A Modified Algorithm for Modelling SDN Data traffic

Haitham Abdelghany (habdelghany@outlook.com)
Mansoura University
Fayez Wanis Zaki
Mansoura University Faculty of Engineering
Mohammed M. Ashour
Mansoura University Faculty of Engineering

Research Article

Keywords: SDN, QoS, Queueing, MMPP, IP traffic model

Posted Date: November 22nd, 2023

DOI: https://doi.org/10.21203/rs.3.rs-3191219/v1

ORIGINAL RESEARCH

A Modified Algorithm for Modelling SDN Data traffic

Haitham M. Abdelghany 1, Fayez. W. Zaki 2, and Mohammed. M. Ashour 3

Abstract

Software Defined Networking (SDN) can allegedly maintain the quality of service (QoS) standards even in the presence of hostile flow, according to recent claims. Queueing theory, and more specifically network models, have been used for a very long time to examine the performance and QoS characteristics of networks in order to investigate this phenomenon. Because of the dependencies between the layers, planes, and components in an SDN architecture, the latter model seems especially well suited to represent the behavior of SDN. Numerous papers have described network models to examine the behavior of various network design applications. Here, we demonstrate how to employ the Markov-modulated Poisson process (MMPP) model to mathematically depict SDN traffic. Many articles had recommended utilizing MMPP to assess and model different types of IP network traffic, and it was widely used to simulate the traffic on traditional IP networks. We assert that MMPP can represent SDN traffic just as if it would in traditional IP networks. Our tests in this study indicate that MMPP is a useful technique for studying SDN data traffic. Starting with the premise that SDN traffic is averaged across multiple experiments and using two different SDN network topologies, we proceed. Emulation tests revealed that MMPP is a good model for SDN data traffic.

Keywords SDN, QoS, Queueing, MMPP, IP traffic model

✉ Haitham M. Abdelghany
habdelghany@outlook.com

Fayez. W. Zaki

fwzaki2017@gmail.com

Mohammed. M. Ashour
mohmoh2@yahoo.com

[1] Electronics and Communication Engineering Department, Faculty of Engineering, Mansoura University, El-Mansoura, Egypt

[2] Electronics and Communication Engineering Department, Faculty of Engineering, Mansoura University, El-Mansoura, Egypt

[3] Electronics and Communication Engineering Department, Faculty of Engineering, Mansoura University, El-Mansoura, Egypt

1. Introduction

The control plane and data plane were both integrated in every device in traditional IP networking systems, allowing each device to choose its own forwarding. The fundamental issue with traditional IP networks is that dispersing the control plane could result in a wide range of issues when the network configuration is upgraded or changed. Software Defined Networking (SDN), on the other hand, separated the control plane from the network into a centralized unit called the controller, which is in charge of making choices on network forwarding while the other devices function as forwarders. In SDN, the controller receives the intelligence of the networks, in other words [1].

Every network is composed of three basic planes: the data plane, the control plane, and the management plane. The network administrator for monitoring, setting up, and troubleshooting [2] uses the management plane. The controller used the control plane to communicate the choice regarding the forwarding of data to other network devices. The controller makes packet control choices in the control plane and sends them to other network nodes. The data plane performs the forwarding of the control plane. The controller uses Open-Flow, the most significant southbound protocol, to interact with forwarders, send data to other devices, and make decisions [3,4].

Because it is a centralized node that handles Packet-In messages from many switches, the controller frequently acts as a bottleneck, delaying table-miss packets. It is difficult to predict the precise frequency of SDN controller overloads in the real world because it depends on traffic volume and the number of switches the controller is in charge of managing. However, several earlier studies [5–14] have shown that an overloaded controller is an issue that has to be solved. Reduced controller load has been the focus of three categories of prior research, including packet scheduling [15,16], multi-controller design [6–10], [17–19], and load reduction [13–14,19,20]. By identifying the difference between traffic from the controller and traffic from other switches, the switches give traffic from the controller priority over traffic from other switches. Since the forwarding rules may be applied to the flow table quickly, the controller receives fewer packets with table-misses. A distributed control-plane design with many load-sharing controllers is known as a multi-controller architecture. Finally, the switch will offer a number of features to reduce the number of Packet-In messages transmitted to the controller in the load reduction area.

These studies aimed to lighten the load on controllers, however many Packet-In messages are still superfluous. This study use the Markov Modulated Poisson Process (MMPP) to model SDN in order to address this problem.

Models of straightforward and precise networking systems are scarce. Even though experts concur that network traffic is not Poisson at any level of aggregation, they disagree on the best methods for describing traffic characteristics. In this paper [21], we provide a method for simulating traffic in SDN using an MMPP representation.

Because it qualitatively captures the time-varying arrival rate and some of the underlying connections between the inter arrival periods while staying analytically tractable, MMPP has been used extensively to describe networking systems. Naor and Yechiali were the first to employ the MMPP in queueing theory, followed by Neuts. Since then, numerous top-notch studies and publications have addressed the problem. [22] contains a few general references.

2. Related Work

Jarschel et al. [23] provide the first reported analytical modelling of SDN using a queueing network model. Without taking into account any particular SDN application, the study focuses on describing the relationship between the SDN switch and the controller. Similar to this, researchers in [24]–[27] use queueing network models to look into various SDN-related topics. [24] uses queueing network models to examine the trade-offs between different buffer sharing strategies. The authors of [25] propose a queueing network model to measure the performance of hardware and software switches in SDN. Using a queueing network model, [26] offers a performance analysis of SDN switches. The authors of [27] also use a queueing network model to examine the effectiveness of SDN with network virtualization functions included into or separate from the controller.

In their paper [28], Yen and Su investigate SDN-based cloud computing architecture using a queueing network model. They demonstrate that the M/M/1 (Kendall's notation) queue-based queueing network model is suitable for simulating the operation of an SDN-based cloud computing architecture. They also show how the proposed SDN-based cloud computing architecture can guarantee the QoS of cloud services. In [29] 4, Chowdhary and Huang use a queueing network model to examine SDN-based network function parallelism in the cloud. Using an M/M/c queue to optimise service function allocation for each service function chain, they demonstrate how service functions with independent action sets can be parallelized to decrease performance overhead.

Ansell et al. [30] provide a solution for forecasting network performance that is based on queueing analytical models and integrated with real-time monitoring. It gives you the ability to examine the effects of changes in traffic load and connection utilization on performance. Muhizi et al. [31] assess the efficacy of SDN using queuing network models to monitor changes in packet processing latency under various parameter settings. Shang et al. [32] model the packet processing delay of SDN switches and controllers. They mainly focus on the impact of message packets on performance. Wang et al. [33] to assess the throughput and delay of the control plane use queuing theory. On throughput and delay, the effect of the number of switches and threads is looked at. Mahmood et al. [34] represent the controller as an M/M/1 queue with either an infinite or a finite buffer. Jackson networks are utilized to define the data plane the queueing estimate model of Haiyan et al. [35] uses queueing theory to represent SDN switches from two perspectives for end-to-end delay analysis. It includes a shared buffer for control plane and data plane traffic and a buffer with two priority queues separating the two. Fahmin et al. [37] to address performance concerns combine SDN and Network Functional Virtualization (NFV). They want to use NFV to mimic SDN with or without the need of a controller. The performance evaluation tool of choice is the M/M/1 queuing model.

Our study is comparable to the performance modelling and analysis of SDN under bursty multimedia traffic [38]. The Markov modulated Poisson process (MMPP) is used by the authors to examine the performance of SDN in the presence of bursty and correlated arrivals. They assume that the packet departure process from the MMPP queue is MMPP in order to provide a tractable analytical model. In contrast to them, software-defined industrial control networks take into consideration network traffic. The results of [39] are utilized to enhance the control plane's packet arrival process model. We also examine the impact of hostile flow on network traffic in software-defined industrial control networks.

We analyses the behavior of network traffic in software-defined industrial control networks using a queueing network model, in contrast to all of the prior research. In industrial control networks with strong periodic patterns of network traffic, we employ MMPP to approximate the arrival process. We also use the findings from [39] to create a plausible description of the link between the data plane and the control plane. The influence of adversarial flow in software defined industrial control networks is then examined using the analytical model.

Any network's design, analysis, and resource planning all depend heavily on traffic modelling and characterization [3]. The following provides a brief survey of the literature on network traffic modelling. The Batch Markovian arrival process is used in Klemm et al.'s new analytical traffic model for the Universal Mobile Telecommunication System (UMTS) [4]. Bozidar Vujicic et al. described the traffic in wireless networks used for public safety. Because of their long-range dependence, they used Weibull and Gamma distributions to represent the statistical distribution for call inter arrival times. Additionally, call holding durations have been modelled using networks with lognormal distribution [5]. Using their statistical features, Will E. Leland et al. showed that Ethernet traffic is self-similar [6]. Shifted gamma distribution is thought to be the best model for describing internet traffic, according to S. Kim et al. [7]. Generalized Extreme Value (GEV) distribution was used by S Bothe et al. to statistically characterize the cellular network traffic and apply it for effective resource planning and optimization [8]. Xin Chen et al. utilizing SDN architecture [9] have examined the Narrow Band Internet of Things (NB-IoT) performance. In their research, the input models for synchronized access and non-synchronized access scenarios, respectively, were Beta and Uniform distributions. The majority of SDN analytical modelling has taken into account the traffic's Poisson structure, allowing them to use classical queueing theory [10], [11], [12]. However, it remains to be verified whether the Poisson approximation is valid in SDN. Therefore, a thorough study and the characterization of traffic in SDN environment is still an open problem. The aim of the paper is to model and characterize the traffic in SDN environment. The contributions of the paper is as follows.

1) We have simulated SDN environment for different topologies and collected data traffic traces for each of them.

2) The best-fit parameter values for different distributions that closely match the inter arrival time distribution of the SDN data traffic is found.

3. Fitting Algorithms for SDN Traffic Models

We suggest and investigate a fitting technique for SDN traffic models in this work. Markov Modulated Poisson Processes (MMPP) are the foundation of the suggested technique for the traffic-fitting model. The suggested algorithm takes into account the measured packet arrival rates over long enough time intervals to model the SDN traffic using three different network topologies.

3.1 Fitting based on cumulative distribution function

The defining parameters of the MMPP(2) model are

$$Q = \begin{pmatrix} -r_1 & r_1 \\ r_2 & -r_2 \end{pmatrix}; \Lambda = \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix} = diag(\lambda)$$

$$p = \frac{1}{\lambda_1 r_2 + \lambda_2 r_1} \begin{pmatrix} \lambda_1 r_2 & \lambda_2 r_1 \end{pmatrix}$$

where Q represents the infinitesimal generator, Λ is the matrix of the Poisson arrival rates and p is the initial probability vector of the underlying Markov process.

The proposed fitting algorithm for SDN traffic model is based on the cumulative distribution of the arrival rate.

The MMPP (2) model is considered as a reference model for the proposed SDN traffic model algorithm

Moreover we studied and evaluated traffic models for MMPP (2), MMPP (3), MMPP (5).

The proposed algorithm for SDN traffic can be used in the characterization of traffic streams and in more generalized applications like interference modeling in wireless communication environment

.

$$A = \Lambda - Q$$

$$F(x) = \left(\int_0^x e^{-Au} du\right)\Lambda \qquad p = \frac{\pi_{ss}\Lambda}{\pi_{ss}\lambda}$$

$$F(x) = \left(I - e^{-Ax}\right)A^{-1}\Lambda \qquad \lambda = \begin{pmatrix}\lambda_1 \\ \lambda_2\end{pmatrix}$$

$$F(\infty) = A^{-1}\Lambda$$

$$\pi_{ss} = (\pi_1 \quad \pi_2)$$

$$\pi_{ss} e = 1; \pi_{ss} Q = 0$$

$$F_c(x) = p e^{-Ax} A^{-1}\Lambda e$$

$$e = \begin{pmatrix}1 \\ 1\end{pmatrix}$$

In this part we assumed a MMPP(2) processes. Let Xi represents the interarrival time between the i th and (i+1)th packets. In this case, the distribution of the interarrival time Xi is a second order hyperexponential distribution (H2), with complementary CDFand PDF as follows:

$$F_c(x) = p e^{-Ax} A^{-1}\Lambda e$$

$$H_2(u_1, u_2, q) = q e^{-u_1 x} + (1-q) e^{-u_2 x}$$

$$pdf = f_X(x) = -\frac{dF_c}{dx} = q u_1 e^{-u_1 x} + (1-q) u_2 e^{-u_2 x}$$

4 Performance Analysis and Results

4.1 Experiment 1

In the first experiment, we compare Mininet simulation and MMPP modelling to the efficiency of MMPP in simulating SDN with a linear topology. The CDF of the transfer is recognized. Matlab was used to run the simulation. Figure 3 shows the analytical model for the network simulation for the linear topology (linear four) utilizing D-MMPP as well as the CDF. Figure 2 displays the transmission of the measured traffic. Figure 4 displays the PDF for the transfer using D-MMPP in both the emulation and analytical modes. The results are highlighted by the fact that the CDF and PDF of the analytical model and the emulation are essentially similar. The pdf deviates greatly from the true SDN traffic because we employed five states in the model to imitate the real traffic, which had only three transition phases.

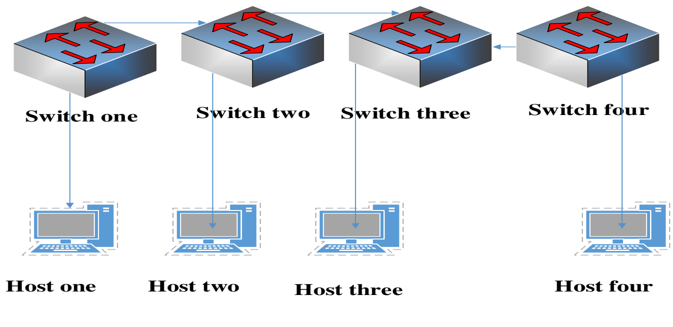


Figure 1 the Linear Topology (linear four).

The three parameters of the hyperexponential distribution, u1, u2 and q, can be related with the MMPP(2) parameters by []:

$$2u_1 = k - \delta; 2u_2 = k + \delta$$

$$k = \lambda_1 + \lambda_2 + r_1 + r_2$$

$$q u_1 + (1-q) u_2 = \frac{\lambda_1^2 r_2 + \lambda_2^2 r_1}{\lambda_1 r_2 + \lambda_2 r_1}$$

$$\delta = \sqrt{(\lambda_1 - \lambda_2 + r_1 - r_2)^2 + 4 r_1 r_2}$$

The auto-covariance function, C[k], 1 k ≥ , is given by:

$$C[k] = E[(X_1 - E[X_1])(X_2 - E[X_2])]$$

$$C[k] = pA^{-2}\Lambda\left(\left(A^{-1}\Lambda\right)^{k+1} - ep\right)A^{-2}\Lambda e$$

$$C[k] = pA^{-2}\Lambda\left(\left(A^{-1}\Lambda\right)^{k+1} - ep\right)A^{-2}\Lambda e = B\sigma^k$$

$$B = \frac{(\lambda_1 - \lambda_2)^2 r_1 r_2}{(\lambda_1 r_2 + \lambda_2 r_1)^2 (\lambda_1\lambda_2 + \lambda_1 r_2 + \lambda_2 r_1)}$$

$$\sigma = \frac{\lambda_1 \lambda_2}{(\lambda_1\lambda_2 + \lambda_1 r_2 + \lambda_2 r_1)}$$

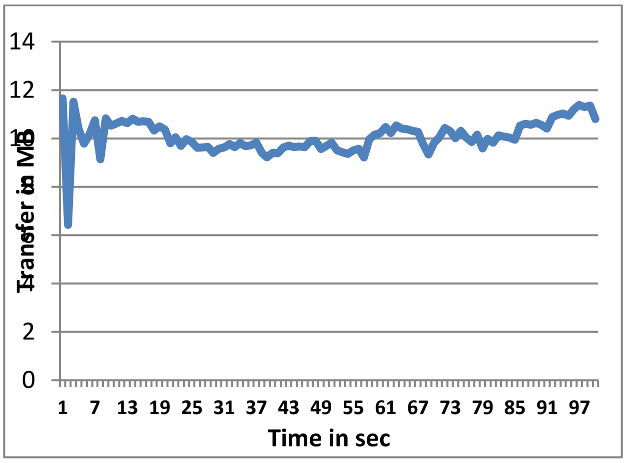


Figure 2 The transfer in MB for the measured traffic

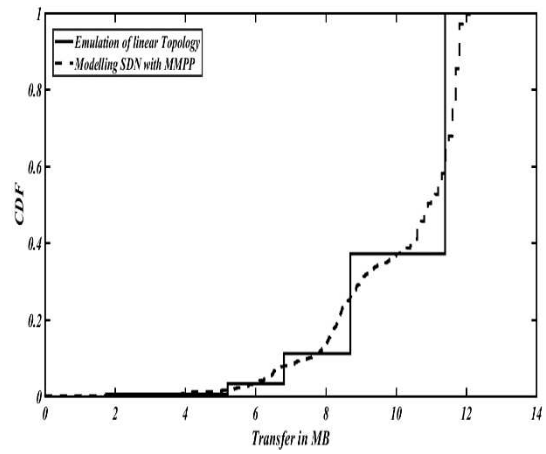


Figure 3 the CDF for transfer in MB for the emulation of linear topology and MMPP modelling

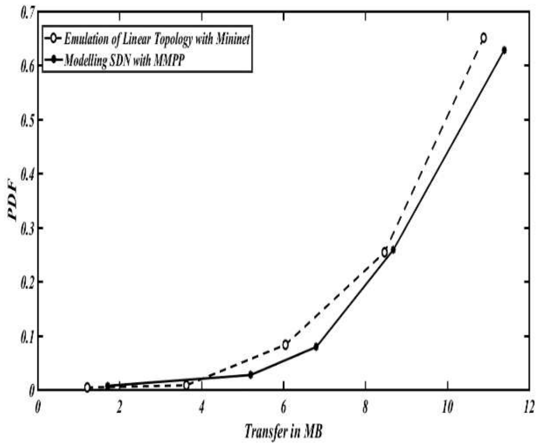


Figure 4 the PDF for transfer in MB for the emulation of linear topology and modelling SDN with MMPP

$$Q = \begin{bmatrix} -1 & 0 & 0.1429 & 0.5714 & 0.2857 \\ 0 & -0.9259 & 0.1481 & 0.4074 & 0.3704 \\ 0 & 0.038 & -0.9114 & 0.3797 & 0.4937 \\ 0.0193 & 0.0656 & 0.1583 & -0.7452 & 0.5019 \\ 0.0032 & 0.008 & 0.0414 & 0.2357 & -0.2882 \end{bmatrix} \quad (10)$$

$$\text{Times in state} = [7 \quad 27 \quad 79 \quad 259 \quad 628] \quad (11)$$

$$\pi_{ss} = \begin{bmatrix} 0.007 \\ 0.027 \\ 0.079 \\ 0.259 \\ 0.628 \end{bmatrix} \quad (12)$$

4.2 Experiment 2

We provide more thorough experimental data in this subsection to show how well D-MMPP models SDN. during the course of 100 seconds. The transfer's CDF is calculated. The simulation has been done through Matlab. Figures 6 and 7 illustrate the CDF and PDF for the transfer using D-MMPP in the emulation and analytical modes, respectively, for the network for the linear topology (linear 6) presented in Figure 5's simulation of the network and the analytical model. The CDF and PDF of the emulation and the analytical model are identical, which sums up the results. Five states are used in both the mathematical model and simulation, thus, it is important to understand that the CDF and the PDF are identical.

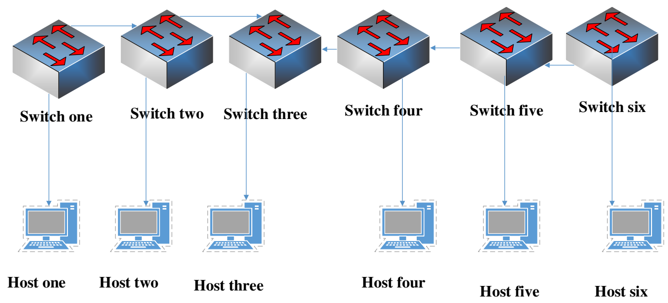


Figure 5 the Linear Topology (linear six).

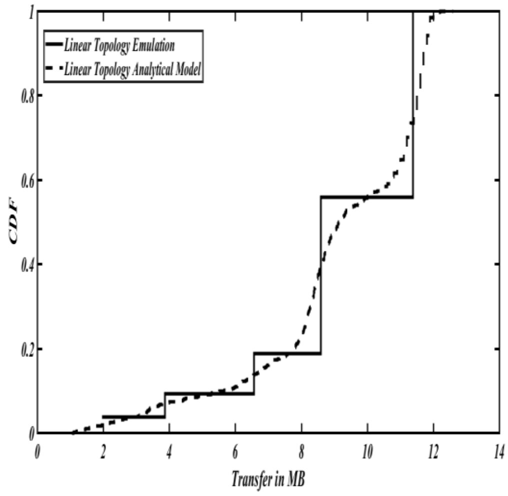


Figure 6 the CDF for transfer in MB for the emulation of linear topology and the analytical model.

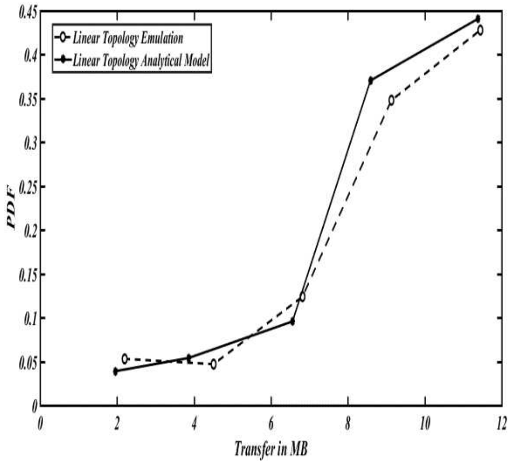


Figure 7 the PDF for transfer in MB for the emulation of linear topology and the analytical model.

$$Q = \begin{bmatrix} -0.9474 & 0.0526 & 0.7105 & 0.0789 & 0.1053 \\ 0 & -0.9815 & 0.1852 & 0.3889 & 0.4074 \\ 0 & 0.0316 & -0.8842 & 0.2632 & 0.5895 \\ 0.097 & 0.1078 & 0.0728 & -0.7574 & 0.4798 \\ 0.0023 & 0.0181 & 0.0452 & 0.5249 & -0.5905 \end{bmatrix} \quad (13)$$

$$\text{Times in state} = [38 \quad 54 \quad 95 \quad 371 \quad 442] \quad (14)$$

$$\pi_{ss} = \begin{bmatrix} 0.039 \\ 0.054 \\ 0.0957 \\ 0.3705 \\ 0.448 \end{bmatrix} \quad (15)$$

4.3 Experiment 3

In this section, we present additional detailed experimental data to demonstrate how accurately D-MMPP predicts SDN. The CDF of the transfer is determined. Matlab has been used to perform the simulation. The CDF for the network simulation for the linear topology (shown in Figure 8) and the analytical model utilizing D-MMPP are both displayed in Figure 9. Figure 10 displays the PDF for the transfer using D-MMPP in both the emulation and analytical modes. The results are summarized by the fact that the CDF and PDF of the analytical model and the emulation are essentially identical. Even though the model employs five stages to simulate SDN whereas real SDN traffic has seven transition states, the CDF and PDF are nearly equal since the flow is the same after the fifth transition.

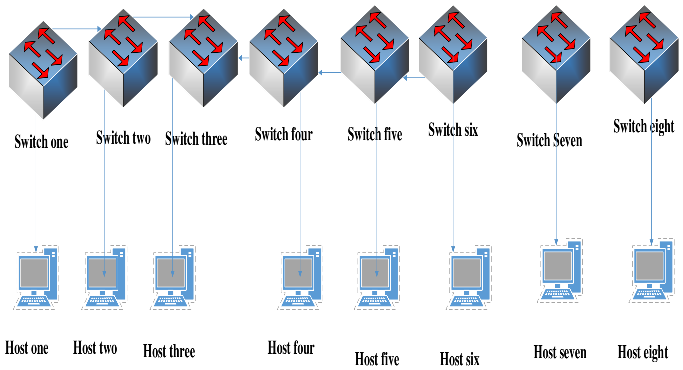


Figure 8 the Linear Topology (linear 8).

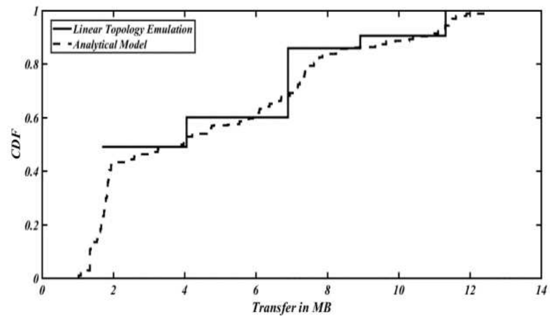


Figure 9 the CDF for transfer in MB for the emulation of linear topology and the analytical model.

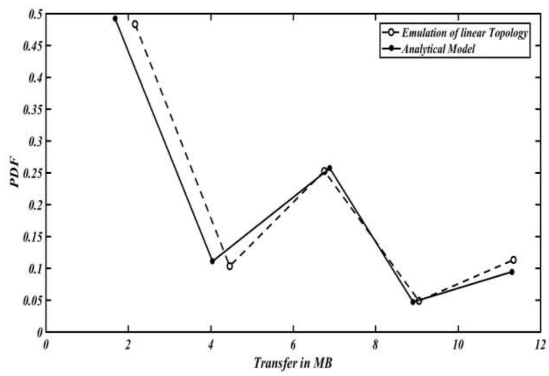


Figure 10 the PDF for transfer in MB for the emulation of linear topology and the analytical model.

$$Q = \begin{bmatrix} -0.0256 & 0.0064 & 0.0128 & 0.0064 & 0 \\ 0.0268 & -0.0893 & 0.0357 & 0.0179 & 0.0089 \\ 0.0263 & 0.0188 & -0.0639 & 0.0113 & 0.0075 \\ 0.0612 & 0.0408 & 0.0612 & -0.2245 & 0.0612 \\ 0 & 0 & 0.0351 & 0.0263 & -0.0614 \end{bmatrix} \quad (16)$$

Times in state = [469 112 266 49 114] (17)

$$\pi_{ss} = \begin{bmatrix} 0.4917 \\ 0.1106 \\ 0.2567 \\ 0.0467 \\ 0.0941 \end{bmatrix} \quad (18)$$

0.1983*exp(-1.825*x) + 0.3935*exp(-7.064*x) + 0.217*exp(-11.06*x) + 0.09144*exp(-9.224*x) + 0.09976*exp(-4.089*x)

4.4 Experiment 4

In order to make sure that MMPP is successful in simulating SDN traffic in this experiment, a tree topology is adopted (as seen in Figure 11). To show that MMPP is the best option for simulating SDN traffic, the CDF and the PDF for both the emulation and the analytical model are measured. Figures 12 and 13 respectively show the CDF and PDF for the analytical model using MMPP and the emulation. They are similar. Because we used five states in the model to simulate the real traffic, which only had three transition phases, the pdf deviates substantially from the true SDN traffic.

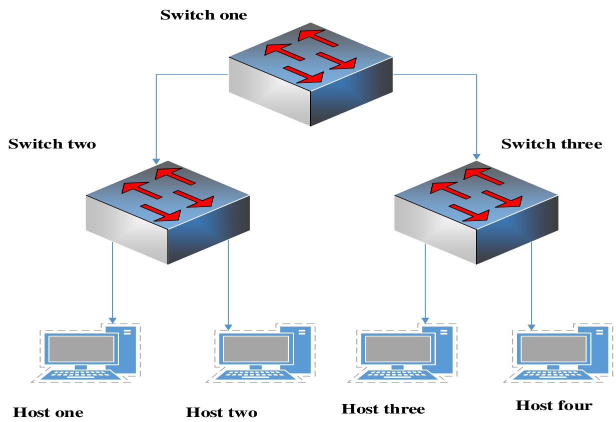
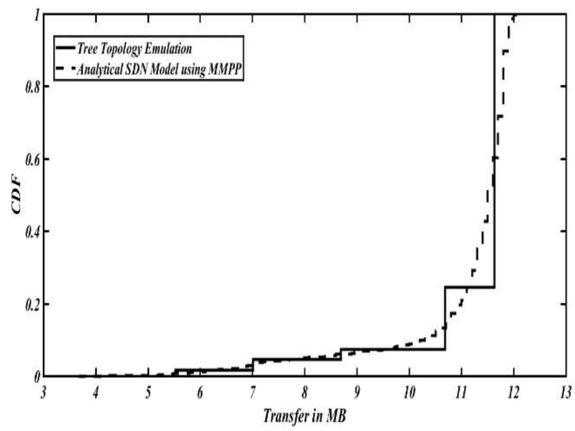


Figure 11 the Tree Topology (two)



.

Figure 12 the CDF for transfer in MB for the emulation of tree topology and the MMPP model

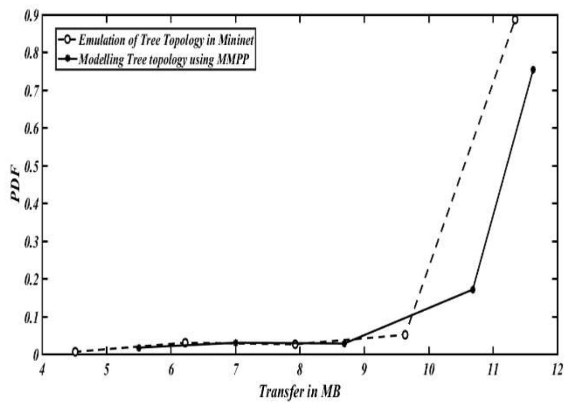


Figure 13 the PDF for transfer in MB for the emulation of tree topology and the MMPP model(19) Q =

$$\begin{bmatrix} -0.7059 & 0.058 & 0 & 0.1765 & 0.4706 \\ 0.1333 & -0.9667 & 0.1333 & 0.0333 & 0.667 \\ 0 & 0 & -0.9286 & 0.1071 & 0.8214 \\ 0.0117 & 0.0292 & 0.0234 & -0.8363 & 0.7719 \\ 0.008 & 0.305 & 0.0239 & 0.1804 & -0.2427 \end{bmatrix}$$

(20) Times in state $= [17 \quad 39 \quad 28 \quad 171 \quad 754]$

() rate vector

(21) $$\pi_{ss} = \begin{bmatrix} 0.017 \\ 0.03 \\ 0.028 \\ 0.1710 \\ 0.754 \end{bmatrix}$$

4.5 Experiment 5

In this experiment, MMPP is utilized to successfully replicate SDN traffic using a tree architecture (as illustrated in Figure 11). We measure the CDF and PDF for the emulation and the analytical model to show that MMPP is the best solution for emulating SDN traffic. Figures 12 and 13 respectively, which are comparable, illustrate the CDF and PDF for the analytical model using MMPP and the emulation. Since the real traffic only had three transition phases, we used five states in the model to simulate it, which greatly deviates from the true SDN traffic.

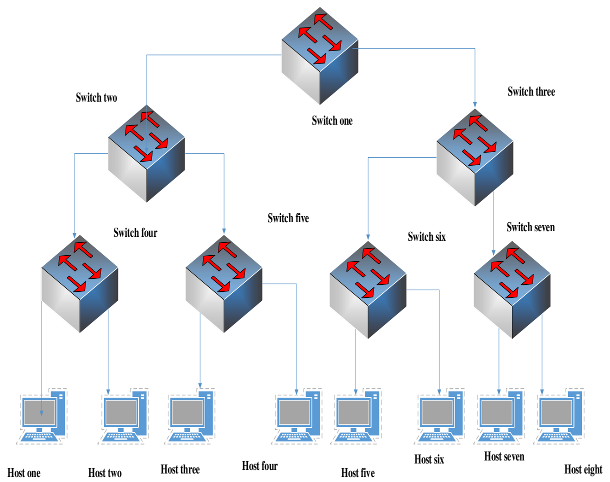


Figure 14 the Tree Topology.

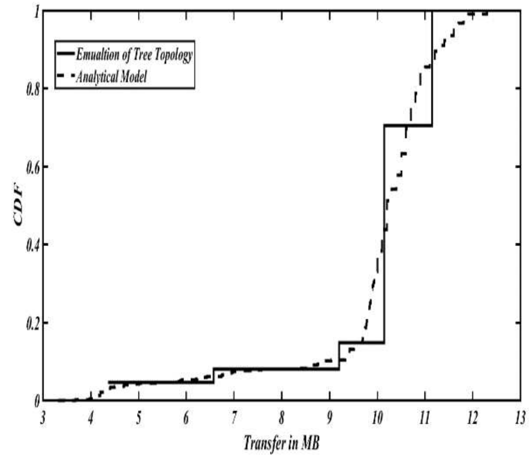


Figure 15 the CDF for transfer in MB for the emulation of tree topology and the analytical model.

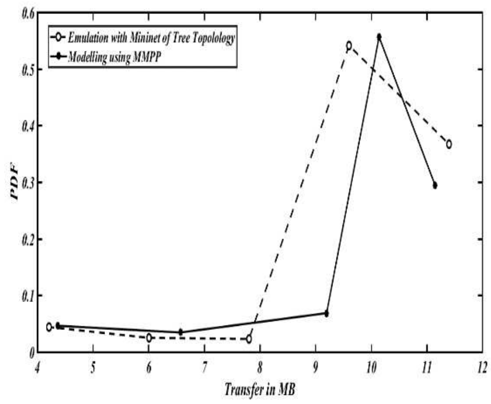


Figure 16 the PDF for transfer in MB for the emulation of tree topology and the analytical model.

Q =

$$\begin{bmatrix} -0.4783 & 0 & 0.0435 & 0.3696 & 0.0652 \\ 0 & -1 & 0.0294 & 0.8235 & 0.1471 \\ 0.029 & 0 & -0.1449 & 0.087 & 0.029 \\ 0.0293 & 0.0548 & 0.0055 & -0.1115 & 0.0219 \\ 0.0132 & 0.0132 & 0.0132 & 0.0362 & -0.0757 \end{bmatrix}$$

(22)

Times in state $= [469 \quad 112 \quad 266 \quad 49 \quad 114]$ (23)

$$\pi_{ss} = \begin{bmatrix} 0.0463 \\ 0.0344 \\ 0.0686 \\ 0.5563 \\ 0.2943 \end{bmatrix} \qquad (24)$$

5 Conclusion

In this paper, D-MMPP was used to simulate SDN and study the transfer in MB, which is related to arrival rate. To enable the controller to properly configure the switch in MMPP, the switch just sends the controller the first table-miss packet of each flow. The successive packets of the flow belonging to the same flow are temporarily stored in the switch until the correct flow entry is added to or modified in the flow table. D-MMPP reduces the number of packets sent to the controller, which lightens the strain on the controller and significantly reduces table-miss packet delay.

We used a simulation to test D-MMPP, using one controller and several switches from two different research. The SDN network should have a lot of switches in a real-world scenario, each with a unique set of dependencies. The controller not only sets the flow entry of the sending switch when a switch sends a table-miss packet to it as a Packet-In message, but it may also set the flow entries of the other switches on the path. In order to get results that are more convincing in the future, it is recommended to use emulation to build a more realistic environment that is made up of one controller and several switches. Allow the packets to actually travel through this false controller and the switch as well.

In simulation and modelling, we ignore the realistic architecture of a physical switch and, instead, simulate a switch as a straightforward processing and forwarding node, as most prior efforts have done. A physical Open-Flow switch's processing for entering and leaving has many flow tables. The switch has a large number of queues as well,

[1] D. Kreutz, F. M. V. Ramos, P. E. Verissimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, “Software-Defined Networking: A Comprehensive Survey”, Proceedings of the IEEE, 103(1), 14-76, Jan. 2015.

[2] Y. Jarraya, T. Madi, and M. Debbabi, “A survey and a layered taxonomy of software-defined networking,” Communications Surveys Tutorials, IEEE, vol. PP, no. 99, pp. 1–1, 2014.

[3] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, “OpenFlow: enabling innovation in campus networks,” SIGCOMM Comput. Commun. Rev., vol. 38, no. 2, pp. 69–74, Mar. 2008.

[5] A. Lara, A. Kolasani, and B. Ramamurthy, “Network innovation using OpenFlow: A survey,” Communications Surveys Tutorials, IEEE, vol. 16, no. 1, pp. 493–512, First 2014.

[6] S.-C.- Lin, P. Wang, I.F. Akyildiz, M. Luo, “Towards Optimal Network Planning for Software-Defined Networks”, IEEE Transactions on Mobile Computing, 2018.

[7] T. Wang, F. Liu, J. Guo, and H. Xu. Dynamic sdn controller assignment in data center networks: Stable matching with transfers. In Proc. IEEE INFOCOM, 2016.

[8] T. Wang, F. Liu, and H. Xu, “An efficient online algorithm for dynamic SDN controller assignment in data center networks,” IEEE/ACM Transactions on Networking, vol. 25, no. 5, pp. 2788–2801, 2017.

[9] N. Correia and F. AL-Tam, “Flow setup aware controller placement in distributed software-defined networking,” IEEE Syst. J., pp. 1–4, 2019.

[10] F. Al-Tam and N. Correia, ‘‘On load balancing via switch migration in software-defined networking,’’ IEEE Access, vol. 7, pp. 95998–96010, 2019.

[11] S. Tariq and M. Bassiouni, “Packet-In Message Control for Reducing CPU Load and Control Traffic in OpenFlow Switches,” In Proceedings of IEEE CCNC, January 2015.

[12] D. Singh, B. Ng, Y.-C. Lai, Y.-D. Lin, and W. K. G. Seah, ‘‘Analytical modelling of software and hardware switches with internal buffer in software-defined networks,’’ J. Netw. Comput. Appl., vol. 136, pp. 22–37, Jun. 2019.

including several queues for various output ports and numerous queues for allocating packet priority between ingress and egress processing.

Funding The authors did not receive support from any organization for the submitted work.

Data Availability Statement The raw data elaborated during the current research are publicly available.

Declarations

Conflict of interest The author declares that he has no conflict of interest.

Ethical approval This article does not contain any studies with human participants performed by any of the authors.

References

[13] A. A. Pranata, T. S. Jun, and D. S. Kim, “Overhead reduction scheme for sdn-based data center networks,” Computer Standards & Interfaces, vol. 63, pp. 1–15, 2019.

[14] A. Fahmin, Y. Lai, M. Hossain, Y. Lin, “Performance modelling and comparison of NFV integrated with SDN: under or aside?,” Journal of Network and Computer Applications, vol. 113, no. 1, pp. 119-129, July 2018.

[15] Z. Shang, H. Wu, and K. Wolter. Buffer management for reducing packet-in messages in OpenFlow networks. In 12th International Conference on Communication Software and Networks. IEEE, 2019.

[16] W. Miao, G. Min, Y. Wu, and H. Wang, “Performance Modelling of Preemption-Based Packet Scheduling for Data Plane in Software Defined Networks,” in IEEE International Conference on Smart City/SocialCom/SustainCom (SmartCity), Chengdu, China, pp. 60–65, 2015.

[17] D. Singh, B. Ng, Y.-C. Lai, Y.-D. Lin, and W. K. G. Seah, “Modelling software-defined networking: Software and hardware switches,” Journal of Computer Network and Computer Applications, vol. 122, pp. 24–36, 2018.

[18] T. Koponen, M. Casado, N. Gude, and J. Stribling, “Distributed control platform for large-scale production networks,” US Patent 8,830,823, 2014.

[19] A. Tootoonchian and Y. Ganjali, “HyperFlow: a distributed control plane for OpenFlow,” in Proceedings of the 2010 internet network management conference on Research on enterprise networking, ser. INM/WREN’10. Berkeley, CA, USA: USENIX Association, pp. 3–3, 2010.

[20] B. P. R. Killi and S. V. Rao, “Controller placement in software defined networks: A comprehensive survey,” Computer Networks, vol. 163, p. 106883, 2019.

[21] M. Yu, J. Rexford, M. J. Freedman, and J. Wang, “Scalable flow-based networking with difane,” SIGCOMM Comput. Commun. Rev., vol. 41, no. 4, pp. –, Aug. 2010.

[22] A. R. Curtis, J. C. Mogul, J. Tourrilhes, P. Yalagandula, P. Sharma, and S. Banerjee, “DevoFlow: scaling flow management for highperformance networks,” Comput. Commun. Rev., vol. 41, no. 4, pp. 254–265, Aug. 2011.

[23] M. Jarschel, S. Oechsner, D. Schlosser, R. Pries, S. Goll, and P. TranGia, “Modeling and performance

evaluation of an OpenFlow architecture," in Teletraffic Congress (ITC), 2011 23rd International, Sept 2011, pp. 1–7.

[24] W. Fischer and K. Meier-Hellstern, "The Markov-modulated Poisson process (MMPP) cookbook," Perf. Eval., vol. 18, pp. 149–171, 1992.

[25] M. Jarschel, S. Oechsner, D. Schlosser, R. Pries, S. Goll, and P. Tran-Gia, "Modeling and performance evaluation of an OpenFlow architecture," in Proc. 23rd Int. Teletraffic Congress (ITC), Sep. 2011, pp. 1–7. [Online]. Available: https://ieeexplore.ieee.org/document/6038457

[26] D. Singh, B. Ng, Y. Lai, Y. Lin, andW. K. G. Seah, "Modelling software defined networking: Switch design with finite buffer and priority queueing," in Proc. IEEE 42nd Conf. Local Computer Networks (LCN), Oct. 2017, pp. 567–570.

[27] D. Singh, B. Ng, Y. Lai, Y. Lin, and W. K. G. Seah, "Modelling software defined networking: Software and hardware switches," J. Network and Computer Applications, vol. 122, pp. 24–36, 2018.

[28] T. Yen and C. Su, "An sdn-based cloud computing architecture and its mathematical model," in Proc. Electronics and Electrical Engineering 2014 Int. Conf. Information Science, vol. 3, Apr. 2014, pp. 1728–1731.

[29] A. Chowdhary and D. Huang, "Sdn based network function parallelism in cloud," in Proc. Networking and Communications (ICNC) 2019 Int. Conf. Computing, Feb. 2019, pp. 486–490.

[30] Ansell, J., Seah, W.K.G., Ng, B., et al.: Making queueing theory more palatable to SDN/OpenFlow-based network practitioners. In: Network Operations and Management Symposium, pp. 1119–1124. IEEE (2016)

[31] Muhizi, S., Shamshin, G., Muthanna, A., Kirichek, R., Vladyko, A., Koucheryavy, A.: Analysis and performance evaluation of SDN queue model. In: Koucheryavy, Y., Mamatas, L., Matta, I., Ometov, A., Papadimitriou, P. (eds.) WWIC 2017. LNCS, vol. 10372, pp. 26–37. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-61382-6 3

[32] Shang, Z., Wolter, K.: Delay evaluation of OpenFlow network based on queueing model. arXiv preprint arXiv:1608.06491 (2016)

[33] Wang, Z., Zhao, S., Fan, Z.,Wan, X.: Performance modeling and analysis of control plane for SDN based on queuing theory. Wirel. Pers. Commun. 1–11 (2017)

[34] Mahmood, K., Chilwan, A., Sterb, O., Jarschel, M.: Modelling of OpenFlow-based software-defined networks: the multiple node case. IET Netw. 4(5), 278–284 (2015)

[35] Haiyan, M., Jinyao, Y., Georgopoulos, P., Plattner, B.: Towards SDN based queuing delay estimation. China Commun. 13(3), 27–36 (2016)

[36] Singh, D., Ng, B., Lai, Y.C., et al.: Modelling software-defined networking: switch design with finite buffer and priority queueing. In: IEEE Conference on Local Computer Networks, pp. 567–570. IEEE Computer Society (2017)

[37] Fahmin, A., Lai, Y.C., Hossain, M.S., Lin, Y.D., Saha, D.: Performance modeling of SDN with NFV under or aside the controller. In: 2017 5th International Conference on Future Internet of Things and Cloud Workshops (FiCloudW), Prague, pp. 211– 216 (2017)

[38] W. Miao, G. Min, Y. Wu, H. Wang, and J. Hu, "Performance modelling and analysis of software-defined networking under bursty multimedia traffic," ACM Trans. Multimedia Comput. Commun. Appl., vol. 12, no. 5s, pp. 77:1–77:19, Sep. 2016. [Online]. Available: http://doi.acm.org/10.1145/2983637.

[39] W. Tian, "Analysis and efficient provisioning of access networks with correlated and bursty arrivals," Int. J. Communication Systems, vol. 27, no. 4, pp. 551–570, 2014.

[40] Y. Lai1, Z. L. Hailemariam1, Y. Chen and Y. Kuo1 "Buffering and prioritization in switches for fast processing table-miss packets in software-defined networks" Int J Commun Syst., vol. 34, pp 1-19, 2021.

[41] D. B. David. "Analyzing Traffic Models Using IP Multimedia Server–Client Systems for Consumer Wireless Multimedia System Devices". Proceedings of the National Academy of Sciences, India Section A: Physical Sciences, 88(2), pp. 309-316, 2018.

[42] Y. Zhu, M. Sheng, J. Li, D. Zhou, and Z. Han, "Modeling and performance analysis for satellite data relay networks using two-dimensional markov-modulated process," IEEE Transactions on Wireless Communications, pp. 3894-3907, Mar. 2020.

[43] Y. Jia, Z. George Zhang, T. Xu. "Improving the Performance of MMPP/M/C Queue by Convex Optimization", A Real-World Application in Iron and Steel Industry. IEEE Access, Vol. 8, pp. 185909–185918, 2020.