Journal of Network and Computer Applications 113 (2018) 119–129

Contents lists available at ScienceDirect

Journal of Network and Computer Applications

journal homepage: www.elsevier.com/locate/jnca

Performance modeling and comparison of NFV integrated with SDN: Under or aside?

Ahmed Fahmin [a], Yuan-Cheng Lai [b], Md. Shohrab Hossain [a,*], Ying-Dar Lin [c]

[a] Department of Computer Science and Engineering, Bangladesh University of Engineering and Technology, Bangladesh
[b] Department of Information Management, National Taiwan University of Science and Technology, Taipei, Taiwan
[c] Department of Computer Science, National Chiao Tung University, Hsinchu, Taiwan

ARTICLE INFO

Keywords:
Software-defined networking
Network function virtualization
M/M/1 queuing model
OpenFlow

ABSTRACT

Software Defined Networking (SDN) focuses on the separation of data and control plane, while network function virtualization (NFV) decouples network functions from underlying hardware. Combining SDN with NFV would have many benefits, but the problem is how to integrate them. There are two possible architectures for such integration: the controller interacts with virtualized network functions (VNFs), or the switch interacts with VNFs. In this paper, the former is referred to as NFV under the controller (NFV_C) while the latter is called NFV aside the controller (NFV_AC). To the best of our knowledge, there is no analytical model for mathematically investigating the performance of such architectures. This paper therefore reports on the analytical modeling of SDN with NFV under or aside the controller. We model and analyze these two SDN + NFV architectures using an M/M/1 queuing model and validate our analysis with various simulations. We show that the analytical results match the simulation results very well. Furthermore, a packet delay reduction of 68.83% can be achieved for NFV_AC over NFV_C, meaning that NFV_AC is a better architecture for integrating SDN with NFV. We also consider feedback from the VNF to the switch. For low loads there is delay increase of 35.42% for high feedback (90%) in comparison with low feedback (10%). In the case of high loads we recorded a delay increase of 66.64%.

1. Introduction

Traditionally a network is built on dedicated hardware, such as routers and switches, with network software provided by the network vendor. A network engineer's capacity to modify the network software is considerably constrained and is for the most part confined by the network vendor. This has prompted the idea of Software Defined Networking (SDN), where adaptability and dynamism have been presented in the virtualization of the control plane (Rowshanrad et al., 2014; Hakiri et al., 2014; Pan et al., 2011; Hu et al., 2014). The fundamental approach is to separate a network into a control plane and a data plane, which then have the capacity to oversee different network devices centrally. The primary advantage of the SDN concept is the programmability of controlling the network devices. This has empowered network engineers to change network configurations and the rationale of data flow as a business requires. A control plane communicates to a data plane through a southbound interface, commonly known as OpenFlow protocol, which was first proposed in (McKeown et al., 2008) and subsequently standardized by the Open Networking Foundation (ONF). Network Function Virtualization (NFV) is however a new approach which deploys or designs various different network functions (Han et al., 2015; Mijumbi et al., 2016). It decouples the network functions, so as NAT, DNS caching and so forth from their proprietary hardware appliances, so that they can be executed in virtual machines, enhancing their service quality.

SDN concentrates on the partition of the network control plane from the physical routers' data plane. SDN is dealt with as the control software that sits atop a cluster of physical devices with which it communicates through interfaces. There are several works (Jarschel et al., 2011; Azodolmolky et al., 2013a, 2013b; Mahmood et al., 2014, 2015; Wang et al., 2015; Xiong et al., 2016; Bozakov and Rizk, 2013; Goto et al., 2016; Miao et al., 2015, 2016) on SDN modeling, but none of these has considered NFV in their analytical modeling. On the other hand, NFV is tied with virtualizing various resources into network functions (in

* Corresponding author.
E-mail addresses: ahmedfahmin@gmail.com (A. Fahmin), laiyc@cs.ntust.edu.tw (Y.-C. Lai), mshohrabhossain@cse.buet.ac.bd (Md.S. Hossain), ydlin@cs.nctu.edu.tw (Y.-D. Lin).

https://doi.org/10.1016/j.jnca.2018.04.003
Received 7 December 2017; Received in revised form 5 March 2018; Accepted 10 April 2018
Available online 14 April 2018

software), so that we need not be concerned about any specific physical devices devoted for specific network functions. SDN virtualizes the control plane and configures the data plane, while NFV virtualizes the data plane that are connected to switches.

Recent works (Lin et al., 2015, 2016) have shown that both SDN and NFV can be consolidated to give more centralized control software and generic hardware where the utilities of SDN can be acknowledged through virtualized robust network functions provided by NFV. Two recent works (Lin et al., 2015, 2016) have investigated the performance of coexisting SDN and NFV architecture using simulation and experimentation. However, no analytical model has been developed for the SDN architecture combined with NFV. This work is the *first* to model the architecture that combines SDN with NFV, as well as analyzing its performance.

There are different ways to deal with consolidating SDN with NFV. Since a controller determines which instance of a virtualized network function (VNF) serves the packets which need network functions (NFV packets for short), the NFV packets can be redirected to the controller from the switch. The controller then advances these packets to the proper VNF to execute the required network functions - the NFV packets will pass through the controller. This approach is called SDN with NFV under the controller (NFV_C). Another approach is that some of the NFV packets are sent to the controller, the controller determines which instance of the VNF will serve these packets, and will set in motion the appropriate actions for the switch. Consequent the NFV packets belonging to the same flow can be directly redirected to the determined instance of VNF without the controller. We term this approach SDN with NFV aside the controller (NFV_AC).

The fundamental *objective* of this work is to carry out analytical modeling of NFV_C and NFV_AC. We also provide a comparison between the two architectures based on the delay of NFV packets. Our work covers: (i) developing the analytical models for the two architectures, (ii) carrying out simulations to validate our analytical model, and (iii) making sensitivity analysis of certain system parameters (such as arrival rates, service rates, etc.) on the performance of these two architectures.

The rest of the paper is organized as follows. In Section 2, we briefly explain the two SDN architectures along with related works on SDN modeling; in Section 3 we present our analytical models for the two SDN architectures, and in Section 4 we present the analytical and simulation results. Finally, Section 5 contains the concluding remarks.

2. Preliminaries

Here we briefly explain the two SDN architectures where NFV is under or aside the controller, followed by the current work on SDN modeling.

In the NFV_C architecture shown in Fig. 1, the controller interacts directly with VNFs. The overall procedure is as follows. First, the NFV packet enters the switch. Then, depending on the action in the flow table, the NFV packet is forwarded to the controller. The controller will determine which instance of VNF serves it and forwards it to the selected instance. After the NFV packet receives its required network function, it returns to the controller and the controller sends it back the switch. Since the action in the flow table is redirected to the controller, all NFV packets belonging to the same flow will be sent to the controller. The main advantage of this architecture is that NFV packets can be dispatched to different instances of VNF for load balancing.

In NFV_AC architecture, shown in Fig. 2 (Lin et al., 2015), the switch interacts directly with VNFs. There is a service chaining module, which selects the proper instances of VNFs and determines the order of chaining. Service chaining module communicates with the controller via northbound interfaces. In this architecture, the controller is responsible for extracting network events, collecting statistics, and analyzing

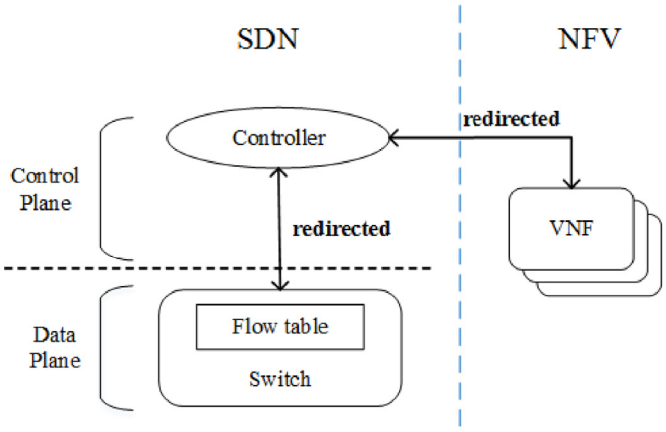

Fig. 1. Traditional SDN architecture where NFV is under controller.

payload to select the proper instances of VNFs and their chaining to support NFV. If table miss of a NFV packet happens, the packet visits the service chaining module through the controller. After receiving a response from the service chaining module, the controller initiates the proper action for the switch and sends this NFV packet back the switch. This packet, experiencing the service chaining module, still needs to go to VNF from the switch to obtain its required network function. Subsequent NFV packets belonging to the same flow can be directly redirected from the switch to determined instances of VNF without the controller. The main advantage of this architecture is that most NFV packets can be directly forwarded to VNF, significantly reducing the controller's loading. However most NFV packets belonging to the same flow will be forwarded to the same instance. Thus the instances of VNFs are not load balanced, resulting in a greater delay in providing network functions. In a real system, there is still a need to have a control plane module on the controller side, doing service chaining to configure the paths among switches and VNF modules for the data plane traffic. The data plane VNF modules of course would be placed to switches on the data plane.

NFV_C serves as the baseline architecture to see how much NFV_AC would differ. Many applications would split their functions into control plane placed on the controller side, i.e., as NFV_C, and data plane placed on the switch side, i.e., as NFV_AC. If a significant percentage of functions need to be placed as NFV_C, the performance impact needs to be evaluated. Thus, NFV_C serves not only as the baseline architecture but also could be used to evaluate the performance impact of functions implemented in the control plane.

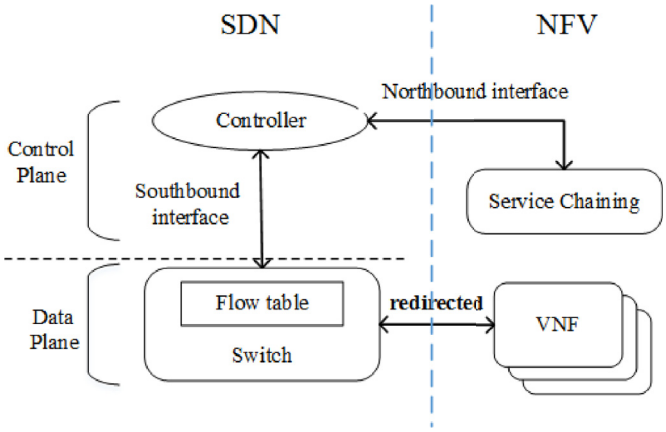

Fig. 2. SDN architecture where NFV is aside controller.

2.1. Related works on SDN modeling

There have been a few works on the analytical modeling of SDN. In Table 1 we summarize the key points of such works, together with their methodologies and performances matrices.

First, modeling on SDN was carried out in (Jarschel et al., 2011), where feedback orientated queueing theory was used to show the interaction between the control plane and the data plane. In (Azodolmolky et al., 2013a), network calculus was used to develop an analytical model of an SDN network. This work developed the buffer size bound and packet delay bound.

Mahmood et al. (2014) presented an improvement on (Jarschel et al., 2011) by modeling SDN as a modified Jackson network. They estimated the packet rate from the controller to the switch, so that the overall packet arrival rate into the switch could actually be obtained. They further extended their previous work to propose an analytical model for the SDN with multiple switches (Mahmood et al., 2015). In this model they calculated the average path delay from the source to the destination, rather than the average packet delay in a switch.

Wang et al. (2015) adopted the concept of hierarchical-controller architecture, which has a root controller and some local controllers for improving flexibility of the control plane. With this architecture, they analyzed the average packet delay spent in the controllers. However, this paper did not consider switches. On the other hand, Xiong et al. (2016) thought that packet arrivals should have batch characteristics, be considered as a Poisson process. Thus, their model is of packet batch arrivals following a Poisson process, with the number of packets in a batch conforming to a Poisson distribution. Finally, they model the behaviors of switches and the controller as the queueing systems M^X/M/1 and M/G/1, respectively.

Bozakov et al. (Bozakov and Rizk, 2013) used a queueing model to characterize the behavior of the control interface between the controller and a switch in terms of the number of serviced messages over different time scales, and provided a calculus-based approach to derive an estimate of the corresponding service curves. They also proposed a simple interface extension for controller frameworks which enabled operators to configure time delay bounds for transmitted control messages. However, the model does not consider the feedback between the data plane and the control plane. This shortcoming of feedback modeling is addressed by Azodolmolky et al. (2013b) who derived a mathematical framework based on network calculus to deal with scalable SDN deployment. Assuming cumulative arriving process at SDN controller, they evaluated the upper bound of the transmission latency and buffer sizing of the root SDN controller.

Goto et al. (2016) considered the switches to have two queues: a high-priority queue for those packets sent back from the controller and a low-priority queue for newly-arrived packets from other switches. Those packets coming back from the controller then have a higher priority for delivery in order to reduce their delay. A three-dimensional state (controller's queue length, high-priority queue length, and low-priority queue length) was created to represent this system. The authors derived state transition probabilities and tried to reduce the complexity of obtaining the steady state probability.

Miao et al. (2015) proposed a pre-emption-based (PQ system) packet-scheduling scheme to improve the global fairness and reduce the packet loss rate in SDN data plane. They also developed an analytical model to quantitatively evaluate the scheduling scheme and pinpoint performances bottleneck in the SDN architecture. They assumed two queues in the switch: one with low priority (with infinite buffer), and another with high priority (with finite buffer); for controller a M/M/1 queue with finite buffer is used.

Miao et al. (2016) considered the realistic nature of multimedia traffic and used a Markov Modulated Poisson Process (MMPP) to model a burst of packet arrivals. They also adopted two queues: a high-priority queue and a low-priority queue in the switch, which is similar to that in (Goto et al., 2016). They solved this problem by using MMPP/M/1 for the high-priority queue and MMPP/M/1/k for the low-priority queue.

Other than the SDN modeling papers (Jarschel et al., 2011; Azodolmolky et al., 2013a, 2013b; Mahmood et al., 2014, 2015; Wang et al., 2015; Xiong et al., 2016; Bozakov and Rizk, 2013; Goto et al., 2016; Miao et al., 2015, 2016) listed in Table 1, there are a few works which had performed SDN implementations and simulations, no mathematical analyses were performed. Kuo et al. (2017) considered a service chain consisting of a sequence of virtualized network functions (VNFs). They showed that the process overhead on a computation node is linear to the total amount of flows processed. They have considered flow based arrival and developed an approximation algorithm of service chain embedding. Hawilo et al. (2014) considered a NFV framework and examined the challenges and prerequisites of its use in cellular net-

Table 1
Related works on SDN modeling.

Paper	Device	#	Methodology	Performance metric
(Jarschel et al., 2011)	Controller	1	M/M/1/k	Avg. packet delay
	Switch	1	M/M/1	Avg. packet delay
(Azodolmolky et al., 2013a)	Controller	1	Net. Calculus	Buffer size bound
	Switch	N	Net. Calculus	Packet delay bound
(Mahmood et al., 2014)	Controller	1	M/M/1	Avg. packet delay
	Switch	1	M/M/1 (adjusted λ)	Avg. packet delay
(Mahmood et al., 2015)	Controller	1	M/M/1/k	Avg. packet delay
	Switch	N	M/M/1	Avg. path delay
(Wang et al., 2015)	Root controller	1	M/M/1	Avg. packet delay
	Local controller	N	M/M/1/k	Avg. packet delay
(Xiong et al., 2016)	Controller	1	M/G/1	Avg. packet delay
	Switch	N	M^X/M/1	Avg. packet delay
(Bozakov and Rizk, 2013)	Controller	1	Net. Calculus	Message delay bound
	Switch	1	Net. Calculus	
(Azodolmolky et al., 2013b)	Controller	1	Net. Calculus	Buffer size bound
	Switch	N	Net. Calculus	Packet delay bound
(Goto et al., 2016)	Controller	1	3D state (controller, HPQ, LPQ)	Avg. packet delay
	Switch	1		Avg. packet delay Packet loss prob.
(Miao et al., 2015)	Controller	1	M/M/1/m	Packet loss probability
	Switch	1	HPQ: M/M/1/m	Avg. packet delay
			LPQ: M/M/1/m	Avg. throughput
(Miao et al., 2016)	Controller	1	MMPP/M/1	Ave. packet delay
	Switch	1	HPQ: MMPP/M/1	Avg. packet delay
			LPQ: MMPP/M/1/k	Avg. throughput

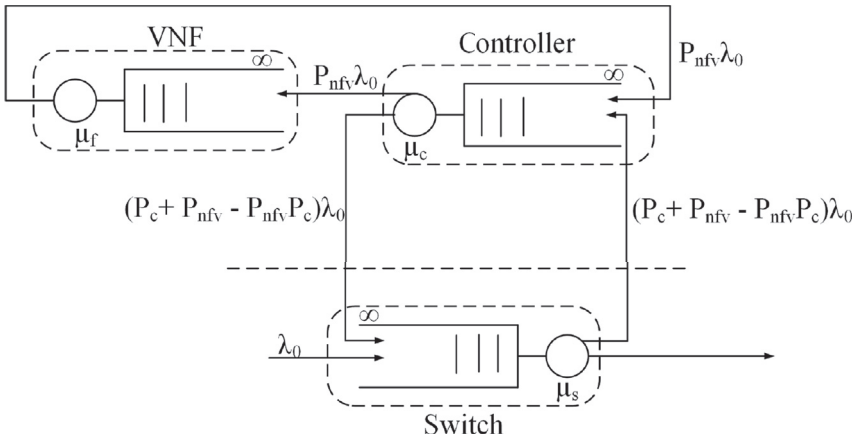

Fig. 3. Case I: queueing model for NFV_C.

works by proposing virtual environment. They proposed a criterion to bundle different functions of a virtualized evolved packet core in a single physical device or a group of adjacent devices to decrease signaling traffic and accomplish better performance. Muhammad et al. (Anan et al., 2016) explained technologies, architectures, applications, underlying protocols, and deployments of SDN network. Omnes et al. (2015) proposed a multi-layered IoT architecture involving SDN and NFV to cope with different challenges in IoT ecosystem. Ferrús et al. (2016) explained how SDN/NFV technologies can be collaborated with satellite communications in order to implement 5G. In the satellite ground segment domain, they proposed three scenarios and showed the possibilities of improvement after introducing SDN/NFV technologies. Basta et al. (2014) explained the functions placement problem within a widely-spanned mobile network. An NFV deployment with all functions virtualized will be helpful for some areas of the mobile network, while for other areas, an SDN deployment with functions decomposition is more favorable in terms of load and delay. They proposed a model to deal with functions placement problem and minimize load overhead of transport network. Yuan et al. (Zhang et al., 2018) explained that in terms of performance and scalability a single controller of SDN has many drawbacks. They investigated the detailed design architectures of SDN with multiple controllers and benefits and challenges of multiple controllers. Lorenz et al. (2017) explained that dynamic resource allocation of the network can cause serious security issue. For enterprise networks, they proposed different architectural design patterns which provide a SDN/NFV-based security solutions. Rahim et al. (Masoudi and Ghaffari, 2016) investigated the three planes of SDN such as data plane, control plane and application plane and examined the challenges and the latest technologies related to SDN. Deval et al. (Bhamare et al., 2016) demonstrated the importance of service function chaining (SFC) as a major enabler for NFV and its importance to improve the performance of NFV. They also developed optimization strategies for SFC architecture providing a flexible and economical approach.

However, none of the SDN modeling works (Jarschel et al., 2011; Azodolmolky et al., 2013a, 2013b; Mahmood et al., 2014, 2015; Wang et al., 2015; Xiong et al., 2016; Bozakov and Rizk, 2013; Goto et al., 2016; Miao et al., 2015, 2016) considered both SDN and NFV in their models. In this paper, we have developed analytical models for combining SDN with NFV. To the best of our knowledge, this work is the *first* that considers the presence of NFV while modeling the performance of SDN.

3. System model

We have used classical queueing theory to develop mathematical models for the NFV_C (case I) and NFV_AC (case II). The queueing

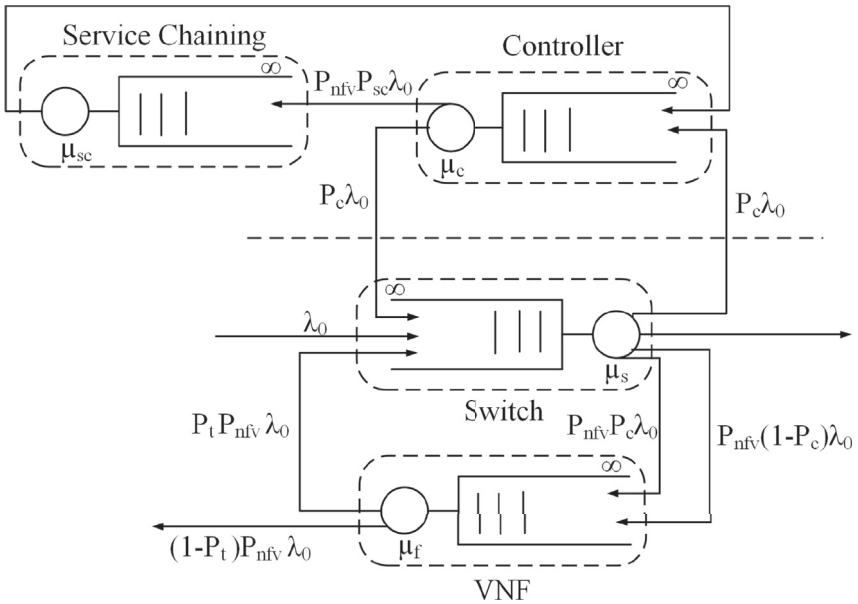

Fig. 4. Case II: queueing model for NFV_AC.

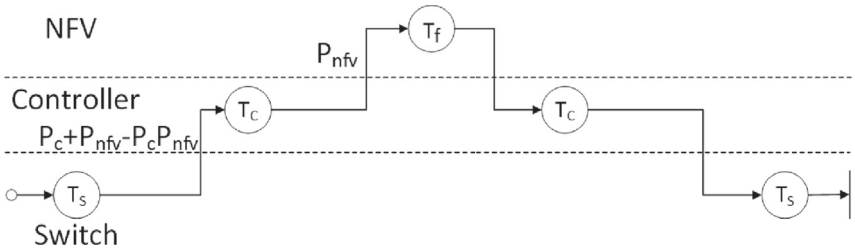

Fig. 5. Phase diagram of delays for NFV packets for NFV_C.

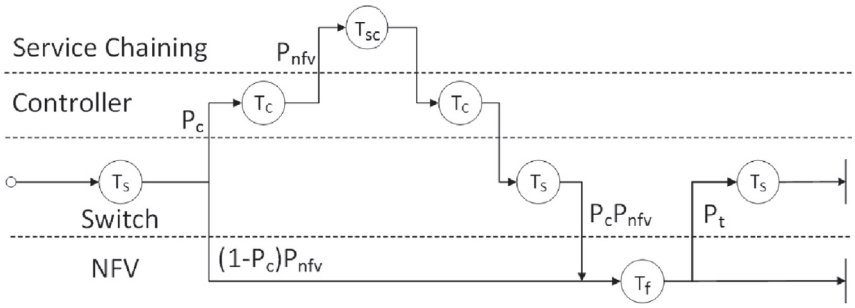

Fig. 6. Phase diagram of delays for NFV packets for NFV_AC.

model of NFV_C is shown in Fig. 3, where it has three M/M/1 queues. For NFV_AC the model has four queues, as shown in Fig. 4.

The flow of NFV packets in the two SDN architectures are illustrated through the phase diagrams in Figs. 5 and 6. These show the way a NFV packet progresses through the system in two cases. T_s, T_c, T_f and T_{sc} represent the average packet delay at the switch, controller, VNF and service chaining module, respectively.

3.1. Assumptions and notations

Following are the assumptions of the model:

- The data arrival process at a switch is a Poisson process.
- The service time of packets in a switch, controller and VNF are assumed to follow exponential distributions.
- For switch, controller, VNF and service chaining module, the queue size is infinite.

For the analytical model we consider M/M/1 queues for switch, controller, VNF and service chaining module independently, but arrival rates of these queues depend on the feedback of other queues.

The notations used in the analysis are listed in Table 2. To denote different parameters, we have used superscript 1 and 2 for NFV_C and NFV_AC, respectively. Moreover, we have used subscript X to indicate devices where X can be replaced by s, c, f and sc for switch, controller, VNF and service chaining module, respectively.

3.2. Analysis for NFV_C

To calculate the average packet delay for NFV packets, we first calculate the average packet delay in different queues. For a packet requiring NFV service, at first the packet will arrive at the switch. From there, the packet will go to the controller. If the packet requires service from the VNF module, the packet will be forwarded to the VNF module from the controller. From VNF module, the packet will return back to the controller and finally from the controller, it returns to the switch.

Fig. 3 shows two types of packets entering the switch: (i) new packets arriving at the switch, and (ii) packets that are redirected from the controller go back to the switch. For the former, the packet arrival rate is λ_0, and for the latter, it can be carefully obtained as follows.

First, we obtain the arrival rate of NFV packets as $P_{nfv}\lambda_0$, because the probability of packets requiring a network function is assumed to be P_{nfv}; then, the rate of packets sent to the controller from the switch (as a result of table miss) is $P_c\lambda_0$. However, these two rates slightly overlap because some NFV packets encounter table miss. Thus, we should subtract the probability of their intersection. Therefore, total arrival rate sent from the switch to the controller is $(P_c + P_{nfv} - P_cP_{nfv})\lambda_0$. This rate is also the rate sent from the controller to the switch because we assume, the controller has an infinite buffer having no loss. Some packets will go to the VNF module from the controller, but those packets will eventually return to the controller from VNF module. Finally, packets will return to the switch. Hence, we have obtained that all the packets that entered the controller from the switch will return to the switch after visiting the controller. Finally, the total packet rate entering into the switch is expressed as follows:

$$\begin{aligned}\Lambda_s^{(1)} &= \lambda_0 + (P_c + P_{nfv} - P_cP_{nfv})\lambda_0 \\ &= (1 + P_c + P_{nfv} - P_cP_{nfv})\lambda_0.\end{aligned} \tag{1}$$

Table 2
Notations used in the analysis.

Symbol	Parameter Name
λ_0	Arrival rate of packets at the switch
$\Lambda_X^{(1)}$	Total arrival rate at X ϵ {s, c, f} for NFV_C
$\Lambda_X^{(2)}$	Total arrival rate at X ϵ {s, c, f, sc} for NFV_AC
P_c	The probability of redirecting to the controller from switch
P_{nfv}	Probability that a packet will require service from VNF module
P_t	The probability of redirecting to the switch from VNF module
μ_s	Service rate at switch
μ_c	Service rate at controller
μ_f	Service rate at VNF
μ_{sc}	Service rate at Service chaining module
$T_X^{(1)}$	Average Packet delay at X ϵ {s, c, f} for NFV_C
$T_X^{(2)}$	Average Packet delay at X ϵ {s, c, f, sc} for NFV_AC
$T_{Total}^{(1)}$	Average packet delay of NFV packets for NFV_C
$T_{Total}^{(2)}$	Average packet delay of NFV packets for NFV_AC

As switch is a M/M/1 queue, average packet delay at switch can be derived using the total arrival rate at switch, $\Lambda_s^{(1)}$, and service rate at switch, μ_s, as

$$T_s^{(1)} = \frac{1}{\mu_s - (1 + P_c + P_{nfv} - P_c P_{nfv})\lambda_0}. \tag{2}$$

Some of the packets from the controller will enter the VNF module. The probability of packets requiring a network function is assumed to be P_{nfv}. Hence, total arrival rate at VNF is as

$$\Lambda_f^{(1)} = P_{nfv}\lambda_0. \tag{3}$$

Therefore, packet delay at VNF can be expressed as

$$T_f^{(1)} = \frac{1}{\mu_f - P_{nfv}\lambda_0}. \tag{4}$$

For the controller, packet arrival occurs in two ways. First, some packets from switch enter the controller. Second, all the packets leaving VNF enter the controller queue. The former has a rate $(P_c + P_{nfv} - P_c P_{nfv})\lambda_0$, as described above. The latter is $P_{nfv}\lambda_0$. Combining the two cases, the total arrival rate at controller can be written as

$$\begin{aligned}\Lambda_c^{(1)} &= (P_c + P_{nfv} - P_c P_{nfv})\lambda_0 + P_{nfv}\lambda_0 \\ &= (2P_{nfv} + P_c - P_c P_{nfv})\lambda_0.\end{aligned} \tag{5}$$

As the controller is a M/M/1 queue, packet delay at controller can be calculated as

$$T_c^{(1)} = \frac{1}{\mu_c - (2P_{nfv} + P_c - P_c P_{nfv})\lambda_0}. \tag{6}$$

Since each NFV packet has to visit switch and controller queue twice and VNF module queue once, we can calculate average packet delay for NFV packets using Eqs. (2), (4) and (6) as

$$T_{Total}^{(1)} = 2T_s^{(1)} + 2T_c^{(1)} + T_f^{(1)}. \tag{7}$$

3.3. Analysis for NFV_AC

In case of NFV_AC, at first NFV packets (requiring NFV service) enters the switch. If there are no information about the packet in the switch (which means table miss for NFV packets), then the packet will be forwarded to the controller. The packet will visit the service chaining module through the controller. After receiving a response from the service chaining module, the controller initiates the proper action for the switch and sends this NFV packet back to the switch. This packet, experiencing the service chaining module, still needs to go to VNF (from the switch) to obtain its required network function. Subsequent NFV packets belonging to the same flow can be directly redirected from the switch to determined instances of VNF (without involving the controller). If there are information about a NFV packet in the switch (that is, table hit for NFV packet), the packet will directly go to the VNF module from the switch. Some of the NFV packets will return to switch from the VNF module after getting required network function.

As shown in Fig. 4, there are three types of packets that enter the switch: (i) new packets to the switch, (ii) packets that are redirected from the controller, and (iii) packets that are redirected from the VNF module. After getting service from VNF module, some packets may return to network (switch) again, depending on the value of P_t. For the former, the packet arrival rate is λ_0, and for the latter, it can be carefully obtained as follows. From the switch depending on characteristic some packet will go the controller, some will go to the VNF module and some packet will leave the switch after service. As the probability of going to the controller is P_c. Therefore, arrival rate from switch to controller is $P_c\lambda_0$. Since controller and service chaining module have infinite buffer, all the packets that enter the controller will return to the switch. All those packet that do not need service from controller but need service from VNF module will directly enter the VNF module from switch with a rate of $P_{nfv}\ (1 - P_c)\lambda_0$. After visiting the controller some packet will be directed to the VNF module from the switch at a rate of $P_{nfv}P_c\lambda_0$ (These packets need service from both controller and service chaining module). From VNF module some packet will return to the switch at the probability of P_t. As total arriving rate of the VNF module is $P_{nfv}\lambda_0$, the rate at which packets will be redirected from the VNF to the switch is $P_t P_{nfv}\lambda_0$. Therefore, total arrival rate at switch can be calculated as

$$\begin{aligned}\Lambda_s^{(2)} &= \lambda_0 + P_c\lambda_0 + P_t P_{nfv}\lambda_0 \\ &= (1 + P_c + P_t P_{nfv})\lambda_0.\end{aligned} \tag{8}$$

Therefore, average packet delay at switch can be expressed as

$$T_s^{(2)} = \frac{1}{\mu_s - (1 + P_c + P_t P_{nfv})\lambda_0}. \tag{9}$$

It is easily observed that the arrival rate of NFV packets, $\Lambda_f^{(2)}$, is

$$\begin{aligned}\Lambda_f^{(2)} &= P_{nfv}P_c\lambda_0 + P_{nfv}(1 - P_c)\lambda_0 \\ &= P_{nfv}\lambda_0.\end{aligned} \tag{10}$$

Therefore, average packet delay at VNF for NFV_AC can be expressed as

$$T_f^{(2)} = \frac{1}{\mu_f - P_{nfv}\lambda_0}. \tag{11}$$

The packets that need processing from the controller have to be sent to it from the switch. Also the packets departing from the service chaining module will enter the controller's queue again. Combining these two cases, the total arrival rate at controller can be written as

$$\begin{aligned}\Lambda_c^{(2)} &= P_c\lambda_0 + P_c P_{nfv}\lambda_0 \\ &= (1 + P_{nfv})P_c\lambda_0.\end{aligned} \tag{12}$$

As the controller is a M/M/1 queue, packet delay at controller can be calculated as

$$T_c^{(2)} = \frac{1}{\mu_c - (1 + P_{nfv})P_c\lambda_0}. \tag{13}$$

In Fig. 4, service chaining module is connected to the controller and NFV packets will visit it depending on the probability P_{nfv}. The arrival rate of the service chaining module can be expressed as

$$\Lambda_{sc}^{(2)} = P_c P_{nfv}\lambda_0. \tag{14}$$

Therefore, average packet delay at service chaining module can be expressed as

$$T_{sc}^{(2)} = \frac{1}{\mu_{sc} - P_c P_{nfv}\lambda_0}. \tag{15}$$

Using Eqs. (9), (11), (13) and (15), we can calculate the average packet delay for NFV packets as

$$\begin{aligned}T_{Total}^{(2)} &= P_c(2T_s^{(2)} + 2T_c^{(2)} + T_{sc}^{(2)} + T_f^{(2)}) \\ &\quad + (1 - P_c)(T_s^{(2)} + T_f^{(2)}) + P_t T_s^{(2)} \\ &= (1 + P_c + P_t)T_s^{(2)} + 2P_c T_c^{(2)} + P_c T_{sc}^{(2)} + T_f^{(2)}.\end{aligned} \tag{16}$$

4. Results

In order to validate our analytical model, we have implemented a custom simulator. For simulation we do not create independent packets for each queue. A new packet arrives only at a switch following Poisson distribution and then, depending on different probability and feedback result, it goes to the controller or the NFV. We used an event queue from which we pop different queueing events and execute them. We

Table 3
Baseline parameters for the analysis and simulation.

Parameter Name	Value
Probability of redirecting to controller, P_c	0.04
Probability of redirecting to NFV, P_{nfv}	0.5
Probability of redirecting to the switch from VNF module, P_t	0.5
Arrival Rate at the Switch, λ_0	55,000 pkts/s
Service rate at switch, μ_s	1,00,000 pkts/s
Service rate at controller, μ_c	70,000 pkts/s
Service rate at NFV, μ_f	95,000 pkts/s
Service rate at service chaining, μ_{sc}	85,000 pkts/s

have created a packet class. For each packet, we keep track of its delay in different queues.

In this section, we present the analytical and simulation results for NFV_C and NFV_AC by varying different system parameters, including the probability of redirecting from switch to controller, P_c, service rate at VNF, μ_f, the probability of NFV packets, P_{nfv} and arrival rate at the Switch, λ_0. We have chosen a set of parameters as baseline parameters listed in Table 3 and varied one at a time while keeping others fix to prove the correctness of our model. To the best of our knowledge, there exists no previous work that discusses the baseline parameters of SDN and NFV while modeling SDN. However, we used the baseline parameters similar to (Jackson, 1957) where the author reported that in a typical OpenFlow network, the probability of new flow arrival is about 4%. Since all the packets of a network do not require service from NFV, we assume 50% packets ($p_{nfv} = 0.5$) will require service from NFV module. We also assume, after getting service from VNF module 50% packets ($p_t = 0.5$) will return to the switch. For simulation, 1.5 million packets are generated for stability. Our main goal is to observe the impact of different parameters in the network. As we used closed formed equation of delay of a M/M/1 queue for analytical purpose, we tried to make sure all parameters satisfy the queueing property.

4.1. Comparing NFV_C with NFV_AC

4.1.1. Impact of P_c

Fig. 7 shows the impact of P_c on the average packet delay of NFV packets. We found that the analytical results match the simulation results, irrespective of the architecture: NFV_C or NFV_AC. This means our model and analysis can correctly emulate real conditions. Also, we observed that packet delay increases with an increase in P_c; the more packets sent to the controller, the heavier the load on the controller, resulting in a longer packet delays. Comparing NFV_C with NFV_AC, the packet delay for NFV_AC is much shorter than that for NFV_C, largely for three reasons. First, each NFV packet in NFV_C has to enter the controller, and then goes to the VNF, and finally returns to the switch via the controller. This is a longer route for NFV packets in NFV_C, compared to that in NFV_AC. Second, more packets enter the controller in NFV_C than in NFV_AC. We can see that the arrival rate of packets entering the controller is $(2P_{nfv} + P_c - P_cP_{nfv})\lambda_0$, which is much larger than $(1 + P_{nfv})P_c\lambda_0$ in NFV_AC. The heavier the load the controller has, the longer the delay packets experience in the controller. Third, more packets sent to the controller from the switch represent more packets sent back to the switch. Although there is feedback from the VNF module to the switch for NFV_AC, the arrival rate in switch for NFV_C is also larger than that for NFV_AC. We can easily see this from Eqs. (1) and (8) for our baseline parameters. The arrival rate in switch for NFV_C is $(1 + P_c + P_{nfv} - P_cP_{nfv})\lambda_0$, which is larger than that for NFC_AC, $(1 + P_c + P_tP_{nfv})\lambda_0$ because we have considered the default value of P_t as 0.5, and to observe the impact of P_c we changed the value from 0.01 to 0.3. For our baseline configuration there is a delay reduction of 68.83% for NFV_AC over NFV_C. The service rate of the controller plays an important role when comparing the two different models. A large portion of the delay for NFV_C is introduced by the controller. If we consider a larger service rate for the controller, then the difference between the two models will decrease.

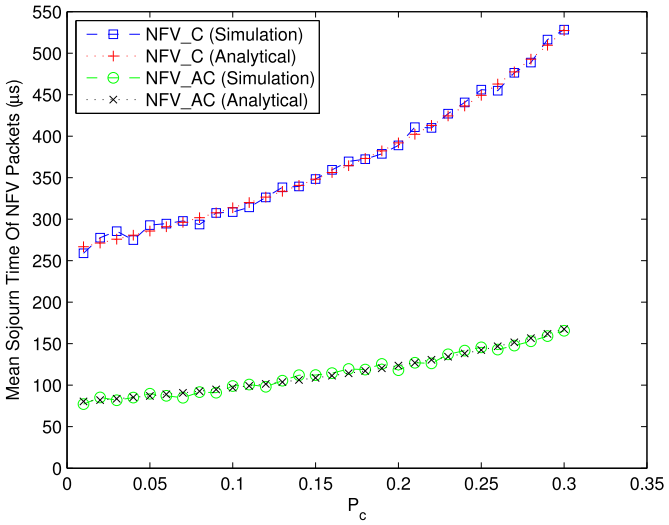

Fig. 7. Average packet delay of NFV packets vs. P_c.

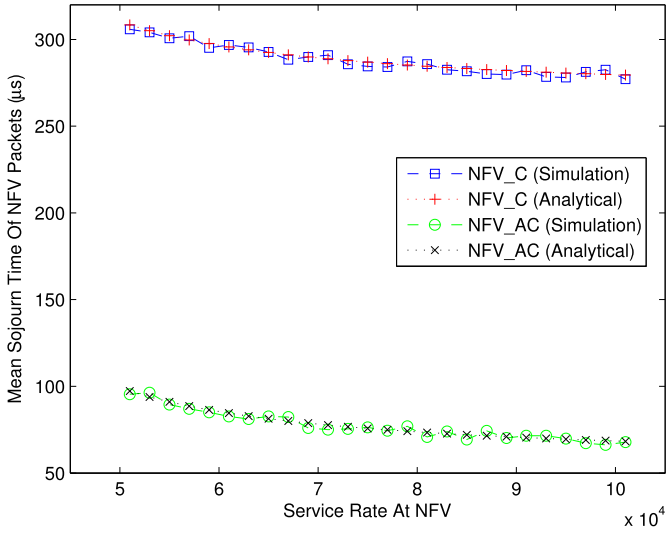

Fig. 8. Average packet delay of NFV packets vs. μ_f.

4.1.2. Impact of service rate at VNF, μ_f

Fig. 8 shows the impact of service rate at VNF, μ_f, on the average packet delay of NFV packets. For both cases, the analytical results match the simulation results very closely. We found that NFV_AC performs better than NFV_C because of the three reasons described above. The delay gap between NFV_C and NFV_AC is constant because μ_f only affects the packet delay in the VNF module. However, the packet arrival rates for NFV_C and NFV_AC are the same, meaning that the packet delays in VNF are also the same for both, irrespective of the value of μ_f. This can be proved from Eqs. (4) and (11) because they have the same formula. The gap is caused by the packet delay differences in the controller, switch, and the service chaining model, but it is not affected by μ_f. It can also be seen from Fig. 8 that, with the increase of the service rate at VNF, there is little variation for either of the models. From this we can get an idea of how network vendors should design a VNF module. Usually fewer packets will enter into VNF module compare to a switch and controller.

4.1.3. Impact of P_{nfv}

Fig. 9 shows the impact of P_{nfv} on the average packet delay of NFV packets. We found that the analytical result matches the simulation results, verifying the correctness of our analysis. we also observed that the average packet delay increases with an increase in P_{nfv}. As described above, μ_f will not affect the gap between NFV_C and NFV_AC. However, P_{nfv} actually affects this gap because a larger P_{nfv} means more packets sent to the controller, especially for NFV_C. That causes that the packet delay for NFV_C to increase faster than that for NFV_AC.

4.1.4. Impact of λ_0

Fig. 10 shows the impact of λ_0 on the average packet delay of NFV packets. We found that the analytical result matches the simulation results, confirming the accuracy about our work. We could also see that the average packet delay increases with an increase of λ_0, and with an increase in the arrival rate at a switch, the load at both switch and controller will also increase. In the case of NFV_C the load at the controller will much higher than for NFV_AC. Our calculations show that for low arrival rates there is delay reduction of 54.02% for NFV_AC compared to NFV_C. For higher arrival rates NFV_AC performs even better. On average there is delay deduction of 72.6% for NFV_AC over NFV_C. Arrival rates at a switch should be one of key concerns in the design of the network equipment. For better performance, in all our experiments the service rate of the switch was almost double of the arrival rate. As there are feedbacks from the controller and VNF module, total arrival rate becomes very high. To hold the queuing property, service rate must be higher than total arrival rate.

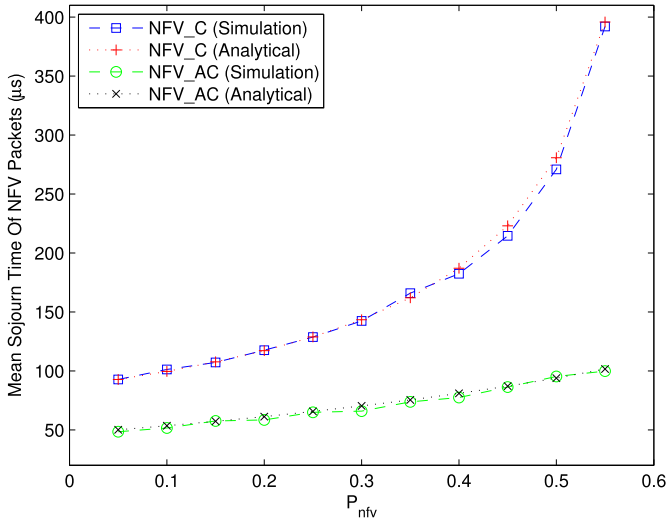

Fig. 9. Average packet delay of NFV packets vs. P_{nfv}.

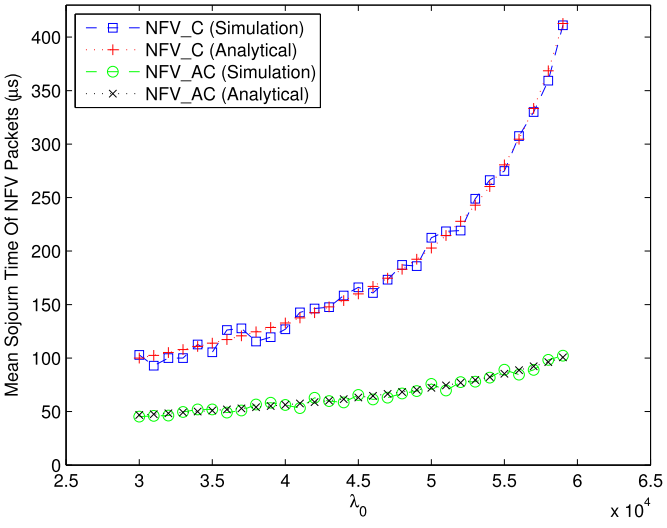

Fig. 10. Average packet delay of NFV packets vs. λ_0.

4.2. The impact of P_t on NFV_AC

In this section, we will observe the impact of P_t on NFV aside the controller model. We use three values for $P_t = \{0.1, 0.5, 0.9\}$ for both analytical and simulation purpose. Other parameters are same as baselines values listed in Table 3.

4.2.1. Impact of P_c

Fig. 11 shows the impact of P_c and P_t on the average packet delay of NFV packets when NFV is aside the controller. We found that the analytical results coordinate the simulation results. This implies that our model and analysis can successfully emulate practical conditions. We also observed that packet delay increases with the increase in P_c. More packets sent to the controller result in a heavier load on the controller, resulting in greater packet delays. For NFV_AC, we have a total of 4 queues. If we observe the arrival rate of these four queues from Eqs. (8), (10), (12) and (14), we see P_t has an impact only on the arrival rate of the switch. For a high value of P_t, the load on the switch will be very high. An increase in the rate for high P_t is much faster than the increase

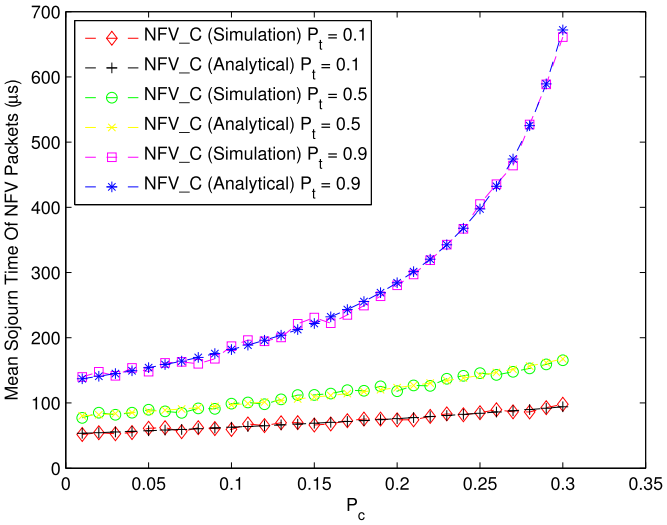

Fig. 11. Average packet delay of NFV packets vs. P_{sc} (Varying P_t).

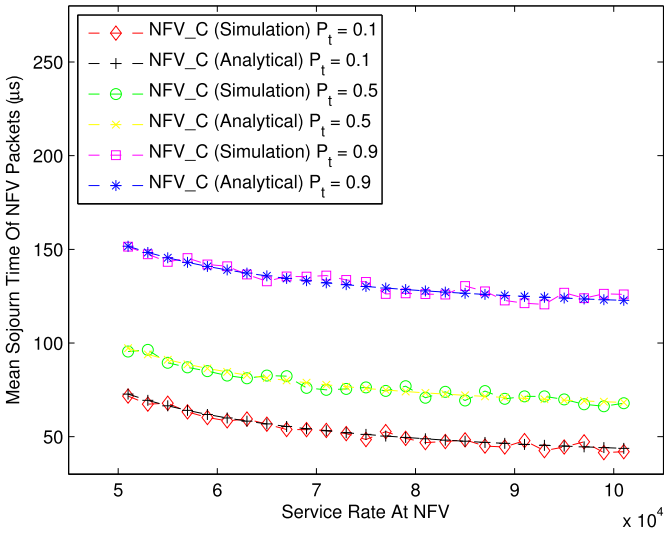

Fig. 12. Average packet delay of NFV packets vs. μ_f (Varying P_t).

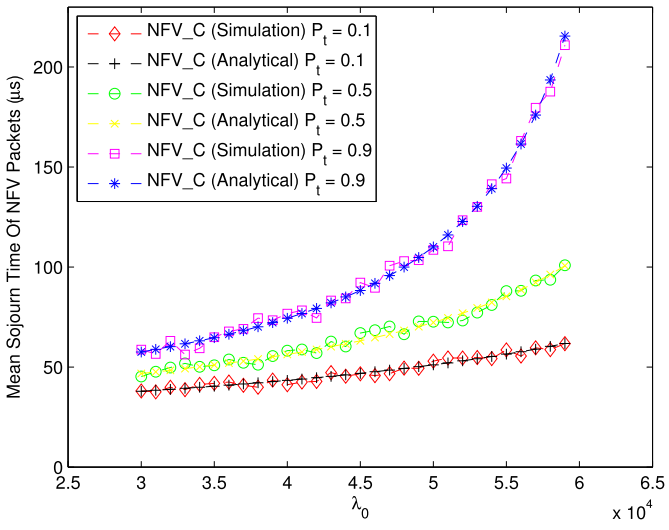

Fig. 14. Average packet delay of NFV packets vs. λ_0 (Varying P_t).

rate for low P_t. In the case of a low P_t, the load at switch is not very high. For a low value of P_c there is a delay increase of 61.79%, comparing low feedback $P_t = 0.1$ with high feedback $P_t = 0.9$. In case of a high value of P_c this delay is very high (82.95%). If we consider high feedback from VNF to the switch, then the service rate of the switch must be very high. We observed another issue here, that the arrival rate of the switch must be comparatively lower then the service rate of the switch. In our model we considered an arrival rate of the switch of 55,000 pkts/s but a service rate of the switch of 1,00,000 pkts/s. Main reason behind this is - As we are considering high feedback ($P_t = 0.9$), then the total arrival rate at the switch (new packet, feedback from VNF and feedback from controller) becomes very high. If we consider a low service rate for the switch then M/M/1 property of the switch can become violated. While designing the switch network vendors should calculate the upper-limit of the arrival rate at the switch and check wether it holds the M/M/1 property or not.

4.2.2. Impact of service rate at VNF, μ_f

Fig. 12 shows the impact of the service rate at NFV, μ_f and P_t on the average packet delay of NFV packets when NFV is aside the controller.

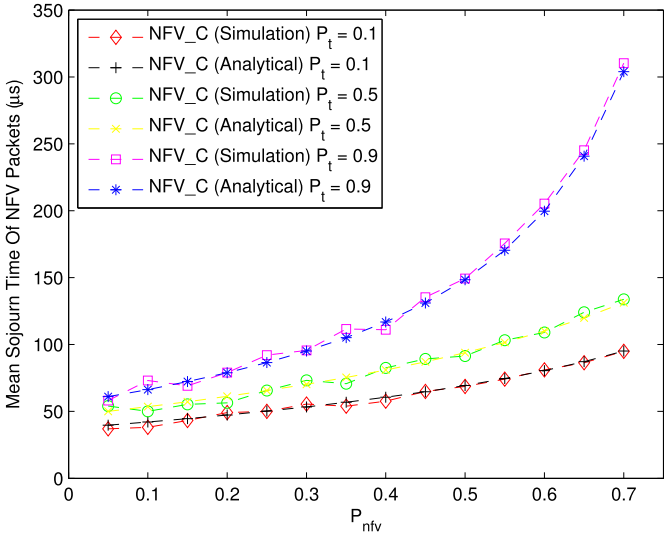

Fig. 13. Average packet delay of NFV packets vs. P_{nfv} (Varying P_t).

For all cases, the analytical results match the simulation results very closely. The delay gaps between different P_t lines are constant because μ_f only affects the packet delay in the VNF module. The gap is caused by the packet delay differences in the switch, as for high P_t large number of packet will be forwarded to switch from VNF. Other than that, all other parameters will remain same. We also observed that the average packet delay of the NFV packets decreases with the increase of service rate at NFV.

4.2.3. Impact of P_{nfv}

Fig. 13 shows the impact of P_{nfv} and P_t on the average packet delay of NFV packets when NFV is aside the controller. We found that the analytical results again matched the simulation results, verifying the correctness of our analysis. We also observed that the average packet delay increases with the increase of P_{nfv}, regardless of P_t. For high values of P_t, the increase in rate is much higher, because for high P_t many packet will be returned to the switch from VNF, which will cause traffic in the switch and increase the total delay. If both P_{nfv} and P_t is high, it means a large portion of the packets will need service from VNF and from there a large portion will be returned to the switch. Since both are very high, this will cause a considerable delay.

4.2.4. Impact of λ_0

Fig. 14 shows the impact of λ_0 and P_t on the average packet delay of NFV packets when the NFV is aside the controller. We again found that the analytical result match the simulation results, verifying the correctness of our analysis. Furthermore, we also observed that the average packet delay increases with an increase of λ_0 regardless of P_t. For high values of both λ_0 and P_t the total load at the switch is very high as there are many new packets and large number of packets are send from the VNF module, which causes a large overall delay. If we consider a large number of feedbacks then we should not allow large numbers of new packets. For low loads there is a delay increase of 35.42%, comparing low feedback $P_t = 0.1$ with high feedback $P_t = 0.9$. In the case of high load this number is very high (66.64%).

5. Conclusion and future works

In this paper, we have presented models for two SDN architectures combined with NFV. The analysis for packet delay of NFV packets is derived using an M/M/1 model. Extensive simulations were conducted to verify our analysis. Results show that analytical results very closely

match simulation results, supporting the correctness of our model and analysis. The packet delay for NFV_AC is significantly less than that for NFV_C. We recorded a significant delay reduction of 68.83% for NFV_AC compared to NFV_C, for three reasons: a shorter route, a smaller controller load, and a smaller switch load.

The service rate at VNF, μ_f, does not affect the delay gap between NFV_AC and NFV_C because packet delays in VNF are the same for both. On the other hand, the probability exists that NFV packets, P_{nfv}, will have some effect. A large P_{nfv} will cause a larger gap in packet delay between NFV_AC and NFV_C because switch and controller become more congested for NFV_C.

If there is a feedback from the VNF to switch for NFV_AC, then with a high value of P_t, the delay of NFV packets will increase compare to low value of P_t. As for high P_t arrival rate at switch will increase, which will cause an overall delay. While designing a switch, network vendors should concentrate on the upper limit of the arrival rate of the switch. If total arrival rate becomes very high it may violate the M/M/1 property of the switch.

Although NFV_AC is obviously better than NFV_C in our model, our model assumes VNF has the same capacity to serve packets. In a real environment, NFV_C has more flexibility to select a lightly-loaded instance to serve NFV packets to reduce packet delay in VNF. This scenario will be further investigated in the future. Besides, Poisson is not realistic compared to more sophisticated arrival models, such as MMPP. But given the combined complexity of the system model, using MMPP would not give us computational affordability. As this is the first work on the modeling of both SDN and NFV, we would leave this as a future work.

References

Anan, M., Al-Fuqaha, A., Nasser, N., Mu, T.-Y., Bustam, H., July 2016. Empowering networking research and experimentation through software-defined networking. J. Netw. Comput. Appl. 70, 140–155.

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

Azodolmolky, S., Wieder, P., Yahyapour, R., October 11-13, 2013. Performance evaluation of a scalable software-defined networking deployment. In: Second European Workshop on Software Defined Networks (EWSDN). IEEE, Berlin, Germany, pp. 68–74.

Basta, A., Kellerer, W., Hoffmann, M., Morper, H.J., Hoffmann, K., Aug 22, 2014. Applying NFV and SDN to LTE mobile core gateways, the functions placement problem. In: 4th Workshop on All Things Cellular: Operations, Applications, & Challenges. ACM, Chicago, Illinois, USA, pp. 33–38.

Bhamare, D., Jain, R., Samaka, M., Erbad, A., November 2016. A survey on service function chaining. J. Netw. Comput. Appl. 75, 138–155.

Bozakov, Z., Rizk, A., October 11-13, 2013. Taming sdn controllers in heterogeneous hardware environments. In: Second European Workshop on Software Defined Networks. IEEE, Berlin, Germany, pp. 50–55.

Ferrús, R., Koumaras, H., Sallent, O., Agapiou, G., Rasheed, T., Kourtis, M.-A., Boustie, C., Gélard, P., Ahmed, T., 2016. Sdn/nfv-enabled satellite communications networks: opportunities, scenarios and challenges. Phys. Commun. 18, 95–112.

Goto, Y., Masuyama, H., Ng, B., Seah, W.K., Takahashi, Y., September 19-21, 2016. Queueing analysis of software defined network with realistic openflow–based switch model. In: IEEE International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS), London, UK, pp. 301–306.

Hakiri, A., Gokhale, A., Berthou, P., Schmidt, D.C., Gayraud, T., 2014. Software-defined networking: challenges and research opportunities for future internet. Comput. Netw. 75, 453–471.

Han, B., Gopalakrishnan, V., Ji, L., Lee, S., 2015. Network function virtualization: challenges and opportunities for innovations. IEEE Commun. Mag. 53 (2), 90–97.

Hawilo, H., Shami, A., Mirahmadi, M., Asal, R., 2014. Nfv: state of the art, challenges, and implementation in next generation mobile networks (VEPC). IEEE Netw. 28 (6), 18–26.

Hu, F., Hao, Q., Bao, K., 2014. A survey on software-defined network and openflow: from concept to implementation. IEEE Commun. Surv. Tutor. 16 (4), 2181–2206.

Jackson, J.R., 1957. Networks of waiting lines. Oper. Res. 5 (4), 518–521.

Jarschel, M., Oechsner, S., Schlosser, D., Pries, R., Goll, S., Tran-Gia, P., September 6-9, 2011. Modeling and performance evaluation of an OpenFlow architecture. In: International Teletraffic Congress, San Francisco, CA, USA, pp. 1–7.

Kuo, J.-J., Shen, S.-H., Kang, H.-Y., Yang, D.-N., Tsai, M.-J., Chen, W.-T., May 1-4, 2017. Service chain embedding with maximum flow in software defined network and application to the next-generation cellular network architecture. In: IEEE Conference on Computer Communications (NFOCOM), Atlanta, GA, USA, pp. 1397–1402.

Lin, Y.-D., Lin, P.-C., Yeh, C.-H., Wang, Y.-C., Lai, Y.-C., May/June 2015. An extended sdn architecture for network function virtualization with a case study on intrusion prevention. IEEE Netw. 29 (3), 48–53.

Lin, P.-C., Lin, Y.-D., Wu, C.-Y., Lai, Y.-C., Kao, Y.-C., November 2016. Balanced service chaining with traffic steering in software defined networks with network function virtualization. IEEE Comput. 49 (11), 68–76.

Lorenz, C., Hock, D., Scherer, J., Durner, R., Kellerer, W., Gebert, S., Gray, N., Zinner, T., Tran-Gia, P., 2017. An sdn/nfv-enabled enterprise network architecture offering fine-grained security policy enforcement. IEEE Commun. Mag. 55 (3), 217–223.

Mahmood, K., Chilwan, A., Østerbø, O.N., Jarschel, M., 2014. On the Modeling of Openflow-based Sdns: the Single Node Case. arXiv preprint arXiv:1411.4733.

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

Masoudi, R., Ghaffari, A., May 2016. Software defined networks: a survey. J. Netw. Comput. Appl. 67, 1–25.

McKeown, N., Anderson, T., Balakrishnan, H., Parulkar, G., Peterson, L., Rexford, J., Shenker, S., Turner, J., 2008. Openflow: enabling innovation in campus networks. ACM SIGCOMM Comput. Commun. Rev. 38 (2), 69–74.

Miao, W., Min, G., Wu, Y., Wang, H., December 19-21, 2015. Performance modelling of preemption-based packet scheduling for data plane in software defined networks. In: International Conference on Smart City/SocialCom/SustainCom (SmartCity). IEEE, Chengdu, China, pp. 60–65.

Miao, W., Min, G., Wu, Y., Wang, H., Hu, J., 2016. Performance modelling and analysis of software-defined networking under bursty multimedia traffic. ACM Trans. Multimed. Comput. Commun. Appl. TOMM 12 (5s) pp. 77:1–19.

Mijumbi, R., Serrat, J., Gorricho, J.-L., Bouten, N., De Turck, F., Boutaba, R., 2016. Network function virtualization: state-of-the-art and research challenges. IEEE Commun. Surv. Tutor. 18 (1), 236–262.

Omnes, N., Bouillon, M., Fromentoux, G., Le Grand, O., February 17-19, 2015. A programmable and virtualized network & it infrastructure for the internet of things: how can nfv & sdn help for facing the upcoming challenges. In: 18th International Conference on Intelligence in Next Generation Networks (ICIN), Paris, France, pp. 64–69.

Pan, J., Paul, S., Jain, R., 2011. A survey of the research on future internet architectures. IEEE Commun. Mag. 49 (7).

Rowshanrad, S., Namvarasl, S., Abdi, V., Hajizadeh, M., Keshtgary, M., 2014. A survey on sdn, the future of networking. J. Adv. Comput. Sci. Technol. 3 (2), 232.

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

Xiong, B., Yang, K., Zhao, WeiLi, J., Li, K., 2016. Performance evaluation of openflow-based software-defined networks based on queueing model. Comput. Netw. 102, 172–185.

Zhang, Y., Cui, L., Wang, W., Zhang, Y., February 2018. A survey on software defined networking with multiple controllers. J. Netw. Comput. Appl. 103, 101–118.

Ahmaed Fahmin received his B.Sc. degree from Bangladesh University of Engineering and Technology (BUET), Dhaka, Bangladesh in 2017 in Computer Science and Engineering. He is working as Graduate Researcher at the department of CSE, BUET. His research interests include Wireless networks, Network function Virtualization and Software defined networking.

Yuan-Cheng Lai received his Ph.D. degree in computer science from National Chiao Tung University in 1997. He joined the faculty of the Department of Information Management at National Taiwan University of Science and Technology in 2001 and has been a professor since 2008. His research interests include wireless networks, network performance evaluation, network security, and social networks.

Md. Shohrab Hossain received his B.Sc. and M.Sc. in Computer Science and Engineering from Bangladesh University of Engineering and Technology (BUET), Dhaka, Bangladesh in the year 2003 and 2007, respectively. He obtained his Ph.D. degree from the School of Computer Science at the University of Oklahoma, Norman, OK, USA in December 2012. He is currently serving as an Associate Professor in the Department of CSE, BUET. His research interests include Mobile malware detections, cyber security, Software defined networking, security of mobile and ad hoc networks, and Internet of Things. He has been serving as the TPC member of IEEE Globecom, IEEE ICC, IEEE VTC, Wireless Personal Communication, (Springer), Journal of Network and Computer Applications (Elsevier), IEEE Wireless Communications. He serves as a member of ITEE Question Making Committee (QMC) for the ITPEC countries. He has published more than 60 technical research papers in leading Journals and conferences from IEEE, Elsevier, Springer.

Ying-Dar Lin is a Distinguished Professor of computer science at National Chiao Tung University (NCTU), Taiwan. He received his Ph.D. in computer science from the University of California at Los Angeles (UCLA) in 1993. He was a visiting scholar at Cisco Systems in San Jose, California, during 2007–2008, and the CEO at Telecom Technology Center, Taipei, Taiwan, during 2010–2011. Since 2002, he has been the founder and director of Network Benchmarking Lab (NBL, www.nbl.org.tw), which reviews network products with real traffic and has been an approved test lab of the Open Networking Foundation (ONF) since July 2014. He also cofounded L7 Networks Inc. in 2002, which was later acquired by D-Link Corp. His research interests include network security, wireless communications, and network cloudification. His work on multihop cellular was the first along this line, and has been cited over 800 times and standardized into IEEE 802.11s, IEEE 802.15.5, IEEE 802.16j, and 3GPP LTE-Advanced. He is an IEEE Fellow (class of 2013), IEEE Distinguished Lecturer (2014–2017), and ONF Research Associate. He currently serves on the editorial boards of several IEEE journals and magazines, and is the Editor-in-Chief of IEEE Communications Surveys and Tutorials (COMST). He published a textbook, Computer Networks: An Open Source Approach (www.mhhe.com/lin), with Ren-Hung Hwang and Fred Baker (McGraw-Hill, 2011).