A

Performance Modelling and Analysis of Software Defined Networking under Bursty Multimedia Traffic

WANG MIAO, University of Exeter
GEYONG MIN, University of Exeter
YULEI WU, University of Exeter
HAOZHE WANG, University of Exeter
JIA HU, University of Exeter

Software Defined Networking (SDN) is an emerging architecture for the next-generation Internet, providing unprecedented network programmability to handle the explosive growth of Big Data driven by the popularisation of smart mobile devices and the pervasiveness of content-rich multimedia applications. In order to quantitatively investigate the performance limits of SDN networks, several research achievements from both simulation experiments and analytical modelling have been reported in the current literature. Among those studies, analytical modelling has demonstrated its superiority in terms of cost-effectiveness in the evaluation of large-scale, e.g., Internet-scale networks. However, for analytical tractability and simplification, existing analytical models are derived based on the unrealistic assumptions that all network traffic follows the Poisson process which is suitable to model nonbursty text data and the data plane of SDN is modelled by one simplified Single Server Single Queue (SSSQ) system. Recent measurement studies have shown that, due to the features of heavy volume and high velocity, the multimedia big data generated by real-world multimedia applications reveals the bursty and correlated nature in the network transmission. With the aim of the capturing such characteristics of realistic traffic patterns and obtaining a comprehensive and deeper understanding of the performance behaviour of SDN networks, this paper presents a new analytical model to investigate the performance of SDN in the presence of the bursty and correlated arrivals modelled by Markov Modulated Poisson Process (MMPP). The Quality-of-Service performance metrics in terms of average latency and average network throughput of the SDN networks are derived based on the developed analytical model. To consider realistic multi-queue system of forwarding elements, a Priority-Queue (PQ) system is adopted to model SDN data plane. To address the challenging problem of obtaining the key performance metrics, e.g., queue length distribution of PQ system with a given service capacity, a versatile methodology extending the Empty Buffer Approximation (EBA) method is proposed to facilitate the decomposition of such a PQ system to two SSSQ systems. The validity of the proposed model is demonstrated through extensive simulation experiments. To illustrate its application, the developed model is then utilised to study the strategy of the network configuration and resource allocation in SDN networks.

CCS Concepts:•**Networks** → **Network performance modeling;** *Network performance analysis;*

Additional Key Words and Phrases: Software defined networking, multimedia big data, performance modelling and analysis, queueing decomposition, resource allocation

ACM Reference Format:
Wang Miao, Geyong Min, Yulei Wu, Haozhe Wang and Jia Hu, 2015. Performance Modelling and Analysis of Software Defined Networking under Bursty Multimedia Traffic. *ACM Trans. Multimedia Comput. Commun. Appl.* V, N, Article A (January YYYY), 20 pages.
DOI: http://dx.doi.org/10.1145/0000000.0000000

This work is supported by the EU FP7 "QUICK" Project, under Grant Agreement No. PIRSES-GA-2013-612652.
Author's addresses: W. Miao, G. Min, Y. Wu, H. Wang and Jia Hu, Harrison Building, Exeter University, North Park Road, Exeter, Devon, United Kingdom, EX4 4QF

DOI: http://dx.doi.org/10.1145/0000000.0000000

1. INTRODUCTION

With rapid innovation in sophisticated Information and Communications Technology (ICT), there has been an explosive growth in the volume of network data over the past several years, driven by the popularisation of smart mobile devices and the pervasiveness of content-rich multimedia applications. According to the latest Cisco Visual Networking Index (VNI) forecast, the global Internet Protocol (IP) traffic will grow threefold from 2014 to 2019 and will reach 2 Zettabytes per year by 2019 [Cisco 2015]. With such an overwhelming amount of data pouring into the future Internet, network domains are embracing an unprecedented wave of traffic flows and are stepping into the era of network Big Data. Therefore, how to handle the ever-increasing network multimedia traffic and provide satisfactory Quality-of-Service (QoS) guarantee has become a timely challenge that stresses the next-generation Internet.

To address this important challenge, Software Defined Networking (SDN) has been attracted significant attention in the networking and telecom community. It was proposed to accelerate network innovation and flexible deployment by decoupling the logic of the network control from its forwarding plane, enabling network operators to gain unprecedented programmability, automation, and network control [Fang et al. 2013] [Congdon et al. 2014][Dong et al. 2015]. SDN architecture has three cooperative layers: infrastructure layer, control layer and application layer. The infrastructure layer provides network forwarding services under the instructions of the control layer. The control layer maintains the network-wide information to manage and optimise the overall network performance. SDN controller uses southbound interfaces such as OpenFlow, OVSDB and OpFlex to communicate with the infrastructure layer, and leverages northbound interfaces, for instance Restful APIs to interact with the application layer. The application layer creates various application services for service providers, e.g., network topology discovery, virtual private network provisioning, and path reservations, based on the received network information from the control layer.

With the logically centralised control of the network infrastructure, SDN enables the software intelligence to program networks via the well-defined programmatic interfaces, which are highly customisable, scalable, and agile to meet the requirements of big data on-demand. For instance, the process of mining the intelligence from the huge amount of big data fundamentally involves three steps: splitting the data into multiple server nodes, analysing each data block in parallel, and merging the computed results. Owing to the Splitting-Merging nature of these parallel computations, the processing speed of big data analytics is highly affected by the efficiency of the data transmission over the underlying networks. According to the study in [Chowdhury et al. 2011], the network transmission latency can account for more than 50% of the overall data processing time. With the logically centralised view of the entire network, SDN has regarded as an ideal architecture to enhance the performance of big data analytics through high-efficient traffic delivery. Specifically, through dynamic configuration of the network devices, SDN is capable of creating secure and reliable end-to-end pathways to satisfy the specific transmission demands for each data processing node, thereby significantly reducing the network transmission latency and hence overall data processing time [Wang et al. 2012] [Sadasivarao et al. 2013] [Das et al. 2013] [Hu et al. 2015] [Qin et al. 2015].

Although SDN enables much precision and efficiency for big data analytics, its design and implementation, on the other hand, encountered new challenges created by big data. Compared with traditional distributed network control, the centralised control of SDN has been criticised for its lack of scalability to support ever-increasing demand for network performance improvement [Yeganeh et al. 2013]. With SDN, the controller is responsible for the transactions of the underlying infrastructure layer,

where switches and routers send the request messages to SDN controller for the routing information when the local flow tables did not cache the forwarding entry for the visiting packets. Flow table management at the forwarding devices and network status collection at the controllers are also bringing frequent communications between SDN controllers and underlying forwarding devices. As a result, the logically centralised controller becomes a key bottleneck for the SDN network to improve the performance and extend the services. With the emerging big data, this issue is becoming even worse than ever, since the massive big data pouring into the data plane places an unprecedented burden on the SDN architecture design in terms of the transmission rate of the southbound interface, the processing capability of the SDN controller, the computation efficiency of the algorithms as well as the storage space of the forwarding devices. Therefore, it is urgent and paramount importance to have a comprehensive and deep understanding of SDN network performance including both data plane and control plane under the traffic patterns exhibited by realistic multimedia applications.

Existing performance studies of the SDN networks have been achieved by both simulation experiments and analytical modelling. For the simulation experiments, most of the studies [Naous et al. 2008] [Bianco et al. 2010] [Antichi et al. 2011] [Khan and Dave 2013], focused on the implementations of the OpenFlow switches on various platforms (e.g., NetFPGA and Linux) to provide high-performance forwarding capabilities, and the design of software-based SDN controllers to facilitate the network management and optimisation. However, with the era of network big data, simulation-based study always become time-consuming and even lose the effectiveness and efficiency when the large-scale network and huge amount of network traffic are considered. In contrast to the simulation, analytical modelling has the potential to offer the reduction in the computation resource and is capable of quantitively capturing the essential characteristics of the network with different system configurations and under various traffic loads [Wu et al. 2013]. Several analytical models have been proposed in the literatures with the aim of investigating the performance of SDN architecture, [Jarschel. et al. 2011] [Azodolmolky et al. 2013] [Mahmood et al. 2015] [Miao et al. 2015]. Nevertheless, for the tractability of the mathematical derivation and performance analysis, existing studies mainly assume that the traffic entering the SDN networks statistically follows nonbursty Poisson process, which is widely adopted to model text data, and the data plane of SDN is modelled using one Single Server Single Queue (SSSQ) system. Many recent measurement studies [Kapoor et al. 2013] [Tate et al. 2013] [Raj et al. 2015] [Liu et al. 2015] have revealed that the multimedia big data results in the higher use of the network in a short time, which exhibits high degree of bursty feature and has significant negative impacts on the improvements of network performance. To bridge this gap, this paper proposes an original comprehensive analytical model for SDN networks 1) to take the realistic nature of multimedia traffic and multi-queue system in the data plane into account, and 2) to develop a systematic and efficient method to derive such an analytical model for the accurate evaluation of SDN performance, in particular its performance bottlenecks. The major contributions of this paper are summarised in the following aspects:

— A novel analytical model is designed in this work for SDN networks subject to the input of bursty multimedia traffic, which is modelled by Markov Modulated Poisson Process (MMPP). The bursty and correlated nature of the traffic characteristics on each link and component of SDN network is captured by the splitting and superposition of multiple bursty sources. The QoS performance metrics in terms of average latency and average network throughput are derived based on the developed analytical model;

— To consider realistic multi-queue system of the forwarding elements, a Priority-Queue (PQ) system is adopted to model SDN data plane. To address the challenging problem of obtaining the key performance metrics, e.g., queue length distribution of a PQ system with the given service capacity, a versatile methodology extending the Empty Buffer Approximation (EBA) method is proposed to facilitate the decomposition of such a PQ system to two SSSQ systems;
— To validate the accuracy of the developed analytical model, extensive OMNet++ simulations are conducted under various network configurations and MMPP parameters, with the aim of capturing different network conditions and degrees of traffic burstiness and correlations;
— To illustrate its applications, the analytical model is then used as an efficient performance evaluation tool to investigate the effects of flow table hit probability and service rates on the performance of SDN networks, acquiring the required resource allocation strategy between the forwarding devices and SDN controller.

The rest of this paper is organised as follows: Section 2 illustrates the working principle of the PQ-based SDN architecture and presents the state-of-the-art related research work on SDN performance studies. Section 3 derives the comprehensive analytical model to investigate the average latency and network throughput of SDN networks subject to bursty and correlated traffic.The accuracy of the developed model is validated in Section 4 through extensive simulation experiments. Section 5 adopts the developed model to conduct the performance analysis of SDN architecture. Finally, Section 6 concludes this study.

2. PRELIMINARIES

2.1. Priority-Queue based SDN System Architecture

This study focuses on the SDN architecture with the PQ system in the data plane, as illustrated in Fig. 1. The PQ system consists of two types of queues: the low priority queue and the high priority queue. The packets transmitted from the SDN controller enter the high priority queue and the ones arriving from the neighbouring forwarding devices enter the low priority queue. During the packet scheduling, the packets in the high priority queue have the priority for receiving the service and the packets belonging to the low priority queue can only receive service when there is no packet in the high priority queue. According to the system design in [Kong et al. 2013], the buffer sizes of the high priority queue, Uplink Channel (UC) queue and Downlink Channel (DC) queue, are considered to be infinite, and the buffer sizes of the SDN controller and the low priority queue are set to be finite, denoted by K_c and K_l, respectively. According to [ONF 2012], the working mechanism of the PQ-based SDN system architecture is as follows: when arriving at the SDN switch, the packet waits in the low priority queue for service if the buffer is not fully occupied and the Switch Server (SS) is busy. Once the SS becomes idle and there is no packets in the high priority queue, the first packet in the low priority queue is popped out and transferred to the SS. If the flow table in the switch holds the corresponding entry, this packet will be served immediately according to the action field of the matching flow entry. Otherwise, if the packet does not match any entry in the flow table, the switch needs to send the whole or partial package (i.e., the packet header) to the SDN controller through the UC to consult how to process the unmatched packet. When the partial transmission strategy is adopted, the unmatched packet is stored in the local cache and waits for the response message from the SDN controller. Once receiving the request message from SDN switches, the controller generates a response message based on a series of routing and forwarding calculations and sends the result out through the DC. When arriving at the SDN switch, the packets of the response message enter the high priority queue

if the SS is busy. Once the SS becomes idle, the first packet in the high priority queue is forwarded to the SS. When receiving the response message, the SDN switch stores the routing information as the entries of the flow table and leverages the action field of the entry to process the packet. Herein, during the whole process, the packet that fails to match the entries of the flow table has to traverse the SS twice.

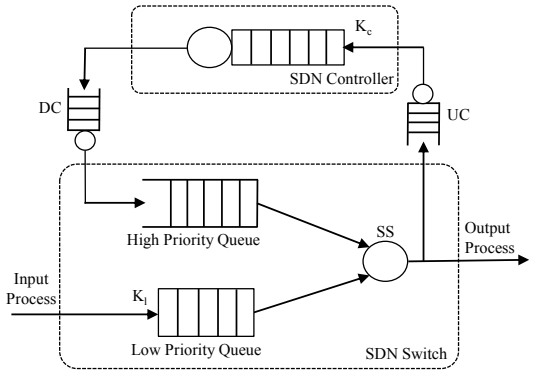


Fig. 1. The PQ-based SDN System Architecture

2.2. Related Work

SDN allows the underlying network to be programmed by the control plane. Recently, several research efforts have been made to leverage this ability to dynamically orchestrate and create large computing platform for big data analytics. In order to jointly optimise application performance and network utilisation, Wang et al. [2012] proposed an application-aware network configuration through SDN controller and optical switching. In order to overcome the limited rule space of SDN devices, Dong et al. [2015] designed a two-level rule space strategy and a novel cache prefetching mechanism to increase the utilisation of network resources. The novelty of this work is the integration of the modification of the flow entries with the prediction of the user mobility to significantly increase the hit probability of the flow table. Authors in [Li et al. 2015] leveraged the capability of SDN architecture to realise the virtualisation of radio access network for the social Internet of Things. In order to provide re-configurable networks for social big data analytics, a big data processing system powered by SDN was designed in [Sadasivarao et al. 2013] through integrating the transmission of SDN networks and the processing of Hadoop architecture. This work exploited the potential of SDN to build a high-performance network infrastructure among different processing units to accelerate the overall data processing; Li et al. [2015] proposed an OFScheduler based on OpenFlow and used as a network optimiser for optimising MapReduce operations in a heterogeneous cluster. Qin et al. [2015] designed a framework that employs SDN elements in Hadoop to reduce the time taken by data to reach the distributed processing nodes. Within each processing nodes, an online deduplication energy approach is developed in [Li et al. 2014] to minimise the energy cost in data centre.

However, due to the explosive growth in the volume of Internet traffic and the ever-increasing QoS requirements of emerging network applications, there have been timely and pressing demands to improve the performance of SDN networks to address the issues brought by big data. Currently, the performance of SDN networks are

mainly investigated by simulation experiments and analytical modelling. For the simulation experiments, Naous et al. [2008] firstly implemented the OpenFlow Ethernet switch on the NetFPGA platform to provide high throughput and low complexity of forwarding switches. Antichi et al. [2011] extended Naous's work by designing a flexible OpenFlow based forwarding architecture through regular expression with the aim of supporting more network applications in data plane. The switches in the proposed architecture are capable of storing up to 200K flow information while satisfying the requirement of line rate processing. Bianco et al. [2010] extended the OpenFlow switch implementation to the Linux platform based on OpenFlow specification, also meeting realiztic line rates. These implementations of OpenFlow switches, however, were only realised on single platform, lacking the support for cross-platform implementation. To fill this gap, in [Khan and Dave 2013], a modular and parameterised implementation of a hardware-based OpenFlow switch was proposed to implement SDN on three different platforms, i.e., NetFPGA, ML605 and DE4. Through performance comparison, this study showed that the switch design can be implemented on different platforms with minor performance variations.

As compared to simulation, analytical modelling has the potential to offer a significant reduction of the computation resource and time required for achieving the system performance, especially when large-scale Internet with high-volume big data is considered [Abawajy 2009]. Azodolmolky et al. [2013] analysed the performance of the SDN architecture through exploiting the network calculus theory and achieved the boundary condition of the transmission latency and queue length. Based on the measurements of the switching time in hardware OpenFlow switch, Jarschel et al. in [2011] proposed an analytical model for SDN networks to estimate the packet sojourn time and packet loss probability. Mahmood et al. in [2015] extended the work in [Jarschel. et al. 2011] to investigate the performance of the SDN architecture with multi-node condition. In this study, both the data plane and control plane were simply modelled as M/M/1 queues. These studies have focused on the performance analysis of SDN networks under the assumption that SSSQ model is adopted to provide service for the arriving packets in SDN data plane. This simplified assumption excludes any necessity for considering multiple queues in the performance analysis. However, according to the OpenFlow specification 1.3.3 [ONF 2013], the queue structure should be designed to have multiple queues in the forwarding devices. Recently, we proposed a new analytical model in [Miao et al. 2015] to investigate the performance of SDN networks in the presence of PQ system in data plane. This work showed that the priority packet scheduling outperforms the First In First Out (FIFO) scheduling in terms of the packet loss probability and quantitatively derived the average packet latency taking the limited buffer size into account. However, for analytical tractability and simplicity, all these existing models for SDN networks were primarily developed under the nonbursty Poisson arrivals. For big data transmission, the traffic patterns typically exhibit bursty and correlated nature. For instance, Tate et al. [2013] revealed that the procedures of the big data analytics lead to the high burstness during the data transmission and more dynamic resource allocation are required to meet the QoS/QoE requirements. The authors in [Kapoor et al. 2013] discovered that the traffic behaviour of the big data exhibits the larger bursty at the range of 10-1000 microsecond. Liu et al. in [2015] analysed the four popular big data applications (Hadoop, Spark, Shark and Impala) running on the experimental clusters. The real traffic was collected from the communications among these applications. The experimental results showed that the big data traffic results in the higher use of the network in a short time and exhibits the high degree of fluctuations and burstiness. Although the transmission of the big data presents the bursty pattern and significantly affects the provisioning of the network services, none of the existing analytical models for the SDN networks is capable

of capturing such realistic characteristics. To fill this gap, the aim of this paper is to design a novel analytical model to comprehensively investigate the performance of the SDN networks with the input of bursty and correlated traffic.

2.3. Markov-Modulated Poisson Process Traffic

In this study, the traffic entering SDN switch is modelled by MMPP, which is a doubly stochastic Poisson process with the arrival rate varying according to an irreducible continuous-time Markov chain. Due to the capability of capturing the time-varying arrival rate, MMPP is an effective tool for modelling bursty and correlated traffic. In addition, the superposition and splitting operations of MMMPs generate a new MMPP [Fischer and Meier-Hellstern 1993] [Lee and Cho 2010], facilitating the derivation of the analytical model in the complex network environment. A two-state MMPP has been widely used to investigate the performance of the network [Mark and Ephraim 2014] [Liu et al. 2008] and is parameterised by an infinitesimal generator matrix Q_i of the Markov chain and an arrival rate matrix Λ_i as follows

$$Q_i = \begin{bmatrix} -\varphi_{1i} & \varphi_{1i} \\ \varphi_{2i} & -\varphi_{2i} \end{bmatrix} \qquad \Lambda_i = \begin{bmatrix} \lambda_{1i} & 0 \\ 0 & \lambda_{2i} \end{bmatrix} \tag{1}$$

where φ_{1i} is the transition rate from state 1 to 2, and φ_{2i} is the rate out of state 2 to 1. λ_{1i} and λ_{2i} are the traffic rates when the Markov chain is in states 1 and 2, respectively. The subscript i denotes the type of the queue the traffic arriving within the SDN architecture, e.g., low priority queue, high priority queue, UC and DC queues and the controller queue. Then, the mean arrival rate, λ_i^m , of $MMPP_i$ can be given by

$$\lambda_i^m = \frac{\lambda_{1i}\varphi_{2i} + \lambda_{2i}\varphi_{1i}}{\varphi_{1i} + \varphi_{2i}} \tag{2}$$

3. DERIVATION OF THE ANALYTICAL MODEL

The average latency in the SDN architecture can be obtained by the weighted sum of the packet latencies, Lat_{hit} if the packet hits the entry of the flow table, and Lat_{miss} if the packet misses the flow entry. The packet arriving at the SDN switch can successfully find the flow entry in the flow table with the probability, ε. Then, the average latency, $Latency$, can be writen as follows:

$$Latency = \varepsilon Lat_{hit} + (1-\varepsilon)Lat_{miss} \tag{3}$$

Based on the system description in Section 2.1, Lat_{hit} and Lat_{miss} are given by

$$Lat_{hit} = D_l \tag{4}$$

$$Lat_{miss} = D_l + D_u + D_c + D_d + D_h \tag{5}$$

where D_l, D_u, D_c, D_d, and D_h are the average delays in the low priority queue, the UC queue, the controller queue, the DC queue and the high priority queue, respectively.

Directly deriving these values is intractable under the input of MMPPs, because it is difficult to calculate the additional delay for the packets in the low priority queue due to the ones in the high priority queue. In order to address this issue, inspired by the well-known Kleinrock's independence approximation [Raj et al. 2015], a novel queueing decomposition approach is developed in this study to transform a PQ system into two SSSQ systems. Therefore, instead of directly modelling the complex PQ system subject to bursty traffic, the task of the performance evaluation of SDN architecture is achieved by analysing the two but relatively simple systems, $SSSQ_l$ and $SSSQ_h$ as

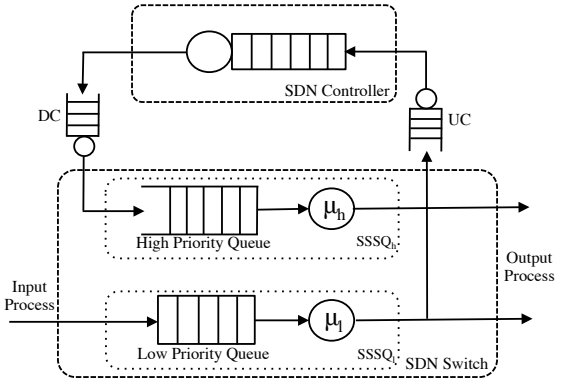


Fig. 2. Decomposition of A PQ System to Two SSSQ Systems in SDN Forwarding Devices

shown in Fig. 2. There have been a few publications appeared in the literature to investigate the queueing decomposition. For instance, Jin. et al. in [2009] proposed a method that can achieve the service rates of the two queues for the decomposed SSSQ system with the input of fractional Brownian motion (FBM) traffic. Liu et al. in [2009] further extended the decomposition method from the traffic of FBM to MMPP. These studies adopted EBA method [Mannersalo and Norros 2002] and, for analytical tractability, assumed that the high priority queue has the negligible impact on the overall queue length of the PQ system and concluded that the overall queue length of the PQ system is almost exclusively formed by the queue length of the low priority queue. As a result, the queue length distribution of the PQ system can be approximated as that of the low priority queue. Although high accuracy is achieved by these studies when the low priority queue is poured by heavy traffic and high priority queue by light traffic, they can hardly capture the comprehensive performance evaluation of the PQ system under various traffic load conditions, especially for the case of the overloaded traffic in the high priority queue. In order to bridge this gap, an enhanced EBA approach is proposed and described in the following section.

3.1. Decomposition of Priority Queue System

In order to successfully decompose a PQ system into two SSSQ systems, let us firstly analyse the relationship among the interdeparture time, the interarrival time and the sojourn time. Let $T_{a,i}$ (the subscript a represents the arrival process) denote the interarrival time between the ith and $(i+1)th$ packets arriving at the PQ system. Let $T_{d,i}$ (the subscript d represents the departure process) denote the interdeparture time between ith and $(i+1)th$ packets leaving the PQ system. Let $T_{s,i}$ and $T_{s,(i+1)}$ (the subscript s denotes the sojourn process) be the time of the ith and $(i+1)th$ packets spending at the PQ system, which includes queueing time and serving time. After simple derivation, the relationship among $T_{a,i}$, $T_{d,i}$ and $T_{s,i}$ can be denoted as $T_{d,i} = T_{s,(i+1)} - T_{s,i} + T_{a,i}$. From this equation, it can be seen that $T_{d,i}$ is determined by two parts: $T_{s,(i+1)} - T_{s,i}$ and $T_{a,i}$. Herein, $T_{a,i}$ is only determined by the input process characterised by $MMPP_l^{in}$ (the subscript l represents the low priority queue and the superscript in denotes the input traffic). As the input traffic does not change, then if the relationship among $T_{a,i}$, $T_{d,i}$ and $T_{s,i}$ is still validate after the decomposition of the PQ system, the sojourn time should be kept unchanged during the the queueing decomposition.

According to the Little's law [Gao et al. 2010], this condition can be equivalently transferred to keep the average number of packets in the PQ system unchanged. The

key issue to satisfy this condition during the PQ decomposition is to achieve the service capabilities of the $SSSQ_l$ and $SSSQ_h$, namely, μ_l and μ_h respectively. Recall that EBA can only achieve high accuracy when the low priority queue is poured with heavy traffic and high priority queue with light traffic, it can hardly capture the comprehensive performance evaluation of the PQ system under various traffic load conditions. In this study, we extend the EBA method to derive μ_l and μ_h. Since the SS provides the absolute priority for the packets in the high priority queue, the newly arriving packets in the low priority queue can hardly have impact on the serving process of the high priority queue. Therefore, the equivalent service rate of the $SSSQ_h$, μ_h, can be achieved as $\mu_h = \mu_s$. Then, the main difficulty in the decomposition of the PQ system transfers to the calculation of the equivalent service rate of the $SSSQ_l$. Inspired by EBA method, the average number of packets in the $SSSQ_l$, L_l, can be calculated by subtracting the average number of packets in the $SSSQ_h$, L_h, from the average number of packets in the PQ system, L_t. This relationship can be expressed as follows:

$$L_l = L_t - L_h \tag{6}$$

The average number of the packets in an $SSSQ_i$ (the subscript i represents different type of the queue systems) system with $MMPP_i$ traffic input can be calculated as follows [Fischer and Meier-Hellstern 1993]:

$$L_i = l_i{}^{(1)} + [\frac{1}{\lambda_i^m}\pi_i\Lambda_i - l_i](e\pi_i + Q_i)^{-1}\Lambda_i e \tag{7}$$

where $e = (1,1)^{-1}$, $\pi_i = (\pi_{1i}, \pi_{2i}) = (\varphi_{1i}, \varphi_{2i})/(\varphi_{1i}+\varphi_{2i})$; Q_i and Λ_i are the infinitesimal generator matrix and the arrival rate matrix of the $MMPP_i$ input traffic. l_i is the average queue length of the $SSSQ_i$ system; $l_i^{(1)}$ is the first moment of l_i; l_i and $l_i^{(1)}$ can be calculated based the method in [Fischer and Meier-Hellstern 1993]. For the low priority queue, after achieving the average number of packets in the $SSSQ_l$ in Eq. (6), the service rate of $SSSQ_l$, μ_l can be calculated based on the Eq. (7) through iterative algorithm, which applies a search over feasible region [0, μ_s]. The search algorithm recursively calculates the average number of packets $L_l{}'$ until satisfies $\left|L_l{}' - L_l\right| < \epsilon$, where ϵ is a small value, e.g., 10^{-8} specifying the stop condition for the recursion loop.

In what follow, we will derive the arrival traffic processes for the $SSSQ_l$, the $SSSQ_h$, and the PQ system, respectively.

3.2. Input Traffic Process of the Low Priority Queue

Since the buffer size of the $MMPP_l^{in}/M/1/K_l$ queueing system is limited as K_l for modelling the low priority queue, the packets arriving at the SDN network will be dropped when the $SSSQ_l$ is full. Let Pb_l indicate the probability that an arriving packet finds $MMPP_l^{in}/M/1/K_l$ full. The traffic effectively entering the queueing system is a fraction $(1 - Pb_l)$ of the total traffic arrived. As the splitting of an $MMPP$ generates a new $MMPP$, let $MMPP_l^{in\to e}$ represent the effective traffic entering the queueing system. $MMPP_l^{in\to e}$ can be calculated by splitting $MMPP_l^{in}$ with the probability $(1-Pb_l)$. Based on the principle of MMPP splitting process [Lee and Cho 2010], the infinitesimal generation $Q_l^{in\to e}$ and rate matrix $\Lambda_l^{in\to e}$ of $MMPP_l^{in\to e}$ can be given by

$$Q_l^{in\to e} = Q_l^{in} = \begin{bmatrix} -\varphi_{1l} & \varphi_{1l} \\ \varphi_{2l} & -\varphi_{2l} \end{bmatrix} \tag{8}$$

$$\Lambda_l^{in\to e} = (1 - Pb_l)\Lambda_l^{in} = \begin{bmatrix} (1 - Pb_l) \times \lambda_{1l} & 0 \\ 0 & (1 - Pb_l) \times \lambda_{2l} \end{bmatrix} \tag{9}$$

To compute the blocking probability, Pb_l, let us first analyse the bivariate Markov chain of the $SSSQ_l$ system. The state (s, n) in this Markov chain denotes that the Markov chain of $MMPP_l^{in}$ is in state s, ($s = \{0, 1\}$), and there are n packets in the $SSSQ_l$, ($0 \leq n \leq K_l$). The transmission rate from the state $(0, n)$ to $(1, n)$ is φ_{1l}, and the rate from state $(1, n)$ to $(0, n)$ is φ_{2l}. φ_{1l} and φ_{2l} are given by Q_l^{in}. The transmission rate from state $(0, n)$ to $(0, n + 1)$ is the packet arrival rate, λ_{1l}, and from state $(1, n)$ to $(1, n + 1)$ is λ_{2l}. λ_{1l} and λ_{2l} can be achieved from Λ_l^{in}. The rate out of the state $(s, n + 1)$ to state (s, n) is the service rate of $SSSQ_l$, μ_l. In order to calculate the blocking probability, the transmission matrix, G, should be built based on the state-transition-rate diagram. Given the number of the state of the bivariate Markov chain, $2K_l$, the dimension of the transmission matrix should be $2K_l * 2K_l$. Based on the balance equations of the bivariate Markov chain, the transmission matrix can be expressed by the transmission rates linking different states. After achieving the transmission matrix, let us calculate the steady-state probability vector $\mathbf{P}$, where $\mathbf{P} = (P(0, 0), P(0, 1), ..., P(0, K_l), P(1, 0), P(1, 1), ..., P(1, K_l))$. Let $\mathbf{e}$ be a $2K_l$ unit vector. The steady-state probability vector, $\mathbf{P}$, satisfies the following equations:

$$\mathbf{PZ} = \mathbf{0} \qquad \mathbf{Pe} = \mathbf{1} \tag{10}$$

Solving the above equations yields the probability $\mathbf{P}$ as

$$\mathbf{P} = \mathbf{u}(\mathbf{I} - \Re + \mathbf{e})^{-\mathbf{1}} \tag{11}$$

where $\Re = \mathbf{I} + \mathbf{Z}/min\{\mathbf{Z}(\rho, \rho)\}$ and $min\{\mathbf{Z}(\rho, \rho)\}$ represents the minimum number in the diagonal line of the matrix $\mathbf{Z}$. $\mathbf{u}$ denotes any row vector of the $\Re$. After achieving $\mathbf{P}$, the probability that there are n packets in the $SSSQ_l$ system, P_n, can be calculated as $P_n = \sum_{s=0}^{1} P_{s,n}$. With P_n, let us calculate, $\dot{P}_n$, which is the probability that an arriving packet observes there are n packets in the $SSSQ_l$ system. $\dot{P}_n$ is given as follows [Wu et al. 2013]:

$$\dot{P}_n = \left(\sum_{n=0}^{K_l} \mathbf{P}_n \times \Lambda_l^{in} \times \mathbf{e} \right)^{-1} \mathbf{P}_n \times \Lambda_l^{in} \times \mathbf{e} \tag{12}$$

The blocking probability is equal to the probability that a packet arrives at the system and observes that the queue is full, therefore, $Pb_l = \dot{P}_{K_l}$.

3.3. Output Traffic Process of the Low Priority Queue

The output process of the PQ system subject to an MMPP arrival process will be partially fed to SDN controller, playing critical role in deriving the input process of the high priority queue. Based on the study in [Feng and Change 2000], the output process of the priority queue no longer possesses the property of MMPP. In order to address this issue to achieve a tractable analytical model, a matching method is leveraged in this section to use an MMPP process characterised by four parameters (λ_1, λ_2, φ_1, and φ_2), to approximately model the output process of the low priority queue. We employ the selection method in [Feng and Change 2000] to choose the four statistics of the interdeparture process to match the four MMPP parameters: the first moment and the third moment of the interdeparture time, $E[T_{d,i}]$ and $E\left[T_{d,i}^3\right]$, the squared coefficient variation of the interdeparture time $c^2(T_{d,i})$, and the covariance of two successive interdeparture times $Cov(T_{d,i}, T_{d,(i+1)})$.

The moments of the inter-departure time, $T_{d,i}$, can be written as

$$E\left[T_{d,i}^{n}\right]=(-1)^{n}\left[\sum_{i=0}^{n-1}\frac{n!}{i!}L^{i}(0)x_{0}U^{-(n-1)}(0)e\right]+(-1)^{n}L^{(n)}(0), \tag{13}$$

where $U(0)=Q-\Lambda$ and $L(s)$ is the Laplace-Stieltjes transform of the service time distribution, given by $L(s)=\mu/(s+\mu)$. Let x_k denote the stationary probability that the number of the packets in the system is k once a departure occurs, and x_0 and x_1 can be achieved based on the method in [Fischer and Meier-Hellstern 1993].Then, the first three moments of the inter-departure time can be easily achieved from Eqs. (13) as follows:

$$E\left[T_{d,i}\right]=l-x_{0}U^{-1}(0)e \tag{14}$$

$$E\left[T_{d,i}^{2}\right]=L^{(2)}(0)-2lx_{0}U^{-1}(0)e+2x_{0}U^{-2}(0)e \tag{15}$$

$$E\left[T_{d,i}^{3}\right]=-L^{3}(0)-3L^{(2)}(0)x_{0}U^{-1}(0)e+6lx_{0}U^{-2}(0)e-6x_{0}U^{-3}(0)e \tag{16}$$

The squared coefficient of variation of the interdeparture time $c^2(T_{d,i})$, can be achieved from Eqs. (14) and (15), given by $c^{2}(T_{d,i})=\left(E\left[T_{d,i}^{2}\right]-\left(E\left[T_{d,i}\right]\right)^{2}\right)/\left(E\left[T_{d,i}\right]\right)^{2}$. The covariance of two successive interdeparture times is given by

$$\begin{aligned}Cov(T_{d,i},T_{d,(i+1)})=&lx_{0}U^{(-1)}(0)e-\left(x_{0}U^{(-1)}(0)e\right)^{2}+x_{1}A_{0}^{'}(0)U^{(-1)}(0)e\\&+x_{0}U^{(-1)}(0)K(0)A_{0}(0)U^{(-1)}(0)e+x_{0}K(0)A_{0}^{'}(0)U^{(-1)}(0)e\end{aligned} \tag{17}$$

where $l=1/\mu$ and $K(0)=(A-\Lambda)^{-1}\Lambda$. Let $\tilde{A}_k(x)$ denote the probability that when a departure happens there is at least one packet in the system and the next departure occurs no later than x with k arrivals during the service time. Then, the transform matrix of $\tilde{A}_k(x)$ can be calculated as follows:

$$A_{k}(s)=\int_{0}^{\infty}e^{-sx}d\tilde{A}_{k}(x)=\int_{0}^{\infty}e^{-sx}P(z,x)d\tilde{L}(x)=\int_{0}^{\infty}e^{-sx}e^{[(Q-\Lambda)+z\Lambda]x}d\tilde{L}(x) \tag{18}$$

where $P(z,x)$ is the z transformation of $P(k,x)$ and is calculated by $P(z,x)=e^{[(Q-\Lambda)+z\Lambda]x}$. $P(k,x)$ is the probability that there are k packets arriving at the system during the length of x time. Given $U(0)=Q-\Lambda$, the cumulative distribution function of exponential distribution, $\tilde{L}(x)$, is achieved by $\tilde{L}(x)=1-e^{\mu x}$. $A_0(0)$ and $A_0^{'}(0)$ can be readily derived from Eq. (18), i.e., $A_0(0)=\mu[\mu I-U(0)]^{-1}$ and $A_0^{'}(0)=-A_0(0)[\mu I-U(0)]^{-1}$. Then the four parameters of the output process of the low priority queue can be derived from the Eqs. (13)-(18) based on the method in [Feng and Change 2000].

3.4. Input Traffic Process of the High Priority Queue

Recall that the packet arriving at SDN switch has the probability, $(1-\varepsilon)$, missing the flow entry in the flow table and a proportional amount of the output traffic will be sent to the SDN controller through the UC. The traffic arriving at the UC, denoted by $MMPP_u^{in}$, is a fraction of output traffic from the $SSSQ_l$. This fraction, f_m, is equal to the miss probability as $f_m=1-\varepsilon$. Based on the Eqs. (8) and (9), the infinitesimal

generator, Q_u^{in}, and the rate matrix, Λ_u^{in} of $MMPP_u^{in}$ can be achieved. Let $MMPP_u^{out}$ be the output process of the UC queue, which is characterised by the infinitesimal generator, Q_u^{out}, and rate matrix Λ_u^{out}. Given the transmission rate of the UC, μ_u, the output process of the UC can be achieved based on the matching approach described in the Section 3.3. Since there is no packet dropped in the transmission of the UC [Kong et al. 2013], the traffic arriving at the SDN controller, $MMPP_c^{in}$ is equal to the output process, $MMPP_u^{out}$ of the UC. Since the buffer size of the SDN controller, K_c, is finite, when the packet arrives at the SDN controller, there is a probability, Pb_c, that the arriving packet is dropped when the queue becomes full. Pb_c can be obtained using the Eqs. (10)-(12). The effective traffic entering the queueing system of SDN controller is a fraction $(1 - Pb_c)$ of traffic arriving at the controller. As the splitting of an $MMPP$ is again a new $MMPP$, let $MMPP_c^{in\rightarrow e}$ denote the effective traffic entering the queueing system. $MMPP_c^{in\rightarrow e}$ can be obtained by splitting $MMPP_c^{in}$ with the probability $(1 - Pb_c)$. Following the matching approach described in Section 3.3, the output process from the $MMPP_c^{in\rightarrow e}/M/1/K_c$ queue, parameterised by Q_c^{out} and Λ_c^{out} can be readily obtained. Similar to the analysis of the UC, the input and output processes of the DC, $MMPP_d^{in}$ and $MMPP_d^{out}$ can be obtained. Again, due to infinite DC queue, the traffic arriving at the high priority queue in SDN architecture, denoted by $MMPP_h^{in}$, is the output process from the DC, $MMPP_d^{out}$.

3.5. Total Traffic Process of the Priority Queue System

Since the superposition of multiple $MMPPs$ is again an $MMPP$ [Fischer and Meier-Hellstern 1993], let $MMPP_t$ denote the superposition of the $MMPP_l$ and $MMPP_h$. The state number of the $MMPP_t$ is the production of those of the $MMPP_l$ and $MMPP_h$, which brings the complex computations in the iterative process to calculate μ_l. In order to achieve an efficient analytical model, inspired by [Feng and Chang 2001], a two-state $MMPP'_t$ is constructed in this study to approximate the $MMPP_t$. For the clarification, let m_j, v_j, p_j and $r_j(t)$ be the mean arrival rate, $E[T_{a,j}]$, the second moment, $E\left[T_{a,j}^2\right]$, the third moment, $E\left[T_{a,j}^3\right]$ and the covariance function, $Cov(T_{a,j}, T_{a,(j+1)})$ of the $MMPP_j$ (where $j = \{l, h\}$). Let τ_j denote the time constant of the $MMPP_j$ process, caculated by $\tau_j = \frac{1}{v_j}\int_0^\infty r_j(t)dt$. Then, the four parameters of $MMPP'_t$, m'_t, v'_t, p'_t and τ'_t, are given by:

$$m'_t = \sum_{j\in\{l,h\}} m_j \quad v'_t = \sum_{j\in\{l,h\}} v_j \quad p'_t = \sum_{j\in\{l,h\}} p_j \quad \tau'_t = \sum_{j\in\{l,h\}} \frac{v_j}{v'_t}\tau_j \tag{19}$$

After obtaining these four parameters, the infinitesimal generator, Q'_t, and rate matrix Λ'_t can be easily obtained based on the matching procedure in [Heffes 1980], which will be used to approximate the infinitesimal generator, Q_t, and rate matrix Λ_t of $MMPP_t$. The accuracy of this approximation will be evaluated in the Section 4. With the $MMPP_l$, $MMPP_h$ and $MMPP_t$, the service rate of $SSSQ_l$ can be calculated from the iteration process in Section 3.1.

According to [Fischer and Meier-Hellstern 1993], the average sojourn time in a $MMPP_i/M/1$ queueing system is given by

$$D_i = \frac{1}{\rho_i}\left\{\frac{1}{2(1-\rho_i)}\left[2\rho_i + \lambda_i^m h_i^{(2)} - 2h_i((1-\rho_i)g_i + h_i\pi_i\Lambda_i)(Q_i + e\pi_i)^{-1}\lambda_i\right] - \frac{1}{2}\lambda_i^m h_i^{(2)}\right\} \tag{20}$$

where ρ_i is the utilisation rate of the server, given by $\rho_i = \lambda_i^m/\mu_i$. h_i and $h_i^{(2)}$ are the mean and the second moment of the service time, given by $h_i = 1/\mu_i$ and $h_i{}^{(2)} = 2/\mu_i{}^2$,

respectively. g_i is the steady state vector of the matrix G_i and can be achieved based on the methods in [Fischer and Meier-Hellstern 1993]. The average sojourn time in the low priority queue, the UC queue, the queue in the SDN controller, the DC queue and the high priority queue, can be computed by the Eq. (20). Finally, the average latency can be achieved from the Eq. (5).

During the whole lifecycle of the packet in the PQ system, packets will be dropped in the queues of SDN switch and controller once the buffers become full. Therefore, the average throughput, $Throughput$, can be obtained as

$$Throughput = \lambda_l^m(1 - Pb_l)(\varepsilon + (1 - \varepsilon)(1 - Pb_c)) \tag{21}$$

3.6. Implementation of the Model

In order to implement the developed analytical model, an algorithm is described in this section for calculating the average latency and the average throughput of PQ-based SDN networks.

4. VALIDATION OF THE MODEL

In this section, we have developed a discrete-event simulator in the Objective Modular Network Testbed in C++ (OMNeT++) simulation environment [OMNet++ 2011] to validate the accuracy of the developed analytical model. 95% confidence intervals is adopted in this study to collect the simulation results when the simulation experiment reaches the steady state. The traffic arriving at the SDN switch follows the $MMPP_l^{in}$ process, characterised by the infinitesimal generator Q_l^{in} and the rate matrix Λ_l^{in}. Extensive simulation experiments have been conducted by varying the network parameters, including buffer sizes, switch service rate, controller service rate, UC and DC capacities, different flow table hit probability, various MMPP traffic inputs. However, for the sake of specific illustration and without loss of generality, the result comparisons between the analytical model and simulation experiments are presented in terms of average latency and average throughput with different combinations of system parameters, which are set as follows:

* Service rate of SDN switch: μ_s = 80, 40, 20 packets/second;
* Service rate of SDN controller: μ_c = 60, 30 packets/second;
* Transmission rates of UC and DC: μ_u = μ_d = 5, 10 packets/second;
* Flow table hit probability: ε = 0.5;
* Buffer sizes of the low priority queue and the controller queue, K_l = 128 packets and K_c = 256 packets;
* The infinitesimal generator, Q_l^{in}, of $MMPP_l^{in}$, representing different degrees of traffic burstiness and correlation.

$$Q_l^{in} = \begin{bmatrix} -0.3 & 0.3 \\ 0.015 & -0.015 \end{bmatrix} \quad Q_l^{in} = \begin{bmatrix} -0.09 & 0.09 \\ 0.06 & -0.06 \end{bmatrix}$$

$$Q_l^{in} = \begin{bmatrix} -0.06 & 0.06 \\ 0.03 & -0.03 \end{bmatrix} \quad Q_l^{in} = \begin{bmatrix} -0.008 & 0.008 \\ 0.004 & -0.004 \end{bmatrix}$$

Figs. 3-6 present the average latency and throughput predicted by the analytical model and simulation experiments for different network configurations. The horizontal axis represents the traffic rate, λ_{1l}^{in}, when the Markov chain of $MMPP$ process stays in state 1. For the clarity of the description, the traffic rate, λ_{2l}^{in} is set to be zero. These figures show that the developed model provides a good degree of matching with the simulation experimental results under different network configurations.

ALGORITHM 1: The procedure for calculating the average latency of PQ-based SDN networks

Input: The service rates of SDN controller, switch, UC, and DC, μ_c, μ_s, μ_u and μ_d; the buffer sizes of low priority queue and controller queue, K_l and K_c; the flow table hit probability, ε; the infinitesimal generator and rate matrix of $MMPP_l^{in}$, Q_l^{in} and Λ_l^{in}; the recursive gap, $\mu_{gap} = \mu_s$; the service rate of $SSSQ_l$, $\mu_l = \mu_{gap}$; and the recursive stop condition, $\epsilon = 10^{-8}$;

Output: The average latency and the average throughput of the PQ-based SDN networks, *Latency* and *Throughput*.

repeat

1. Calculate the blocking probability for the low priority queue, Pb_l, using Eq. (12);
2. Compute the infinitesimal generator and rate matrix for effective input process of the low priority queue, $Q_l^{in\to e}$ and $\Lambda_l^{in\to e}$, using Eqs. (8)-(9);
3. Calculate the four matching parameters, the infinitesimal generator, and the rate matrix for the output process of the low priority queue, $E\left[T_{d,i}\right]$, $E\left[T_{d,i}^2\right]$, $E\left[T_{d,i}^3\right]$, $Cov(T_{d,i}, T_{d,(i+1)})$, Q_l^{out} and Λ_l^{out}, using Eqs. (13)-(18);
4. Calculate the infinitesimal generator and the rate matrix for the input process of the UC, Q_u^{in} and Λ_u^{in}, using Eqs. (8)-(9);
5. Calculate the infinitesimal generator and the rate matrix for the output process of the UC, Q_u^{out} and Λ_u^{out}, using Eqs. (13)-(18);
6. Calculate the blocking probability of the controller queue, Pb_c, using Eq. (12);
7. Compute the infinitesimal generator and rate matrix for the effective input process of the controller queue, $Q_c^{in\to e}$ and $\Lambda_c^{in\to e}$, using Eqs. (8) and (9);
8. Calculate the infinitesimal generator and the rate matrix for the output process of the controller queue, Q_c^{out} and Λ_c^{out}, using Eqs. (13)-(18);
9. Calculate the infinitesimal generator and the rate matrix for the input process of the DC, Q_d^{in} and Λ_d^{in}, using Eqs. (8) and (9);
10. Calculate the infinitesimal generator and the rate matrix for the output process of the DC, Q_d^{out} and Λ_d^{out}, using Eqs. (13)-(18);
11. Calculate the infinitesimal generator and the rate matrix for the total traffic process of the PQ system, Q_t and Λ_t, using Eq. (19);
12. Calculate the average queue lengths of the low priority queue, the high priority queue, and the PQ system, L_l, L_h and L_t, respectively, using Eq. (7);
13. Calculate the value of the queue length difference, dif, using Eq. (6);
14. Update the recursive gap, $\mu_{gap} = \mu_{gap}/2$, and calculate the service rate of the $SSSQ_l$, μ_l;

if $dif > 0$ **then**

$\mu_l = \mu_l + \mu_{gap}$;

else

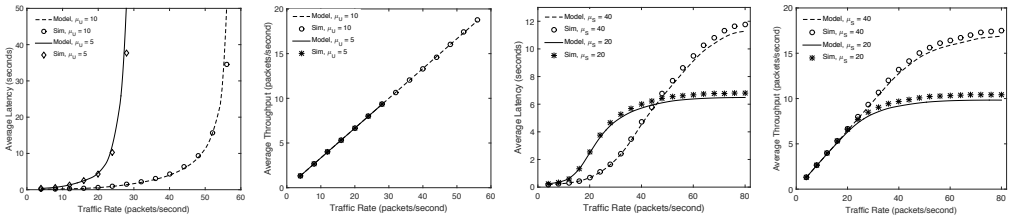
$\mu_l = \mu_l - \mu_{gap}$;

end

until $|dif| > \epsilon$;

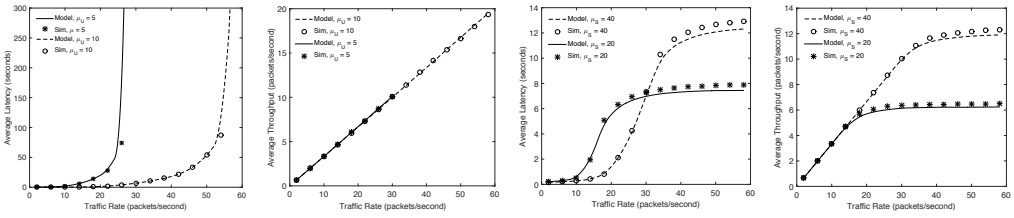
14. Calculate the average sojourn times in the low priority queue, the UC, the SDN controller, the DC, and the high priority queue, D_l, D_u, D_c, D_d, and D_h, respectively, using Eq. (20);
15. Calculate the average latency in the PQ-based SDN architecture, $Latency$, using Eqs. (3)-(5);
16. Calculate the average throughput in the PQ-based SDN architecture, $Throughput$, using Eq. (21).

In addition, from the subfigures, it can be seen that the average latency and average throughput significantly increase when the arrival traffic rate goes up (subfigures (a) and (b)). In this case, the uplink and downlink channels become the bottleneck for the system performance improvement even though SDN switch and controller have adequate service capacity. When sufficient transmission capabilities are allocated to the uplink and downlink channels, with the practical configuration of the finite buffer in the forwarding devices, the average latency and average throughput would reach a



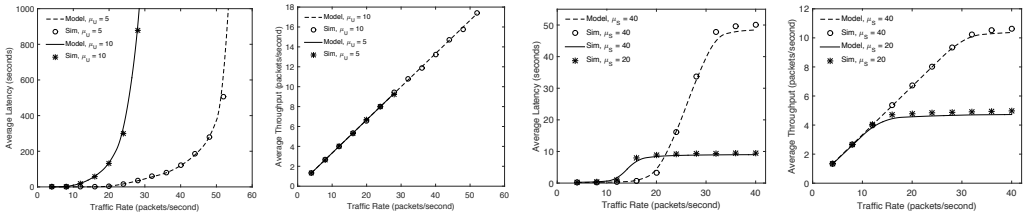
(a) (b) (c) (d)

Fig. 3. Average latency and throughput predicted by the model and simulation with $\varphi_{1l} = 0.3$, $\varphi_{2l} = 0.15$: (a) and (b) μ_u = μ_d = {5, 10}, μ_s = 80, μ_c = 60, ϵ =0.5, K_l = 128, K_c = 256; (c) and (d) μ_u =10, μ_d = 10, μ_s = {40, 20}, μ_c = 30, ϵ =0.5, K_l = 128, K_c = 256.



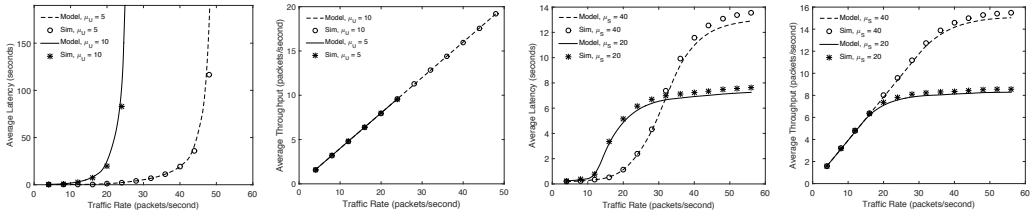
(a) (b) (c) (d)

Fig. 4. Average latency and throughput predicted by the model and simulation with $\varphi_{1l} = 0.06$, $\varphi_{2l} = 0.03$: (a) and (b) μ_u = μ_d = {5, 10}, μ_s = 80, μ_c = 60, ϵ =0.5, K_l = 128, K_c = 256; (c) and (d) μ_u =10, μ_d = 10, μ_s = {40, 20}, μ_c = 30, ϵ =0.5, K_l = 128, K_c = 256.



(a) (b) (c) (d)

Fig. 5. Average latency and throughput predicted by the model and simulation with $\varphi_{1l} = 0.008$, $\varphi_{2l} = 0.004$: (a) and (b) μ_u = μ_d = {5, 10}, μ_s = 80, μ_c = 60, ϵ =0.5, K_l = 128, K_c = 256; (c) and (d) μ_u =10, μ_d = 10, μ_s = {40, 20}, μ_c = 30, ϵ =0.5, K_l = 128, K_c = 256.



(a) (b) (c) (d)

Fig. 6. Average latency and throughput predicted by the model and simulation with $\varphi_{1l} = 0.09$, $\varphi_{2l} = 0.06$: (a) and (b) μ_u = μ_d = {5, 10}, μ_s = 80, μ_c = 60, ϵ =0.5, K_l = 128, K_c = 256; (c) and (d) μ_u =10, μ_d = 10, μ_s = {40, 20}, μ_c = 30, ϵ =0.5, K_l = 128, K_c = 256.

stable point once the incoming traffic exceeds the service capacity that the SDN switch can provide (subfigures (c) and (d)). In this case, packets will be dropped as the buffer is full, the service capacity of the switch becomes the performance bottleneck for SDN network. Therefore, in order to avoid the service degradation and Service Level Agreement (SLA) violation, the developed model can be used as a practical and cost-effective tool to gain insights into the performance of SDN networks in the presence of bursty and correlated multimedia traffic.

5. PERFORMANCE ANALYSIS

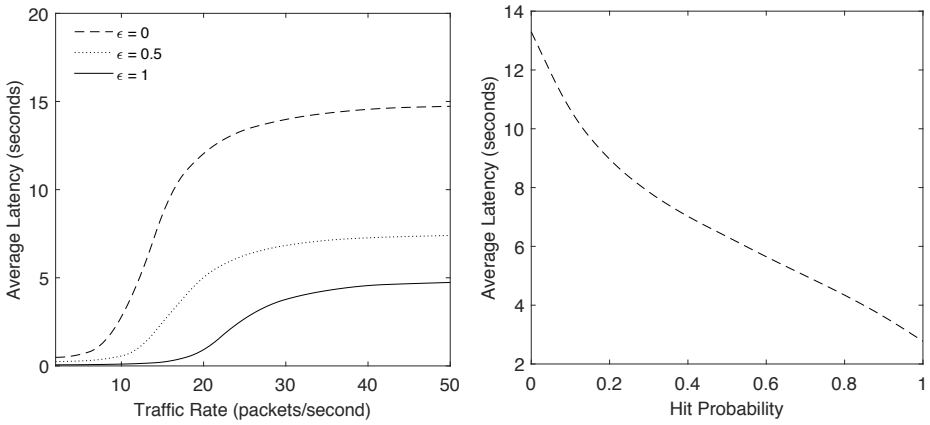
The accuracy of the proposed analytical model has been investigated in the above section. In this section, the developed model is adopted to conduct the performance evaluation of the SDN architecture.

5.1. The Effects of Flow Table Hit Probability

In order to investigate the impact of flow table hit probability on the performance of SDN networks, Fig. 7 (a) presents the average latency predicted by the developed analytical model against varying traffic arrival rates with different flow table hit probabilities (ε = 0, 0.5 and 1); the service rates of the SS, DC, SDN controller and UC, μ_s, μ_d, μ_c and μ_u are set to be 20 packets/second, 10 packets/second, 30 packets/second and 10 packets/second, respectively; and the buffer sizes of the low priority queue and SDN controller queue, K_s and K_c, are set to be 128 and 256 packets. When $\varepsilon = 1$, each new arriving packet can find the forwarding rule in the flow table, without the loop communication with the SDN controller. This case can be used to approximate the forwarding mechanism of the traditional network architecture, where network control resides in the forwarding devices. When $\varepsilon = 0$, no flow entry is stored in the flow table and the SDN network is in its initiation stage. For each arriving packet failing to match the entry in the flow table, the header of the new arrival packet will be forwarded to the controller for requesting the necessary forwarding rule. $\varepsilon = 0.5$ represents the case that 50% arriving packets can match the rules in the flow table. In addition, Fig. 7 (b) presents the average latency obtained from the model against varying flow hit probability from 0 to 1 with the fix step of 0.1. From these two figures, it can be seen that the average latency of SDN networks becomes better with the increase in the flow table hit probability and reaches the highest level when the hit probability is equal to 1. This relationship shows that the analytical model is very useful for the practical network deployment and management. For instance, in order to avoid the disruptive QoS degradation of network service in the early stage of network deployment, network routing information would be cached in the flow table in advance. During this installation process, the analytical model can be used as a cost-effective tool to quantitatively calculate the threshold of the flow table to satisfy a required network latency.

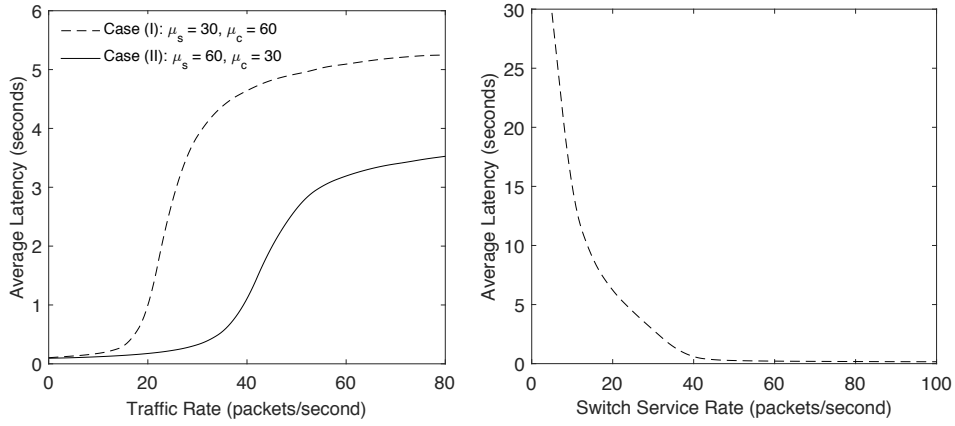
5.2. The Effects of the Resource Allocation

The efficient resource allocation plays an important role in the SDN network to provide services with the required QoS guarantee. In what follows, the impact of the resource allocation on the performance of SDN networks in terms of average latency will be investigated. For the sake of illustration, a PQ-based SDN system with two scenarios of resource allocations is firstly considered: Case (I) $\mu_c > \mu_s$ (e.g., $\mu_c = 60$, $\mu_s = 30$) and Case (II) $\mu_c < \mu_s$ (e.g., $\mu_c = 30$, $\mu_s = 60$). The infinitesimal generator of the MMPP arrivals is set as $\varphi_{1l} = 0.09$ and $\varphi_{2l} = 0.06$; the transmission rates of both the UC and DC, μ_u, μ_d, are set to be 20 packets/second; and the buffer sizes of the SDN switch queue and controller queue, K_l, K_c, are set to be 128 packets and 256 packets, respectively. The flow table hit probability, ε, is set to be 0.5. Fig. 8 (a) depicts the results of the average latency under the two cases and shows that Case (II) provides the lower



(a) (b)

Fig. 7. Impact of the flow table hit probability on the average latency: (a) $\varphi_{1l} = 0.09$, $\varphi_{2l} = 0.06$, μ_s = 20, μ_c = 30, μ_u = μ_d = 10, ϵ ={0, 0.5, 1}, K_l = 128, K_c = 256; and (b) $\varphi_{1l} = 0.09$, $\varphi_{2l} = 0.06$, μ_s = 20, μ_c = 30, μ_u = μ_d = 10, λ_{1l}^{in} = 25, λ_{2l}^{in} = 0, K_l = 128, K_c = 256.



(a) (b)

Fig. 8. Impact of the service capacity of SDN switch on the average latency: (a) $\varphi_{1l} = 0.09$, $\varphi_{2l} = 0.06$, μ_s = {30, 60}μ_c = {60, 30}, μ_u = μ_d = 20, ϵ = 0.5, K_l = 128, K_c = 256; and (b) $\varphi_{1l} = 0.09$, $\varphi_{2l} = 0.06$, μ_c = 30, μ_u = μ_d = 20, λ_{1l}^{in} = 25, λ_{2l}^{in} = 0, K_l = 128, K_c = 256.

average latency compared with that by Case (I). From the analytical model, the effective traffic entering the queue of the SDN controller, $MMPP_c^{in\rightarrow e}$, is derived through two splitting procedures of the $MMPP_l^{in}$ (one splitting is in the queue of SDN switch and the other one is in that of SDN controller) using Eqs. (8)-(12). The input of the SS is the superposition of the traffic from the $SSSQ_h$ and $SSSQ_l$, where the input of $SSSQ_h$ is statistically the output traffic from the SDN controller as there is no packet lose during the transmission from the controller to the $SSSQ_h$. It is therefore readily to see that the traffic load of the SDN switch is heavier than that of the SDN controller in the PQ-based SDN system architecture, leading to the phenomenon shown in the figure that the higher capacity allocated to the SDN switch could bring the lower average latency of the whole system. From the above analysis, we can conclude that the

SDN switch should be given more consideration on the resource allocation, i.e., the more network resources are placed in the data plane than that in the control plane, the better network performance can be obtained. Furthermore, in order to achieve a deeper understanding of the impact of the data plane on the overall system performance, the quantitative relationship between the service rate of the SDN switch and the average latency is achieved through the developed model, presented in Fig. 8 (b). It can be seen that the average latency significantly reduces with the increase in the service rate of the SDN switch from 5 packets/s to 40 packets/s. While after the point of 40 packets/s, the average latency reaches a stable point. Herein, the other parts of the SDN networks, such as the UC, DC and the SDN controller hold the dominance to the overall average latency. Therefore, in order to further improve the network performance, more resources should be allocated to the UC, DC and SDN controller. These findings are very useful in the stage of the network deployment for network operators with constrained Capital Expenditure (CAPEX) and Operational Expenditure (OPEX).

6. CONCLUSIONS

This paper has proposed an analytical model for the Software Defined Network (SDN) architecture in the presence of Markov-Modulated Poisson Process (MMPP) arrivals capturing the traffic characteristics of multimedia applications. A Priority-Queue (PQ) system has been adopted to model SDN data plane to capture the multi-queue nature of forwarding devices. A versatile methodology extending the Empty Buffer Approximation (EBA) method has been proposed to facilitate the decomposition of such a PQ system to two Single Server Single Queue (SSSQ) systems in order to facilitate the derivation and improve the tractability of the analytical model. The key performance metrics including average latency and average network throughput have been derived by the model. The accuracy of the proposed model has been validated through extensive OMNeT++ simulation experiments. The validation results have revealed that the average latency and the average throughput predicted by the developed analytical model reasonably match those obtained from the simulation experiments. The analytical model has been adopted as a cost-effective tool to study the impact of flow table hit probability and the service resource allocation in the SDN controller and the switch on the system performance.

REFERENCES

J. Abawajy. 2009. An Efficient Adaptive Scheduling Policy for High-performance Computing. *Future Gener. Comput. Syst.* 25, 3 (March 2009), 364–370. DOI:http://dx.doi.org/10.1016/j.future.2006.04.007

G. Antichi, A. A. Pietro, S. Giordano, G. Procissi, and D. Ficara. 2011. Design and Development of an OpenFlow Compliant Smart Gigabit Switch. In *Global Telecommunications Conference (GLOBECOM 2011), 2011 IEEE*. 1–5. DOI:http://dx.doi.org/10.1109/GLOCOM.2011.6133995

S. Azodolmolky, R. Nejabati, M. Pazouki, P. Wieder, R. Yahyapour, and D. Simeonidou. 2013. An Analytical Model for Software Defined Networking: A Network Calculus-based Approach. In *Global Communications Conference (GLOBECOM), 2013 IEEE*. 1397–1402. DOI:http://dx.doi.org/10.1109/GLOCOM.2013.6831269

A. Bianco, R. Birke, L. Giraudo, and M. Palacin. 2010. OpenFlow Switching: Data Plane Performance. In *Communications (ICC), 2010 IEEE International Conference on*. 1–5. DOI:http://dx.doi.org/10.1109/ICC.2010.5502016

M. Chowdhury, M. Zaharia, J. Ma, M. Jordan, and I. Stoica. 2011. Managing Data Transfers in Computer Clusters with Orchestra. *SIGCOMM Comput. Commun. Rev.* 41, 4 (2011), 98–109. DOI:http://dx.doi.org/10.1145/2043164.2018448

Cisco. 2015. Visual Networking Index: Forecast and Methodology, 2014–2019. (2015). http://www.cisco.com/c/en/us/solutions/collateral/service-provider/ip-ngn-ip-next-generation-network/white_paper_c11-481360.html

P. Congdon, P. Mohapatra, M. Farrens, and V. Akella. 2014. Simultaneously Reducing Latency and Power Consumption in OpenFlow Switches. *Networking, IEEE/ACM Transactions on* 22, 3 (June 2014), 1007–1020. DOI:http://dx.doi.org/10.1109/TNET.2013.2270436

A. Das, C. Lumezanu, Y. Zhang, V. Singh, G. Jiang, and C. Yu. 2013. Transparent and Flexible Network Management for Big Data Processing in the Cloud. In *Presented as part of the 5th USENIX Workshop on Hot Topics in Cloud Computing*. USENIX, San Jose, CA. https://www.usenix.org/conference/hotcloud13/workshop-program/presentations/Das

M. Dong, H. Li, K. Ota, and J. Xiao. 2015. Rule Caching in SDN-enabled Mobile Access Networks. *IEEE Network* 29, 4 (July 2015), 40–45. DOI:http://dx.doi.org/10.1109/MNET.2015.7166189

S. Fang, Y. Yu, C. Heng Foh, and K.M.M. Aung. 2013. A Loss-Free Multipathing Solution for Data Center Network Using Software-Defined Networking Approach. *Magnetics, IEEE Transactions on* 49, 6 (June 2013), 2723–2730. DOI:http://dx.doi.org/10.1109/TMAG.2013.2254703

H. Feng and J. Chang. 2001. Departure Processes of BMAP/G/1 Queues. *Queueing Syst. Theory Appl.* 39, 2/3 (Oct. 2001), 109–135. DOI:http://dx.doi.org/10.1023/A:1012786932415

H. Feng and J. Change. 2000. Connection-Wise End-to-End Delay Analysis in ATM Networks. *Communications, IEICE Transactions on* 83, 3 (March 2000), 659–671. http://ci.nii.ac.jp/naid/110003218675/en/

W. Fischer and K. Meier-Hellstern. 1993. The Markov-modulated Poisson process (MMPP) Cookbook. *Performance Evaluation* 18, 2 (1993), 149 – 171. DOI:http://dx.doi.org/10.1016/0166-5316(93)90035-S

P. Gao, S. Wittevrongel, K. Laevens, D. Vleeschauwer, and H. Bruneel. 2010. Distributional Little's Law for Queues with Heterogeneous Server Interruptions. *Electronics Letters* 46, 11 (May 2010), 763–764. DOI:http://dx.doi.org/10.1049/el.2010.3207

H. Heffes. 1980. A Class of Data Traffic Processes-covariance Function Characterization and Related Queuing Results. *Bell System Technical Journal* 59, 6 (July 1980), 897–929. DOI:http://dx.doi.org/10.1002/j.1538-7305.1980.tb03039.x

H. Hu, Y. Wen, Y. Gao, T. Chua, and X. Li. 2015. Toward an SDN-enabled big data platform for social TV analytics. *Network, IEEE* 29, 5 (September 2015), 43–49. DOI:http://dx.doi.org/10.1109/MNET.2015.7293304

M. Jarschel., S. Oechsner, D. Schlosser, R. Pries, S. Goll, and P. Tran-Gia. 2011. Modeling and Performance Evaluation of an OpenFlow Architecture. In *Teletraffic Congress (ITC), 2011 23rd International*. 1–7.

R. Kapoor, A. Snoeren, G. Voelker, and G. Porter. 2013. Bullet Trains: A Study of NIC Burst Behavior at Microsecond Timescales. In *Proceedings of the Ninth ACM Conference on Emerging Networking Experiments and Technologies (CoNEXT '13)*. ACM, New York, NY, USA, 133–138. DOI:http://dx.doi.org/10.1145/2535372.2535407

A. Khan and N. Dave. 2013. Enabling Hardware Exploration in Software-Defined Networking: A Flexible, Portable OpenFlow Switch. In *Field-Programmable Custom Computing Machines (FCCM), 2013 IEEE 21st Annual International Symposium on*. 145–148. DOI:http://dx.doi.org/10.1109/FCCM.2013.15

X. Kong, Z. Wang, X. Shi, X. Yin, and D. Li. 2013. Performance Evaluation of Software-defined Networking with Real-life ISP Traffic. In *Computers and Communications (ISCC), 2013 IEEE Symposium on*. 000541–000547. DOI:http://dx.doi.org/10.1109/ISCC.2013.6755002

H. Lee and D. Cho. 2010. Capacity Improvement and Analysis of VoIP Service in a Cognitive Radio System. *Vehicular Technology, IEEE Transactions on* 59, 4 (May 2010), 1646–1651. DOI:http://dx.doi.org/10.1109/TVT.2009.2039503

H. Li, M. Dong, X. Liao, and H. Jin. 2014. Deduplication-Based Energy Efficient Deduplication-Based Energy Efficient Storage System in Cloud Environment. *Computer Journal* (December 2014). DOI:http://dx.doi.org/10.1093/comjnl/bxu122

H. Li, M. Dong, and K. Ota. 2015. Radio Access Network Virtualization for the Social Internet of Things. *IEEE Cloud Computing* 2, 6 (November 2015), 42–50. DOI:http://dx.doi.org/10.1109/MCC.2015.114

K. Liu, X. Ling, and J.W. Mark. 2008. Performance Analysis of Prioritized MAC in UWB WPAN With Bursty Multimedia Traffic. *Vehicular Technology, IEEE Transactions on* 57, 4 (July 2008), 2462–2473. DOI:http://dx.doi.org/10.1109/TVT.2007.912139

Z. Liu, X. Wang, W. Pan, B. Yang, X. Hu, and J. Li. 2015. Towards Efficient Load Distribution in Big Data Cloud. In *Computing, Networking and Communications (ICNC), 2015 International Conference on*. 117–122. DOI:http://dx.doi.org/10.1109/ICCNC.2015.7069326

K. Mahmood, A. Chilwan, O. Østerbø, and M. Jarschel. 2015. Modelling of OpenFlow-based software-defined networks: the multiple node case. *Networks, IET* 4, 5 (2015), 278–284. DOI:http://dx.doi.org/10.1049/iet-net.2014.0091

P. Mannersalo and I. Norros. 2002. A Most Probable Path Approach to Queueing Systems with General Gaussian Input. *Computer Networks* 40, 3 (2002), 399 – 412. DOI:http://dx.doi.org/10.1016/S1389-1286(02)00302-X

B. Mark and Y. Ephraim. 2014. Explicit Causal Recursive Estimators for Continuous-Time Bivariate Markov Chains. *Signal Processing, IEEE Transactions on* 62, 10 (May 2014), 2709–2718. DOI:http://dx.doi.org/10.1109/TSP.2014.2314434

W. Miao, G. Min, Y. Wu, and H. Wang. 2015. Performance Modelling of Preemption-based Packet Scheduling for Data Plane in Software Defined Networks. In *Proceedings of the 2015 IEEE International Conference on Smart City*. IEEE.

J. Naous, D. Erickson, G.A. Covington, G. Appenzeller, and N. McKeown. 2008. Implementing an OpenFlow Switch on the NetFPGA Platform. In *Proceedings of the 4th ACM/IEEE Symposium on Architectures for Networking and Communications Systems (ANCS '08)*. ACM, New York, NY, USA, 1–9. DOI:http://dx.doi.org/10.1145/1477942.1477944

OMNet++. 2011. *OMNeT++ Network Simulator*. http://www.omnetpp.org/

ONF. 2012. *Software-Defined Networking: The New Norm for Networks*. Technical Report. Open Network Foundation. https://www.opennetworking.org/

ONF. 2013. *OpenFlow Switch Specification Version 1.3.3*. Technical Report. Open Network Foundation. www.opennetworking.org/

P. Qin, B. Dai, B. Huang, and G. Xu. 2015. Bandwidth-Aware Scheduling With SDN in Hadoop: A New Trend for Big Data. *Systems Journal, IEEE* PP, 99 (2015), 1–8. DOI:http://dx.doi.org/10.1109/JSYST.2015.2496368

P. Raj, A. Raman, D. Nagaraj, and S. Duggirala. 2015. *High-Performance Big-Data Analytics*. Springer International Publishing. DOI:http://dx.doi.org/10.1007/978-3-319-20744-5

A. Sadasivarao, S. Syed, P. Ping, C. Liou, I. Monga, G. Chin, and A. Lake. 2013. Bursting Data between Data Centers: Case for Transport SDN. In *High-Performance Interconnects (HOTI), 2013 IEEE 21st Annual Symposium on*. 87–90. DOI:http://dx.doi.org/10.1109/HOTI.2013.20

J. Tate, P. Beck, P. Clemens, S. Freitas, J. Gatz, M. Girola, J. Gmitter, H. Mueller, R. Hanlon, and J. Walker. 2013. *IBM and Cisco: Together for a World Class Data Center*. IBM Redbooks publication.

G. Wang, T.S. Ng Eugene, and A. Shaikh. 2012. Programming Your Network at Run-time for Big Data Applications. In *Proceedings of the First Workshop on Hot Topics in Software Defined Networks (HotSDN '12)*. ACM, New York, NY, USA, 103–108. DOI:http://dx.doi.org/10.1145/2342441.2342462

Y. Wu, G. Min, and L. T. Yang. 2013. Performance Analysis of Hybrid Wireless Networks Under Bursty and Correlated Traffic. *Vehicular Technology, IEEE Transactions on* 62, 1 (January 2013), 449–454. DOI:http://dx.doi.org/10.1109/TVT.2012.2219890

S. Yeganeh, A. Tootoonchian, and Y. Ganjali. 2013. On Scalability of Software-defined Networking. *Communications Magazine, IEEE* 51, 2 (February 2013), 136–141. DOI:http://dx.doi.org/10.1109/MCOM.2013.6461198