SAKARYA UNIVERSITY JOURNAL OF COMPUTER AND INFORMATION SCIENCES
http://saucis.sakarya.edu.tr/
e-ISSN: 2636-8129
Publisher: Sakarya University
Vol. 8, No. 3, 382-391, 2025
DOI: 10.35377/saucis...1599100
RESEARCH ARTICLE

Load Balancing and Quality of Experience in Software-Defined Networks

Gürhan Çoban[1,*], Murtaza Cicioğlu[2], Ali Çalhan[3]

[1]Bursa Uludağ University, 16059 Görükle Nilüfer Bursa/Türkiye, ror.org/03tg3eb07
[2]Bursa Uludağ University, 16059 Görükle Nilüfer Bursa/Türkiye, ror.org/03tg3eb07
[3]Düzce University, 81620 Orhangazi District Düzce/Türkiye, ror.org/04175wc52

Corresponding author:
Gürhan Çoban, Bursa Uludağ University
gurhancoban@gmail.com

Article History:
Received: 10.12.2024
Revised: 12.08.2025
Accepted: 13.08.2025
Published Online: 24.09.2025

ABSTRACT
Software-defined networks (SDN) aim to eliminate the disadvantages of traditional network environments by separating the control plane and the data plane. As a result, the controllability of the network is increased. This study proposes to examine the relationship between load balancing and Quality of Experience (QoE) in SDNs in terms of Structural Similarity Index Method (SSIM), Round Trip Time (RTT), delay and hop count measurements. In addition, the data loss parameter is also examined. A comparison is made between clients and servers in a video streaming test. Five scenarios, namely no-load balancing, Round Robin, average RTT, delay-based and the proposed load balancing method, are tested. The average SSIM value of 0.9678 is obtained with the proposed load balancing method. When the data loss is examined, the average data loss value is 0.25 Mbytes higher than the Round Robin method, but better results are obtained than the other scenarios. High SSIM value and reduction in data loss indicate a better QoE. It is observed that load balancing operations performed by considering the load status of the servers and the statistical data of the network give better results.

Keywords: Load Balancing, Quality of Experience, SDN

1. Introduction

The popularity of SDNs is increasing amazingly because their structure provides information about the current status of the network and supports advanced traffic engineering by controlling network traffic [1]. SDN draws attendance because it is programmable, allows for network application development, and separates the data and control planes that traditionally coexist [2]. In the SDN architecture, the network controller provides an overall view of the network status and facilitates network management [3]. Today, network traffic mostly consists of video-based traffic, so video streaming is the most important traffic type for network traffic engineering [4], [5]. Quality of Service (QoS) is defined as increasing the network performance by controlling the network traffic in limited network capacity [6]. QoS measures how data is transmitted over the network serviced by certain standards. QoS concept includes technical parameters such as network bandwidth, delay, and packet loss. QoS ensures that network resources are used effectively and efficiently. Quality of Experience (QoE) is a parameter that evaluates how the user or customer perceives a service or application. The user utilizes abstract concepts such as good, medium, and bad when expressing satisfaction with the service or service they receive [7]. QoE takes into account not only technical performance but also the user's emotional and psychological reactions. The International Telecommunication Union (ITU) has standardized QoE with the Mean Opinion Score (MOS) to move from subjectivity to objectivity while evaluating QoE. This score is valued between 1 and 5 (1 is the worst and 5 is the best) [8]. In 2016, Huawei used the Video Mean Opinion Score (vMOS) as a QoE measure on network video streaming. Also, human factor engineering studies that examined the capacity and limits of people have been considered. Thus, the relationship between the results obtained and the capacity and limits of people was revealed [9].

In this study, video streaming tests have been implemented and the Structural Similarity Index Method (SSIM) has been used to determine how good the quality of the transferred video is. SSIM is based on assumptions that the human visual system adapts to losses of structural information rather than perceived errors. It uses structural information loss instead of perceived errors and finds the level of perceived structural disorder by proportioning it to the structural information loss [10]. This value estimates the perceived quality of videos by detecting similarities between the original and the transmitted video in video

Cite as: G. Çoban, M. Cicioğlu and A Çalhan. (2025). Load balancing and quality of experience in software-defined networks. *Sakarya University Journal of Computer and Information Sciences*. vol. 8, no. 3, pp. 382-391, 2025 doi: 10.35377/saucis...1599100

streaming [11]. As the obtained SSIM value approaches 1, the similarity between the source video and the transferred video will increase, and it can be said that a higher-quality video transfer will occur, thus increasing the QoE.

The main contributions of the study can be summarized as follows:

- The effects of load balancing on network statistics are examined.
- The effects of network statistics on QoE are examined.
- The determination of QoE with the Structure Similarity Index Method (SSIM) is proposed.

In Section 2, various studies on QoS and QoE are examined. In Section 3, information is given about the prepared scenarios, the testing process, the proposed method and algorithm. The results obtained from the tests are given in detail in Section 4. The results obtained from the examined models and the results of the simulation study are evaluated in the last section.

2. Literature Review

Many studies have been conducted to solve the load-balancing problem in SDNs and different methods have been proposed. Kumar and Anand used the dataset published on Kaggle by Universidad Del Cauca Popayan in Colombia in their study and developed a machine learning-based load-balancing algorithm using the data obtained from CIC Flowmeter (a tool that can collect network statistics). When there is a request, the load-balancing algorithm is first run. The statistical information obtained with the CIC Flowmeter is processed with the K-Means algorithm and the cluster value is calculated. If the cluster value is below the threshold value, the relevant server is redirected [12]. Babayiğit and Ulu proposed a deep learning technique in their study to perform load-balancing in Data Center Networks based on SDN. Load values between connections were used for training the deep learning network. The deep learning technique was compared with machine learning algorithms such as Artificial Neural Network (ANN), Logistic Regression and Support Vector Machine, and as a result, it was seen that deep learning and ANN were better than logistic regression and support vector machine in terms of response time. It was also seen that the deep learning technique gave better results than ANN [13]. Begam et al. proposed a multiple regression-based search algorithm for path routing and server selection in data center networks. The multiple regression-based search algorithm selects the server based on server parameters such as load, response time, bandwidth, and server utilization. It reduced the latency by 45% and achieved 83% better server utilization compared to traditional methods [14]. Wilson Prakash and Deepalakshmi showed the imbalance in the load on the virtual machines used in cloud computing causes inefficient use of resources in their study. In order to prevent the problem, Back Propagation Artificial Neural Network (BPANN) was used to perform dynamic agent-based Load Balancing on the software-defined network. According to the results of the study, it was shown that SDN increased its efficiency and optimized the use of resources by estimating the load under heavy load conditions. In their study, lower data flow time was obtained compared to multipath TCP and heuristic algorithm [15]. Filali et al. proposed migration between data plane elements for load balancing among distributed SDN controllers. The load on the controllers is first estimated and then the migration processes are optimized to give better results under the delay constraint. Two models based on Auto-Regressive Integrated Moving Average (IRM) and Long Short-Term Memory approach were tested for load estimation. It was observed that the Long Short Term Memory approach gave 55% better results in long-term predictions. In the second stage, the reinforcement learning method was used for migration of components [16]. Liang et al. proposed an IoT-based load balancing algorithm on the data center in the SDN architecture and used a Bayesian network to predict load congestion. Reinforcement learning was used to decide the optimal movement [17].

3. Material and Method

In this study, various scenarios are designed and analyzed to determine the relationship between load balancing and QoE in SDN architecture. The analyses of the scenarios are performed on a virtual machine installed on a laptop computer with an Intel i-7 11th Generation 2.8 GHz processor and 8.00 GB RAM. Oracle VM Virtual Box was used as the virtual machine software. A 40 GB hard drive and 5.00 GB RAM are allocated for the created virtual machine. The operating system installed is Ubuntu 22.04.3 LTS distribution in the Linux ecosystem. Mininet software is used to create the SDN environment. Mininet is a software that supports Python and allows easy creation of virtual networks [18]. For the controller in SDN, the Floodlight controller that supports OpenFlow protocols is used. This software is developed as open source. The Floodlight controller, which is a Java-based software, has API support. In this way, it is easy to obtain information about the network, and flow routing operations can be performed through the controller [19]. The NSFNET topology, which is widely used in the literature, is preferred in order to create more realistic topologies. This topology is designed by the National Science Foundation (NSF) of the United States to improve collaboration and resource sharing [20]. The proposed network structure is shown in Figure 1. In the study, computers labeled h10 and h14 are used as servers, and computers labeled h1, h2, h3, h4, h5 h6, h7, h8, h9 h11, h12 and h13 were used as clients.

Table 1. Literature Studies

Authors	Aim - Method	Result
A.Kumar And D. Anand [12]	Load balancing for Software Defined Network with Machine Learning	KMeans perform better than DBSCAN
B. Babayiğit and B. Ulu [13]	Comparison of Deep Learning and Machine Learning algorithms for load balancing in SDN-based data center networks	Deep Learning technique has shown better performance than machine learning algorithms.
G. S. Begam, M. Sangeetha, and N. R. Shanker [14]	A multiple regression-based search algorithm is proposed for load balancing in data center networks.	45% better latency was achieved compared to traditional algorithms
S. WilsonPrakash and P. Deepalakshmi [15]	Back Propagation Neural Network (BPANN) was used to perform dynamic agent-based load balancing on the software defined network.	Lower data flow time was achieved compared to multipath TCP and heuristic algorithm
A. Filali, Z. Mlika, S. Cherkaoui, and A. Kobbane [16]	Proposed migration between data plane elements for load balancing among distributed SDN controllers	IRM and LSTM approach were tested for load forecasting. LSTM approach provided 55% better results in long-term forecasting.
S. Liang, W. Jiang, F. Zhao, and F. Zhao [17]	IoT-based load balancing algorithm was used in the data center in SDN architecture and Bayesian network was used to predict the load congestion.	Average latency reduced by 73%

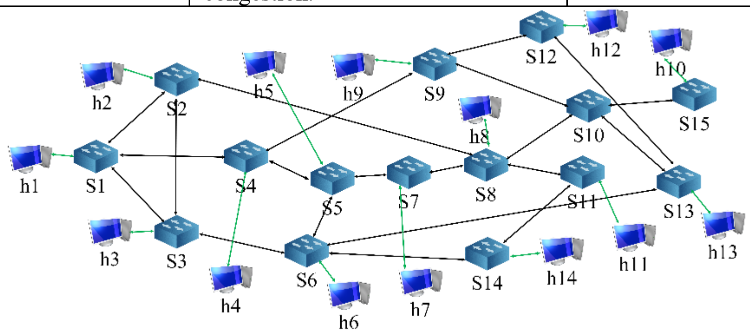


Figure 1. NSFNET Based Network

Five different scenarios were designed for SDN simulation in the study. The scenario where no load balancing is performed is shown in Figure 2. The commonly used load balancing methods are symbolized in Figure 3. Scenario 2 where the round robin method is used, scenario 3 where load balancing is performed according to the average Round Trip Time(RTT) value and scenario 4 where load balancing is performed according to the delay time are expressed in Figure 3. Scenario 5, which is the method proposed in our study, is shown in Figure 4. 10 tests were performed for each scenario. These scenarios are;Scenario 1: All clients are directed to h10 or h14 server with no load-balancing.

Scenario 2: Clients are directed to h10 and h14 servers respectively using the Round Robin method, thus attempting to provide load-balancing. Here, the route with the lowest latency among routes from the client to the server is selected using Floodlight RestAPI when determining the routes.

Scenario 3: Clients are directed to servers H10 and H14 based on the average RTT between the client and the servers. The client is directed to the server with the lower RTT, aiming to achieve load balancing. Here, the route with the lowest latency between the client and the server is selected using the Floodlight REST API during route determination.

Scenario 4: When clients are directed to servers H10 and H14, the latency between the client and the servers is taken into account. The client is directed to the server with the lower latency to achieve load balancing. In this process, the route with the lowest latency between the client and the server is selected using the Floodlight REST API during route determination.

Scenario 5: Clients are routed according to the proposed load-balancing algorithm. This algorithm takes the number of packets on the switch device to which the servers are connected from the Floodlight controller and makes a decision. Routing is done according to the obtained value.

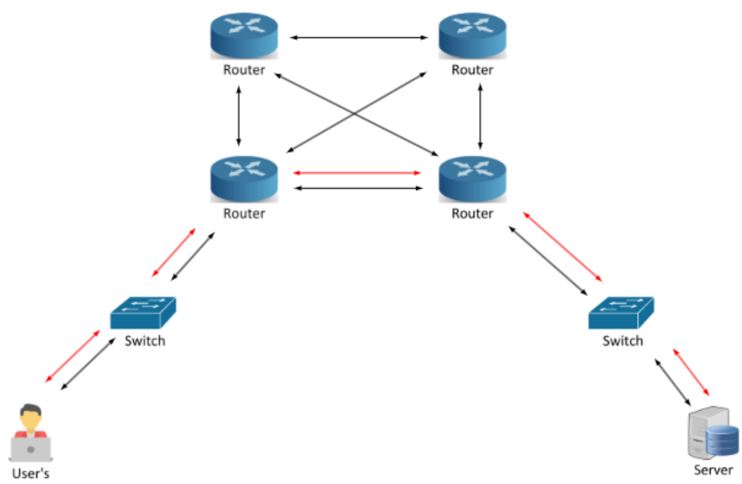


Figure 2. No Load Balancing

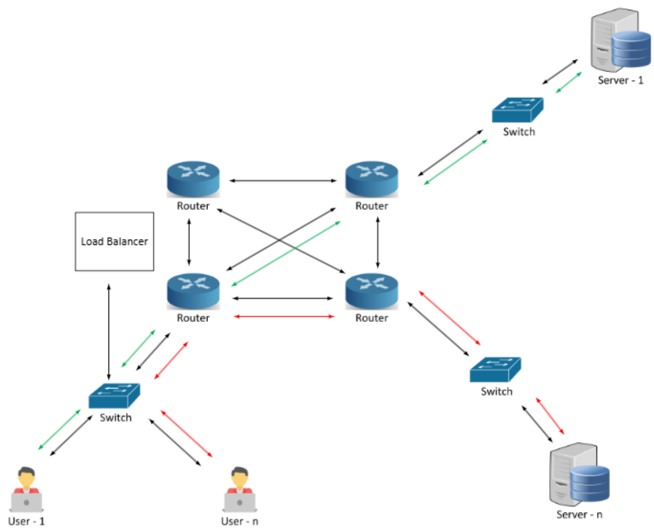


Figure 3. Commonly Used Load Balancing Methods

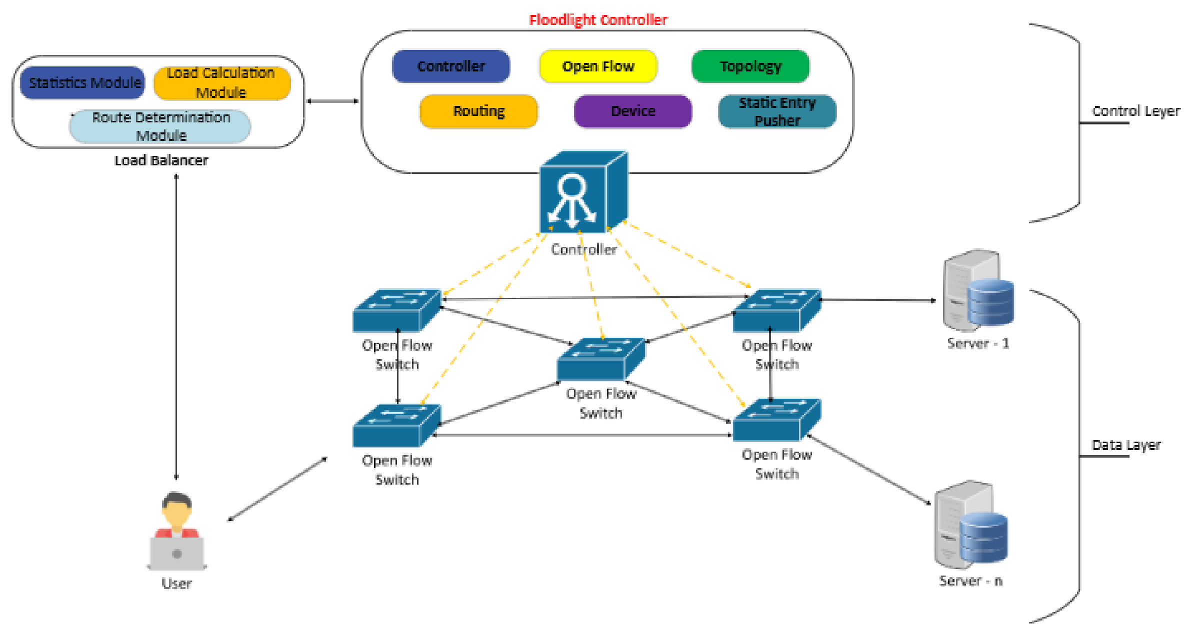


Figure 4. Proposed Method

According to 2022 data, 65% of internet traffic is video streaming [5]. For this reason, we use a 9-minute 56-second, and 14310-frame video that is free video prepared by the Blender Foundation. There are no subtitles in the video. UDP is used for the video stream. The servers are enabled to broadcast video using the ffmpeg library. This video stream is recorded on the client computers and the recorded videos are compared with the source video. Peak Signal to Noise Ratio (PSNR) value is generally used to measure the quality value of the obtained video. Although PSNR value is used to measure the quality of the transferred image and video in most of the studies, the Structure Similarity Index Method is used in this study. SSIM accepts image distortion as a perception change. Here, the perceived quality of images and videos is estimated. It measures the similarity between the original video at the source, and the recorded video at the destination [11].

The QoE is satisfactory when the video at the source and the video at the destination are similar after video streaming. In this study, the SSIM value is tried to improve by the proposed load-balancing method.

The h6-labeled client is used for testing purposes. The computers labeled h10 and h14 are used as video servers. The other clients are used to create the load on the network. In each scenario, the h6-labeled client is the last client to send a video request in order to be in the worst case.

In the proposed approach, the algorithm receives network statistics from the controller when a request comes to the load-balancing algorithm. The number of packets passing through the switch ports to which the servers are connected is compared and routing is done to the server with the lowest packet number. In the meantime, the statistics of the switch devices in the network are also examined and a route is created to provide the lowest delay. Route information is updated in the flow tables. Thus, load-balancing and routing are performed simultaneously to achieve higher performance. The pseudo-code of the load-balancing algorithm is given in Algorithm 1.

Algorithm 1. Load Balancing Algorithm

```
Input:  IPv4 addresses of clients and servers
Output: Add Server with minimum load and Route with
        minimum delay to flow table.
Function LoadBalancingAlgorithm(client_ipv4,
serverList):
Load ← empty array declaration
  For server in serverList:
    serverIstatistic←ServerIstatisticCollect(ServerIpv4)
    Load←incomingPacket+ongoingPacket
  End for
lowestLoadServer←Min(load)
Route← bestRoute(lowestLoadServer)
addFlowTable(Route)
routeResponse(lowestLoadServer)
```

Table 2 shows the relationship between SSIM and MOS given in the study conducted by A.B. Letaifa et al. [10]. SSIM value is taken as the basis for determining QoE value. Although the PSNR value is used to measure the quality of transmitted images and video in most of the studies on images and video, the Structure Similarity Index Method is used in this study. SSIM accepts image distortion as a perception change. Here, the perceived quality of images and videos is estimated. It measures the similarity between the original image and the recorded image [11]. A high SSIM value indicates a better QoE value.

Table 2. SSIM Value - MOS - QoE Relationship

SSIM Value	**MOS**	**Quality**
>= 0.99	5	Perfect
[0.95, 0.99)	4	Good
[0.88, 0.95)	3	Acceptable
[0.55, 0.88)	2	Poor
<0.55	1	Bad

4. Findings

In our study, tests were performed using mininet simulation and network statistics were collected. The ffmpeg library, which is frequently used for video streaming, was used. Similarly, by using the ffmpeg library, the original video and the video sent to the target client were compared using SSIM and the SSIM score was obtained. In all scenarios, the delay between each node in the network was set to 25ms and the bandwidth was set to 10Mbit. Ten different tests are performed for each scenario.

Figure 5 shows the average SSIM values obtained according to the tests performed in each scenario. The worst results are obtained in each test in scenario 1, where no load balancing is performed. Although some tests in other scenarios produce below-average results, a significant improvement is observed in the average SSIM values compared to scenario 1.

Figure 6 shows the average Round Trip Time values according to the tests performed for each scenario. The worst average values are obtained in scenario 1. Although improvement is observed with load balancing in other scenarios, it is seen that there is an increase in the average RTT value in some tests. When Figure 4 is examined, it is seen that there is a decrease in the average SSIM value in parallel with this situation. When the Average RTT and SSIM values are examined together in Figure 5 and Figure 6, it can be said that there is an inverse relationship between these two values.

The SSIM value of the test in which Figure 7-a was obtained is 0.4695. According to Table 2, in this case, we can describe the obtained QoE as "bad". The SSIM value of the test in which Figure 7-b was obtained is 0.9975. According to Table 2, in this case, we can describe the obtained QoE as "excellent". When Figure 7-b is compared with Figure 7-c of the original video, the similarity between them will be seen. In this case, when a high SSIM value is obtained, we can talk about a better service and therefore a high QoE.

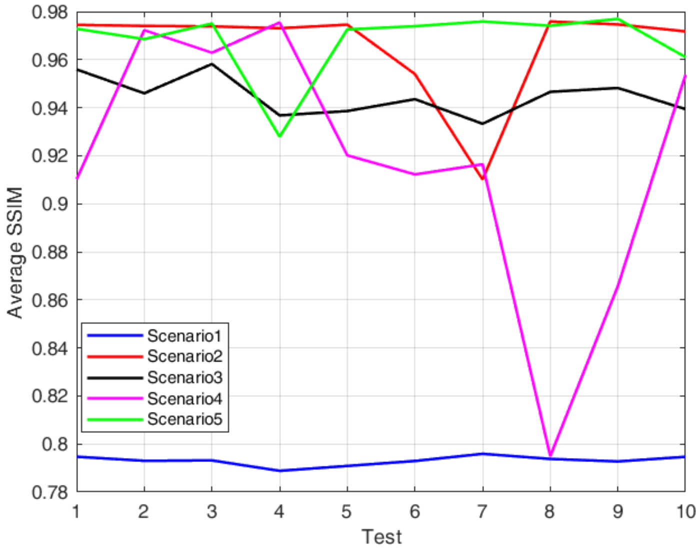


Figure 5. Average SSIM in Tests

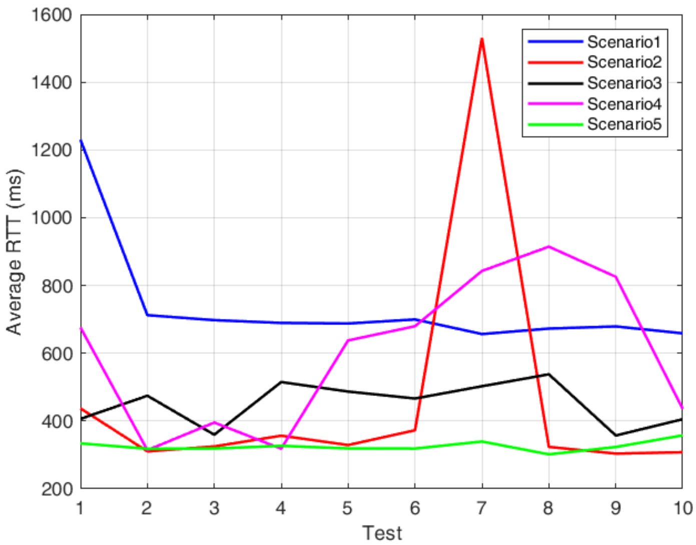
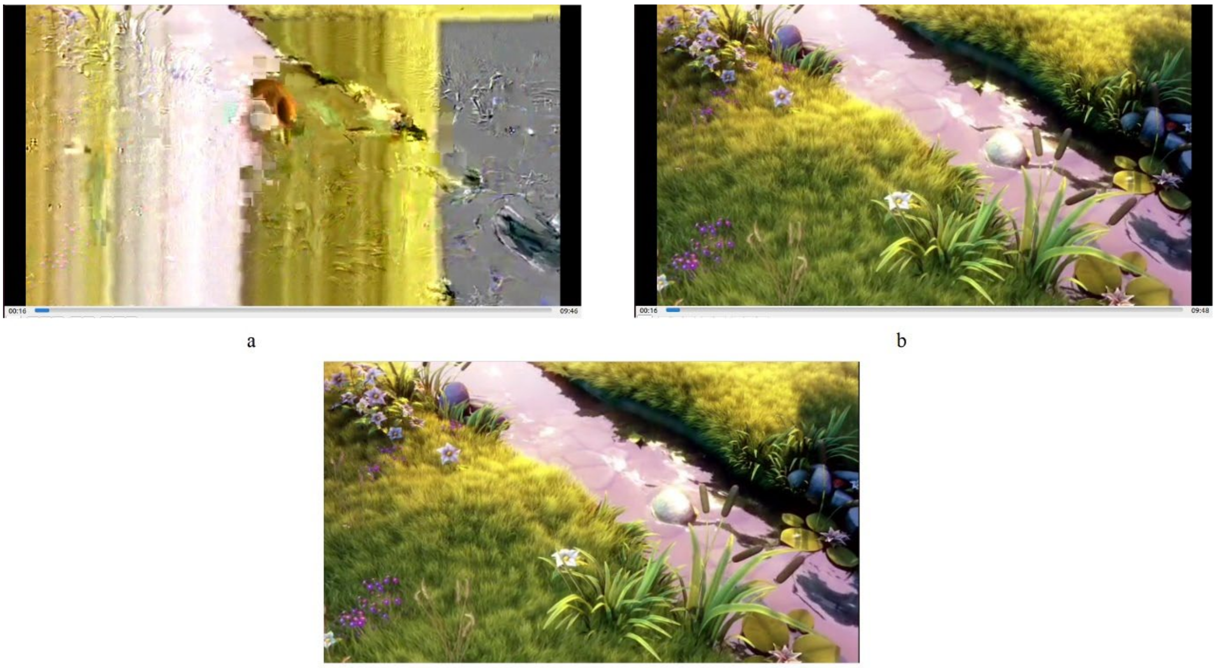


Figure 6. Average Round Trip Time in Tests



a b

c

Figure 7. Sample Screenshot
(Figure 7-a Low SSIM Value, Figure 7-b High SSIM Value Figure 7-c Original Video)

Table 3 shows the average SSIM values and packet losses of the clients for each scenario. As expected in Scenario 1 (no load balancing scenario), the h6 tagged client gives the worst performance when SSIM value and packet loss are examined. Although Scenario 2 (Round Robin is used for load balancing) and Scenario 4 (latency value is used for load balancing) generally perform better than Scenario 1, they remain below the average SSIM value for this scenario. As can be seen when Table 3 is examined, the h6 tagged client obtains the best SSIM average value in Scenario 3 (RTT value is used for load balancing), but there is no significant difference between it and Scenario 5 (Recommended load balancing method). When examined in general, it is seen that load balancing operations have a positive effect on SSIM. When examined in terms of packet loss, it is seen that the best results for the h6 tagged client are in Scenario 3 and Scenario 5, respectively. When the general average of the system is examined, Scenario 5 gave better results than Scenario 3. When examined over the general average of the system, Scenario 2 gave the best result. The second best result is in Scenario 5.

Table 4 shows the average latency value of each scenario and the average latency value of the h6-labeled client. Although the h6-labeled client has a latency value above average in each scenario, it achieved the best value in Scenario 5.

Table 5 shows the average Round Trip Time for each scenario and the average Round Trip Time for the h6 labeled client. Although the h6 labeled client performs better than average in all scenarios except Scenario 4, its best result is in Scenario 5.

Table 6 shows the average hop value for each scenario and the average hop value for the h6 labeled client. There is no significant difference in the system-wide scenarios 1 and 2. Scenario 3 is the scenario with the best result. When evaluated by the h6 labeled client, Scenario 5 has the best result.

Table 7 shows the average SSIM Value and QoE relation for each scenario and the h6 labeled client. When the system is examined, the best QoE is obtained in Scenario 2 and Scenario 5. When examined in terms of the test client labeled H6, the best QoE is obtained in Scenario 3 and Scenario 5. While the best QoE value is obtained in Scenario 5 for both the system as a whole and the test client, there is a difference between the system as a whole and the H6 labeled client in the other scenarios.

5. Discussion

In this study, the effects of load balancing on QoE in software-defined networks were investigated. The results obtained showed that the proposed algorithm was particularly effective in increasing the overall performance. It showed that the SSIM value could be used as a parameter in determining QoE. Technical parameters affecting QoE were examined and the relationship between the obtained results and these parameters was determined. On the other hand, user-based subjective parameters that could affect QoE were ignored in the study, and performance on large-scale networks could not be tested due to the limitations of the hardware used for simulation. Since a database was not used, a performance comparison could not be made with a load balancing study performed using machine learning algorithms.

This study showed the effectiveness of load balancing algorithms in improving QoE. However, testing the method used on larger networks will better reveal the effects of load balancing on QoE. Combining it with load balancing approaches based on machine learning algorithms may provide a better solution.

In future studies, a study can be presented in which a database is created in light of the results obtained in this study and load balancing operations are combined with machine learning algorithms.

Table 3. Average SSIM Values of Clients and Loss Value

Clients	Scenarios									
	Scenario 1		Scenario 2		Scenario 3		Scenario 4		Scenario 5	
	SSIM	Loss	SSIM	Loss	SSIM	Loss	SSIM	Loss	SSIM	Loss
h1	0.7708	21.85	0.9152	5.25	0.9370	4.8	0.9290	5.65	0.9198	8.65
h2	0.8170	16.2	0.9685	4.47	0.9343	5.57	0.9210	6.62	0.9724	3.35
h3	0.8476	12.77	0.9687	3.27	0.9273	5.65	0.8909	9.5	0.9764	3.17
h4	0.8712	9.35	0.9769	1.95	0.9719	2.0	0.9292	6.37	0.9756	3.3
h5	0.8864	6.4	0.9819	0.87	0.9199	7.4	0.8925	9.35	0.9659	3.15
h6	**0.6891**	**31.2**	**0.9418**	**7.82**	**0.9598**	**5.65**	**0.8500**	**16.05**	**0.9512**	**7.6**
h7	0.8412	10.15	0.9832	1.0	0.9296	6.22	0.90280	7.97	0.9734	2.97
h8	0.8050	14.05	0.9795	1.57	0.9309	6.02	0.9123	7.62	0.9693	2.75
h9	0.7742	17.77	0.9715	2.4	0.9884	0.5	0.9848	0.9	0.9793	2.47
h11	0.7529	20.7	0.9711	3.6	0.9348	5.47	0.9344	4.62	0.9791	2.85
h12	0.7368	23.25	0.9651	4.77	0.9229	6.52	0.9283	6.17	0.9779	2.625
h13	0.7240	25.5	0.9636	5.97	0.9781	1.67	0.9398	5.22	0.9741	3.35
Average	0.7930	17.43	0.9656	3.60	0.9446	4.79	0.9184	7.17	0.9678	3.85

Table 4. Average Delay

Scenarios	Average Delay (ms)	h6 Average Delay (ms)
Scenario 1	54.92	40.8
Scenario 2	62.8	42.4
Scenario 3	63.81	69.3
Scenario 4	73.25	51.2
Scenario 5	71.89	38.4

Table 5. Average Round Trip Time

Scenarios	Average Round Trip Time (ms)	h6 Average Round Trip Time (ms)
Scenario 1	735.41	297.28
Scenario 2	463.66	386.16
Scenario 3	451.60	303.56
Scenario 4	604.01	686.84
Scenario 5	326.00	196.28

Table 6. Average Hop Count

Scenarios	Average Hop Count	h6 Average Hop Count
Scenario 1	2.9	2
Scenario 2	2.89	2
Scenario 3	2.47	2.60
Scenario 4	2.54	1.60
Scenario 5	2.75	1.40

Table 7. Average SSIM Value - QoE

Scenarios	Average SSIM Value	QoE	h6 Average SSIM Value	QoE
Scenario 1	0.7930	2	0.6891	2
Scenario 2	0.9656	4	0.9418	3
Scenario 3	0.9446	3	0.9598	4
Scenario 4	0.9184	3	0.8500	2
Scenario 5	0.9678	4	0.9512	4

6. Conclusion

The relationship between load-balancing and QoE in the SDNs has been investigated in this study. Higher-quality video stream performance has been provided with server load-balancing, and the QoE is improved. In the tests, when the average SSIM value of the h6-labeled client in the proposed algorithm is examined, it is determined that there is an improvement in scenario 5. Also, there are improvements in the average delay, Round Trip Time, and hop counts of the h6-labeled client for testing purposes. It is seen that the workload of the server is an important criterion in load-balancing. When the SSIM values are examined, it is seen that there is a general improvement in performance with the load-balancing algorithm. A data set with network statistics using Scenario 5 in different network topologies will be created and load-balancing performed with the machine learning algorithm in future studies.

References

[1] X. Huang, M. Zeng, and K. Xie, "Intelligent traffic control for QoS optimization in hybrid SDNs," *Computer Networks*, vol. 189, Apr. 2021, doi: 10.1016/j.comnet.2021.107877.

[2] M. Cicioğlu and A. Çalhan, "Yazılım Tanımlı Ağlar – YTA," *Karaelmas Fen ve Mühendislik Dergisi*, vol. 7, no. 2, pp. 684–695, Jun. 2017.

[3] T. Abar, A. Ben Letaifa, and S. El Asmi, "Machine Learning Based QoE Prediction in SDN Networks," in *2017 13th International Wireless Communications and Mobile Computing Conference (IWCMC)*, IEEE, Jun. 2017, pp. 1395–1400.

[4] L. Cantor, "A word from Sandvine CEO," 2023. Accessed: Jan. 29, 2024. [Online]. Available: https://www.sandvine.com/global-internet-phenomena-report-2023

[5] Q. Wang, H. N. Dai, H. Wang, and D. Wu, "Data-driven QoE analysis on video streaming in mobile networks," in *Proceedings - 15th IEEE International Symposium on Parallel and Distributed Processing with Applications and 16th IEEE International Conference on Ubiquitous Computing and Communications, ISPA/IUCC 2017*, Institute of Electrical and Electronics Engineers Inc., May 2018, pp. 1115–1121. doi: 10.1109/ISPA/IUCC.2017.00168.

[6] Fortinet, "What is Quality of Service QoS in Network." Accessed: Apr. 21, 2023. [Online]. Available: https://www.fortinet.com/resources/cyberglossary/qos-quality-of-service

[7] A. Raake *et al.*, "Qualinet White Paper on Definitions of Quality of Experience," Novi Sad, Mar. 2013.

[8] International Telecommunication Union, "T-REC-P.800-199608." Accessed: Apr. 21, 2023. [Online]. Available: https://www.itu.int/rec/T-REC-P.800-199608-I

[9] Huawei mLab and iLab, "Technical White Paper on Mobile Bearer Network Requirements for Mobile Video Services Joint Release by Huawei mLab and iLab."

[10] A. Ben Letaifa, G. Maher, and S. Mouna, "ML based QoE enhancement in SDN context: Video streaming case," in *2017 13th International Wireless Communications and Mobile Computing Conference (IWCMC)*, IEEE, Jun. 2017, pp. 103–108. doi: 10.1109/IWCMC.2017.7986270.

[11] F. Artuğer and F. Özkaynak, "Görüntü Sıkıştırma Algoritmalarının Performans Analizi İçin Değerlendirme Rehberi," *International Journal of Pure and Applied Sciences*, vol. 8, no. 1, pp. 102–110, Jun. 2022, doi: 10.29132/ijpas.1012013.

[12] A. Kumar and D. Anand, "Load balancing for Software Defined Network using Machine learning," *Turkish Journal of Computer and Mathematics Education*, vol. 12, no. 2, pp. 527–535, Apr. 2021.

[13] B. Babayigit and B. Ulu, "Deep learning for load balancing of SDN-based data center networks," *International Journal of Communication Systems*, vol. 34, no. 7, May 2021, doi: 10.1002/dac.4760.

[14] G. S. Begam, M. Sangeetha, and N. R. Shanker, "Load Balancing in DCN Servers through SDN Machine Learning Algorithm," *Arab J Sci Eng*, vol. 47, no. 2, pp. 1423–1434, Feb. 2022, doi: 10.1007/s13369-021-05911-1.

[15] S. WilsonPrakash and P. Deepalakshmi, "Artificial Neural Network Based Load Balancing On Software Defined Networking," in *2019 IEEE International Conference on Intelligent Techniques in Control, Optimization and Signal Processing (INCOS)*, Tamilnadu, India: IEEE, Apr. 2019, pp. 1–4. doi: 10.1109/INCOS45849.2019.8951365.

[16] A. Filali, Z. Mlika, S. Cherkaoui, and A. Kobbane, "Preemptive SDN Load Balancing with Machine Learning for Delay Sensitive Applications," *IEEE Trans Veh Technol*, vol. 69, no. 12, pp. 15947–15963, Dec. 2020, doi: 10.1109/TVT.2020.3038918.

[17] S. Liang, W. Jiang, F. Zhao, and F. Zhao, "Load Balancing Algorithm of Controller Based on SDN Architecture Under Machine Learning," *Journal of Systems Science and Information*, vol. 8, no. 6, pp.

578–588, Dec. 2020, doi: 10.21078/JSSI-2020-578-11.
[18] Mininet.org, “Mininet.” Accessed: Dec. 25, 2023. [Online]. Available: mininet.org
[19] floodlight, “Floodlight Controller.” Accessed: Dec. 25, 2023. [Online]. Available: https://floodlight.atlassian.net/wiki/spaces/floodlightcontroller/overview
[20] Tracy L. LaQuey, “NSFNET,” in *The User’s Directory of Computer Networks*, Elsevier, 1990, pp. 247–250. doi: 10.1016/B978-1-55558-047-6.50032-1.

Article Information Form

Author(s) Contributions

Gürhan ÇOBAN: Conceptualization, Methodology, Writing – Original Draft, Software, Murtaza CİCİOĞLU: Methodology, Writing – review & editing, Ali ÇALHAN: Methodology, Writing – review & editing.

Acknowledgments

This work was supported by Bursa Uludağ University Scientific Research Projects Unit under Grants FGA 2023–1557

Ethical Approval and Informed Consent

It is declared that during the preparation process of this study, scientific and ethical principles were followed, and all the studies benefited from are stated in the bibliography.

Artificial Intelligence Statement

No artificial intelligence tools were used while writing this article.

Plagiarism Statement

This article has been scanned by iThenticate™.