


ITC 1/51 Information Technology and Control Vol. 51/ No. 1/ 2022 pp. 5-17 DOI 10.5755/j01.itc.51.1.29408	Reliable Multipath Flow for Link Failure Recovery in 5G Networks Using SDN Paradigm	
	Received 2021/07/05	Accepted after revision 2021/08/28
	 http://dx.doi.org/10.5755/j01.itc.51.1.29408	

HOW TO CITE: Yasin, Q., Iqbal, Z., Khan, M. A., Kadry, S., Nam, Y. (2022). Reliable Multipath Flow for Link Failure Recovery in 5G Networks Using SDN Paradigm. *Information Technology and Control*, 51(1), 5-17. <https://doi.org/10.5755/j01.itc.51.1.29408>

Reliable Multipath Flow for Link Failure Recovery in 5G Networks Using SDN Paradigm

Qadeer Yasin

Department of Computer Science, University of Engineering and Technology Taxila; e-mail: qadeer26@gmail.com

Zeshan Iqbal

Faculty of Computer Science Department, University of Engineering and Technology, Taxila;
e-mail: Zeshan.iqbal@gmail.com

Muhammad Attique Khan

Department of Computer Science, HITEC University, Taxila, Pakistan; e-mail: attique@ciitwah.edu.pk

Seifedine Kadry

Department of Applied Data Science, Noroff University College, Norway; e-mail: skadry@gmail.com

Yunyoung Nam

Department of Computer Science and Engineering, Soonchunhyang University, Asan, South Korea;
e-mail: ynam@sch.ac.kr

Corresponding author: ynam@sch.ac.kr

In modern networks and cloud evolution, as new nodes to internet network growing rapidly and use of gaming and video streaming over the network require high availability with very small latency rate. 5G networks provides much faster services than 4G but link failure occurrence can affect the quality of service. In 5G networks environmental factors also affect the efficiency of wireless signals. To overcome such type of issues, base stations are placed in distributed manner around urban areas when decision is required for the placement. However, in some scenarios we can have few similarities, as in general, highways are same. In existed systems signals distribution is performed homogenously, so it will be generating issues like fractal and environmental. It will be cause of great economic loss. However, being an emerging network paradigm, Software defined Network (SDN) is easy to manage due to logical separation of control plane and data plane. SDN supports numerous advantages, one of them is link robustness to avoid service unavailability. To cope with network link failures there are many mechanisms existed like proactive and reactive mechanisms, but these mechanisms calculate multiple paths and stored in flow tables without considering reliability of link. Therefore, it will be cause of high latency rate due to calculation of many multiple paths and increased traffic overhead too. To overcome these issues, we proposed a new approach in which multipath numbers depend on reliability of primary path. Number of alternative paths decreased as reliability of link increased it leads to less time required to calculate alternative path. In addition, traffic overhead decreases as compared to existing approaches. Secondly, we also integrate the shortest distance factor with reliability factor, and we get better results than existing approaches. Our proposed system will be helpful in increasing the availability of services in 5G network due to low latency rate and small traffic overhead required in link failure recovery.

KEYWORDS: Software Defined Networking (SDN), OpenFlow Enabled Switches, OpenFlow, POX.

1. Introduction

In traditional networks, control and management functions along with data were performed on network devices e.g. switches, gateways, and routers. That is why many problems were faced such as misconfiguration, network fluctuation, etc. Mostly problems in traditional networks were due to manual configuration of devices, low level programming commands and no less involvement of operating system in it [22]. To overcome these issues software defined network (SDN) introduced as a new architecture of network in which new manner used with separate control and management plane at logically centralized point known as SDN controller [21]. Data plane of forwarding devices separation improve the working of SDN controller due to easy control and management of network. Due to centralized controller, a network can efficiently configure with load balancing, traffic engineering, security enforcement, etc. Central controller in SDN make easy to maintain tasks like reachability map, enforcing Access Control List (ACL) at a single logical point [40]. SDN divides in three moduli. First, *application plane* in which Northbound API use to forwarding updates to *control plane*. Secondly control plan controlled the *data plane* by decision making by itself use southbound API. (Commonly used open flow protocol). Each device in

a network manual register itself with SDN controller and regularly update the central controller by providing latest link state information. Thus, it is cleared that SDN controller has complete view of overall network and this quality makes SDN more efficient as compare to traditional network [23]. SDN introduced as easy management of network by decoupling the planes.

Application plane is also known as management plane because end user is controlling it by using SDN programming languages like Frenetic [7]. This plane interrelate with control plane like security application is answerable to counter attacks and load balancer is responsible for distribution of traffic among given links. Application plane can be customized by the network administrator for modifying the performance of network. Thus, this unit makes network easy to program and modify by the developers [11]. In the control plane perform decision logic at data packets which is depend on application plane and topology which is used. Control plane sends the specific action to the centralize controller and controller apply these decisions on data packets. These flow rules installed at switch located in communication path. Network administrator can obtain complete view of network topology at centralize controller [26].

The third module in SDN is data plane consist of network nodes and devices all these devices are controlled by controller using protocol(like open flow) until packet arrived at destination point. Data plane provide secure communication with controller and memorize all pattern and corresponding actions in form of table. When data packet arrived, controller perform action on it by taking information from flow table. Corresponding action performed on given packet where pattern matched with flow table's pattern. In case no similar pattern found then switch sends idle message to controller through OpenFlow protocol to calculate the flow rule for given data packet [23]. SDN has many advantages as compare to traditional internet protocol networks. Easy to update the policies at single point (i.e. controller) and quite easier for network administrator to manage the whole network. In traditional network difficult to find failure specially when network consist of large number of nodes but in case of SDN controller has view of whole network and have awareness about failures that occurs in network. Due to these advantages in this era SDN adopted by many organizations e.g. Huawei, Google, VMware and Microsoft [20, 30, 35]. These organizations using SDN parallelly with already established functional network. As we discussed that data plane and controlled plane decoupled in SDN and it provide better programming capabilities, easy to manage flow of data packets and network virtualization, etc. [32]. All these advantages are admirable however, separation of data and control plane also cause of some difficulties. In which fast link failure recovery is also included because whole network is depending on SDN controller for failure handling, which is cause of large delay in between [15]. Failure recovery delay is also cause of packet loss and badly effect the network services.

Challenges in 5G networks

In this era, the growing usage of video streaming online gaming and smart technologies are the main causes for development of 5G network system [1]. To overcome these challenges 5G technology introduced which is providing ten times more data than 4G network system [4]. 5G network set to provide the services with high availability in cost effective way. Different technologies such as SDN and NFV currently using by many cloud service provider for providing high throughput, Resilience and reliability with low latency rate in 5G [24]. As we discussed

that video streaming and gaming, etc. Are causes of utilization of bandwidth and delay in service availability. To overcome these issues multi path flow protocol and forwarding devices with latest technology for fast communication between different planes of SDN paradigm while it is integrated with 5G network. SDN paradigm works very efficiently using Multipath flow protocol for achieving better utilization of 5G resources with high throughput with low latency.

In 5G networks many environmental factors which can affect the efficiency of wireless signals. To overcome this issue radio, microwaves distributed Heterogenous based on urban development and user distribution territory. If decision about place of base station made. However, in different scenarios few similarities can be occur. Because mostly developed cities, highways have same architecture. In existed systems signals distribution performed homogenously so, it will be generating issues like fractal and environmental [6, 38].

Besides these advantages of integration of 5G and SDN in large number of end users, requirement of high bandwidth, and reliability can be badly affect by link/node failure. In SDN occurrence of link failure not only in centralized point but also can be effect data plane as well. To overcome these issue 5G network should be able to predict/detect the failure occurrence either in central point or in data plane. To recover these failures in very small period which will almost negligible. In our proposed system SDN and 5G integrated with multipath flow protocol but in modified form of it. Multipath flow protocol using reliability of given path it will be reduce the failure recovery time and effectively reduced the bandwidth in 5G network.

2. Related Work

In SDN failure recovery possible with two mechanisms. Now in this section we will discuss some existing techniques and their deficiencies.

Proactive Failure Recovery Mechanisms

FF is failure recovery mechanism which is mostly suitable for port failure detection and recovery. In group tables few actions are predefine in buckets. Watch group detect a failure occur at any post that

flag down indicator then any alternative port with liveliness will be used instead of failed one [9, 3].

In SDN by adopting Bidirectional Forwarding Detection (BFD) failure detection performed by using control messages and echo in between two nodes. Link current state checked by control messages and these messages send to each node. Nodes can be making judgement for status of existing session by echo messages. In FF group less involvement of controller after computation of primary path. However, drawback is alternative paths have not predefined, so if primary path fails there is no alternative path for failure recovery.

SPIDER project is the one where researchers overcome the problems faced in FF group. In SPIDER, failure recovered without communicating to controller when there is no alternative path available [27]. By using link probing failure detected and can be resend with low latency without involvement of Controller. However, this solution is completely based on data plane.

It is also a proactive solution in which two flow entries must be installed for each switch for every incoming packet for its associated path. One of them is used as active and other one is alternative path when failure occurred [19]. However, it is suitable only for small networks where number of failures is very small and another issue is TCAM memory limitation which can overflow when number of matching and actions increased in network.

In searching for an alternating path if congestion factor considered then alternative paths can be computed with low packet loss rate [33]. In this technique back paths predefined for each primary path. Any flow in which failure occurs it can be retransmitted by using alternative path. In Congestion aware techniques researchers have overcome these issues like less involvement of controller, Reducing flow entries, etc. However, if alternative path calculated for each link, it will be cause of traffic overhead [34, 28].

Challenges in Proactive Failure Recovery Mechanisms

- 1 SDN Switches available with limited number of flow entries. E.g. 8000 flow rules can store in State of art switches. Cost increased when more switches required [13].

- 2 In large scale SDN network when number of flow entries increased then in flow entries matching process (for alternative paths) will be cause of greater latency rate [5].
- 3 Proactive approach is suitable for small scale network, because when number of flow entries increased then Data plane scales upward.
- 4 In dynamic conditions may be possible that backup path may be fail earlier than first configured path. In this case, there will be no alternative path when failure occurs.

Reactive Failure Recovery Mechanisms

In reactive failure recovery consists of the following steps.

- 1 Monitoring the status of network by heartbeat mechanism.
- 2 Detection of failure based on heartbeat messages.
- 3 Controller computation for alternative path for failure recovery.
- 4 Replacement of old entries with new flow entries for updating path.

Challenges in Reactive Failure Recovery Mechanisms

Shortest distance mechanism was proposed in [8]. In which priority-based flow used. A packet with highest priority takes minimum delay for failure recovery. Due to avoidance of congestion this mechanism is not suitable for large scale SDN. Because when size of network increased it will be cause in increasing complexity of algorithm. This technique has not been applied on standard topology.

As number of flow operations increases then average failure recovery delay increased. Thus, to overcome this issue flow operations minimized as described in [37]. If alternative path selected with low cost (small number of flow operations) then overall failure recovery delay can be reduced. In described that a single flow entry consumes 11 ms. [17]. In realistic SDN minimum 200 to 300 ms required to recover a failure. In following table, a comparative view of proactive and reactive failure recovery techniques.

As we have already discussed two methods of link failure recovery in integration of SDN with 5G. In these existing systems in which proactive mecha-

Table 1
Comparison of Proactive and reactive Approaches

Issues	Proactive	Reactive
Routing Tables Updates	Heartbeat and Echo messages	In data plane (Failure occurs)
TCAM memory	High Consumption	Low Consumption
Flow operation Matching	High, due to backup paths	Fewer matching
Network Configuration	Difficult to backup paths	Easy to find backup paths
Processing Load on Switches	High load (more flow rules)	Low Overhead traffic on switch
Latency Rate	Small (predefined paths)	High (Controller involved)
Scalability (large/small)	Suitable for small	Large networks

nism used computation of alternative paths cause of overburden at centralized controller specially when number of nodes in data plane increased and utilization of switch's memory also increased. Specifically, when end users of 5G demands high service availability with minimum ten times faster as compare to 4G [2]. In predefined mechanisms mostly researchers focused on how we can reduce the utilization of memory or by reducing the load at SDN controller (less involvement in failure recovery). However, these defined mechanisms can work more efficiently if multipath flow protocol calculate different number alternative paths after measuring the reliability of primary path. Like If reliability of primary path is maximum then no need to compute alternate paths for this link. It will be creating great impact on memory consumptions with low latency rate because number of alternative paths decreases.

3. Proposed Methodology

Reliable multipath flow mechanism proposed in which in first step controller compute a primary path between sender and receiver node when sender node send a request to SDN controller for path computa-

tion as we discussed in above section. After that Using proposed methodology calculate reliability of primary path on basis of predefined factors. Then how many numbers of alternate paths will be stored in forwarding table is depend on reliability ratio. After first phase then we also include distance calculation and find the shortest path using minimum spanning tree or Dijkstra can be used to find shortest path. It will be more effective when reliability and distance both attributes integrate in proposed method. Dijkstra algorithm is also used for calculating shortest path.

Bootstrapping process

When an OpenFlow channel is established between controller and switch, Symmetric packets like Hello, Echo request, and Echo response are exchanged among the controller and all switches. Controller initiates a Feature-Request message for the switch. In response to said request, switch generate an asynchronous message Feature-Reply for Controller [31]. Multiple packets are exchanged using OpenFlow channel which is initiated by the controller for switch states inspection, state modification, interface statistics, flow rule statistics, and capabilities. By caching these response packets in the proposed methodology, controller maintains the network-wide view dynamically and periodically [16].

Graph Composition process

Controller C in our proposed methodology has an application for transforming the data plane information and attributes into weighted undirected graph G . Controller periodically updates graph connectivity in the response of end node and devices discovery events. It also updates the nodes joining like end-user and forwarding devices in data plane as vertices $V_{\text{END_USER}}$, links $l_{\text{END_USER}}$ their attributes l -attribute and V_{SWITCHES} , links l_{SWITCHES} . Reliability inquired by controller reflects the stability of link among forwarding devices and end-user connectivity with devices and proposed approach use procedural programming fashion for its processing reliable flow rule installation.

Algorithm 1

Initialize undirected Graph = \hat{G}

1 **Procedure** Activate (C (event_publisher, event_subscriber)) \rightarrow Control functionality for getting data plane event and handling.

- 2 Procedure** Data Plane (link_event, Publisher)
C_handler (link_event, Subscriber)
 Return devices_dictionary (V_{END_USER} ,
 $l_{END_USER} [l_attribute]$)
- 3 Procedure** Data_Plane (end_user_event,
 Publisher):
C_handler (end_user_event, Subscriber)
 Return end_user_dictionary ($V_{SWITCHES}$,
 $l_{SWITCHES} [l_attribute]$)
- 4 UPDATE** ($\hat{G} \{ [V_{END_USER}: l_{END_USER} [l_attribute]],$
 $V_{SWITCHES}: l_{SWITCHES} [l_attribute] \}$)

Primary path computation

In this phase, after the graph composition, now network can start working. At First stage, flow tables will be empty. After that switch 1 receives a packet from host A. Switch s1 sending a message to SDN controller for path computation after checking its own forwarding table. If entry $\mathfrak{P} - \check{\mathfrak{e}}$ matched with forwarding table entries then corresponding action will be performed, otherwise now SDN controller decides whether this packet will be forwarded or not according to already defined network policy(A).

If $\mathfrak{P} - \check{\mathfrak{e}} \rightarrow$ Permit or Deny (1)

else

$A \ni \mathfrak{P} \quad \forall (A_{SRC}, A_{DST}) \ni (\mathfrak{P}_{SRC}, \mathfrak{P}_{DST})$ (2)

$\hat{S} = \mathfrak{P}_{SRC} + l_1 + l_2 +$
 $l_3 \dots \dots \dots + l_n + \mathfrak{P}_{DST}$ (3)
 OpenFlow Command [$\mathfrak{P} - \check{\mathfrak{e}} \rightarrow \mathfrak{S}\omega$]

Reliability Computation process

Controller application use periodically probe packets for inspection of link failure frequency within a time slot (10 seconds) and compute the link reliability. Higher the number of failures φ cause in lower the reliability percentage and introduce signal of flow rule installation for controller. φ presents the failure frequency between 0 and 1, where λ is recovery constant of link for all re-channelized links in topology heuristically.

$\Phi = (1 - \varphi)$ (4)

$\Phi = (1 - \varphi)$ (4)

$1 - \Phi \times \lambda$ where $\lambda=10$ (5)

Φ in Equation 4 presents the link aliveness time in stipulated 10 seconds time slot while $\Phi \times \lambda$ is reliability. A path \hat{S} computed from source to destination is a set of all intermediate forwarding devices. Equations 6-7 present the path from source to destination along with associated aliveness time between source and destination and the sum of path aliveness time of any path.

$\hat{S} = \Phi_1 l_1 + \Phi_2 l_2 + \Phi_3 l_3 \dots \dots \dots \Phi_n l_n$ (6)

$\hat{S} = \sum_{i=0}^n \Phi_i l_i$ (7)

$i = \{0, 1, 2, 3, \dots, n\}$

Reliable multipath flow

In this phase, first primary path will be computed based on highest reliability which is computed in last phase D. All alternative paths will be calculated according to rules defined as follows.

$\hat{S}_{reliable} = \sum_{i=0}^n \text{Max } \Phi_i l_i$ (8)

- 1 IF $\hat{S}_{reliable} > 90\%$ Then no alternative path.
- 2 IF $\hat{S}_{reliable} > 80\%$ Then 2 alternative paths.
- 3 IF $\hat{S}_{reliable} > 70\%$ Then 3 alternative paths.
- 4 IF $\hat{S}_{reliable} > 60\%$ Then 4 alternative paths.
- 5 IF $\hat{S}_{reliable} > 50\%$ Then 5 alternative paths.
- 6 IF $\hat{S}_{reliable} < 50\%$ Then all alternative paths.

Distance Based Reliable multipath flow

It is a modified form of multipath flow in which primary path calculated not only reliability based but shortest distance attributes also involved in it. It will be decreasing the latency rate better as compare to RMF.

$\hat{S} = \sum_{i=0}^n \text{Max } \Phi_i \text{ Min } l_i$ (9)

Algorithm 2**Input**

1 $(\hat{G}\{V_{\text{END_USER}} : \mathcal{L}_{\text{END_USER}}[\mathcal{L}\text{-attrib}],$
 $V_{\text{SWITCHES}} : \mathcal{L}_{\text{SWITCHES}}[\mathcal{L}\text{-attrib}]\})$

2 Source_node

3 Destination_node

For i, j in Zip($\mathcal{L}_{\text{END_USER}}[\mathcal{L}\text{-attrib}],$
 $V_{\text{SWITCHES}} : \mathcal{L}_{\text{SWITCHES}}[\mathcal{L}\text{-attrib}]),$

For $\text{time}_{\text{stamp}}$ in $\mathcal{L}_{\text{END_USER}}[\mathcal{L}\text{-attrib}]$:
 $i.\text{get}(\mathcal{L}_{\text{END_USER}}[\mathcal{L}\text{-attrib}][\text{time}_{\text{stamp}}][\text{reliability}]),$

For $\text{time}_{\text{stamp}}$ in $\mathcal{L}_{\text{SWITCHES}}[\mathcal{L}\text{-attrib}]$:
 $j.\text{get}(\mathcal{L}_{\text{SWITCHES}}[\mathcal{L}\text{-attrib}][\text{time}_{\text{stamp}}][\text{reliability}]),$

Case::1

if $(i > \text{threshold})$ and $(j > \text{threshold})$:

call Procedure RMF(\hat{G}, i, j)

if $(i < \text{threshold})$ and $(j > \text{threshold})$:

call Procedure ALT(\hat{G}, i, j)

Case::2

if $(i > \text{threshold})$ and $(j > \text{threshold})$:

call Procedure DRMF(\hat{G}, i, j)

Procedure RMF (\hat{G}, i, j):

For level in $\text{reliab}_{\text{dictionary}}$:

If $i, j = \text{reliab}_{\text{dictionary}}[\text{value}]$:

$K.\text{get}(\text{reliab}_{\text{dictionary}}[\text{alternate_paths}])$

For path in K:

OpenFlow_Configuration(path)

Procedure2_ALT(\hat{G}, i, j):

OpenFlow_Configuration(All_paths)

Procedure DRMF(\hat{G}, I, j):

For level in $\text{reliab}_{\text{dictionary}}$:

If $I, j = \text{reliab}_{\text{dictionary}}[\text{value}]$:

$K.\text{get}(\text{reliab}_{\text{dictionary}}(\text{Min}([\text{alternate_paths}])))$

For path in K:

OpenFlow_Configuration(path)

Flow of Proposed Methodology

In working of reliable multiple path flow starting by controller take a network view of all nodes (switches) after that primary reliable path computed by controller when switch requested for it. Reliability measuring algorithm computed reliability if calculated reliability is less than threshold then all alternative paths updated in flow table but in case more than threshold then MRF cases decides how many alternative paths assigned. Primary path calculation based on reliability and shortest distance. In our proposed system we have used dijkstra algorithm for calculating shortest path. Flow diagram repeated for n number of paths and each time update latest paths in flow entries.

4. Simulation and Results**Simulation**

Perform series of experiments of proposed approaches POX To controller and Mininet simulator have used in it.

Mininet Simulator

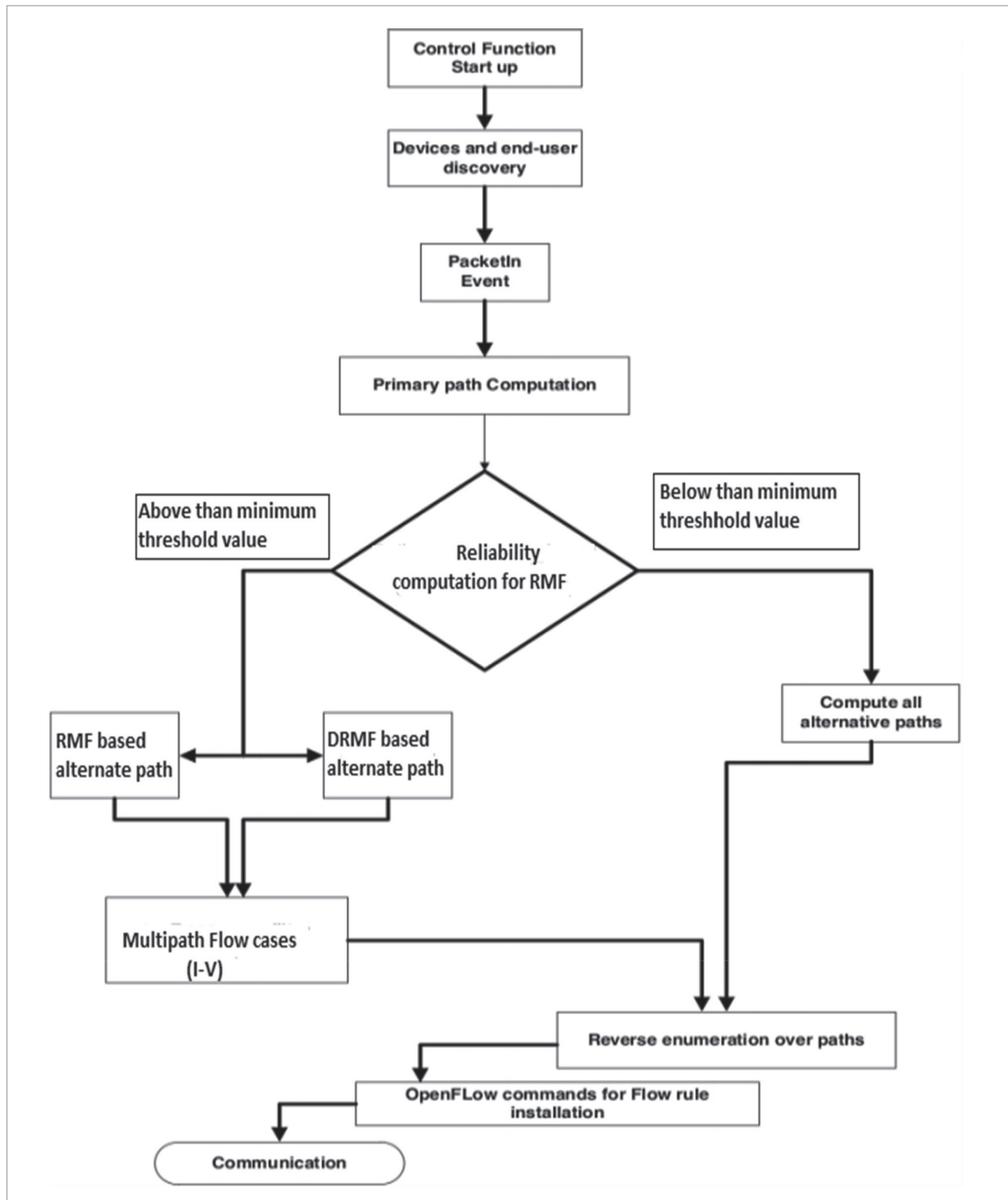
Mininet simulator have used in experimentation because it is feasible for both large scale and small-scale simulation of network. Hundreds and thousands of nodes can be tested easily using simple tools for command line and API. The simulator has the benefit of interface for multiple SDN controller like Pox, Floodlight, Ryu and Open daylight regardless of topology development programming interface level as a programming model in SDN comprises of low, mid, and high-level programming [8, 37, 31]. Mininet provide easy customization, sharing, and testing nodes of SDN [19]. It also provides a virtually separate interface for any host node for processing of host granular applications. Mininet simulator suitable for both (real and simulated controller).It can also use to simulate connections between different types of controllers like POX, Ryu, etc. [40]. Various types of switches can be created and modify using Mininet according to required simulation. NASA, ICSI, and many other researchers used in world used Mininet for multi controller simulations [14].

POX Controller

Stateless switch communication based OpenFlow protocol can be controlled by POX framework [12].

Figure 1

Flow Diagram of Proposed Methodology



Python language can used POX for design a SDN controller. It is efficient tool used in research for developing a basic SDN controller [8]. By adding more components, a complex SDN controller can be designed. POX can support 1.0 and 1.3 versions of OpenFlow switches. POX also provide interface for Mininet, open source availability, and integration with other simulators like NS3 [36].

Experimental Results

In our emulated network following components used.

Table 2

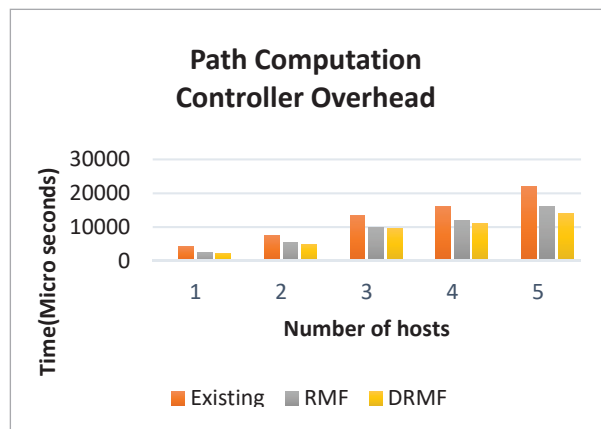
Components of Network

Virtual end hosts	25
OVS switches	9
OVS switches	9
Packet generated per host	10000 UDP
Size of average packet	62 bytes

Better results obtained generated by 5 hosts because of limited resources. All experiments performed using POX controller (*version 2.2*).Mininet simulator improves our work performance and competitively better results than existing approaches ./we have used python for scripting due to commutability of python with POX and Mininet. Figure 2 presents comparison between overheads of existing and new system.

Figure 2

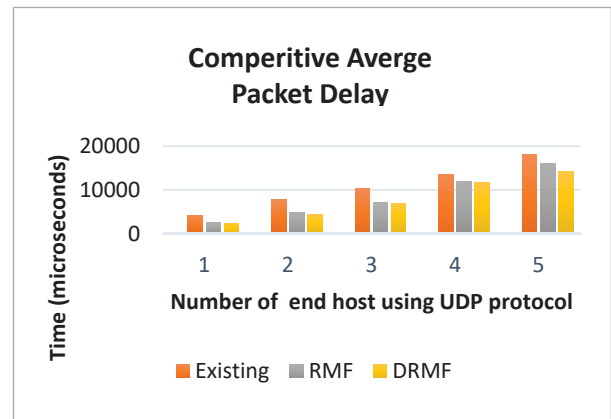
Overhead comarison between existing and RMF



Overhead of existing system greater due to installing large number of alternative paths which is cause of increase in communication between controller and intermediate switches. Figure 3 elaborates delay comparison of packet transmission. RMF has smaller delay due to less involvement of controller.

Figure 3

Average packet transmission delay



In Figure 4, elaboration of flow rule installation is shown. If length of path increased, then number of alternative paths also increase which will be cause of large number of flow rules installation in existing system. In RMF/DRMF consumes less memory for flow rules installations.

Figure 4

Flow rules Installation

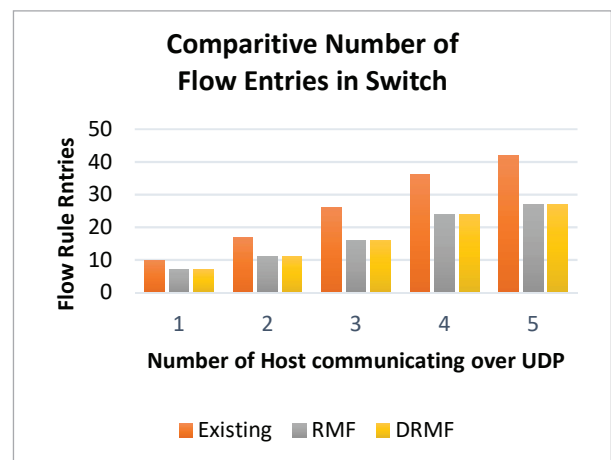
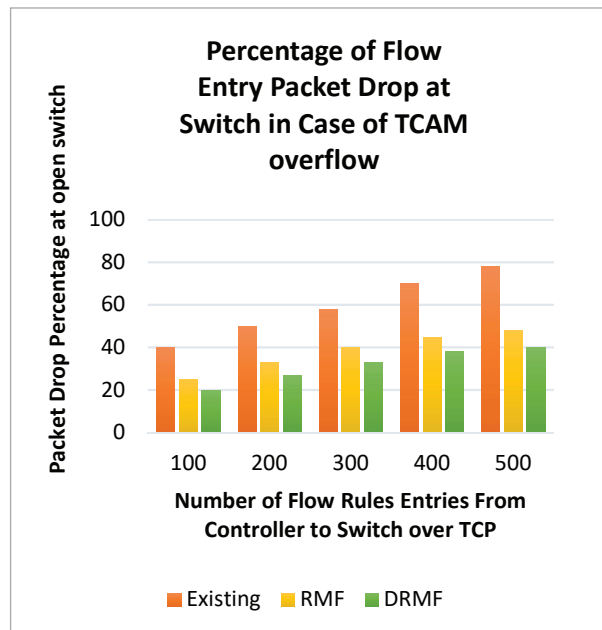


Figure 5 illustrates the observation of flow entry encapsulated packet drop at switch. The reason behind this behavior of OpenFlow switch has limited memory. These memory constraints insist on the mechanism of flow rule installation wisely by considering the forwarding devices' capacity. In this situation, our proposed solution is relatively more suitable in the SDN production network.

Figure 5
Switch flow entry drop percentage in case of switch memory Overflow



Observations regarding the average delay of flow rule entries installation in Figure 6 conclude that as the number of entries increases, the time of their installation also increases proportionally. When the controller computes the higher frequency of these flow entries along with another application interface then control traffic for configuration approaches to delay. RMF and DRMF are relatively less delayed oriented and perform configurations according to the network policies timely.

All or selective path installation procedures in the existing approach install the useless entries in switch memory. Figure 7 presents the experimental results by accessing the packets and number of bytes en-

Figure 6
Average delay computed at controller for installation of computed Flow Rule Entries

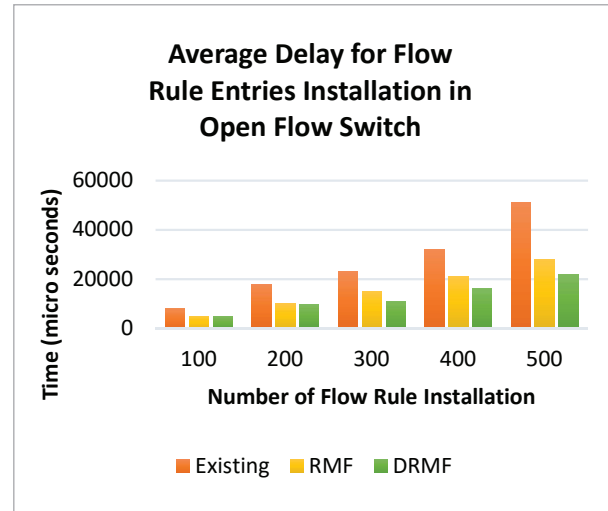
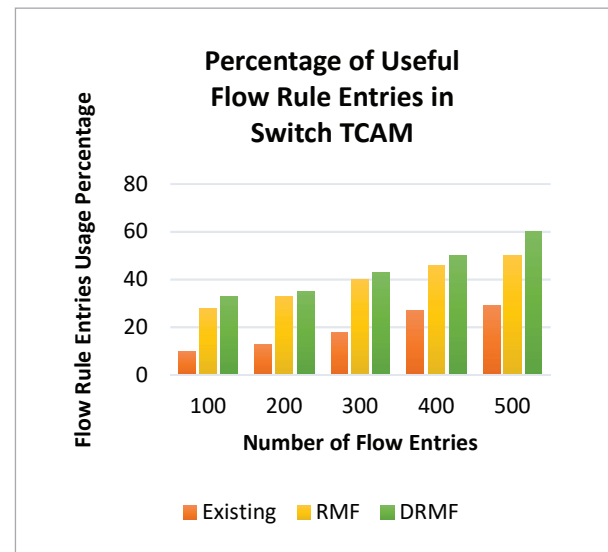


Figure 7
Flow Rule Entries entertaining the production packets according to specified action in a flow rule entry



tertained by flow rule entries in switches and it has yielded that RMF and DRMF are much efficient for said situation. The proposed approach presents the highest percentage comparative to the existing approaches and useful entries in switches.

5. Conclusion

As we have discussed that 5G networks used to provide high availability to end users in services (such as video streaming, gaming, etc). However, different types of network failures will be cause of a great economic loss if services not provided with high speed and reliability. To overcome this issue many techniques designed for failure recovery but without using reliability factor. Thus, as shown in results existing system produce large traffic overhead due to computation of many alternative paths and latency rate increased too. SDN is and emerging paradigm in which control plane decoupled by data plane. Due to this reason SDN easy to control by centralized controller (SDN Controller). In our proposed system we have used reliability factor to and number of alternative paths depend on this reliability. As reliability of primary path increased number of alternative paths will reduce. Therefore, traffic overhead

decreased due to less involvement of controller, low latency rate in failure recovery because small number of alternative paths will be computed for failure recovery. After that, we have involved shortest distance factor which improves our results as shown in results. In future, we can improve our work by using machine learning algorithm for reliability computation and including more factors like shortest distance. It will be beneficial in future 5G networks.

Acknowledgment

This research was supported by a grant of the Korea Health Technology R&D Project through the Korea Health Industry Development Institute (KHIDI), funded by the Ministry of Health & Welfare, Republic of Korea (grant number: HI21C1831) and the Soonchunhyang University Research Fund.

References

1. Aggarwal, R. Kompella, K. Nadeau, T. Swallow, G. Bidirectional Forwarding Detection (BFD) for MPLS Label Switched Paths (LSPs); Internet Engineering Task Force: Fremont, CA, USA, 2017, RFC 5884, RFC 7726.
2. Barakabitze, A., Sun, L., Mkwawa, I.-H., Ifeachor, E. A Novel QoE-Centric SDN-based Multipath Routing Approach of Multimedia
3. Services over 5G Networks. IEEE International Conference on Communications, May, 2018, 191-209.
4. Capone, A., Cascone, C. Nguyen, A. Q. T., Sanso, B. Detour Planning for Fast and Reliable Failure Recovery in SDN with OpenState. In Proceedings of the International Conference on Design of Reliable Communication Networks, 24-27 March, 2015, 25-32. <https://doi.org/10.1109/DRCN.2015.7148981>
5. Cascone, C., Sanvito, D., Pollini, L., Capone, A., Sansò, B. Fast Failure Detection and Recovery in SDN with Stateful Data Plane. International Journal of Network Management, 2017, 1957. <https://doi.org/10.1002/nem.1957>
6. Chang, D. F., Govindan, R., Heidemann, J. The Temporal and Topological Characteristics of BGP Path Changes in Network Protocols. In Proceedings of the 11th IEEE International Conference on Network Protocols, Atlanta, GA, USA, 4-7 November, 2003, 190-199.
7. Chen, J., Ge, X., Zhong, Y., Li, Y. A Novel JT-CoMP Scheme in 5G fractal Small Cell Networks. In IEEE Wireless Communications and Networking Conference (WCNC), 2019 April 15, 1-7. <https://doi.org/10.1109/WCNC.2019.8886024>
8. Cheng, Z., Xiaoning, Z., Li, Y., Yu, S., Lin, R., He, L. Congestion Aware Local Reroute for Fast Failure Recovery in Software-Defined Networks. IEEE/OSA Journal of Optical Communications and Networking, 2017, 9(11), 934-944. <https://doi.org/10.1364/JOCN.9.000934>
9. Francois, P., Bonaventure, O., Decraene, B., Coste, P.A. Avoiding Disruptions During Maintenance Operations on BGP Sessions. IEEE Transactions on Network and Service Management, 2017, 221-233.
10. Huang, L. Shen, Q. Shao, W. Congestion Aware Fast Link Failure Recovery of SDN Network Based on Source Routing. KSII Transactions on Internet and Information Systems, 2017, 11, 5200-5222. <https://doi.org/10.3837/tiis.2017.11.002>
11. Jin, C., Lumezanu, C., Xu, Q., Zhang, Z. L., Jiang, G. Telekinesis: Controlling Legacy Switch Routing with OpenFlow in Hybrid Networks. In Proceedings of the 1st ACM SIGCOMM Symposium on Software Defined Networking Research, Santa Clara, CA, USA, 17-18 June, 2015, 20. <https://doi.org/10.1145/2774993.2775013>

12. Kandan, M., Valliyammai, V., Deepa, C. Switch Failure Detection in Software-Defined Networks. *Advances in Big Data Cloud Computing*, 2019, 750, 155-162. https://doi.org/10.1007/978-981-13-1882-5_13
13. Kitsuwon, N., Payne, D. B., Ruffini, M. A Novel Protection Design for OpenFlow-based Networks. *Proceedings of 16th International Conference on Transparent Optical Networks*, July, 2014, 1-5. <https://doi.org/10.1109/ICTON.2014.6876515>
14. Kreutz, D., Ramos, P. E., Rothenberg, C. E., Azodolmolky, S., Uhlig, S. Software-Defined Networking: A Comprehensive Survey. *IEEE Proceedings*, 2015, 103, 14-76. <https://doi.org/10.1109/JPROC.2014.2371999>
15. Lee, S. S. W., Li, K. Y., Chan, K. Y., Lai, G. H., Chung, Y. C. Software-based Fast Failure Recovery for Resilient OpenFlow Networks. In *Proceedings of 7th International Workshop on Reliable Networks Design and Modeling*, October 5-7, 2017, 194-200.
16. Lin, Y. D., Teng, H. Y., Hsu, C. R., Liao, C. C., Lai, Y. C. Fast Failover and Switchover for Link Failures and Congestion in Software Defined Networks. In *Proceedings of the 2016 IEEE International Conference on Communications (ICC)*, 22-27 May, 2019, 1-6. <https://doi.org/10.1109/ICC.2016.7510886>
17. Lin, Y. D., Teng, H. Y., Hsu, C. R., Liao, Y. C. Fast Failover and Switchover for Link Failures and Congestion in Software Defined Networks. In *Communications (ICC), IEEE International Conference on 2016*, 1-6. <https://doi.org/10.1109/ICC.2016.7510886>
18. Malik, A., Aziz, B., Adda, M., Ke, C. H. Smart Routing: Towards Proactive Fault Handling of Software-Defined Networks. *Computer Networks*, 2020, 1655-1668. <https://doi.org/10.1016/j.comnet.2020.107104>
19. Martínez, R. Experimental SDN Control Solutions for Automatic Operations and Management of 5G Services in a Fixed Mobile Converged Packet-Optical Network. In *International Conference on Optical Network Design and Modeling (ONDM)*, 2018, 214-219. <https://doi.org/10.23919/ONDM.2018.8396133>
20. Martínez, R. Integrated SDN/NFV Orchestration for the Dynamic Deployment of Mobile Virtual Backhaul Networks Over a Multilayer (Packet/Optical) Aggregation Infrastructure. *IEEE/OSA Journal of Optical Communications and Networking*, 2017, A135-A142. <https://doi.org/10.1364/JOCN.9.00A135>
21. Montero, R. et al. Supporting QoE/QoS aware End-to-End Network Slicing in Future 5G-Enabled Optical Networks, Metro and Data Center Optical Networks and Short-Reach Links II. *Int'l. Society for Optics and Photonics*, 2019, 10946, 289-305. <https://doi.org/10.1117/12.2508579>
22. Nam, T. M., Phong, P. H., Khoa, T. D., Huong, T. T., Nam, P. N., Thanh, N. H., Thang, L. X., Tuan, P. A., Loi, V. D. Self-Organizing Map-Based Approaches In DDOS Flooding Detection Using SDN. *International Conference on Information Networking (ICOIN)*, IEEE, 2018, 249-254. <https://doi.org/10.1109/ICOIN.2018.8343119>
23. Ndiaye, M. Hancke, G. P. Abu-Mahfouz, A. M. Software Defined Networking for Improved Wireless Sensor Network Management: A Survey. *Sensors*, 2019, 5200-5222.
24. Oliveira, T. et al. SDN-Based Architecture for Providing QoS to High Performance Distributed Applications, *Proceedings of 2018 IEEE Symposium on Computers and Communication (ISCC)*, IEEE, 2018, 602-607. <https://doi.org/10.1109/ISCC.2018.8538694>
25. Padma, V., Yogesh, P. Proactive Failure Recovery in OpenFlow Based Software Defined Networking, In *Proceedings of the 3rd International Conference on Signal Processing, Communication and Networking (ICSCN)*, Chennai, India, 26-28 Marchm 2017, 1-6.
26. Paramonov, A., Muthanna, A., Aboulola, O. I., Elgendy, I. A., Alharbey, R., Tonkikh, E., Koucheryavy, A. Beyond 5G Network Architecture Study: Fractal Properties of Access Network. *Applied Sciences*, 2020, 238-255. <https://doi.org/10.3390/app10207191>
27. Qiu, K., Yuan, J. Zhao, J. Wang, X. Secci, S. Fu, X. Efficient Recovery Path Computation for Fast Reroute in Large-scale Software Defined Networks. *IEEE*, 2018, 37, 1755-1768. <https://doi.org/10.1109/JSAC.2019.2927098>
28. Ramos, R. M. Martinello, M. Rothenberg, C. E. Slick-Flow Resilient Source Routing in Data Center Networks Unlocked by OpenFlow. In *Proceedings of the 38th Annual IEEE Conference on Local Computer Networks*, October 2018, 606-613.
29. Rawat, D., Reddy, B. Software Defined Networking Architecture, Security, and Energy Efficiency, A survey. *Environment*, 2017, 3(5), 6.
30. Rezaee, M., Moghaddam, M. H. Y. SDN-Based Quality of Service Networking for Wide Area Measurement System, *IEEE Transactions of Industrial Informatics*, May, 2020, v16, 3018-3028. <https://doi.org/10.1109/TII.2019.2893865>
31. Sgambelluri, A. et al. Orchestrating QoS-Based Connectivity Services in a Multi-Operator Sandbox. *Journal of Optical Communications and Networking*, 2019, 11(2), A196-A208. <https://doi.org/10.1364/JOCN.11.00A196>

32. Shantharama, P., LayBack, SDN Management of Multi-Access Edge Computing (MEC) for Network Access Services and radio Resource Sharing. *IEEE Access* 6, 2018, 57545-57561. <https://doi.org/10.1109/ACCESS.2018.2873984>
33. Sharma, S. Staessens, D., Colle, D., Pickavet, M., De-meester, P. Openflow: Meeting Carrier-Grade Recovery Requirements. *Computer Communication*, 2018, 36, 656-665. <https://doi.org/10.1016/j.comcom.2012.09.011>
34. Siminesh, C. N., Grace, M. K. E. Ranjitha, K. A Proactive Flow Admission and Re-routing Scheme for Load Balancing and Mitigation of Congestion Propagation in SDN Data Plane. *International Journal of Computer Networks & Communications*, 2019, 314-329.
35. Soliman, M., Nandy, B. Lambadaris, I. Ashwood-Smith, P. Exploring Source Routed Forwarding in SDN-based WANs. In *Proceedings of IEEE International Conference on Communications*, June, 2018, 3070-3075.
36. Van Andrichem, N. L. M., Van Asten, B.J., Kuipers, F.A. Fast Recovery in Software-Defined Networks. In *Proceedings of the Third European Workshop on Software Defined Networks*, London, UK, 1-3September, 2019.
37. Venkatesh, K., et al. QoS Improvisation of Delay Sensitive Communication Using SDN Based Multipath Routing for Medical Applications. *Future Generation Computer Systems*, 2019, 93, 256-265. <https://doi.org/10.1016/j.future.2018.10.032>
38. Vissicchio, S., Vanbever, L., Pelsser, C., Cittadini, L., Francois, P., Bonaventure, O. Improving Network Agility with Seamless BGP Reconfigurations. *IEEE/ACM Transactions on Networking (TON)*, 2013, 21, 990-1002. <https://doi.org/10.1109/TNET.2012.2217506>
39. Wang, D., Liu, S., Zhao, Y. A Preliminary Study on the fractal Phenomenon Disconnected+ Disconnected= Connected. *Fractals*, 2017 February 27, 25(01), 1750004. <https://doi.org/10.1142/S0218348X17500049>
40. Yao, J., Han, Z., Sohail, M., Wang, L. A robust Security Architecture for SDN-based 5g Networks. *Future Internet*, 2019, 11(4), 85. <https://doi.org/10.3390/fi11040085>
41. Ye, J., Cheng, X., Zhu, J., Feng, L., Song, L. A DDOS Attack Detection Method Based on SVM in Software Defined Network. *Security and Communication Networks*, 2018, 309-324. <https://doi.org/10.1155/2018/9804061>

