Performance Modeling and Analysis of TCP Connections over Software Defined Networks

Yuan-Cheng Lai[1], Ahsan Ali[2], Md. Mahadi Hassan[2], Md. Shohrab Hossain[2] and Ying-Dar Lin[3]
[1]Department of Information Management, National Taiwan University of Science and Technology, Taipei, Taiwan
[2]Department of Computer Science and Engineering, Bangladesh University of Engineering and Technology, Bangladesh
[3]Department of Computer Science, National Chiao Tung University, Hsinchu, Taiwan
Email: laiyc@cs.ntust.edu.tw, {ahsanali.buet, sibathasan}@gmail.com, mshohrabhossain@cse.buet.ac.bd, ydlin@cs.nctu.edu.tw

***Abstract*—Software Defined Networking (SDN) decouples the control plane from the data plane, thereby enhancing flexibility in network management. Earlier works on SDN modeling only focused on packet-level arrivals without considering flow-level arrivals. However, a model without considering flow-level arrivals cannot correctly reflect the probability of sending packets to the controller. In this paper, we propose an analytical model of SDN considering flow-level (TCP connection) arrivals and packet-level arrivals simultaneously. We use an analytical method termed as 4D state model, which uses four-dimensional states. We have derived the state transition rates of the model and also the packet delay and packet loss probability. We have conducted numerical analysis and extensive simulations. Our results show good matches, which verify the suitability and correctness of our analysis. Error ratios of the analytical results for 4D state model and M/M/1 model against simulation results are also given. Results show that for data packet delays, 4D state can achieve error ratios of 1.16∽3.30%, which is much better than 8.57∽35.7% attained by M/M/1 model, which only considers packet-level arrivals.**

***Index Terms*—Software Defined Networking (SDN), Performance Modeling, Queueing Model, TCP.**

I. Introduction

Software Defined Networking (SDN) is currently seen as a promising approach for future Internet [1]–[4]. SDN is a concept of disassociating the control plane (decision maker) from the data plane (packet forwarder) of network devices, facilitating dynamic programmability to forward packets in highly distributed networks [5], [6]. SDN also enhances flexibility in network routing and provides the ability to change the behavior of a part of the network by isolating it from other parts [7]. In SDN architecture, packet forwarding devices (switches) are controlled by a logically centralized controller. The control plane and the data plane communicate through a south-bound interface, commonly known as OpenFlow protocol [8] that was standardized by the Open Networking Foundation (ONF).

There are several works on the performance issues of SDN. Jarschel et al. [7] used feedback oriented queueing theory to evaluate the performance of SDN architecture, considering a single switch and a single controller. Azodolmolky et al. [9] used network calculus to develop a mathematical model to evaluate the performance bound of SDN controller and switches in a worst case scenario. Few other works [1], [10], [11] also developed analytical models for SDN considering only packet-level arrivals.

Transmission Control Protocol (TCP) is a very popular protocol in today's Internet. In TCP, three-way handshaking (through SYN and ACK packets) is carried out to establish a connection, and then only the data packets are transmitted. Packet arrivals considering the flow (in TCP connection) arrivals is quite different from those cases that do not consider the TCP connection. In the former, no data packets will be sent before the SYN packet has been handled, while in the latter, there are always data packet arrivals. Hence, a model without considering flow-level arrivals will result in larger analytical errors. Therefore, it is essential to consider flow-level arrivals while modeling TCP over SDN. However, none of the previous works [1], [7], [9]–[11] focused on flow-level arrivals while performing SDN modeling. This is the *first work* that has developed a performance model for TCP connections over SDN, and makes use of flow-level arrivals for TCP connections.

In this work, we have developed a model for TCP connections over SDN, considering only one switch and one controller. In the model, we have considered the switch and controller jointly as a system and create a four-dimensional state to evaluate steady-state probability of the system states. The *contributions* of this work are: (i) developing a queueing model of TCP connections over SDN; (ii) developing an analytical method, termed the 4D state model, to obtain performance metrics according to the queueing model; (iii) performing extensive simulations to validate our analytical results, and (iv) comparing the analytical results of the 4D state model with the M/M/1 model [7] on various system parameters.

The remainder of this paper is organized as follows. Section II discuses TCP connection establishment in SDN architecture and related works on SDN modeling. In Section III, we describe our system model and present our analysis. In Section IV, we compare the analytical results of the 4D state model, M/M/1 model [7], and the simulation results. Finally, Section V has the concluding remarks.

II. Background and Related works

In Subsection II-A, we briefly cover SDN architecture and some previous works, and in Subsection II-B we explain the TCP connection establishment process in SDN architecture [5], [6].

978-1-5090-5019-2/17/$31.00 ©2017 IEEE

TABLE I
EXISTING WORKS ON SDN MODELING.

Paper	Component	#	Methodology	Performance Metrics
[7]	Switch	1	M/M/1	Average packet delay
	Controller	1	M/M/1/m	
[9]	Switch	N	Net. Calculus	Buffer size bound,
	Controller	1	Net. Calculus	Packet delay bound
[10]	Local Controller	N	M/M/1/m	Average packet delay
	Root Controller	1	M/M/1	
[11]	Switch	N	M/M/1	Avg. path delay,
	Controller	1	M/M/1/m	Avg. packet delay
[1]	Switch	N	M^X/M/1	Avg. packet delay,
	Controller	1	M/G/1	Avg. packet sojourn time
Our Model	Switch	1	Approx. MMPP/M/1	Avg. data packet delay, Avg. SYN packet delay,
	Controller	1	Approx. MMPP/M/1	Packet loss probability

A. Background

SDN disassociates the control plane from the data plane and then connects them by an open interface, the OpenFlow protocol. The control plane is implemented in the controller with software. A secure channel is used to communicate between the switch and the controller. The SDN switch contains a flow table for storing flow entries for every packet. When a packet arrives at the SDN switch, the header information of the packet is matched against the header part of the flow table entries. If no match is found with any flow table entries, the packet is forwarded to the controller, which makes a decision to handle packet and updates the statistics of the flow table. Again, if there is a match in the flow table, SDN switch takes appropriate action from the flow table entry.

There are only a few works on SDN modeling listed in Table. I. Jarschel et al. [7] used feedback orientated queueing theory to show the interaction between the control plane and the data plane. They used Markovian servers for both systems, i.e. an M/M/1 for the switch and an M/M/1/m for the controller. Azodolmolky et al. [9] used network calculus to develop the analytical model of a SDN network. The main difference between [9] and [7] is that queuing theory calculates average behaviors of the system when it is in equilibrium, whereas network calculus deals with worst cases.

Wang et al. [10] implemented the concept of multistage controllers for improving the flexibility of the control plane. However, this model increases mean delay time as there are multiple controllers. Mahmood et al. [11] aimed at expanding the data plane for more than one node. They proposed an analytical model for performance analysis of SDN, where one controller responds to multiple switches in the data plane. Jackson network [11] is used to model the data plane to support more than one node, whereas the controller is separately modeled as a single M/M/1 queue. Mahmood et al. [11] also derived the closed form expression for the probability density function (PDF) and cumulative density function (CDF) of the time spent by a packet in a SDN network. The most recent work on SDN, of Xiong et al. [1] modeled the packet forwarding of OpenFlow switches and the packet-in message processing of SDN controller as the queueing systems M^X/M/1 and M/G/1 respectively, where M^X means each arrival follows a Poisson process, but has a batch of packets. They designed a queueing model of OpenFlow networks in terms of packet forwarding performance and derived its closed-form expression of average packet sojourn time and the corresponding probability density function. However, none of these works [1], [7], [9]–[11] developed any analytical model for SDN for TCP connections.

B. TCP connections in SDN

In case of a TCP connection, the source first sends a connection request (SYN packet) to the destination through the SDN switch. When a SYN packet arrives at the SDN switch, its header portion is matched against the flow table entries. If there is a flow table entry for a corresponding SYN packet, the switch forwards the packet to its destination. Otherwise, the SDN switch forwards the SYN packet to the controller, which decides the route of the SYN packet, updates the switch flow table and sends it back to the switch. The switch then forwards the SYN packet to its destination. In response to the SYN packet, destination also sends a request plus acknowledgement (SYN+ACK packet) to the source through SDN switch. Again, the SYN+ACK packet is forwarded to the source if it has a flow table entry in the switch; otherwise, it is forwarded to the controller. In response to the SYN+ACK packet, the source finally sends an ACK packet to the destination through the switch, and this time the switch directly forwards the ACK packet to its destination. Thus, a TCP connection is established between the sender host and the destination host. Subsequent data packets of this TCP connection are directly forwarded by the SDN switch to the destination, without the involvement of the controller.

III. System Architecture

In our SDN queuing model, we have considered the switch and the controller jointly and modeled it using continuous time Markov chain. We have used queueing theory to develop a mathematical model of TCP connections over SDN. We first explain this queueing model in Subsection III-A, and in Subsection III-B, we describe the transition of system states, followed by the performance metrics in Subsection III-C.

A. Queueing model of the system

Fig. 1 shows the queueing model of SDN for TCP connections. It contains one switch and one controller, each of which has its own queue. The assumptions of the model are: (i) TCP connection arrival and packet arrival in each TCP connection is a Poison process; (ii) service times in both switch and controller follow exponential distributions; (iii) TCP connection duration follows an exponential distribution; (iv) switch queue size and controller queue size is finite; (v) The uni-direction from the source to the destination is considered and (vi) For each TCP connection, no data packets

will be sent before the SYN packet has been handled, that is, the table entry for this TCP connection has been set up in the flow table of the switch. The notations used in the queueing model (Fig. 1) are listed in Table II.

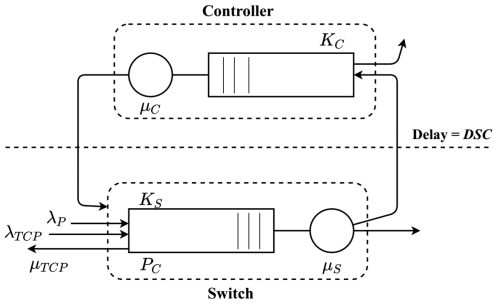

Fig. 1. Queuing model of SDN for TCP connections.

TABLE II
NOTATIONS USED IN THE QUEUEING MODEL

Symbol	Description
λ_{TCP}	TCP connection (SYN packet) arrival rate in switch
λ_P	Packet arrival rate per TCP connection in switch
P_C	The probability of redirecting SYN packet to controller
μ_S	Service rate of switch
μ_C	Service rate of controller
μ_{TCP}	TCP connection termination rate
K_S	Switch queue size
K_C	Controller queue size
DSC	The propagation delay between switch and controller
N_{TCP1}	The number of TCP connections sending SYN to the controller
N_{TCP2}	The number of other TCP connections
q_S	Switch queue length
q_C	Controller queue length

Since TCP connections whose SYN packets being handled by the controller will not send any data packets, we must know its number to determine an accurate packet arrival rate. Thus, we classify TCP connections into two types: TCP1 and TCP2, where TCP1 is the TCP connections whose SYN packets are handled by the controller, and TCP2 is other TCP connections. In our model, we used a four-dimensional state $(N_{TCP1}, N_{TCP2}, q_S, q_C)$, where $N_{TCP1}, N_{TCP2}, q_S, q_C$ represent the number of TCP1 connections, the number of TCP2 connections, the queue length in the switch, and the queue length in the controller, respectively.

B. State transition of system model

Fig. 2 shows seven possible transitions (labeled from 1 through 7) of state $(N_{TCP1}, N_{TCP2}, q_S, q_C)$ in our SDN system model for TCP connections. The followings are the explanation of each state transition:

- First transition occurs when a SYN packet arrives at the switch which has no flow table entry and it is redirected to the controller. So, N_{TCP1} and q_S are increased by 1. The transition rate is the product of λ_{TCP} and P_C.

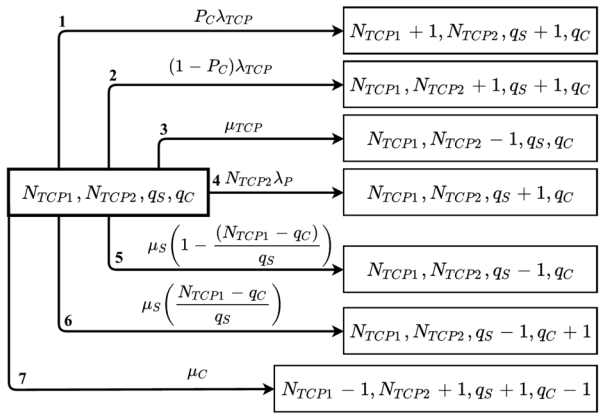

Fig. 2. State transition diagram of the SDN TCP model.

- Second transition occurs when a SYN packet arrives at the switch which has a flow table entry and it is forwarded to the destination. So, N_{TCP2} and q_S are increased by 1. The transition rate is the product of λ_{TCP} and $(1-P_C)$.
- Third transition occurs when a TCP connection terminates. So, N_{TCP2} is decreased by 1. The transition rate is μ_{TCP}.
- Fourth transition occurs when a data packet of an established TCP connection arrives at the switch. So, q_S is increased by 1. The transition rate is the product of λ_P and N_{TCP2}.
- Fifth transition occurs when switch serves a SYN packet (with flow table entry) or a data packet which is not redirected to the controller. The probability of redirecting a SYN packet to controller is $(N_{TCP1} - q_C)/q_S$. Hence, the probability of not redirecting a SYN packet (with flow table entry) or a data packet to controller is $(1 - (N_{TCP1} - q_C)/q_S)$. So, the transition rate is $(1-(N_{TCP1}-q_C)/q_S)\mu_S$, and switch queue length q_S is decreased by 1.
- Sixth transition occurs when switch serves a SYN packet (with no flow table entry) which is redirected to controller. The probability of redirecting a SYN packet to controller is $(N_{TCP1} - q_C)/q_S$. As SYN packet is redirected to controller from switch, q_S is decreased by 1 and q_C is increased by 1. The transition rate is $((N_{TCP1} - q_C)/q_S)\mu_S$.
- Seventh transition occurs when controller serves a SYN packet and forwards it to the switch. So, both N_{TCP1} and q_C are decreased by 1 and both N_{TCP2} and q_S are increased by 1. The transition rate is μ_C.

C. Performance metrics

According to [12], the mean queue length of the switch can be determined as

$$\overline{q_S} = \sum q_S \times P(N_{TCP1}, N_{TCP2}, q_S, q_C), \qquad (1)$$

and the mean queue length of the controller can be determined as

$$\overline{q_C} = \sum q_C \times P(N_{TCP1}, N_{TCP2}, q_S, q_C), \qquad (2)$$

where $P(N_{TCP1}, N_{TCP2}, q_S, q_C)$ is the steady state probability of state $(N_{TCP1}, N_{TCP2}, q_S, q_C)$.

(I) Average data packet delay in the switch, D_{data} :
D_{data} can be calculated as

$$D_{data} = \left(\frac{\overline{q_S} + 1}{\mu_S}\right). \quad (3)$$

(II) Average delay of SYN packet sent to the controller, D_{SYN1} :
In our SDN system model, SYN packet is queued twice at the switch and once at the controller. SYN packet delay in the switch is same as Eqn. (3) and SYN packet delay in the controller D_C is $(\overline{q_C} + 1)/\mu_C$. So, D_{SYN1} can be calculated as $D_C + 2 \times D_{data} + 2 \times DSC$. Hence, we get

$$D_{SYN1} = \left(\frac{\overline{q_C} + 1}{\mu_C}\right) + 2\left(\frac{\overline{q_S} + 1}{\mu_S}\right) + 2DSC. \quad (4)$$

(III) Packet loss probability in the switch, L_S :
In our SDN system model, incoming packet drops at the switch when queue is full (i.e., $q_S = K_S$). So, L_S can be determined as

$$L_S = \sum P(N_{TCP1}, N_{TCP2}, q_S = K_S, q_C). \quad (5)$$

In SDN architecture, SYN packet drops at the controller when controller queue is full (i.e., $q_C = K_C$). So, SYN packet loss probability in the controller L_C can be obtained as

$$L_C = \sum P(N_{TCP1}, N_{TCP2}, q_S, q_C = K_C). \quad (6)$$

(IV) Loss probability of SYN packet sent to the controller, L_{SYN1} :
In our SDN system model, SYN packet is queued twice at the switch and once at the controller. So, SYN packet can drop either at the switch or at the controller. So, L_{SYN1} can be determined as

$$L_{SYN1} = 1 - (1 - L_S)^2(1 - L_C). \quad (7)$$

IV. Results

In this section, we present the simulation and analytical results of 4D state and M/M/1 models [7] while varying different system parameters. We discuss the impact of the different parameters on average packet delay and packet loss probability for the two analytical models. We also calculate their error ratios by comparing with the simulation results. Finally, we summarize major observations.

A. Simulation

In order to validate our analytical method, we developed a JAVA program (to implement packet-based simulation) which supports both flow-level arrivals and packet-level arrivals, following the assumptions of our queueing model detailed in Subsection III-A. The baseline parameter values (used for analysis and simulation) of λ_P, λ_{TCP}, μ_S, μ_C, μ_{TCP} and P_C are 4000 packets/s, 20 flows/s, 30000 packets/s, 20000 packets/s, 25 flows/s and 0.25 respectively. We considered both switch queue size K_S and controller queue size K_C as 8. We compared the simulation results by varying different system parameters: λ_{TCP}, P_C, and λ_P; and observed the impact of those parameters on performance metrics. While varying a system parameter, we kept other baseline parameters fixed. Here, we simulated 3000 packets for every set of values and ran the simulation 2000 times.

B. M/M/1 model

M/M/1 model is the most commonly used model on analyzing the SDN performance [7], [10], [11]. Actually, M/M/1 can achieve correctness to a certain extend for SDN which only considers packet-level arrivals [7]. So, we compared the analytical results of M/M/1 model with the simulation results to show the inappropriateness of the M/M/1 model. The M/M/1 model assumes that the average packet arrival rates at the switch and at the controller follow a Poisson process with mean λ_S and λ_C, respectively. We use M/M/1 formula to derive the average packet delay in the switch, D_S, and in the controller, D_C, as

$$D_S = \left(\frac{1}{\mu_S - \lambda_S}\right), D_C = \left(\frac{1}{\mu_C - \lambda_C}\right). \quad (8)$$

However, in our model, since the number of TCP2 connections is varied, λ_S and λ_C are difficult to obtain. To fair comparison, we measure λ_S and λ_C from the simulation, and then using Eq, 8 we calculate D_S and D_C. Afterwards, we can calculate the D_{data} and D_{SYN1} as

$$D_{data} = D_S, \quad (9)$$

$$D_{SYN1} = D_C + 2D_S + 2DSC. \quad (10)$$

Paper [7] assumes an infinite buffer (for switch and controller) and uses the M/M/1 model, and thus does not investigate the issue of packet loss probability. Referring to [13], [14], the buffer is modeled as an infinite queue model and the queue is truncated at some finite integer K, so that the loss probability is obtained. We obtain the loss probability in the switch, L_S, and in the controller, L_C, as

$$L_S = \left(\frac{\lambda_S}{\mu_S}\right)^{K_S}, \; L_C = \left(\frac{\lambda_C}{\mu_C}\right)^{K_C}. \quad (11)$$

Finally, we obtain the loss probability of SYN packets sent to controller in M/M/1 model as

$$L_{SYN1} = 1 - (1 - L_S)^2(1 - L_C). \quad (12)$$

C. Impact of λ_{TCP}

Fig. 3 shows the impact of λ_{TCP} on average data packet delay, D_{data}, for 4D state model and M/M/1 model. Fig. 3 shows that the analytical results of 4D state model are very close to the simulation results. Here, D_{data} increases with the increase of λ_{TCP} in both methods because the number of connections in the switch increases with increasing λ_{TCP}, resulting in an increase in data packet arrival rate. However, the gap between 4D state model and M/M/1 model result of D_{data} is almost constant. Thus, using M/M/1 will overestimate D_{data} and obtain a higher error rate. We also found that the

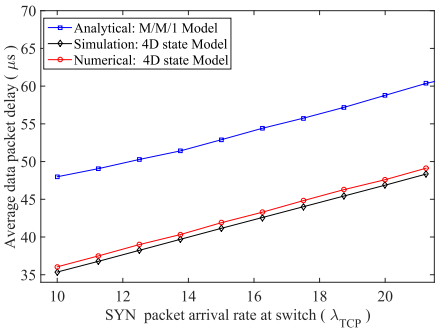

Fig. 3. Average data packet delay vs. λ_{TCP}

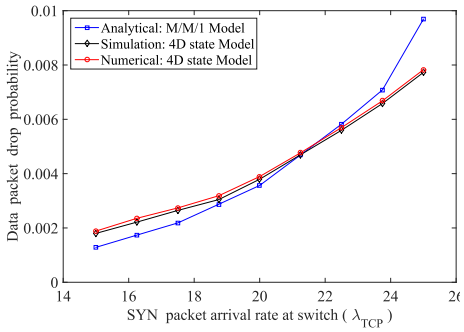

Fig. 4. Data packet drop probability vs. λ_{TCP}

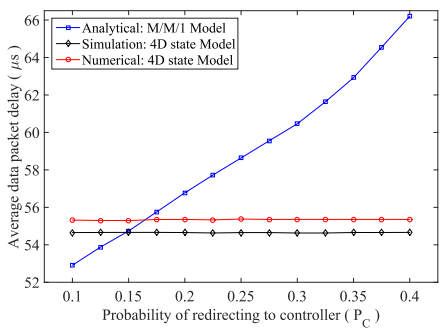

Fig. 5. Average data packet delay vs. P_C

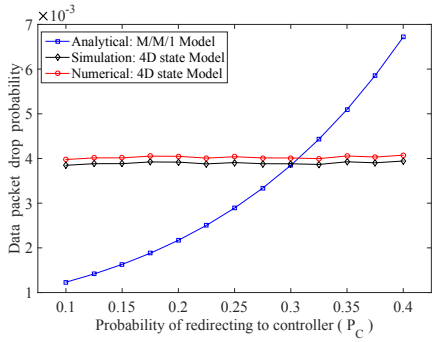

Fig. 6. Data packet drop probability vs. P_C

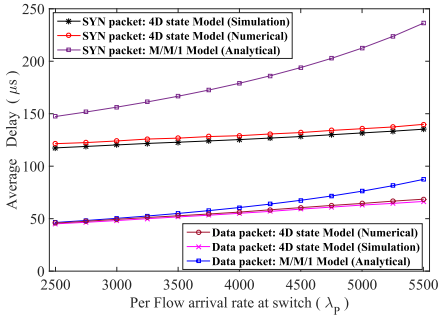

Fig. 7. SYN packet and data packet delay vs. λ_P

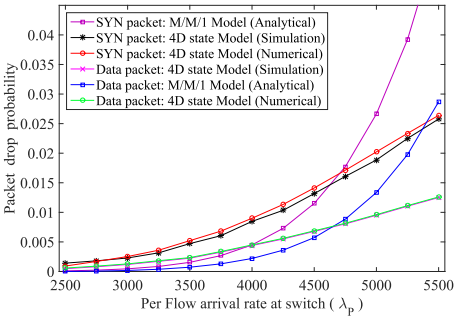

Fig. 8. SYN packet and data packet loss probability vs. λ_P

error ratios of 4D state model and M/M/1 model against the simulation results on average data packet delay are 1.78% and 28.9%, respectively.

Fig. 4 shows the impact of λ_{TCP} on data packet loss probability, L_S, for 4D state model and M/M/1 model. Fig. 4 shows that the analytical results of 4D state model are very close to the simulation results, which can verify our correctness of 4D state model. Here, L_S increases with the increase of λ_{TCP} in both methods, because more connections in the switch appear, resulting in the increase in packet arrival rate. Initially data packet loss probability L_S, for M/M/1 model is lower than 4D state model but it becomes higher for M/M/1 model as λ_{TCP} crosses 21 flows/sec. We also found that the error ratios of 4D state model and M/M/1 model against the simulation results on data packet loss probability are 3.11% and 13%, respectively.

D. Impact of P_C

Fig. 5 shows the impact of P_C on average data packet delay, D_{data}, for 4D state and M/M/1 model. Fig. 5 shows that the analytical results of 4D state are very close to the simulation results, confirming its correctness. It was also found that, D_{data} for M/M/1 model increases with increasing P_C. As P_C increases, more data packets are redirected to the controller from the switch, which generates more packets to the switch from the controller, resulting in larger D_{data}. But in our model, with increasing P_C, more SYN packets are redirected to the controller which does not affect D_{data}. We also found that the error ratios of 4D state model and M/M/1 model against the simulation results on average data packet delay are 1.26% and 8.5%, respectively.

Fig. 6 shows the impact of P_C on data packet loss probability, L_S, for 4D state model and M/M/1 model. Fig. 6 shows that the analytical results of 4D state model are very close to the simulation results, which verifies the correctness of 4D state model. Here, L_S increases with the increase of P_C for the M/M/1 model because more packets are sent to the controller, resulting in an increase in data packet arrivals from the controller to the switch, which increases packet loss probability. Initially, data packet loss probability L_S, for M/M/1 model is lower than 4D state model but it becomes higher for M/M/1 model as P_C crosses 0.3. We also found that the error ratios of 4D state model and M/M/1 model on data packet loss probability are 3.33% and 40%, respectively.

E. Comparison of SYN packets with data packets

Fig. 7 shows the average delay of SYN packets sent to the controller, D_{SYN1}, and data packets, D_{data}, in 4D state model and M/M/1 model by varying λ_P. Fig. 7 shows that the analytical results of 4D state model and the simulation results are very close. It was found that D_{SYN1} and D_{data} increase with increasing λ_P because of more packet arrivals. However, they increase rapidly for M/M/1 model but slowly for the 4D state model. As a SYN packet visits the controller once and the switch twice, its delay is higher than the average delay of data packets. That means the time required to create a TCP

connection is much longer than the transmission time of data packets. Fig. 8 shows the loss probability of SYN packets, L_{SYN1}, and data packets, L_S in 4D state model and M/M/1 model by varying λ_P. Fig. 8 shows that the analytical results of 4D state model and the simulation results are very close. It was found that L_{SYN1} and L_S increase as λ_P is increased because of more packet arrivals. However, they increase rapidly for M/M/1 model but slowly for 4D state model. As a SYN packet can drop either at the switch or at the controller, L_{SYN1} is higher than the drop probability of data packets, L_S.

Finally, the error ratios of the analytical results of 4D state model and M/M/1 model on average data packet delay are 2.41% and 12.61%, respectively and on packet loss probability are 5% and 54%, respectively. On the other hand, the error ratios of 4D state model and M/M/1 model on SYN packet delay are 3.17% and 45.5%, respectively and on SYN packet loss probability are 10.1% and 60%, respectively. We can see the error ratios of 4D state model are significantly lower than that of M/M/1 model. Also the error ratios for SYN packets are larger than that for data packets. A SYN packet is queued twice at the switch and once at the controller. Therefore, the errors caused by the calculation for each queue is accumulated to a larger error ratio. On the other hand, a data packet is queued at the switch once, so its error caused by the calculation is only for the switch queue, resulting in a lower error ratio.

F. Summary of results

The following are the major observations from the results.

- 4D state model obtains significantly lower error ratios than M/M/1 model on delay and loss probability.
- The error ratios of SYN packets on delay and loss probability are larger than those of data packets.
- The error ratio of packet loss probability is larger than that of packet delay.
- When the network becomes more congested (larger λ_{TCP} or larger P_C), the error ratios become larger.

V. Conclusion and future works

In this paper, we have presented an analytical model for the performance analysis of TCP connections over SDN architecture considering one switch and one controller. We considered the control plane and the data plane jointly and created four-dimensional states, and calculated their steady-state probabilities, validated by extensive simulation. We have compared the analytical results of 4D state model with the simulation results on D_{data} and L_S. Result shows that 4D state model can make a correct analysis and obtain low error ratios, ranging between 1.16% and 3.30% on average packet delay, and between 0.8% and 8.45% on packet loss probability. However, using an M/M/1 model will generate high error ratios, ranging between 8.5% and 35.7% on average packet delay, and between 9.83% and 54% on packet loss probability. This is because a 4D state model considers both flow-level arrivals and packet-level arrivals while M/M/1 model considers only packet-level arrivals.

We also observed the delay of SYN packets sent to the controller is significantly larger than the delay of data packets. Further, the loss probability of SYN packets is larger than that of data packets. However, 4D state model can still generate correct analytical results for SYN packets. The error ratios of SYN packets are 3.17% on delay and 10.1% on loss probability. From the results, our analytical method (4D state model) can achieve excellent accuracy on packet delay and packet loss probability under the SDN environment that has the flow-level (TCP connection) arrivals and packet-level arrivals. On the other hand, M/M/1 model generates high error ratios, meaning that, it is not applicable to the SDN environment.

References

[1] B. Xiong, K. Yang, J. Zhao, W. Li, and K. Li, "Performance evaluation of openflow-based software-defined networks based on queueing model," *Computer Networks*, vol. 102, pp. 172–185, June 2016.

[2] S. Rowshanrad, S. Namvarasl, V. Abdi, M. Hajizadeh, and M. Keshtgary, "A survey on SDN, the future of networking," *Advanced Computer Science & Technology*, vol. 3, no. 2, pp. 232–248, 2014.

[3] A. Hakiri, A. Gokhale, P. Berthou, D. C. Schmidt, and T. Gayraud, "Software-defined networking: Challenges and research opportunities for future internet," *Computer Networks*, vol. 75, no. PA, pp. 453–471, December 2014.

[4] J. Pan, S. Paul, and R. Jain, "A survey of the research on future internet architectures," *IEEE Communications Magazine*, vol. 49, no. 7, pp. 26–36, 2011.

[5] ONF, "Software-Defined Networking: The new norm for networks," April 2012. [Online]. Available: https://www.opennetworking.org/images/stories/downloads/sdn-resources/white-papers/wp-sdn-newnorm.pdf

[6] ——, "SDN architecture overview," December 2013. [Online]. Available: https://www.opennetworking.org/images/stories/downloads/sdn-resources/technical-reports/SDN-architecture-overview-1.0.pdf

[7] M. Jarschel, S. Oechsner, D. Schlosser, R. Pries, S. Goll, and P. Tran-Gia, "Modeling and performance evaluation of an OpenFlow architecture," in *International teletraffic congress*, 2011, pp. 1–7.

[8] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "Openflow: Enabling innovation in campus networks," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 2, pp. 69–74, 2008.

[9] S. Azodolmolky, R. Nejabati, M. Pazouki, P. Wieder, R. Yahyapour, and D. Simeonidou, "An analytical model for software defined networking: A network calculus-based approach," in *IEEE Global Communications Conference (GLOBECOM)*, Atlanta, GA, USA, Dec 9-13, 2013, pp. 1397–1402.

[10] G. Wang, J. Li, and X. Chang, "Modeling and performance analysis of the multiple controllers' approach in software defined networking," in *23rd International Symposium on Quality of Service*. Portland, OR, USA: IEEE, June 15-16, 2015, pp. 73–74.

[11] K. Mahmood, A. Chilwan, O. Østerbø, and M. Jarschel, "Modelling of OpenFlow-based software-defined networks: the multiple node case," *IET Networks*, vol. 4, no. 5, pp. 278–284, 2015.

[12] B. Ciciani, A. Santoro, and P. Romano, "Approximate analytical models for networked servers subject to MMPP arrival processes," in *IEEE International Symposium on Network Computing and Applications*, July 12-14, 2007, pp. 25–32.

[13] R. J. Simcore and T.-B. Pei, "Perspectives on atm switch architecture and the influence of traffic pattern assumptions on swtich design," *SIGCOMM Computer Communications Review*, vol. 25, no. 2, pp. 93–105, 1995.

[14] S. C. Liew, "Performance of various input-buffered and output-buffered atm switch design principles under bursty traffic: Simulation study," *IEEE Transactions on Communications*, vol. 42, no. 234, pp. 1371–1379, 1994.