bandwidth, latency, packet loss, and jitter [28]. The SDN controller has also controlled the flow of network packets to integrate the network communication which dynamically assigns routes between senders and receivers for facilitating service usage. Besides, the SDN controller has decided to assign routes between the sender and receiver. This decision is changed by several parameters that are used to decide about the assigning of the routes. When the controller chooses the primary path and recovery path for routing packets, the time can be adjusted depending on the network structure and characteristics. However, when multipath routing is included, the controller faces challenges, especially if several paths are chosen as the main and recovery paths. Moreover, the decision of multiple routes is based on network configuration changes and SDN routing settings. When these settings are not properly configured the end user's path selection takes longer time and the end user is dissatisfied with the application service.

The recent studies explained the mechanisms that are used for solving the issue of network routing in different scenarios. Priyanka et al. proposed a scheme that consists of three phases; flow splitting, multipath routing, and flow reordering. In the first phase, they proposed a flow-splitting scheme to decide how to split the incoming flows to enable multipath routing in the network. In the second phase, a cost function formulation is designed for the cost routing problem as an integer linear program (ILP). Finally, in the third phase, they proposed a flow reordering scheme for the received sub-flows through multiple paths to maintain the desired flow sequence at the destination [15]. Moreover, S.A. et al. proposed a cuckoo search technique to designate an optimum routing path based on individual nodes' residual energy. Their purpose was to balance the routing overhead among the individual nodes participating in routing [2]. Furthermore, Hemalatha et al. proposed stable node prediction, stability determination, route

exploration, and packet dissemination for routing networks. At first, the stable nodes are identified using a recurrent neural network along with a modified seagull optimization approach. Through Garson's pruning-based recurrent neural network accompanied by modified seagull optimization (RMSG) algorithm, the stable neighbors are chosen. Network routing is formed by interconnecting the stable nodes from the source to the destination. When a routing link failure happens, the route recovery process will be initiated. Thus, the data packets are broadcasted in multi-paths without any intervention [9]. The concept of Quality of Service (QoS)-based routing is relatively new, with just a few protocols addressing it. The QoS is addressed using the clustering approach. Each cluster has a cluster head which is picked using the Random Selected Leader (RSL) based optimizer algorithm [35].

The main contribution of this paper is to develop a smart algorithm for an SDN controller. The proposed algorithm relies on two strategies; the first consider selecting an adaptive packet depending on the adaptive packet size, accurate packet numbers, and adequate time interval. The second strategy works on how long time is required for sniffing the QoS- link capacity and estimates the number of hops.

The remainder of the article is organized as follows: Section 2 surveys the state-of-the-art routing algorithms in SDNs. Section 3 addresses the Methodologies, Materials and describes the testbed design. While in Section 4, the experiments and test results are provided. A comparison of test results is discussed in Section 5. Finally, in Section 6 conclusions and future works are presented.

2 Literature review

key issues of SDN routing protocols are network resource utilization and delay time of path selection. There have been several studies that have been developed based on how the SDN controller can choose the best and fastest routes. The main purpose was to minimize the delay time and reduce network congestion. This section reviews the recently mentioned articles to identify the gaps that are causing the problem to the worse case as well as the suggested mechanisms based on the decision of delay time and network resource utilization. Kannan et al. proposed a congestion controller for SDNs which is capturing and analyzing the information of sharing traffic of each switch. Although, the proposed method reduced the probability of occurring congestion by redirecting the flows to alternative flow routes. Nevertheless, the weak point of this work is the lack of information collected in SDN switches which leads to an inaccurate decision in selecting a path [16]. Liu et al. focused on the bandwidth-constrained multi-path optimization problem in SDN. A genetic algorithm (GA) based SDN is presented to optimize the flow routing in the critical network environment. Shortest path and bandwidth parameters have been taken for the basic framework of GA. After implementing the proposed mechanism, the SDN controller could confirm the viability and effectiveness of routing packets. Therefore, the GA for solving the problem only searches the optimal paths and is not able to realize parallel optimization for all resources [22]. Maris et al. proposed a multipath routing strategy that uses alternate paths through a network at the same time. A high network latency problem for Spanning Tree Protocol (STP) in SDN is introduced with a multipath routing and admission control-based system. The admission system control was used to control

the amount of traffic entering into network flows and the multipath routing was used to load the balance of the flows. They aimed to minimize network latency and maximize throughput. Moreover, the proposed approach outperformed STP but the system is still fragile to provide fair resource utilization and load balancing [24]. Xu et al. proposed SDN based multi-path traffic scheduling mechanism for the imbalance network traffic of cloud data centers which leads to network congestion. Scheduling mechanisms and fat-tree approaches are used for adapting the data flows and multipath routes. The proposed mechanism managed the traffics, the loaded balance of cloud data traffic, and improved the utilization efficiency of the quality of services (QoS). However, a tremendous delay time has been produced which is estimated at 6 ms [20, 38]. M. Jothish et al. improved a multipath routing strategy in Wireless Sensor Networks (WSN). The multipath routing strategy has taken the advances of several routes from sender to receiver, and it is essentially useful to consider the traffic and reduce the network congestion in WSN. Their stated method is based on predicting the link congestion over a specific interval of time. The results showed the raise of the throughput and overcome the routing overhead in WSN. Even though, the approach has not given an accurate result [18]. Table 1 Compares the state-of-the-art SDN techniques in terms of experiments and delay time. Different strategies are studied, as explained in Table 1, and they aimed to minimize the delay time of handling links and resource usage.

Based on the gap that has been found in this survey, a smart and efficient approach is proposed which overcomes the above-mentioned issues. The adaptive packet size is used to calculate the network link weight and the different parameters considered to monitor the network traffic. The goal is to reduce the delay time of selecting paths and minimize the usage of network resources.

Table 1 Comparison of the state-of-the-art SDN approaches

Related works	Techniques	Experiments	Resource usage
Ref. [30] by S. Sharma et al.	Based on the new Address Resolution Protocol (ARP) request	The controller was able to restore the routes at 12 ms.	Median
Ref. [29] by A. Sgambelluri et al.	Based on the OSP scheme enabling fast recovery in networks	The controller decided to activate the backup route at 64 ms.	High
Ref. [7] by N. Dorsch et al.	Based on a centralized approach for both failure detection and traffic recovery bidirectional forwarding detection	The proposed method changed the failure paths at 4.5 ms	Median range of the packets is lost
Ref. [21] by Y. D. Lin et al.	Based on the static link weighting function.	The experiment results showed that the recovery time was less than 100 ms.	gigantic
Ref. [1, 11]	Based on SDN network monitoring	The goal of minimizing the amount of switchover which took ~40 ms.	High
Ref. [10] by H. H. Hsieh and K. Wang	Based on a multi-controller in SDN architecture.	The experiment results showed that handling routes took 7 ms	Low

3 Materials and methods

This section has three subsections: including; the definition of SDN Architecture, and then presenting the Proposed Algorithm, and finally, the Testbed of the system is described.

3.1 Toward SDN architecture

A traditional network has been implemented using control-plane routing protocols alongside data-plane forwarding functionalities. A network layer in a traditional router performs two important functions: routing and forwarding. A router is a device that sends packets from one interface to another. This hardware-based procedure is manageable, and it takes only a few nanoseconds to complete. The routing function provides end-to-end packet transportation from the source to the destination. Forwarding works as a routing function but with a difference; it does not manage moving data from one device to another. To calculate the distance of paths from source to destination, the routing algorithm is designed to find the best path for network routing. Traditionally, a routing algorithm runs in the network layer of the routers, therefore, the best route is determined by the network layer through which packets can be transmitted. To compute the values of the forwarding table, the routing algorithm function communicates with the routing algorithm of another router. The forwarding table deployed a decision to find the proper egress network interface to which the input interface should forward the packets. The data and control planes are developed and separated in SDN. In SDN, new protocols can be performed by software without requiring any network hardware changes. The control-plane functions are implemented in an independent controller service. Therefore, the forwarding process is performed by the routing device, while the remote controller calculates and assigns forwarding tables [23]. The remote controller and switches communicate by exchanging messages containing forwarding tables and other parts of routing information. In addition, forwarding tables are computed by the SDN controller wherever the network plane is software-defined. The controller interacts with switches through the SDN software [3]. The controller's functionality is instructed to forward packets however the data plane is unable to take forwarding decisions [31]. The open-flow messages protocol is used to provide communication between the control plane and the data plane. A secured channel is utilized to connect the controller to the switches for any rule deployments [4]. Details of the SDN layers architecture are presented in Fig. 1. Generally, three layers compromise the SDN structure. The infrastructure layer which is the first layer is corresponding to where the network forwarding equipment, it relies on the control layer to provide forwarding instructions and configuration. The second is the control layer, which is responsible for configuring the infrastructure layer. It receives service requests from the third layer (application layer). Moreover, the control layer maps the service requests to the infrastructure layer in the most optimal manner. The third layer is where Internet (cloud) applications or management applications place their demands for the network onto the control layer. In SDN, each of these layers is designed to provide agility by logically centralizing the full configuration of networks. Furthermore, The Network Services Abstraction Layer (NSAL) gives access to the services

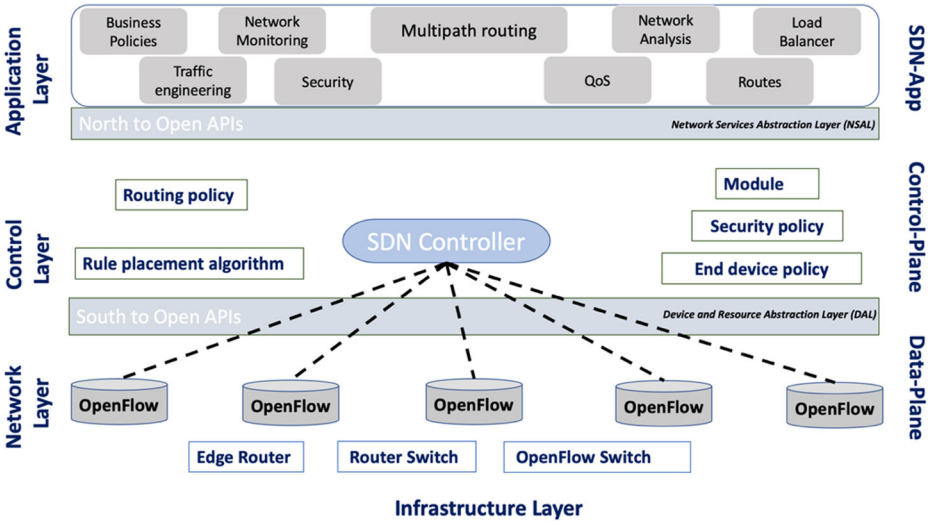

Fig. 1 SDN Architecture

between the controller and application planes. DAL (Device and resource Abstraction Layer) is one or more models that are used to construct the device's resource abstraction layer. The device's forwarding and operating resources are concentrated in DAL. Service Interfaces can take many forms, including RESTful APIs, NETCONF, inter-process communications, and CORBA interfaces. Additionally, the services residing in the Application Plane may implement services to other services. The applications that reside in the application plane through the service interface, include network topology discovery, network provisioning, and path reservation.

3.2 Proposed controller's algorithm

The most important part of the SDN controller is the decision on path selection. The SDN algorithm provides an appropriate path selection with minimum cost in the application layer. The suggested algorithm analyzes the paths using critical metrics based on connection conditions in order to select the optimum path. A high-speed adaptive routing technique selects the optimum path among several options. Based on the network topology, the optimum primary and backup pathways are computed from source to destination. This is done by the OpenFlow Controller (OFC) that manages all network information. The controller selects the best path due to the important metrics which include; QoS link capacity (Link bandwidth) and the path length (number of hops) shortest path. In this proposal, the cost of each link has been calculated by considering the link weight. Accordingly, each OpenFlow switch has a table that lists the primary and secondary pathways. Moreover, the incoming packets are transmitted to the destination through the main path where no fault occurs during the transfer. If the main path is neglected, the OpenFlow controller acquires the information from the OpenFlow switch, and the controller rearranges the flow automatically to the backup path. The

main purpose of proposing a smart mechanism is to provide a fast and adaptive routing packet. The SDN Controller uses the Link Layer Discovery Protocol (LLDP) to collect information about links and the controller has been implemented to inform about any changes that occur in the network topology. Furthermore, If a link goes down or the network topology changes, the controller's algorithm keeps all network nodes up to date. Consequently, the network adapts to the conditions when a new flow is detected. The suggested approach utilizes Depth-first search (DFS) as a simple and cost-effective algorithm to achieve the feature of finding multiple routes between two nodes, and it determines a graph to locate all available paths from source to destination. After finding all the paths between the source and destination, the DFS algorithm explores possible vertices in a graph by finding the deepest vertex in the graph first before it backtracks to find other possible vertices using a stack. In addition, the proposed method is utilized to determine the cost of available resources between two nodes (source and destination). The cost of each path connection in the SDN topology network is calculated as shown in Eq. 1.

$$Transmission\ Cost = \left(\frac{Av.BW}{Max.Thr.Intf} \right), \tag{1}$$

The (maximum throughput) absolute value that the switch's ports can provide to the network is referred to as (Av. BW) available bandwidth. The amount of throughput that can be used to transfer a data file is referred to as interface throughput denoted by *Max*. *Thr*. *Intf*. The controller has defined each switch's flow packet and executes the rule deployment of the activities to indicate the forwarding or dropping process. Also, it adds values to the SDN switches after getting all active paths. CP (Capacity path) as the bucket weight is defined in the algorithm which is used to find out the weight of the available links that evenly distribute path loads. Finally, the controller shifted traffic to an alternate path when a particular port went down instead of dropping packets. This method eliminates the downed path's disconnecting interval time. Equation 2 shows how the weight of the path in the suggested algorithm is calculated.

$$Link\ Weight = \left(\left(1 - \frac{\mathrm{CP}y}{\sum_{y}^{\mathrm{n}} \mathrm{CP}_{y}} * 10 \right) \right), \tag{2}$$

Where the bucket weight of each path is a metric used to calculate the Link Weight, which is defined in the algorithm, the number of the paths indicated by y, where n is the total number of available switches in the network topology of the SDN. CP is the cost of the path between every two nodes. In addition, the general description of the proposed model for the SDN controller is given in Algorithm 1 and Fig. 2.

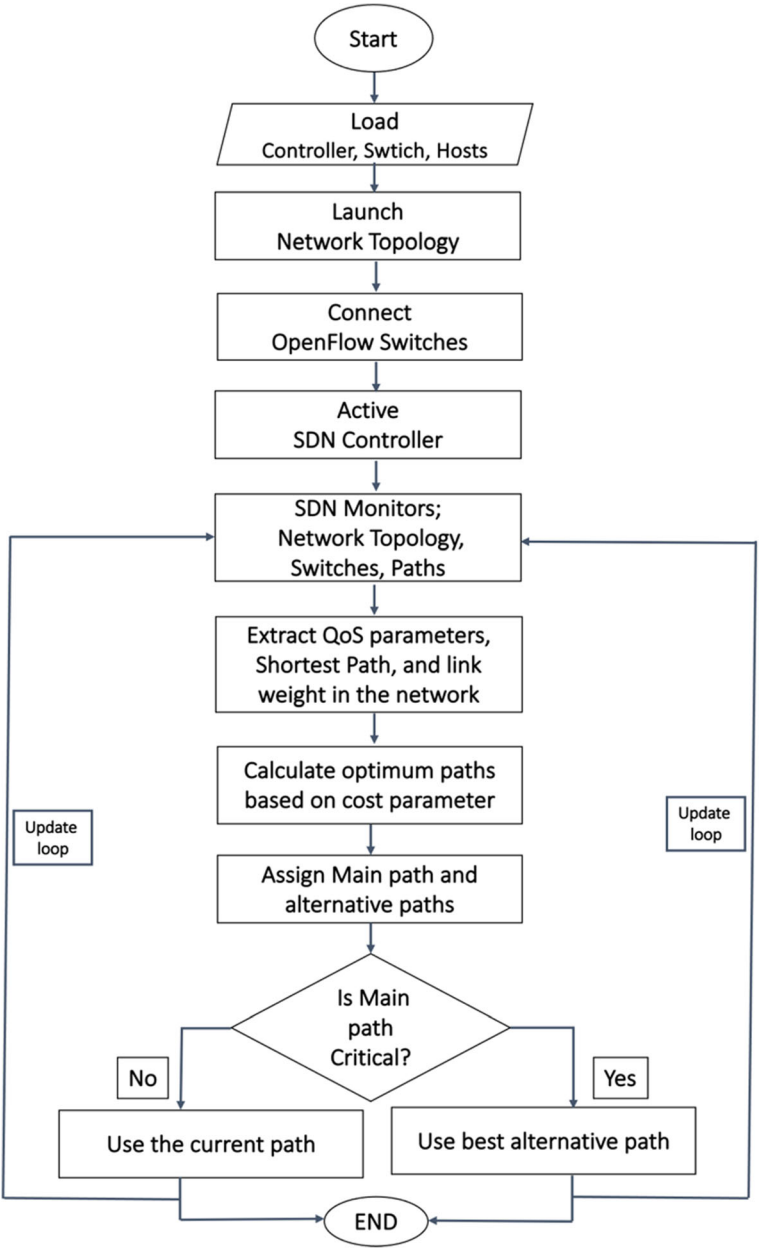

Fig. 2 Flow Diagram of System Model

Algorithm 1 Smart Routing algorithm for SDN controller

```
STEP 1: Input: Host j = Host1, Host2,..., Host n; Switch i = S1, S2..., SN ; Controller = C0; Network
topology;  N= Current Switch (S s SrcSwitch), Destination Switch (D s DesSwitch)
STEP 2: Output: Optimum path (P)
STEP 3: link_cost ← Ø
STEP 4: Optimum path (P) ← Ø
STEP 5: Datapath ← Ø
STEP 6: Path []←Ø
STEP 7: i = 0;
STEP 8:  While C0 do:
STEP 9:    If Src && Des address is not in Mactable:
STEP 10:     add Src && Des to Mactable
STEP 11:    For S in N do:          // Number of Switches between Ss and S_D
STEP 12:      For each path [i, i+1]   // Counting number of paths
STEP 13:       cost += get_link_cost(path[i], path[i+1])  // Calculate link cost
STEP 14:       return cost    // return updated cost
STEP 15:      End for
STEP 16:      path[i] ← get min(cost) // Select the minimum cost of all selected paths
STEP 17:     If Datapath ≠ Ø:            // comparison of available paths
STEP 18:      put path[i] to Datapath // Set the path to be used by end-to-end connection
STEP 19:     End if
STEP 20:    Adjust Weight(P(Datapath)) //use the best path
STEP 21:    End for
STEP 22:   End if
STEP 23: End while
STEP 24: Return P   // The controller assigns the best path
```

3.3 Testbed description

Details of the testbed design that has been used to prepare and handle the experiments are described in this section.

In the process of designing the network testbed, various mechanisms have been used. These mechanisms include; experimental network design, performing routing algorithm, monitoring the network activity, and assuming the delay time of the packets.

The SDN network environment is designed by using a virtualization system based on Ubuntu 20.04.3 LTS [17]. Therefore, the implementation of the SDN scenario is virtualized by installing the Mininet software [19]. Additionally, the virtual machine device is considered to simulate the SDN network scenario. The SDN controller in the network topology is defined as a centralized device, and entire switches are connected through a secured channel to the controller in order to protect the security connection between switches and the controller. The application interface (API) [8] and command-line interface are admitted for the configuration of processes and implementation of the system design. The Ryu Controller is an open-source, software-defined networking (SDN) controller which is considered to improve network agility by simplifying the management and adaptation of traffic. In this way, creating new network management and control applications becomes an easy approach. Ryu controller in this research is implemented to control the process of routing protocols. Moreover, Internet control message protocol (ICMP) protocol is used to monitor the link capacity and sends a sequence of

packets to measure the information of the links such as latency and path length [41]. The real-time transmission process is measured by using the local clock of the real device.

In the system testbed, to provide better and more accurate test experiments, the interval time between each packet is set to 1 millisecond and the default packet size is 56 bytes with 8 bytes of the packet header. Besides, testing multipath routing efficiency between sender and receiver is performed by using a video streaming application. The main purpose of using this video application is to get accurate results as the video application requires a high QoS. Since the vision through human eyes is much more sensitive to detect good and bad quality video services, people participate in decision-making tests. To apply the video streaming experiment, we use FFMPEG, VLC, and FFPlay open-source applications to encode and decode the streamed video [5]. Moreover, we use a Macbook_Device to do the test experiments, the device must have these characteristics; 2,4 GHz Quad-Core Intel Core i7, 8 GB 1600 MHz DDR3, NVIDIA GeForce GT 650 M 1 GB - Intel HD Graphics 4000 1536 MB, and 500GB of the hard disk.

4 Tests and results

In this section, the details of test experiments are given using different features such as; Performance, Controller's delay, and Video streaming Test.

4.1 Performance test

The SDN implementation is provided through Ryu controller installation. A dynamic network topology is made up of several switches and nodes. The network topology also includes several switches and hosts, as well as one centralized controller, designated as C0. In the presented scenario, the limited number of nodes is considered to launch. In this situation, launching a specific architecture is considered to make monitoring and management of the activity scenario easier and less time consumption. Therefore, the simulation can handle a wide range of larger network topologies. The number of switches in the topology entailed fourteen switches which are s_a, s_b, s_c, s_d, s_e, s_f, s_g, s_h, s_i, s_j, s_k, s_l, s_m, s_n respectively. Each switch is associated by a name and all of them are linked together. To establish connectivity, most of the switches employ three or four ports. Figure 3 shows the network topology. Each network has its Internet Protocol (IP) address range and they are unique. The programmed Ryu controller and the network topology are configured to their optimum state. The remote SDN controller is configured under Ryu_C0, and the switches are configured with Open Flow. Thus, as shown in Fig. 3, each switch is labeled with its corresponding name. In the network topology, there are 31 paths shared by 14 switches (This topology is changed according to scenario selection). In addition, a node in the Sa network can connect to a node in the Sn network through a variety of routes. Each path has a different links capacity and QoS such as link speed in Mbps, delay in milliseconds, and packet loss in ratio. Furthermore, depending on the network scenario's design, the link's capacity can be changed as presented in Table 2. When the Controller detects multipath, the parameters of QoS are randomly configured to select an alternative pathway. The proposed algorithm is implemented to achieve multipath routing and reduce the link handling time in an SDN environment. Therefore, we apply different experiments to evaluate the algorithm. Different tests are performed to show the system performance and to find the best path from a number of the available paths of the network. The controller can handle the

link that failed in the shortest time without any loss of packets. Hence, the accuracy and efficiency of the method are provided by using video streaming tests. Further, a comparison is to be performed between the transferred video and the original video to show how the path handing has affected the quality of the video streaming.

4.2 Controller's latency test

In this experiment, 31 paths are generated for the networks as explained in Fig. 3. The first switch as denoted by (Sa) is corresponding to the source node (Sender) which can flow its packets over multi-paths. Therefore, the sender has the probability of using 31 paths and selecting the optimum flow path which is identified by the SDN controller. In this test, the process is started through monitoring the network flows by the controller to select the routes depending on different parameters such as availability of bandwidth, delay of the packets, packet loss rate, and length of the route. Moreover, the process followed by extracting all the obtained results of the flow paths and saving them by the controller. Later, the ICMP protocol is configured with different parameters; packet sizes, numbers, and time intervals. As indicated in Table 2, various packet values were trained to discover the best routes in the suggested scenario. The link weight is determined using a mathematical method. Various range of packet size (small, medium, and large) is selected to provide the test experiment including 32, 64, and 128 packets as a datagram. Also, an extra 8-bit header for each packet is attached. The intervals of time of sending the packets are 1, 2, and 0.5 milliseconds respectively, and the sequence of the number of the packets are chosen as 10, 20, and 40. According to the test experiment, using less than and more than the above mention set ranges can lead to inefficient and inaccurate results. This may result in overhead and additional calculating effort for the controller. Before launching the Mininet simulation and running the network scenario, 33 network links (including two paths for the nodes) and 14 switches in the network topology are randomly configured with their values as shown in Table 2. Each link has its QoS value which is different from the other links. A node from the source can use the optimum link to send the packets to the destination device. Furthermore, the SDN controller is programmed with the aforementioned parameters.

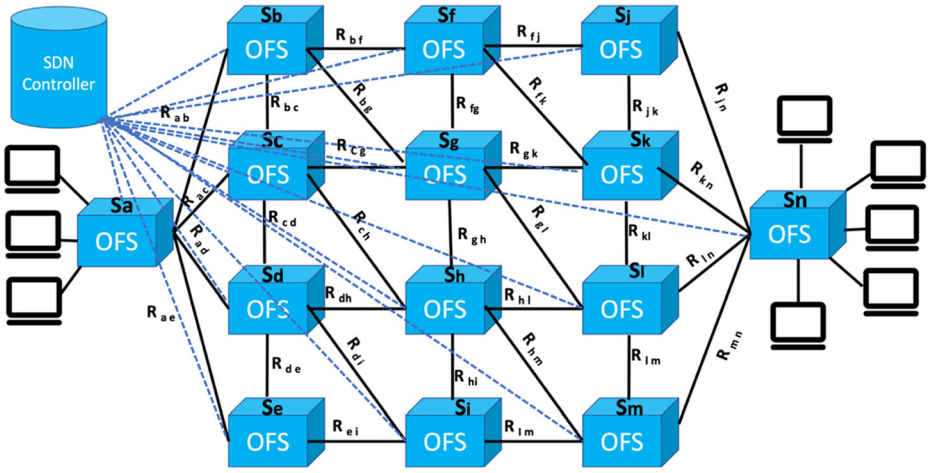

Fig. 3 Simulated Network Topology

Table 2 QoS parameters for the network topology

Route Name	Bandwidth (Mbps)	Delay (ms)	Packet loss (%)
OFS (S_a)			
ROUTE $_{ab}$	10	1	0.1
ROUTE $_{ac}$	20	10	0.2
ROUTE $_{ad}$	30	100	0.3
ROUTE $_{ae}$	40	100	0.4
OFS (Sb)			
ROUTE $_{ab}$	50	1	0.5
ROUTE $_{bc}$	60	10	0.6
ROUTE $_{bg}$	70	100	0.7
ROUTE $_{bf}$	80	100	0.8
OFS (Sn)			
ROUTE $_{jn}$	90	1	0.9
ROUTE $_{kn}$	100	10	1
ROUTE $_{lm}$	10	100	0.9
ROUTE $_{mn}$	20	1000	0.8
OFS (Sm)			
ROUTE $_{jn}$	30	1	0.7
ROUTE $_{kn}$	40	10	0.6
ROUTE $_{lm}$	50	100	0.4
ROUTE $_{mn}$	60	1000	0.5

The dataset has been generated from the case studies presented in Tables 3, 4, and 5. For selecting the optimum and accurate value (as shown in the tables in bold) from the test experiments of the case studies, a method is used to specify which value is appropriate for feeding the algorithm. Results observation revealed that finding the optimum packet size is depended on the short and/or large packet size. The established route takes a short time when the packet sequence/size is small, while the enlarged range takes a longer time. To prove which range of packet size is better for feeding the algorithm, another experiment was performed. In this test, the QoS parameters of the current path are down. This is leading the controller to select the appropriate alternative paths in the network. It is shown in Fig. 4 that the current link is down where the controller is using the most suitable alternative path. The experiment of link drain was repeated 10 times for each test to obtain more accurate results.

Test results showed that the small packet number provided oscillation in selecting the paths with less delay in time whereas the larger packet number produced a higher delay in time. Experiment results of 20 packets give 0.85, 1.14, and 1.23 milliseconds of delay in time respectively which can consider as the better result for feeding the proposed algorithm in selecting the paths.

4.3 Video streaming test

To provide the video streaming tests, the network testbed is prepared to broadcast the streamed videos as shown in Fig. 5. This test aims to prove how the proposed algorithm influences the quality of video streaming services. In this test, four videos are streamed in the given network scenario and each video has 60 seconds of duration, the purpose behind selecting a long video duration is to get an accurate result. The videos datasets are obtained from different sources; BigBuckBunny has taken from (https://peach.blender.org/), (http://ftp.itec.aau.at/datasets/DASHDataset2014/Valkaama/) for Valkaama and Red Bull PlayStreets, and Sintel [36],

with X.264 tool library [13, 33], are encoded to an acceptable degree of quality resolution while FFmpeg is used to stream the dynamic video encoding. FFplay and VLC have been utilized to playback the streamed videos on the client side with a bitrate parameter of 1000 to 1300 Mbps. Each video has more or less than 1100 Mbps of bitrate and also it depends on the size of Intra-Frames [26, 39]. Furthermore, transportation protocol based on UDP is used to transmit the videos to handle the network congestion due to the high packet request. The time consumption of path selection for video streaming is investigated in this research. The video is sent from the server side taking the best route by the SDN controller. Normally, the server initiates the video transmission process, and the video data is received on the client side using VLC or FFPlay applications. A simple method is established to achieve failure among the connectedness of the paths between sender and receiver. When the current route has poor transmission or the QoS values are not sufficient, the SDN controller picks another path to complete the video streaming process. The video bitrate and transferred number of bits are monitored via the SDN controller.

In the first test of this scenario, the current link which is used for transmitting the videos is drained at a different time of the video playback. Choosing three ranges of different times to drain the routing is related to the I-frame size. The drain time of routing will occur at the beginning, middle, and end of the playback of the video. This is aimed to detect the impact of the link failure on the video transmission.

Consequently, as shown in Figs. 6 and 7, the current route is down at the 6th second of video time, and the SDN controller proactively takes the optimal alternative path to integrate the transmission. The decision time of the path selection took less than a second when using the presented algorithm, which means that the video only stalled for less than a second on the client side. Also, this value can be changed according to the network conditions, however, in the traditional approach, for the drained route at the beginning of the playout of the video, the process takes 3 seconds, which means the video is frozen for 3 seconds and the client waits for a long time until the buffer's client application received the entire packets. This test presented that the traditional programmable Ryu-SDN controller indicated a worse case of QoE to the end user. Thus, the proposed mechanism provided better results of QoE, less time consumption for recovering path, and reduced congestion. These tests are done for both high and low-motion video streaming to measure the number of stalls and stall duration as shown in Figs. 6 and 7.

In another test, the packet loss rate is measured. As previously mentioned, the videos are streamed over the given topological network. The techniques that have been surveyed in the state of the art are used to program the SDN controller. Each network link's packet loss

Table 3 A small range of packets number (Packet Number = 10)

Test No.	Packet size (Byte)	Time Interval (sec)	Delay time (ms)
1	32	0.5	**0.43**
2	32	1	0.45
3	32	2	0.61
4	64	0.5	**0.57**
5	64	1	0.64
6	64	2	0.55
7	128	0.5	**0.61**
8	128	1	0.64
9	128	2	1.01

Table 4 Medium range of packet numbers (Packet Number = 20)

Test No.	Packet size (Byte)	Time Interval (sec)	Delay time (ms)
1	32	0.5	**0.85**
2	32	1	0.87
3	32	2	1.23
4	64	0.5	**1.14**
5	64	1	1.28
6	64	2	1.29
7	128	0.5	**1.23**
8	128	1	1.60
9	128	2	2

parameter is configured, the packet loss range includes 0.1, 0.5, and 1. These values are randomly given to the network links. The server streams the videos over optimal paths. When the QoS of the network path is degraded or the connectivity is interrupted, the controller finds the best alternative path for the end-user. Figures 8, 9, 10 and 12 show the test results of packet loss rates for high and low video motions when the end-user played back the videos. When the controller is programmed with the proposed algorithm, the receiver loses fewer packets compared to the state-of-the-art results. The results presented in Fig. 8 depict that 24 packets are lost when the traditional routing algorithm is configured in the Ryu SDN controller, however, 5 packets were lost when selecting the optimal route based on the proposed algorithm for streaming slow-motion BigBuckBunny.

Although, The result varies for the Valkaama sequence. In this test, the number of packet losses is 9 while in the conventional method 36 packets were lost. This shows that less packets are lost when the Ryu Controller is programmed with the proposed approach as compared to the packet loss of the conventional algorithm, this is presented in Fig. 9.

The last test of packet loss has been done for high-motion video streaming (Red Bull PlayStreets) and (Sintel). The results presented that a high number of packet loss is observed when the videos are streamed based on choosing the optimal path. The proposed method has taken the best path with less packet loss as compared to the conventional method as illustrated in Figs. 10 and 11.

However, the number of packet loss for high-motion video streaming is higher than that of slow-motion video streaming, since in high-motion videos there are higher artifacts, the frames are moved faster and it is sensitive to packet loss. The presented results prove that the

Table 5 The higher range of packet numbers (Packet Number = 40)

Test No.	Packet size (Byte)	Time Interval (sec)	Delay time (ms)
1	32	0.5	**2.40**
2	32	1	2.59
3	32	2	3.65
4	64	0.5	**3,44**
5	64	1	3,82
6	64	2	3,30
7	128	0.5	**3,67**
8	128	1	3.72
9	128	2	5.95

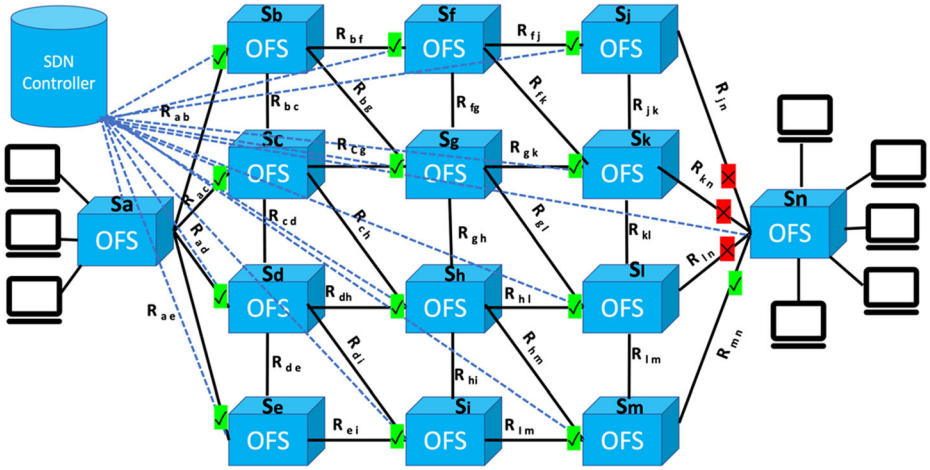

Fig. 4 Optimal path selection in SDN controller

suggested algorithm can effectively reduce the packet loss rate which improves the utilization efficiency of the network resource and provides a better QoE to the end user.

5 Discussion and benchmark comparison

In this section, the effect of delay on selecting the multipath routes for the SDN controller is studied. According to the obtained results that have been presented in Tables 3, 4, and 5, various values are obtained for calculating the consumption time of selected paths which are 0.43 ms for 10 packets, 0.85 ms for 20 packets, and 2.40 ms for 40 packets. The optimal result was 0.85 ms and it can be integrated into the proposed algorithm to calculate the weight of the paths. Moreover, the small range of packets has given inaccurate results even though the duration of delay was short but the controller was not able to get enough knowledge to find the

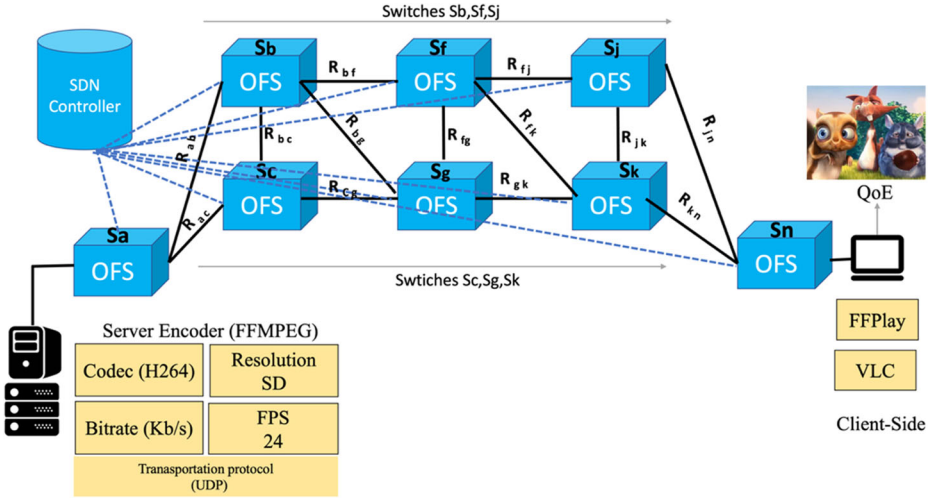

Fig. 5 Network scenario for video streaming tests

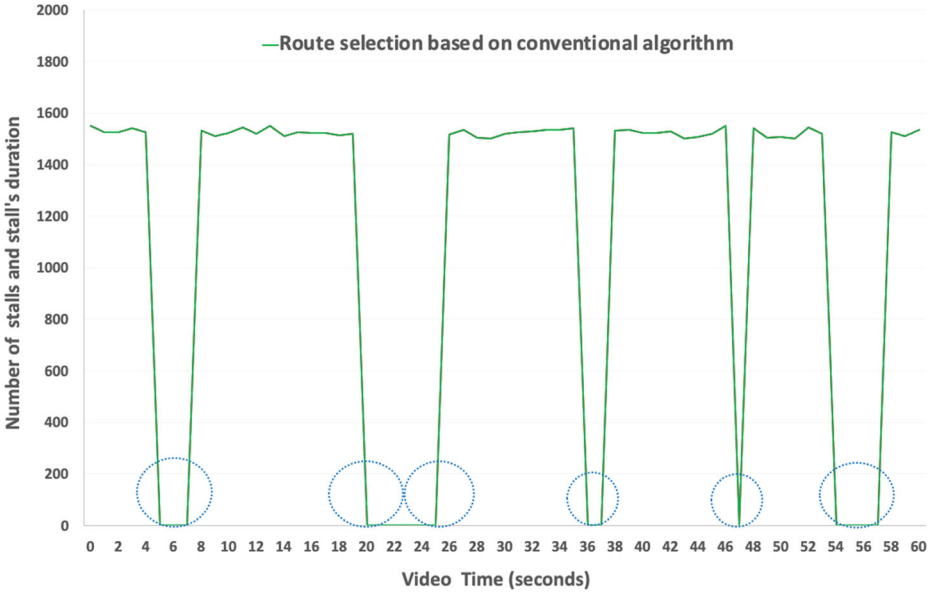

Fig. 6 Impact of conventional routing algorithm on video streaming

best route in a short period due to many path selections that faced the SDN controller. Details of different packet sizes and sequences are presented in Table 6. The large sequence is not useful for deploying the decision of path selection because of the complexity of delay among the nodes.

A comparison is presented between various state-of-the-art mechanisms and the suggested algorithm for sending a large number of network packets. As depicted in Fig. 12, the

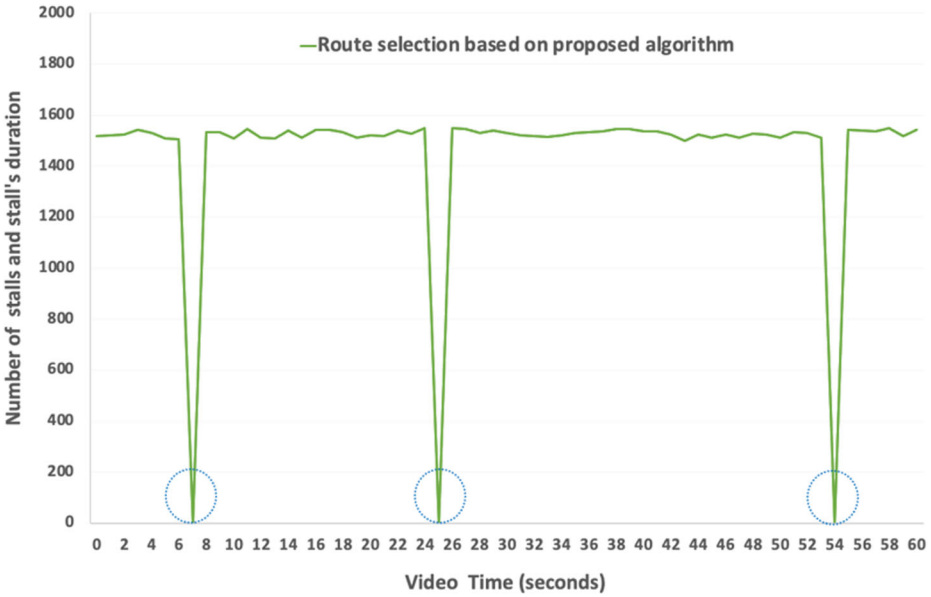

Fig. 7 Impact of proposed routing algorithm on video streaming

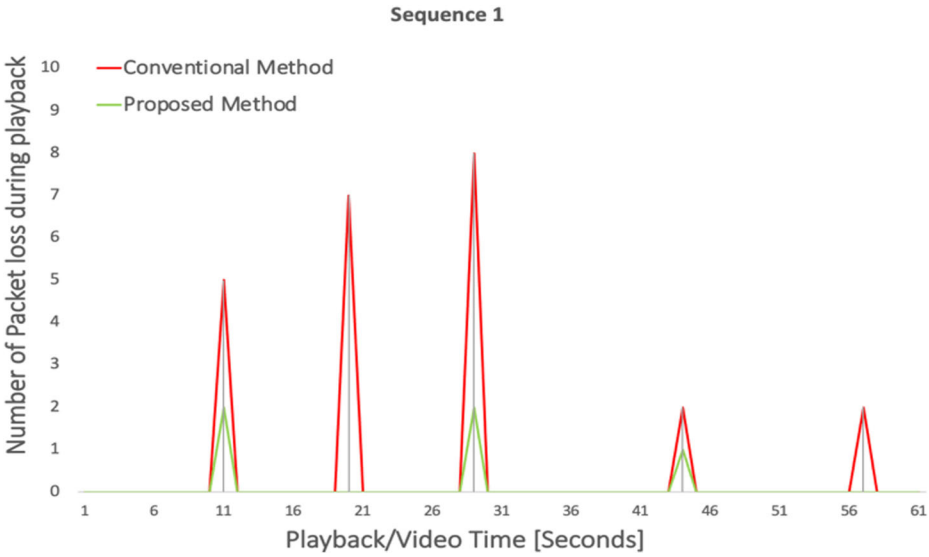

Fig. 8 Packet loss in slow-motion sequence (BigBuckBunny)

comparison showed how many packets will be discarded when a node flows 50 packets per second to the destination. The result explains that packet loss is minimized when the controller is programmed with the proposed algorithm as compared to other mechanisms.

Furthermore, another test experiment for the proposed routing algorithm is explained. In this test, QoE based on the objective metric – Peak Signal to Noise Ratio (PSNR) is calculated for video streaming applications [14]. In this test, low and high-motion videos are streamed using the designed technique. The SDN controller has captured the information of the available bandwidth and length of the paths from the source to the destination. The shortest

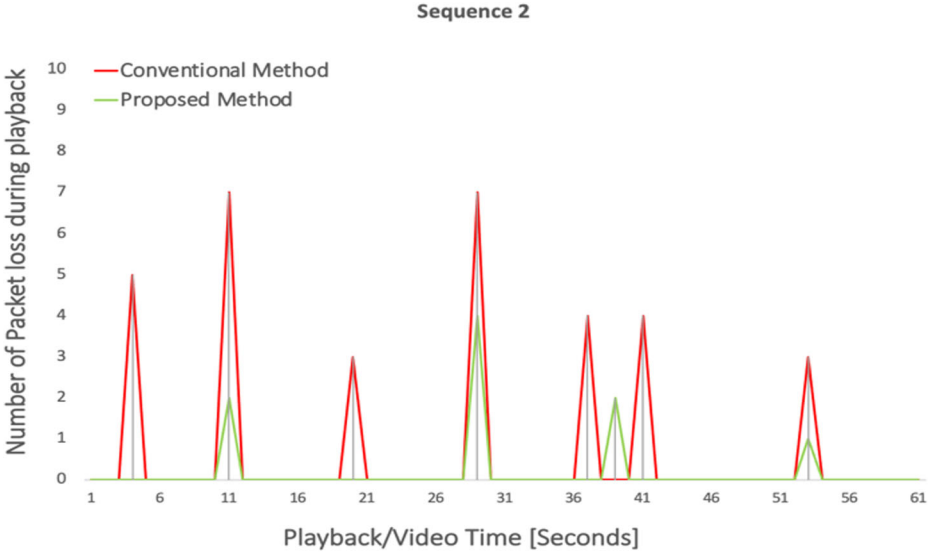

Fig. 9 Packet loss in slow-motion sequence (Valkaama)

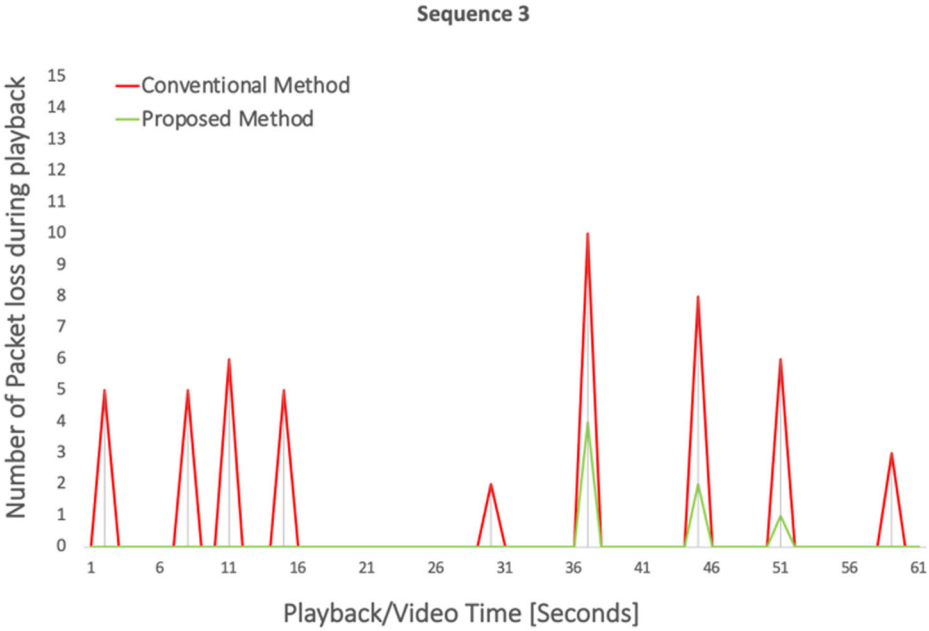

Fig. 10 Packet loss in high-motion sequence (Red Bull PlayStreets)

path is considered to have a minimum delay time. Test results of Tables 7 and 8 indicate that the proposed scheme outperforms the other research schemes for presenting better QoE.

The effectiveness of using minimum network resources is to select the optimal path whenever it is available. If it's not available, the model searches for an alternative path based

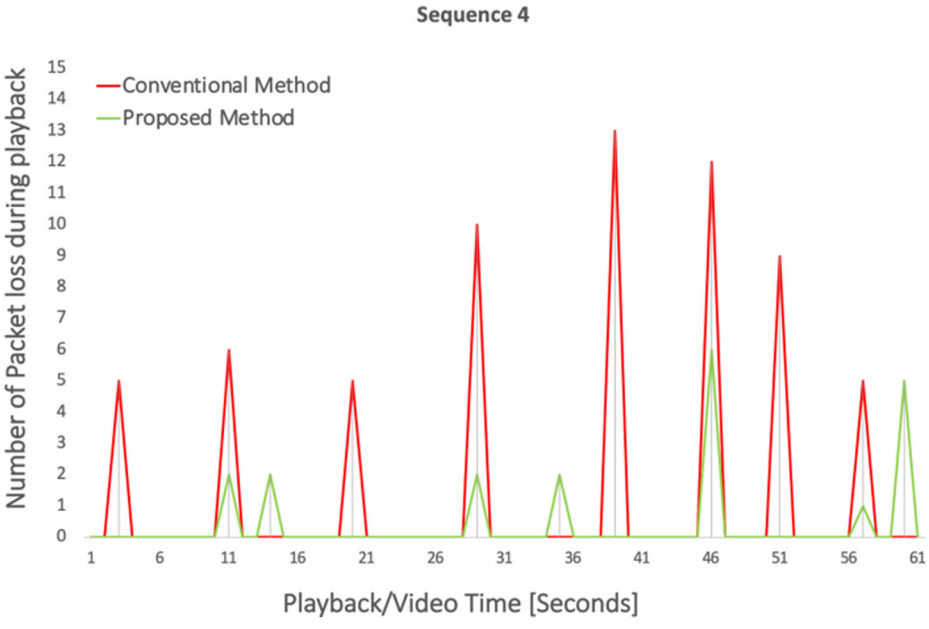

Fig. 11 Packet loss in high-motion sequence (Sintel)

on network QoS parameters. Finally, the benchmark comparison is presented in Table 9 which includes; the type of the SDN controller, simplicity and complexity of the network topology, consumption time to select an optimum path, congestion and resource usage, and applicability of the tests for video streaming. Results outperformed other techniques due to the following factors: 1- selecting the ideal packet size, 2- packet number, 3- time interval, and 4- The right decision for multiple path selection.

The system is designed for streaming on-demand and progressive videos, it cannot support segmentation of adaptive bitrate due to the incapability of the platform. A new platform needs to be updated. However, the software-defined network topology can launch only a few hundred OpenFlow switches. Significant resource utilization is necessary to virtualize complicated routes.

6 Conclusions

In this paper, a smart and efficient algorithm has been developed for the SDN controller, which calculates the best path between the sender and the receiver. It is based on using optimal network

Table 6 Pros and Cons of packet size for SDN routing algorithm

Sequence packet numbers	Delay time	Routing selection	Congestion	Advantage /Disadvantage
10	0.43	High oscillation in route selection	Depend	Fast/ Inaccurate
20	0.85	Optimum in route selection	Low	Optimal
40	2.40	High delay path selection and network congestion	High	Valid/ Complex

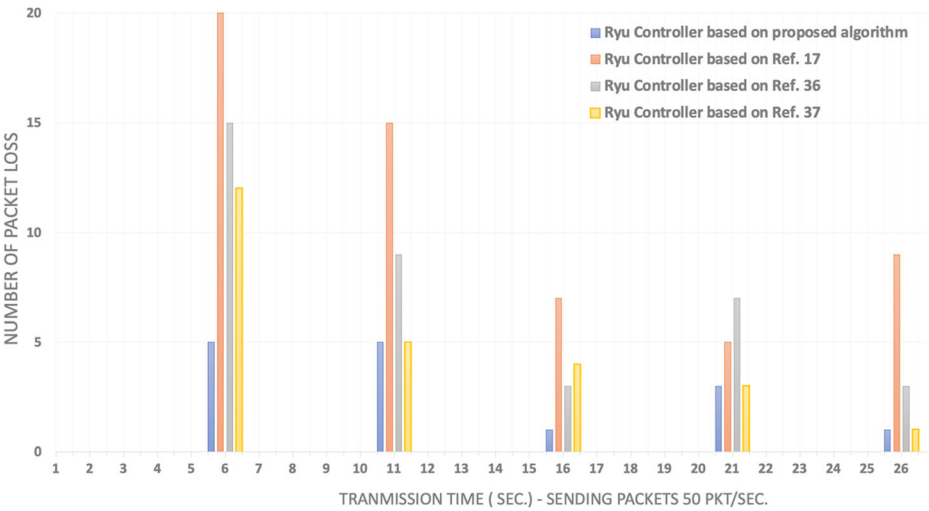

Fig. 12 Comparison between mechanisms of the state-of-the-art and proposed algorithm

Table 7 QoE Comparison Low Motion Video

Mechanisms	Route selection	PSNR (dB)
Ref. [1]	Optimum path with high packet loss rate	30
Proposed Scheme	Optimum path with less Packet loss rate	36

Table 8 QoE Comparison High Motion Video

Mechanisms	Route selection	PSNR (dB)
Ref. [10]	Optimum path with high packet loss rate	27
Proposed Scheme	Optimum path with less Packet loss rate	34

packet size and OpenFlow information to complete the process of routing. A range of packet sizes, packet numbers, and interval times are considered to determine the time that is taken for selecting the ideal routes. Important metrics such as adaptive packet size, observing the network state, shortest path, and link capacity have been used to design the algorithm. Furthermore, the controller's algorithm can decide to switch the flow traffic from the main path to the optimal recovery path wherever the main link is down or QoS parameters are critical. As a result, the test experiments showed that the link handling time has approximately consumed less than one millisecond. On the other hand, the proposed mechanism has provided a better QoE for end-users of video streaming applications. To conclude, the model reduced the delay time of route selection to 96.3% as compared to the conventional algorithm which is estimated by 68.7%. This leads to producing an efficient result for selecting the routes and delay tolerance. This can efficiently solve the problem of decision time for selecting a recovery path when the main path is failed or is critical. This method could be further improved by using deep learning techniques to deploy the most accurate decision for multipath routing and using sophisticated network topology (Details of the abbreviation are described in Appendix Section.

Table 9 Benchmark comparison

Approaches	Type of controller	Resource usage	Network Congestion	Decision Time (ms)	Tests based on multimedia streaming
Ref. [7] by N. Dorsch et al.	Floodlight	High	Very High	4.5	NO
Ref. [21] by Y. D. Lin et al.	RYU	High	Very High	<40	NO
Ref. [1] by Y. Aldwyan and R. O. Sinnott	Not Specified	Minimum	High	1.155	NO
Ref. [10] by H. H. Hsieh and K. Wang	Open Daylight	Minimum	High	1.087	NO
Ref. [6] by F. Chahlaoui et al.	RYU	High	High	1.033	NO
Ref. [34] by W. Tan W et al.	RYU	Minimum	High	1.01	NO
Proposed Algorithm	RYU	Minimum	Less	**0.85**	YES

Appendix

Acronym	Description
AP	Access Point
APIs	Application Programming Interface
ARP	Address Resolution Protocol
AVC	Advanced Video Coding
BR	Bit-rate
BVIHFR	Bristol Vision Institute High Frame Rate
CORBA	Common Object Request Broker Architecture
CP	Capacity path
DAL	Device and resource Abstraction Layer
DFS	Depth-first search
DMOS	Differential Mean Opinion Score
FDSP	Flexible Dual TCP-UDP Streaming Protocol
FFMPEG	Fast Forward MPEG
FFPLAY	Fast Forward Play
FHD	Full High Definition
HD	High Definition
HFRs	High Frame Rates
HTB	Hierarchy Token Bucket
HTTP	Hyper Text Transfer Protocol
HVS	Human Vision system
ICMP	Internet control message protocol
IEEE	Institute of Electrical and Electronics Engineers
ILP	Integer Linear Program
IP	Internet Protocol
IPTV	Internet Protocol Television Service
ISP	Internet Service Providers
LAN	local area network
LLDP	Link Layer Discovery Protocol
LTE	Long Term Evolution
MAN	Metropolitan Area Network
MB	Megabit
MOS	Mean Opinion Score
ms	Milliseconds
MSE	Mean Square Error
NETCONF	Network Configuration
NETEM	Network Emulator
NoRF	Number of Reference Frames
NSAL	Network Services Abstraction Layer
OSP	OpenStack Platform
OSPF	Open Shortest Path First
PC	Personal Computer
PSNR	Peak Signal to Noise Ratio
QoE	Quality of Experience
QoS	Quality of Service
RESTful	Representational State Transfer
RMSG	Recurrent Neural Network Accompanied by Modified Sea gull Optimization
RSL	Random Selected Leader
SDN	Software Defined Network
SSIM	Structure Similarity
STP	Spanning Tree Protocol
TCP	Transmission Control Protocol
TV	Television
UDP	User Datagram Protocol
UHD	Ultra-High Definition

Acronym	Description
VBR	Variable Bitrate Error
VLC	VideoLAN Client
VMAF	Video Multi-Method Assessment Fusion
VOD	Video on Demand
VQM	Video Quality Monitoring
Wi-Fi	Wireless Fidelity
WPAN	Wireless Personal Area Network
WSN	Wireless Sensor Networks

Acknowledgments This research is a part of the research work of the University of Sulaimani in Kurdistan Region of Iraq. Special thanks to the College of Science at University of Sulaimani for providing a healthy environment to fulfill this project. We would also like to express our deep gratitude for the generous support and funds from the presidency of university of Sulaimani.

Data availability The datasets generated during and/or analyzed during the current study are available from the corresponding author upon reasonable request.

Declarations

Conflict of interest The author certifies that there is no actual or potential conflict of interest concerning this article.

References

1. Aldwyan Y, Sinnott RO (2019) Latency-aware failover strategies for containerized web applications in distributed clouds. Futur Gener Comput Syst 101:1081–1095. https://doi.org/10.1016/j.future.2019.07.032
2. Alghamdi SA (2022) Cuckoo energy-efficient load-balancing on-demand multipath routing protocol. Arab J Sci Eng 47:1321–1335. https://doi.org/10.1007/s13369-021-05841-y
3. Ali J, Lee S, Roh BH (2018) Performance analysis of POX and Ryu with different SDN topologies. ACM Int. Conf. Proceeding Ser., pp 244–249. https://doi.org/10.1145/3209914.3209931
4. Alsaeedi M, Mohamad MM, Al-Roubaiey AA (2019) Toward adaptive and scalable OpenFlow-SDN flow control: a survey. IEEE Access 7:107346–107379. https://doi.org/10.1109/ACCESS.2019.2932422
5. Big Buck Bunny. Available from: https://www.peach.blender.org. Last accessed 16 Apr 2021
6. Chahlaoui F, Dahmouni H, El Alami H (2022) Multipath-routing based load-balancing in SDN networks. In: 2022 5th Conference on Cloud and Internet of Things (CIoT). IEEE, pp 180–185
7. Dorsch N, Kurtz F, Girke F, Wietfeld C (2016) Enhanced fast failover for software-defined smart grid communication networks. IEEE Glob. Commun. Conf. GLOBECOM 2016 - Proc., pp 1–6. https://doi.org/10.1109/GLOCOM.2016.7841813
8. Fernandez C, Muñoz JL (2016) Software Defined Networking (SDN) with OpenFlow 1.3, Open vSwitch and Ryu. UPC Telematics Department, pp 183
9. Hemalatha R, Umamaheswari R, Jothi S (2022) An efficient stable node selection based on Garson's pruned recurrent neural network and MSO model for multipath routing in MANET. Concurr Comput: Pract Exp 34(21):e7105
10. Hsieh HH, Wang K (2019) A simulated annealing-based efficient failover mechanism for hierarchical SDN controllers. IEEE Reg. 10 Annu. Int. Conf. Proceedings/TENCON, vol. 2019-Octob, pp 1483–1488. https://doi.org/10.1109/TENCON.2019.8929249
11. Hwang R-H, Tang Y-C (2016) Fast failover mechanism for sdn-enabled data centers. International Computer Symposium (ICS), Chiayi, Taiwan. IEEE, pp 171–176. https://doi.org/10.1109/ICS.2016.0042
12. Jiawei W, Xiuquan Q, Guoshun N (2018) Dynamic and adaptive multi-path routing algorithm based on software-defined network. Int J Distrib Sens Netw 14(10). https://doi.org/10.1177/1550147718805689
13. Jin H et al (2019) TALON: tenant throughput allocation through traffic load-balancing in virtualized software-defined networks. 2019 International Conference on Information Networking (ICOIN). IEEE

14. Jin H et al (2019) FAVE: bandwidth-aware failover in virtualized SDN for clouds. 2019 IEEE 12th International Conference on Cloud Computing (CLOUD). IEEE
15. Kamboj P, Pal S, Bera S, Misra S (2022) QoS-aware multipath routing in software-defined networks. IEEE Trans Netw Sci Eng
16. Kannan A, Vijayan S, Narayanan M, Reddiar M (2018) Adaptive routing mechanism in SDN to limit congestion. In: Information systems design and intelligent applications. Springer, Singapore, pp 245–253. https://doi.org/10.1007/978-981-13-3329-3_23
17. Keti F, Askar S (2015) Emulation of software defined networks using mininet in different simulation environments. Proc. - Int. Conf. Intell. Syst. Model. Simulation, ISMS, vol 2015-Octob, pp 205–210. https://doi.org/10.1109/ISMS.2015.46
18. Kumar MJ, Ramachandran B (2020) Multipath routing strategy for reducing congestion in WSNS. In: Intelligent computing in engineering. Springer, Singapore, pp 561–567. https://doi.org/10.1007/978-981-15-2780-7_61
19. Lee S, Ali J, Roh BH (2019) Performance comparison of software defined networking simulators for tactical network: mininet vs. OPNET. 2019 Int. Conf. Comput. Netw. Commun. ICNC 2019, pp 197–202. https://doi.org/10.1007/s10586-019-02996-0
20. Lihua L (2020) Multi-path allocation scheduling optimization algorithm for network data traffic based on SDN architecture. IMA J Math Control Inf 37(4):1237–1247. https://doi.org/10.1093/imamci/dnaa011
21. Lin YD, Teng HY, Hsu CR, Liao CC, Lai YC (2016) Fast failover and switchover for link failures and congestion in software defined networks. IEEE Int. Conf. Commun. ICC 2016. https://doi.org/10.1109/ICC.2016.7510886
22. Liu Y, Pan Y, Yang M, Wang W, Fang C, Jiang R (2015) The multi-path routing problem in the software defined network. 11th International Conference on Natural Computation (ICNC). IEEE, 254, p 250. https://doi.org/10.1109/ICNC.2015.7377999
23. Nisar K, Welch I, Hassan R, Sodhro AH, Pirbhulal S (2020) A survey on the architecture, application, and security of software defined networking. Internet Things 12:100289. https://doi.org/10.1016/j.iot.2020.100289
24. Ramdhani MF, Hertiana SN, Dirgantara B (2016) Multipath routing with load balancing and admission control in Software-Defined Networking (SDN). 4th International Conference on Information and Communication Technology (ICoICT). IEEE, pp 1–6. https://doi.org/10.1109/ICoICT.2016.7571949
25. Rego A, Sendra S, Jimenez JM, Lloret J (2019) Dynamic metric OSPF-based routing protocol for software defined networks. Clust Comput 22(3):705–720. https://doi.org/10.1007/s10586-018-2875-7
26. Rezende P et al (2019) An SDN-based framework for routing multi-streams transport traffic over multipath ne works. ICC 2019–2019 IEEE International Conference on Communications (ICC). IEEE
27. Rhamdani F, Suwastika NA, Nugroho MA (2018) Equal-cost multipath routing in data center network based on software defined network. 6th Int. Conf. Inf. Commun. Technol. ICoICT, vol 0, no c, pp 222–226. https://doi.org/10.1109/ICoICT.2018.8528730
28. Sendra S, Rego A, Lloret J, Jimenez JM, Romero O (2017) Including artificial intelligence in a routing protocol using software defined networks. IEEE International Conference on Communications Workshops (ICC 2017), Paris, France. https://doi.org/10.1109/ICCW.2017.7962735
29. Sgambelluri A, Giorgetti F, Cugini FP, Castoldi P (2013) OpenFlow-based segment protection in Ethernet networks. J Opt Commun Netw 5(9):1066–1075. https://doi.org/10.1364/JOCN.5.001066
30. Sharma S, Staessens D, Colle D, Pickavet M, Demeester P (2011) Enabling fast failure recovery in OpenFlow networks, pp 164–171. https://doi.org/10.1109/DRCN.2011.6076899
31. Shi Y, Cao Y, Liu J, Kato N (2019) A cross-domain SDN architecture for multi-layered space-terrestrial integrated networks. IEEE Netw 33(1):29–35. https://doi.org/10.1109/MNET.2018.1800191
32. Singh SK, Das T, Jukan A (2015) A survey on internet multipath routing and provisioning. IEEE Commun Surv Tutor 17(4):2157–2175. https://doi.org/10.1109/COMST.2015.2460222
33. Taha M, Garcia L, Jimenez JM, Lloret J (2017) SDN-based throughput allocation in wireless networks for heterogeneous adaptive video streaming applications. 13th International Wireless Communications and Mobile Computing Conference (IWCMC). IEEE, pp 963–968. https://doi.org/10.1109/IWCMC.2017.7986416
34. Tan W, Xiong Z (2022) Energy efficient multipath routing approach in WMSN. International conference on electronic information engineering, big data, and computer technology (EIBDCT 2022), vol 12256. SPIE
35. Venkatasubramanian S (2022) Improvement of QoS and selection of cluster head using RSL algorithm with multipath routing protocol in MANET. In: 2022 3rd International Conference on Electronics and Sustainable Communication Systems (ICESC). IEEE, pp 569–576
36. VideoLAN, a project and a non-profit organization. Available from: https://www.videolan.org/developers/x264.html. Last accessed 16 Apr 2021

37. Wang R, Mangiante S, Davy A, Shi L, Jennings B (2017) QoS-aware multipathing in datacenters using effective bandwidth estimation and SDN. In: 2016 12th International Conference on Network and Service Management (CNSM). IEEE, vol 20, pp 342–347. https://doi.org/10.1109/CNSM.2016.7818444
38. Xiaolong X, Yun C, Liuyun H, Anup K (2019) MTSS: multi-path traffic scheduling mechanism based on SDN. J Syst Eng Electron 30(5):974–984. https://doi.org/10.21629/JSEE.2019.05.14
39. Yan J et al (2015) HiQoS: an SDN-based multipath QoS solution. China Commun 12(5):123–133
40. Yang Z, Yeung KL (2020) Sdn candidate selection in hybrid ip/sdn networks for single link failure protection. IEEE/ACM Trans Networking 28(1):312–321. https://doi.org/10.1109/TNET.2019.2959588
41. Zhang ZH, Chu W, Huang SY (2019) The ping-pong tunable delay line in a super-resilient delay-locked loop. Proc. - Des. Autom. Conf., pp 90–91. https://doi.org/10.1145/3316781.3322479