SERBIAN JOURNAL OF ELECTRICAL ENGINEERING
Vol. 21, No. 2, June 2024, 235-250
UDC: 004.735:62-55 **DOI: https://doi.org/10.2298/SJEE2402235N**

A Proactive Controller Failure Recovery Mechanism in SD-WAN with Multiple Controllers

Sminesh Choorkunnu Narayanan[1], Viji Varghese[1]

Abstract: The separation of data and control plane is a key feature of Software-defined networking (SDN), which makes network administration more intelligent. The control plane is realized using a logically centralized controller. In SD-WAN, as the network becomes larger, multiple controllers are needed to manage the network. In this scenario, there is a chance for controller failure due to overload. Once the controller fails, the switches lose connection with the controller. The load of the failed controller is to be re-distributed among other controllers. Sometimes this load transfer may cause many other problems like controller chain failure. It may consistently undermine the trustworthiness of the network. In the proposed technique, the multiple controller deployment based on affinity propagation clustering in SD-WAN is extended to include a proactive controller failure recovery mechanism. Whenever the controller load exceeds a pre-defined threshold, a set of switches under the bottleneck controller will be reassigned to a neighbouring controller without exceeding its capacity. The simulation results show that when network traffic increased, the proposed proactive controller failure recovery method balanced the controller load, resolved a cascading controller failure, improved the average throughput, and reduced the average end-to-end delay and packet loss effectively.

Keywords: Multi-controller failure recovery, Cascading controller failure, SD-WAN.

1 Introduction

In SDNs, the data plane consists of packet-forwarding switches, and the control plane decides where and how to forward the packets. The control plane controls the data plane elements through a well-defined application programming interface [1]. The adequacy of a single controller is challenging the size of the network increases [2, 3]. A better solution for this problem is to deploy multiple controllers to work together as logically centralized controllers, this method provides flexibility and efficiency in managing large-scale SDN.

[1]Department of Computer Applications, Government Engineering College Thrissur, Kerala, India;
Emails: smineshcn@gectcr.ac.in, vijivarghese.mca@gectcr.ac.in

In a multi-controller Software Defined Wide Area Network (SD-WAN), the failure of one controller results in the reassignment of switches to the nearby controller. If the controller gets overloaded due to the redistribution of switches it may eventually fail. This repeated transfer of load without considering the backup controller capacity is known as a cascading controller failure problem [4]. Cascading failure is considered one of the main problems in SD-WAN with multiple controllers. Consider a three-controller scenario where each controller has a capacity of four switches. If controller C1 fails, switches S1 and S2 are reassigned to controller C2 which fails from the resultant overload. Similarly, controller C3 also fails, as shown in Fig. 1. To overcome this problem of cascading failure controller failure recovery can make use of the load balancing approaches in SD-WAN.

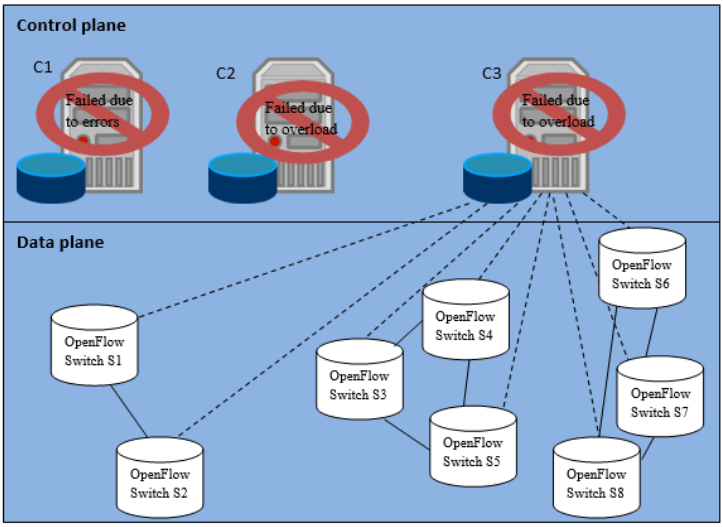

Fig. 1 – *Cascading Controller Failure (Three Controller Scenario).*

The proposed method is a proactive strategy for cascading controller failure recovery in SD-WAN with multiple controllers by considering controller load. In this method, the controller will not get failed due to its overload. The load on the bottleneck controller is transferred to the adjacent, lightly loaded controllers. The proposed algorithm checks whether it exceeds the capacity of the controller to which the load is transferred before reassigning the switches. Then the switches are reassigned to the nearest neighbouring selected controllers. The switch reassignment process is repeated until the load on the bottleneck controller reaches a lower threshold. The deployment of multiple controllers based on non-parametric clustering in the SD-WAN [5] is further extended to include a failure recovery algorithm.

Mininet [6] is used to simulate the proposed system with the POX controller as the external controller. The SWITCH, GEANT and BICS network topologies selected from the Internet Topology Zoo [7] are used to simulate the proposed algorithm. Simulation results show that reassigning switches to neighbouring controllers without exceeding their residual capacity is a better way to prevent cascading controller failure. The Distributed Internet Traffic Generator (D-ITG) produces the network traffic, and the log files are analysed to observe QoS metrics. It is noted that in the SWITCH network, the average throughput is improved by 36.6%while the average end-to-end delay and average packet loss are reduced by 34.7% and 25.4%, respectively. In the GEANT network, the average throughput is improved by 30.5%, and the average end-to-end delay and average packet loss are reduced by 38.7% and 24.6%, respectively. Similarly, in the BICS network, the average throughput is improved by 15.4%, and the average end-to-end delay and average packet loss are reduced by 39.2% and 22.5%, respectively. When the network traffic increases, the proposed proactive failure recovery mechanism improves network efficiency and Quality of Service (QoS) parameters as well.

The rest of this paper is structured as follows. Section 2 discusses the related work and Section 3 introduces the proposed proactive approach. Simulation and results are presented in Section 4 followed by a conclusion in Section 5.

2 Related Works

A centralized controller is appropriate for medium and small-sized networks, with the chief concern being how to ensure its resiliency and reliability. An SD-WAN with multiple controllers also enhances the complexity with several new problems affecting network management. A controller that fails in a multi-controller scenario should have its load re-transmitted to the neighbouring controllers [8, 9]. Hu et al. [4] proposed a method named Hyperflow that transfers the load of the failed controller to the nearest neighbouring controller without considering its current workload. In such a case, a centralized controller is ideal for medium and small-sized networks for added resiliency and reliability.

Isong et al. [10] proposed a method for fault tolerance that comprises an FT manager (FTM) and three controllers. The FTM has different parts that contribute to FT by observing and detecting faults using heartbeat messages and recovering from failure using checkpointing.

Most controller failure recovery strategies assign a minimum of one backup controller to every primary controller in the network. Using a higher number of backup controllers will provide better survivability in the network. However, there will be a chance for resource wastage. Zhang et al. [11] published a resource-saving replication method to reduce the number of backup controllers. The authors split the set of controllers into Ordinary controllers and Important

controllers. In the case of important controllers, a method is provided to find out a proper place to fix the backup controllers. For ordinary controllers, controllers of other domains are used to transfer the load.

The unfavorable allocation of slave controllers may cause cascading controller failure and break down the entire network, an Adaptive Slave Controller Assignment method proposed by Hu, Tao et al. [4] resolves this problem. It provides tolerance to the fault by allocating switches rationally to the backup controllers. The ASCA chooses slave controllers by checking the status of the network, processing capability of the controllers, and cost assignment. Thereafter the switches are allocated to different slave controllers for load balancing.

Aly et al. [12] provided an effective fault tolerance approach, whenever a controller experience failure, this model redistributes the load of the failed controller to the nearest controller/s. However, this method did not consider the load of the backup controllers.

Zhao et al. [13] proposed a method based on affinity propagation clustering, the similarity measure for the algorithm is the minimum distance between the switches and controllers. The authors compute the shortest path propagation delay proportional to the Euclidean distance between the nodes for the clustering.

Sminesh et al. [5] proposed a deployment of multiple controllers based on the affinity propagation clustering (MCPAP) method in which the identification of the subnetwork number is automated by applying an exemplar-based, iterative message exchange method. The proposed algorithm proceeds to compute two real-valued messages, responsibility and availability, which are updated until the election of exemplars. The exemplar-based message passing is carried out iteratively until the candidate exemplar locations are stable. The exemplars and their locations are stored in a set Caffinity which is retrieved as the controller placement for the selected network topology.

Yang et al. [14] proposed a solution for WAN connection failures and network congestion using MPTCP. The authors improved the throughput and fault tolerance in the SD-WAN network effectively.

Guo et al. [15] proposed a method to manage flows that are made offline due to controller failure. The authors assign such offline flows to the neighbouring active controllers using a heuristic algorithm which in turn improves the flow recovery. However, the cascading controller failure problem is unaddressed in the experiment.

Radam, et al. [16] construct the network using graph theory which in turn results in improved delay and throughput parameters. However, customizing WAN topologies shall lead to unreliable results.

Varsha et al. [17] posited a multi-mapping technique where the switches are mapped to multiple controllers. When the primary controller fails the secondary controller takes the role of the failed controller hence the switching time can be reduced considerably. The multiple controller failure scenario is not addressed in this paper.

Guo et al. [18] created a switch-level programmability recovery solution in a multi-controller failure scenario by smartly mapping the recovered switches to active controllers. The proposed heuristic model achieves good recovery performance with low computation time and limits the propagation delay. Path programmability for flows with stringent QoS requirements is not addressed.

Dou et al. [19] proposed an adaptive solution to recover offline flows under multiple controller failures in the SD-WAN by jointly changing offline flow paths and establishing active controllers to switch mappings by designing a heuristic algorithm. However, the link failure probability between a controller and a switch is not incorporated.

Isyaku et al. [20] studied various SD-WAN fault-tolerant mechanisms. The authors made a comparison of static and dynamic routing approaches, path protection and restoration schemes. They recommended dynamic routing-based approaches and hybrid methods for failure recovery for future SD-WANs.

It is observed that the proactive failure recovery methods are more suitable for multi-controller SD-WANs than its reactive approaches from the perspective of failure recovery time. The multi-controller placement algorithm, MCPAP, may be further extended to include proactive failure recovery. Reassigning the switches of the overloaded controller to neighbouring controllers without exceeding their capacity will resolve cascading controller failure in the SD-WAN.

3 The Proposed System

The objective of the proposed method is to develop a proactive method for cascading failure recovery in multi-controller-based SD-WANs by considering the controller load. In this method, the load on the bottleneck controller is transferred to other less-loaded adjacent controllers to avoid the bottleneck controller failure due to its overload. While transferring the load it proactively checks whether it exceeds the capacity of the controller to which the load is to be transferred. Thereafter the switches are reassigned to adjacent controllers that can hold the load without getting overloaded. The algorithm computes suitable controllers for each switch until the load on the bottleneck controller reaches a lower threshold.

The proposed technique is developed as four submodules: network topology creation, computation of controller load, controller failure detection, and

reassignment of switches. In module 1, the network topology is simulated using Mininet [6] and the POX controller [21]. The SWITCH, GEANT and BICS network topologies from the Internet Topology Zoo are used for the simulation. With the multi-controller placement obtained from the MCPAP algorithm [5], the custom topology is created with the controllers and switches. The links between the switches and the hosts associated with them are added to the custom topology. The second module will calculate the controller flow by using the ofp_flow_stats_request. It can be periodically calculated by setting a hard timeout. Then the web flows on switches under the same controller can be added to get the controller flow. Then the third module is responsible for detecting the bottleneck controller and finding out the reassigning switches and their corresponding controllers. The last module will reassign the switches to backup controllers as per the result obtained from the second module. A detailed block diagram of the proposed method is depicted in Fig. 2.

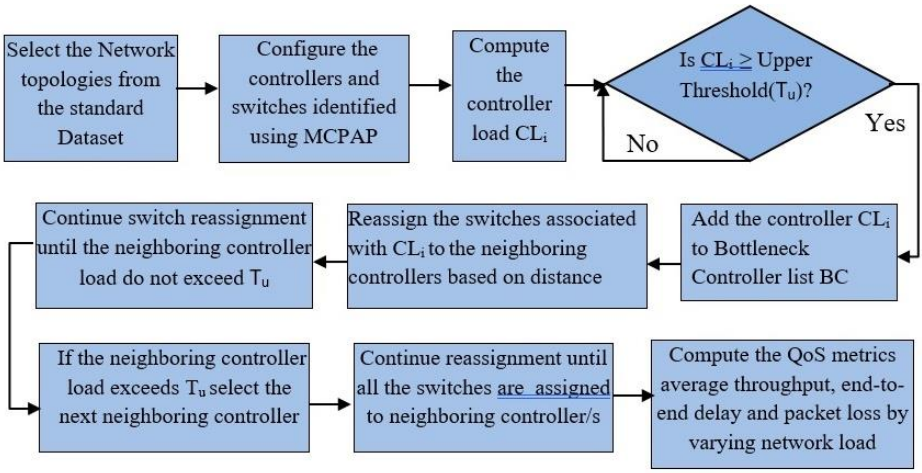

Fig. 2 – *The proposed system.*

4 Simulation and Results

The simulation is carried out using SD-WAN topologies from the Internet Topology Zoo (ITZ) [7]. The ITZ dataset contains country/continent-wide network topologies. The metadata includes geographic location information and link bandwidth details. The MCPAP method, using affinity propagation, computes the multi-controller placement in the SD-WAN. The number of controllers identified using MCPAP in the SWITCH, GEANT and BICS topologies is shown in **Table 1**.

The MCPAP algorithm computes the required controllers and the switches are assigned to it. In the selected SD-WAN topologies SWITCH, GEANT and BICS network, the computed number of controllers are four, six and five respectively.

Table 1
Network topologies.

Serial Number	Network Topology	Nodes	Edges	Number of Controllers Computed by MCPAP
1	SWITCH	30	51	4
2	GEANT	40	61	6
3	BICS	32	48	5

The deployment of multiple controllers using MCPAP is extended to include the proactive failure recovery algorithm. Mininet is used to emulate the MCPAP and MCPAP With Failure Recovery (MCPAP-WFR) methods. The performance of the networks is analyzed from the perspective of the quality-of-service parameters of average throughput, end-to-end delay, and packet loss. The proposed MCPAP-WFR computes the above-mentioned QoS metrics for the selected SWITCH, GEANT and BICS network topologies. The proposed MCPAP-WFR is given in Algorithm 1.

Algorithm 1: MCPAP With Failure Recovery (MCPAP-WFR)

Input: Multi-Controller Placement using Affinity Propagation in SWITCH, GEANT and BICS Initialize $T_u \leftarrow$ Controller load upper threshold and $T_l \leftarrow$ Controller load lower threshold

Procedure: MCPAP-WFR()

Configure Controllers and switches in Mininet based on MCPAP in SWITCH, GEANT and BICS

Configure D-ITG to generate the required number of flows in the network

for <each Controller C_i in the selected network topology> **do**

Compute the controller load CL_i

if CL_i => the upper threshold T_u

Add C_i to the Bottleneck Controller List BC

Add each switch associated with C_i to the Reassigned Switches list RS

Add the remaining controllers to the admissible controller list AC

Sort the list in the increasing order of distance from C_i

end if

end for

for <each switch j in RS >**do**
Calculate the load from each switch j: LS_j
for <each controller k in RC >**do**
Calculate the load of the controller LRC_k
Calculate the load on the reassignment of switch j on controller k: $LRC_k + LS_j$
if $T_l <= LRC_k + LS_j <= T_u$
Reassign switches to the controller k and **goto** step 12
end if
end for
end for
Compute the QoS metrics average throughput, end-to-end delay, and packet loss for a varying number of flows from the D-ITG log files.

Output: Average Throughput, End-to-end delay, and Packet loss

In the above algorithm, the controller failure can be observed by checking whether the controller load exceeds a pre-defined upper threshold Tu. The upper threshold is set as 80% of the capacity of the controller. The bottleneck controllers identified are added to a bottleneck controller list. After computing the bottleneck controller, each switch under this controller is selected and the distance between the switch and other controller/s is computed. The remaining controllers are added to the list in increasing order of the distance calculated from the switch. Check whether the first controller in the list can hold the switch or not. If the sum of the backup controller load and the assigned switch load is less than the pre-defined upper threshold, the switch will be reassigned to the particular controller. The process is repeated until the controller load exceeds the lower threshold and the next controller in the bottleneck controller list is selected for the reassignment. The process will continue until the load on the bottleneck controller reaches below the lower threshold Tl. The lower threshold is set as 60% of the capacity of the controller. Thus, the proposed method computes the list of bottleneck controllers and reassigned switches, along with the corresponding controller to which they are reassigned.

In the simulation, the network traffic is increased by changing the number of traffic flows in each connection. The performance metrics averaged over the number of flows are observed directly from the D-ITG log file [22]. The flow characteristics in the D-ITG are shaped to follow exponential distribution in both inter-departure time and packet size. The multi-flow mode enables ITGSend to generate several flows simultaneously. A single thread manages each flow separately, with another thread acting as master and coordinating the others.

Exploiting a multi-threaded design, ITGSend can send multiple parallel traffic flows toward multiple ITGRecv instances and vice versa. The simulation runs are repeated by increasing the number of flows to 10, 20, 40, 50 and 100. The QoS metrics computed for each simulation run are shown in the following section.

4.1. Average end-to-end delay

The average end-to-end delay observed in the SWITCH, GEANT, and BICS network topologies using MCPAP and MCPAP-WFR is shown in **Tables 2–4**, respectively.

Table 2
Average delay in the SWITCH topology.

Number of flows	Delay in MCPAP [s]	Delay in MCPAP-WFR [s]
10	4.35	3.8
20	10.37	8.73
40	27.68	10.62
50	29.47	16.68
100	43.95	26.29

Table 3
Average delay in the GEANT topology.

Number of flows	Delay in MCPAP [s]	Delay in MCPAP-WFR [s]
10	6.85	5.83
20	14.55	12.27
40	27.82	12.32
50	36.15	13.16
100	43.15	24.17

Table 4
Average delay in the BICS topology.

Number of flows	Delay in MCPAP [s]	Delay in MCPAP-WFR [s]
10	5.81	4.83
20	10.62	8.20
40	23.56	12.91
50	34.12	14.30
100	41.18	19.43

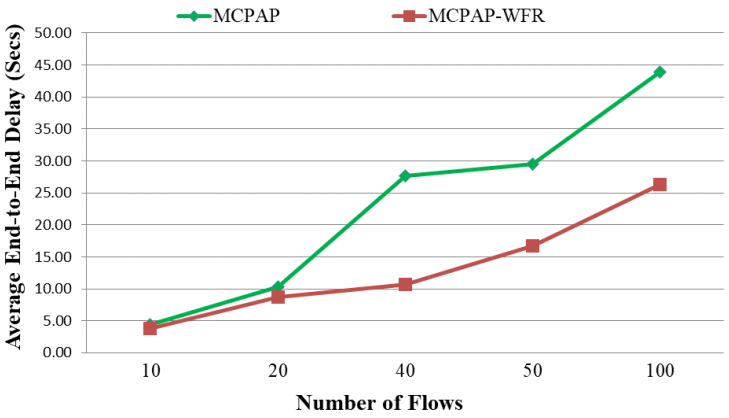

Fig. 3 – *Average end-to-end delay in the SWITCH topology.*

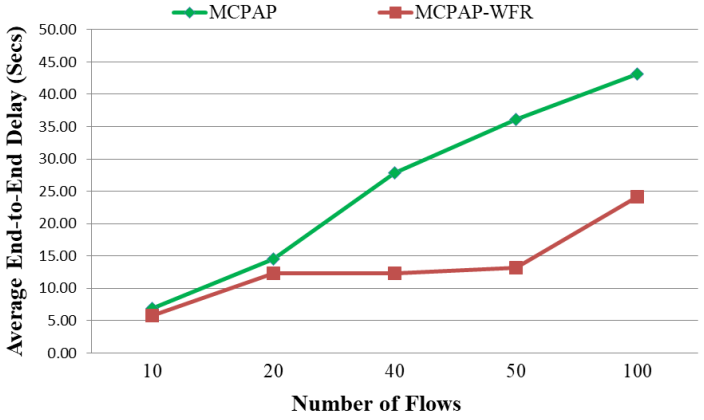

Fig. 4 – *Average end-to-end delay in the GEANT topology.*

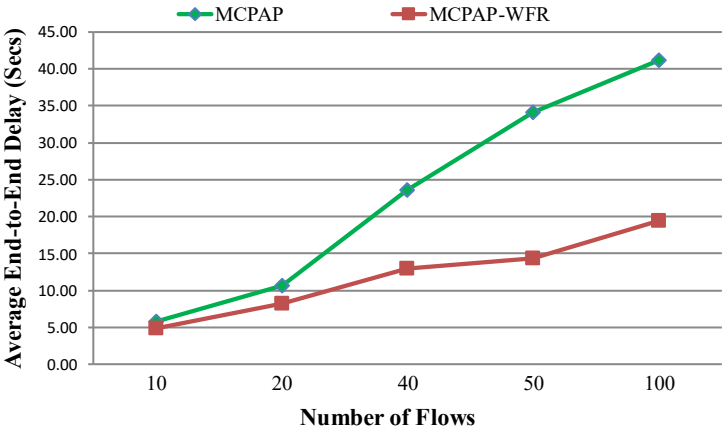

Fig. 5 *– Average end-to-end delay in the BICS topology.*

In the SWITCH, GEANT, and BICS topologies, the average end-to-end delay will increase with the number of flows transferred. The simulation results show that the average end-to-end delay incurred for packets in the proposed MCPAP-WFR is less than in the MCPAP scheme. The proposed system reassigns the switches and reduces queuing delay proactively. It is observed that in the proposed MCPAP-WFR, the average end-to-end delay is reduced by 34.7%, 38.7% and 39.2% in the SWITCH, GEANT, and BICS topologies, respectively. Figs. 3, 4 and 5 show the observed average end-to-end delay with the increasing number of flows in the selected topologies.

4.2. Average throughput

The observed average throughput in the SWITCH, GEANT, and BICS network topologies in the multi-controller scenarios using MCPAP and MCPAP-WFR are shown in **Tables 5**–**7**, respectively.

Table 5

Average throughput in the SWITCH topology.

Number of flows	Throughput in MCPAP[Kb/s]	Throughput in MCPAP-WFR[Kb/s]
10	2084.35	2455.18
20	2171.44	2920.11
40	2251.06	3965.84
50	2408.08	4634.12
100	2914.15	5959.19

Table 6

Average throughput in the GEANT topology.

Number of flows	Throughput in MCPAP[Kb/s]	Throughput in MCPAP-WFR[Kb/s]
10	1022.2	1152.29
20	1475.13	1665.83
40	1789.38	2954.73
50	2013.67	3852.29
100	2354.54	4100.31

Table 7

Average throughput in the BICS topology.

Number of flows	Throughput in MCPAP[Kb/s]	Throughput in MCPAP-WFR([Kb/s]
10	2012.84	2139.89
20	2543.02	2835.56
40	2785.38	3239.82
50	3015.94	3935.91
100	3361.94	4391.59

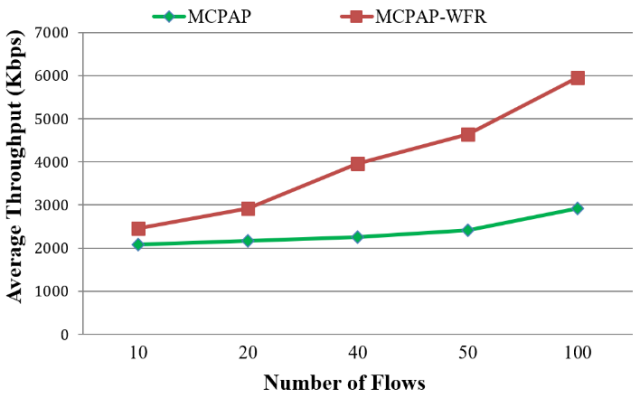

Fig. 6 – *Average throughput in the SWITCH topology.*

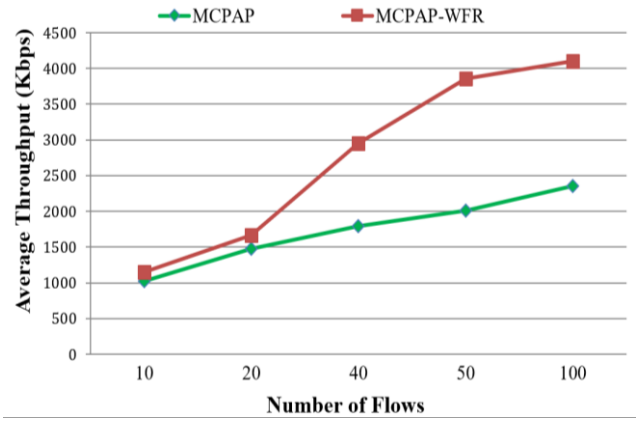

Fig. 7 – *Average throughput in the GEANT topology.*

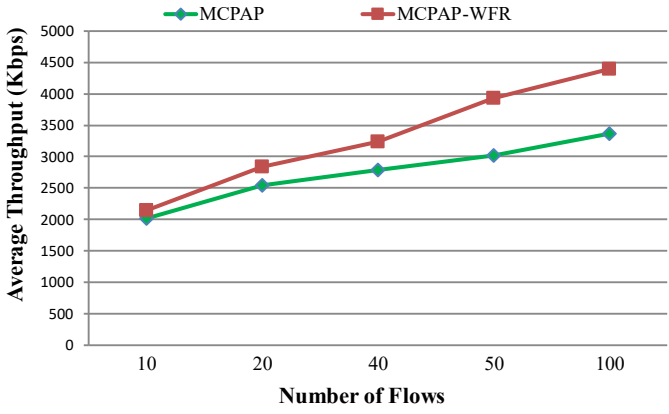

Fig. 8 – *Average throughput in the BICS topology.*

The simulation results show that when the number of flows and network load increases, the proactive failure recovery in MCPAP-WFR effectively reduces the possibility of congested controllers and packet drops, resulting in higher average throughput. It is observed that the average throughput is improved by 36.6%, 30.5% and 15.4% in the SWITCH, GEANT, and BICS topologies, respectively. The average throughput observed in the SWITCH, GEANT, and BICS topologies is depicted in Figs. 6, 7 and 8, respectively.

4.3 Average packet loss

The average packet loss over the number of flows in the SWITCH, GEANT, and BICS network topologies using MCPAP and MCPAP-WFR multi-controller SD-WAN is shown in **Tables 8**–**10**.

Table 8

Average packet loss in the SWITCH topology.

Number of flows	Packet Loss in MCPAP (#packets)	Packet Loss in MCPAP (#packets)
10	510	439
20	4646	3213
40	22281	15301
50	24043	18448
100	55441	40026

Table 9

Average packet loss in the GEANT topology.

Number of flows	Packet Loss in MCPAP (#packets)	Packet Loss in MCPAP (#packets)
10	583	455
20	7406	6542
40	16297	12254
50	28631	20748
100	50457	31566

Table 10

Average packet loss in the BICS topology

Number of flows	Packet Loss in MCPAP (#packets)	Packet Loss in MCPAP (#packets)
10	985	748
20	2431	1864
40	12541	10154
50	29301	20389
100	41387	34982

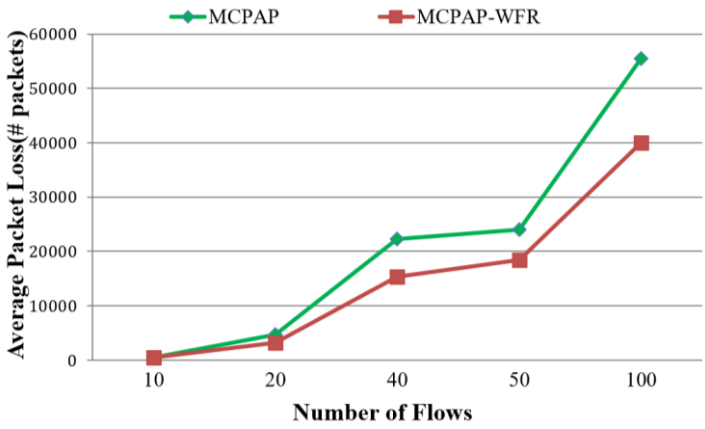

Fig. 9 – *Packet loss in the SWITCH topology.*

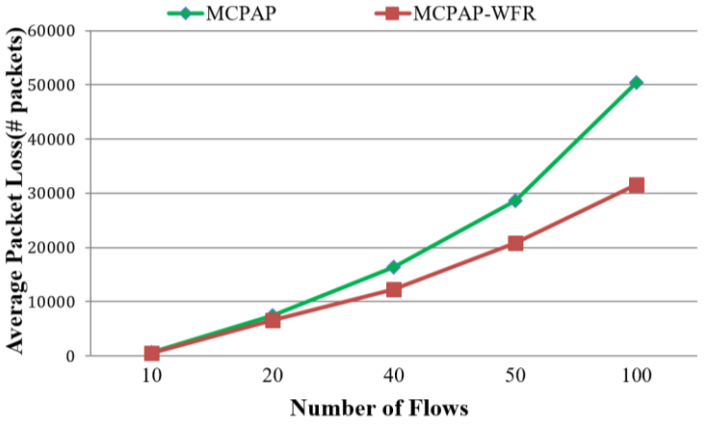

Fig. 10 *– Packet loss in the GEANT topology.*

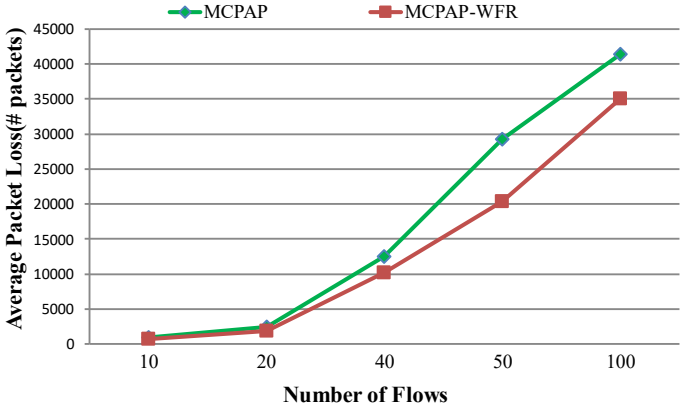

Fig. 11 – *Packet loss in the BICS topology.*

The simulation results show that the average packet loss escalates with the number of flows. When the network traffic grows, the proposed MCPAP-WFR balances the controller load effectively to reduce the packet drop. The proposed MCPAP-WFR reduces the average packet loss by 25.4%, 24.6% and 22.5%, respectively, in the SWITCH, GEANT and BICS networks selected in the simulation. The variation in packet loss with different numbers of flows in the network topologies SWITCH, GEANT and BICS network topologies is shown in Figs. 9, 10 and 11, respectively.

It is observed that the proposed MCPAP-WFR reassigns switches proactively, based on the controller load, and averts the cascading failure problem. When the network traffic escalates, the proposed method augments network efficiency from the QoS perspective, the average throughput, end-to-end delay, and packet loss.

5 Conclusion

The proposed proactive controller failure recovery mechanism MCPAP-WFR monitors the network traffic and proactively reassigns the switches under the bottleneck controller to their neighbouring controllers, while reassigning the switches it also checks the load of these controller/s. Hence reducing the chance of cascading controller failure. The switches are reassigned to the existing adjacent controllers in the SD-WAN, thus bypassing the need for additional backup controllers. However, the reassignment is done manually since the simulation tool Mininet is not capable of automatically reassigning switches to the nearest controllers. When the network load escalates in comparison with MCPAP the proposed MCPAP-WFR method substantially improved the QoS metrics and network efficiency. In this research, since all the flows have equal priority, the objective was to minimize offline flows in the event of failure. In terms of future research, flows with stringent QoS requirements may be classified and assigned priorities. The failure recovery of such flows can be carried out without compromising QoS.

6 References

[1] M. Jammal, T. Singh, A. Shami, R. Asal, Y. Li: Software Defined Networking: State of the Art and Research Challenges, Computer Networks, Vol. 72, October 2014, pp. 74 – 98.

[2] I. F. Akyildiz, A. Lee, P. Wang, M. Luo, W. Chou: A Roadmap for Traffic Engineering in SDN-OpenFlow Networks, Computer Networks, Vol. 71, October 2014, pp. 1 – 30.

[3] Y. Zhang, L. Cui, W. Wang, Y. Zhang: A Survey on Software Defined Networking with Multiple Controllers, Journal of Network and Computer Applications, Vol. 103, February 2018, pp. 101 – 118.

[4] T. Hu, Z. Guo, J. Zhang, J. Lan: Adaptive Slave Controller Assignment for Fault-Tolerant Control Plane in Software-Defined Networking, Proceedings of the IEEE International Conference on Communications (ICC), Kansas City, USA, May 2018, pp. 1 – 6.

[5] C. N. Sminesh, E. G. M. Kanaga, A. G. Sreejish: A Multi-Controller Placement Strategy in Software Defined Networks Using Affinity Propagation, International Journal of Internet Technology and Secured Transactions, Vol. 10, No. 1-2, January 2020, pp. 229 – 253.

[6] B. Lantz, N. Handigol, B. Heller, V. Jeyakumar: Introduction to Mininet, Available at: https://github.com/mininet/mininet/wiki/Introduction-to-Mininet.

[7] S. Knight, H. Nguyen, N. Falkner, M. Roughan: Realistic Network Topology Construction and Emulation from Multiple Data Sources, The University of Adelaide, Tech. Rep. 2012.

[8] I. F. Akyildiz, A. Lee, P. Wang, M. Luo, W. Chou: Research Challenges for Traffic Engineering in Software Defined Networks, IEEE Network, Vol. 30, No. 3, May-June 2016, pp. 52 – 58.

[9] Y. Zhang, L. Cui, W. Wang, Y. Zhang: A Survey on Software Defined Networking with Multiple Controllers, Journal of Network and Computer Applications, Vol. 103, February 2018, pp. 101 – 118.

[10] B. Isong, I. Mathebula, N. Dladlu: SDN-SDWSN Controller Fault Tolerance Framework for Small to Medium Sized Networks, Proceedings of the 19th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD), Busan, South Korea, June 2018, pp. 43 – 51.

[11] L. Zhang, Y. Wang, X. Zhong, W. Li, S. Guo: Resource-Saving Replication for Controllers in Multi Controller SDN Against Network Failures, Proceedings of the IEEE/IFIP Network Operations and Management Symposium, Taipei, Taiwan, April 2018, pp. 1 – 7.

[12] W. H. Fouad Aly, A. M. Ali Al-anazi: Enhanced Controller Fault Tolerant (ECFT) Model for Software Defined Networking, Proceedings of the Fifth International Conference on Software Defined Systems (SDS), Barcelona, Spain, April 2018, pp. 217 – 222.

[13] J. Zhao, H. Qu, J. Zhao, Z. Luan, Y. Guo: Towards Controller Placement Problem for Software-Defined Network Using Affinity Propagation, Electronics Letters, Vol. 53, No. 14, July 2017, pp. 928 – 929.

[14] Y. Zhang, J. Tourrilhes, Z.- L. Zhang, P. Sharma: Improving SD-WAN Resilience: From Vertical Handoff to WAN-Aware MPTCP, IEEE Transactions on Network and Service Management, Vol. 18, No. 1, March 2021, pp. 347 – 361.

[15] Z. Guo, S. Dou, W. Jiang: Improving the Path Programmability for Software-Defined WANs Under Multiple Controller Failures, Proceedings of the IEEE/ACM 28th International Symposium on Quality of Service (IWQoS), Hang Zhou, China, June 2020, pp. 1 – 10.

[16] N. S. Radam, S. T. Faraj Al-Janabi, K. Sh. Jasim: Multi-Controllers Placement Optimization in SDN by the Hybrid HSA-PSO Algorithm, Computers, Vol. 11, No. 7, July 2022, p. 111

[17] V. Varsha, C. N. Sminesh: QoS Aware Multi Mapping Technology in SD-WAN, Proceedings of the International Conference on Intelligent Data Communication Technologies and Internet of Things (ICICI), Coimbatore, India, August 2020, pp. 421 – 433.

[18] Z. Guo, S. Dou, W. Wu, Y. Xia: Toward Flexible and Predictable Path Programmability Recovery Under Multiple Controller Failures in Software-Defined WANs, IEEE/ACM Transactions on Networking, Vol. 31, No. 5, October 2023, pp. 1965 – 1980.

[19] S. Dou, G. Miao, Z. Guo, C. Yao, W. Wu, Y. Xia: Matchmaker: Maintaining Network Programmability for Software-Defined WANs Under Multiple Controller Failures, Computer Networks, Vol. 192, June 2021, p. 108045.

[20] B. Isyaku, K. Bin Abu Bakar, F. A. Ghaleb, A. Al-Nahari: Dynamic Routing and Failure Recovery Approaches for Efficient Resource Utilization in OpenFlow-SDN: A Survey, IEEE Access, Vol. 10, 2022, pp. 121791 – 121815.

[21] H. M. Noman, M. N. Jasim: POX Controller and Open Flow Performance Evaluation in Software Defined Networks (SDN) Using Mininet Emulator, Proceedings of the 3rd International Conference on Sustainable Engineering Techniques (ICSET), Baghdad, Iraq, April 2020, pp. 1 – 9.

[22] R. Salam, A. Bhattacharya: Performance Evaluation of SDN Architecture Through D-ITG Platform for Distributed Controller over Single Controller, Proceedings of the 12th International Conference on Computing Communication and Networking Technologies (ICCCNT), Kharagpur, India, July 2021, pp. 1 – 6.