Queueing Analysis of Software Defined Network with Realistic OpenFlow–based Switch Model

Yuki Goto*, Hiroyuki Masuyama*, Bryan Ng†, Winston K.G. Seah† and Yutaka Takahashi*

*Department of Systems Science, Graduate School of Informatics, Kyoto University, Kyoto 606-8501, Japan
Email: goto@sys.i.kyoto-u.ac.jp, {masuyama,takahashi}@i.kyoto-u.ac.jp

†School of Engineering and Computer Science, Victoria University of Wellington, Wellington 6140, New Zealand
Email: {winston.seah,bryan.ng}@ecs.vuw.ac.nz

***Abstract*—Software Defined Networking (SDN) is the latest network architecture that does for networking what virtualisation did for servers in data centres. In SDN, separation of the control plane from the data plane brought about new flexibility in the routing of flows through the network. Closely associated with SDN is OpenFlow, the most widely used protocol governing the information exchange between the data plane (switching devices) and the control plane (controller). The ease of implementing and testing new schemes in SDN has prompted many researchers to adopt the experimental and prototyping approach to validate their ideas. Consequently, there has been very little work done to evaluate the performance of SDN and/or OpenFlow–based networks analytically. While the experimentation approach in validation has merits, analytical modelling provides valuable insights by making explicit the dependence of SDN performance on chosen parameters. In this paper: (i) we propose a queueing model of an OpenFlow–based SDN that takes into account classful treatment of packets arriving at a switch and (ii) derive an exact analysis of the proposed queueing model.**

***Index Terms*—Queueing analysis, Software Defined Networks, OpenFlow, packet transfer delay, loss probability.**

I. Introduction

Software Defined Networking (SDN) became prominent in 2007 after a period of research that started as early as 2003 [1]. It gives carriers, service providers and enterprises greater control over the way packets are moved around their networks. This is achieved by decoupling the control plane (that makes decisions about where network traffic is sent) from the data plane (the underlying systems that forward traffic to the selected destinations.) Network providers can make centralized and optimized decisions on how traffic should flow (move) from one point to another within their networks, and disseminate these flow decisions to the networking devices (routers and switches) that make up their infrastructures; in current networks, these traffic flow decisions are made by routers/switches in a distributed manner using information and policy rules that are configured into them.

To date, analytical modelling research for SDN typically uses two different approaches, viz. queueing theory and network calculus. Proper adaptation of queueing theoretic modelling techniques and analytical methods can contribute to accurate performance evaluation of SDN-based production networks that are exposed to new user demands continually, facilitating early identification of potential traffic hotspots and/or bottlenecks which carriers and network providers can quickly remedy before they become problems. With this goal in mind, we propose a queueing model of an OpenFlow–based software defined network that aims to model the operation of the network devices as accurately as possible.

A critical functionality that has not been addressed by queueing analytic modelling research is the different treatment of packets arriving at a switch from a controller, and this is our key consideration. In the next section, we discuss the relevant research on analytical modelling of SDN and OpenFlow–based networks, and then present our queueing model in Section III. Following that, in Section IV, we derive the average packet transfer delay in the whole system and packet loss probabilities of the two different types of packets arriving at a switch, together with a numerical example to validate our analysis in Section V. Lastly, we summarize our contributions and briefly discuss ongoing and future research.

II. Related Work

SDN's flexibility allows and encourages researchers to quickly implement prototypes and testbeds for experimental validation and performance measurements [2]. As the focus of this paper is on mathematical performance analysis and modelling, we shall only discuss related work that utilizes mathematical techniques, and refer the reader to recent surveys, e.g. [3, 4], for details on other related work.

The first attempt to study the performance of OpenFlow–based networks analytically modelled the network as a feedback-oriented queueing model, where the switch is modelled as an M/M/1 queue and the controller as a feedback queueing system of the delay-loss type M/M/1-S [5]. In a recent extension of that work [6], a Jackson network is used to model the data plane while the controller is separately modelled as an M/M/1 queue (considering both infinite and finite buffer scenarios.) Based on the OpenFlow standard, traffic arriving at a switch that comes from the controller should not be sent back to the controller again. However, this important aspect was not accounted for by both models [5, 6]; they did not distinguish between traffic from the controller and other traffic, thus providing only an approximation of the interaction between the controller and switches.

With the consolidation of the control plane in a logically centralized location, viz. the controller, the performance and scalability of the controller architectures became a critical problem [7]. Analytical performance models have been proposed to study the response times of three typical SDN control plane architectures, viz., centralized, decentralized and hierarchical [8]. In [9], flow set-up requests from switches to controller are modelled as a batch arrival process M[X]/M/1 to derive the average flow service time. Given a limited flow set-up time, the number of switches can be determined, thus providing a method to evaluate the capacity of a single controller, extendable to multiple controllers.

While extensive research has been done on SDN and OpenFlow, the use of analytical performance evaluation methods remains lacking. The preceding discussion, which is by no means comprehensive, has summarized the notable work in this aspect. As noted above, current queueing models do not accurately model the OpenFlow protocol's operation on the switch. We aim to address this shortcoming by separating traffic coming from the controller from the rest of the traffic into different queues, so that traffic arriving from the controller are not sent back to the controller.

III. Queueing Model

Figure 1 shows the queueing model for a simple OpenFlow switch-controller topology. The system has one switch and one controller. External packets arrive at the switch, which

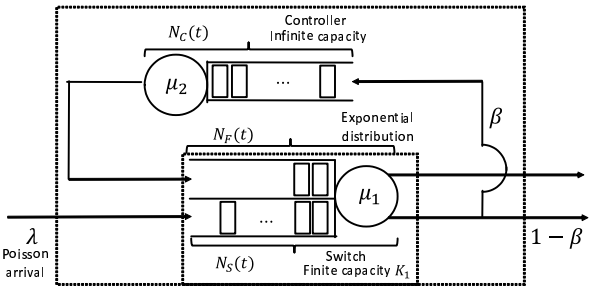

Fig. 1. Queueing model of simple network with an OpenFlow switch connected to a single controller.

we call **class S** packets, according to a Poisson process with parameter λ. The switch checks the flow table to match the forwarding methods of class S packets. The forwarding time of class S packets follows an exponential distribution with parameter μ_1. If the forwarding methods of class S packets do not exist in the flow table, these packets are directed to the controller, that is, the switch makes inquiries to the controller about the forwarding methods of those packets (acomplished via a *packet_in message)* . We assume this event occurs with probability β. If the forwarding methods of class S packets are found in the flow table, the packets are forwarded to the appropriate output port. We assume this event occurs with probability $1-\beta$. Namely, the probability of the first packet of a flow is β and that of the succeeding packets is $1-\beta$. Thus, the packet flow size follows a geometric distribution with average $1/\beta$.

Packets whose forwarding information is missing in the flow table are forwarded to the controller, and we refer to them as **class C** packets. Upon receiving class C packets, the controller confirms the forwarding methods of class C packets and updates the flow table in the switch. The processing time of class C packets follows an exponential distribution with parameter μ_2. After being processed by the controller, class C packets are redirected (back) to the switch and we call them **class F** packets. As for forwarding discipline of the switch, class F packets have priority over class S packets because class F packets were originally received by the switch earlier than class S packets in the buffer. After being processed by the switch, class F packets are directed to the output port. We assume that the controller has buffer of infinite capacity while the switch has finite buffer that can enqueue no more than K_1 packets.

IV. Analysis

In the following discussions, we derive a continuous-time Markov chain and the transition rate matrix in subsection IV-A. Section IV-B provides numerical calculation methods in order to derive stationary state distribution. In Section IV-C, we derive performance measures.

A. Derivation of continuous-time Markov chain

Let $N_C(t)$ (respectively, $N_{\mathrm{F}}(t)$ and $N_{\mathrm{S}}(t)$) denote the number of class C (respectively, class F and class S) packets. These variables meet the following condition.

$$0 \le N_{\mathrm{F}}(t) + N_{\mathrm{S}}(t) \le K_1,\ 0 \le N_{\mathrm{C}}(t). \tag{1}$$

We consider the stochastic process $\{(N_{\mathrm{C}}(t), N_{\mathrm{F}}(t), N_{\mathrm{S}}(t)); t \ge 0\}$, with state space $\mathbb{F}$ given as follows:

$$\mathbb{F} = \{(i,j,k) \mid i \in \mathbb{Z}_+, \\ (j,k) \in \{0,1,\ldots,K_1\} \times \{0,1,\ldots,K_1 - j\}\}.$$

As described in Section III, external packets arrive according to a Poisson process and the forwarding times in the switch and the controller are distributed according to independent exponential distributions. From the assumptions for the model, the stochastic process $\{(N_{\mathrm{C}}(t), N_{\mathrm{F}}(t), N_{\mathrm{S}}(t));\ t \ge 0\}$ is a continuous-time Markov chain with infinite state space $\mathbb{F}$. Now let $\boldsymbol{Q}$ denote the transition rate matrix of the Markov chain, and taking $N_{\mathrm{C}}(t) \in \mathbb{Z}_+$ as the level variable, $\boldsymbol{Q}$ is given as follows:

$$\boldsymbol{Q} = \begin{array}{c} \\ \mathbb{F}_0 \\ \mathbb{F}_1 \\ \mathbb{F}_2 \\ \vdots \end{array} \begin{array}{c} \begin{array}{cccccc} \mathbb{F}_0 & \mathbb{F}_1 & \mathbb{F}_2 & \cdots & & \end{array} \\ \begin{pmatrix} \boldsymbol{B}_1 & \boldsymbol{A}_0 & \boldsymbol{O} & \boldsymbol{O} & \cdots & \cdots \\ \boldsymbol{A}_2 & \boldsymbol{A}_1 & \boldsymbol{A}_0 & \boldsymbol{O} & \ddots & \ddots \\ \boldsymbol{O} & \boldsymbol{A}_2 & \boldsymbol{A}_1 & \boldsymbol{A}_0 & \boldsymbol{O} & \ddots \\ \boldsymbol{O} & \boldsymbol{O} & \boldsymbol{A}_2 & \boldsymbol{A}_1 & \boldsymbol{A}_0 & \ddots \\ \vdots & \ddots & \boldsymbol{O} & \boldsymbol{A}_2 & \boldsymbol{A}_1 & \ddots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots \end{pmatrix} \end{array} \tag{2}$$

where $\mathbb{F}_i = \{(i,j,k) \in \mathbb{F} \mid i \in \mathbb{Z}_+\}$, and matrices $\boldsymbol{A}_0$, $\boldsymbol{A}_1$, $\boldsymbol{A}_2$ and $\boldsymbol{B}_1$ also have block structures. Thus, taking $N_{\mathrm{F}}(t)$ as the second level variable, $\boldsymbol{A}_0$, $\boldsymbol{A}_1$, $\boldsymbol{A}_2$ and $\boldsymbol{B}_1$ also have similar structures.

1) Elements of block matrix $\boldsymbol{A}_0$: If $N_{\mathrm{C}}(t)$ increases by 1, $N_{\mathrm{F}}(t)$ is 0 because class F packets have priority over class S packets when being processed and forwarded by the switch. Thus, $\boldsymbol{A}_0$ has the following structure:

$$\boldsymbol{A}_0 = \begin{array}{c} \\ \mathbb{F}_{i,0} \\ \mathbb{F}_{i,1} \\ \vdots \\ \mathbb{F}_{i,K_1} \end{array} \begin{array}{c} \begin{array}{cccc} \mathbb{F}_{i,0} & \mathbb{F}_{i,1} & \cdots & \mathbb{F}_{i,K_1} \end{array} \\ \begin{pmatrix} \boldsymbol{C} & \boldsymbol{O} & \cdots & \boldsymbol{O} \\ \boldsymbol{O} & \boldsymbol{O} & & \vdots \\ \vdots & & \ddots & \vdots \\ \boldsymbol{O} & \cdots & \cdots & \boldsymbol{O} \end{pmatrix} \end{array}$$

where

$$\mathbb{F}_{i,j} = \{(i,j,k) \in \mathbb{F} | i \in \mathbb{Z}_+, j = 0, 1, \ldots, \ K_1\},$$
$$\boldsymbol{C} = \left(C_{k,k'}\right)_{(k,k') \in \mathbb{Z}_+^{\leqslant K_1} \times \mathbb{Z}_+^{\leqslant K_1}}, \text{ and}$$
$$\mathbb{Z}_+^{\leqslant i} := \{0, 1, \ldots, i\}.$$

If $N_{\mathrm{C}}(t)$ increases by 1 and $N_{\mathrm{F}}(t) = 0$, $N_{\mathrm{S}}(t)$ decreases by 1 and thus the elements of matrix $\boldsymbol{C}$ become,

$$C_{k,k'} = \begin{cases} \beta\mu_1, & k^{'} = k-1, \\ 0, & \text{otherwise.} \end{cases}$$

2) Elements of block matrix $\boldsymbol{A}_2$: If $N_{\mathrm{C}}(t)$ decreases by 1, $N_{\mathrm{F}}(t)$ increases by 1, and therefore $\boldsymbol{A}_2$ is given as follows:

$$\boldsymbol{A}_2 = \begin{array}{c} \\ \mathbb{F}_{i,0} \\ \mathbb{F}_{i,1} \\ \vdots \\ \vdots \\ \mathbb{F}_{i,K_1} \end{array} \begin{array}{c} \begin{array}{ccccc} \mathbb{F}_{i,0} & \mathbb{F}_{i,1} & \cdots & \cdots & \mathbb{F}_{i,K_1} \end{array} \\ \begin{pmatrix} \boldsymbol{O} & \boldsymbol{D}^{(0)} & \boldsymbol{O} & \cdots & \boldsymbol{O} \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & \boldsymbol{O} \\ \vdots & & & \ddots & \boldsymbol{D}^{(K_1-1)} \\ \boldsymbol{O} & & \cdots & \cdots & \boldsymbol{O} \end{pmatrix} \end{array}$$

where $\boldsymbol{D}^{(j)} = \left(D_{k,k'}^{(j)}\right)_{(k,k') \in \mathbb{Z}_+^{\leqslant K_1 - j} \times \mathbb{Z}_+^{\leqslant K_1 - j + 1}}$.

If $N_{\mathrm{C}}(t)$ decreases by 1 and $N_{\mathrm{F}}(t)$ increases by 1, $N_{\mathrm{S}}(t)$ does not change; hence, matrix $\boldsymbol{D}^{(j)}$ becomes

$$D_{k,k'}^{(j)} = \begin{cases} \mu_2, & k^{'} = k, \\ 0, & \text{otherwise.} \end{cases}$$

3) Elements of block matrix $\boldsymbol{A}_1$ and $\boldsymbol{B}_1$: If $N_{\mathrm{C}}(t)$ does not change, $N_{\mathrm{F}}(t)$ does not change or decreases by 1. Thus, $\boldsymbol{A}_1$ is given as follows:

$$\boldsymbol{A}_1 = \begin{array}{c} \\ \mathbb{F}_{i,0} \\ \mathbb{F}_{i,1} \\ \vdots \\ \vdots \\ \mathbb{F}_{i,K_1} \end{array} \begin{array}{c} \begin{array}{ccccc} \mathbb{F}_{i,0} & \mathbb{F}_{i,1} & \mathbb{F}_{i,2} & \cdots & \mathbb{F}_{i,K_1} \end{array} \\ \begin{pmatrix} \boldsymbol{E}^{(0)} & \boldsymbol{O} & \cdots & \cdots & \boldsymbol{O} \\ \boldsymbol{F}^{(1)} & \boldsymbol{E}^{(1)} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \boldsymbol{O} \\ \vdots & \ddots & \boldsymbol{F}^{(K_1-1)} & \boldsymbol{E}^{(K_1-1)} & \boldsymbol{O} \\ \boldsymbol{O} & \cdots & \boldsymbol{O} & \boldsymbol{F}^{(K_1)} & \boldsymbol{E}^{(K_1)} \end{pmatrix} \end{array} \quad (3)$$

where

$$\boldsymbol{E}^{(j)} = \left(E_{k,k'}^{(j)}\right)_{(k,k') \in \mathbb{Z}_+^{\leqslant K_1 - j} \times \mathbb{Z}_+^{\leqslant K_1 - j}},$$
$$\boldsymbol{F}^{(j)} = \left(F_{k,k'}^{(j)}\right)_{(k,k') \in \mathbb{Z}_+^{\leqslant K_1 - j} \times \mathbb{Z}_+^{\leqslant K_1 - j + 1}}.$$

Besides, the diagonal block elements of $\boldsymbol{B}_1$ are not $\boldsymbol{E}^{(j)}$ but $\boldsymbol{E}^{(0,j)}$ because matrix $\boldsymbol{B}_1$ is different from matrix $\boldsymbol{A}_1$ only in the diagonal block elements. The entries $\boldsymbol{F}^{(j)}$ corresponds to the case that $N_{\mathrm{C}}(t)$ and $N_{\mathrm{S}}(t)$ remain unchanged and $N_{\mathrm{F}}(t)$ decreases by 1. Then $\boldsymbol{F}^{(j)}$is given by:

$$F_{k,k'}^{(j)} = \begin{cases} \mu_1, & k^{'} = k, \\ 0, & \text{otherwise.} \end{cases}$$

Next, we derive $\boldsymbol{E}^{(j)}$, which represents state transitions that $N_{\mathrm{C}}(t)$ and $N_{\mathrm{F}}(t)$ remain unchanged and that $N_{\mathrm{S}}(t)$ changes. If $N_{\mathrm{F}}(t) = 0$, $N_{\mathrm{S}}(t)$ increases or decreases by 1. If $N_{\mathrm{F}}(t) \neq 0$, $N_{\mathrm{S}}(t)$ increases by 1. Thus, $\boldsymbol{E}^{(j)}$ is given by:

$$E_{k,k'}^{(j)} = \begin{cases} \lambda, & k^{'} = k+1, \\ (1-\beta)\mu_1, & j = 0, \ k^{'} = k-1, \\ 0, & \text{otherwise.} \end{cases}$$

The diagonal elements of $\boldsymbol{E}^{(j)}$ are given as follows:

$$E_{k,k}^{(j)} = \begin{cases} -\mu_2 - \lambda, & j = 0, \ k = 0, \\ -\mu_1 - \mu_2 - \lambda, & j = 0, \ 0 < k < K_1, \\ -\mu_1, & 0 \leq j \leq K_1, \ k = K_1 - j, \\ -\mu_1 - \mu_2 - \lambda, & 0 < j \leq K_1, \ 0 \leq k < K_1 - j. \end{cases}$$

The diagonal elements of $\boldsymbol{B}_1$, that is, the diagonal elements of $\boldsymbol{E}^{(0,j)}$ are given by

$$E_{k,k}^{(0,j)} = \begin{cases} -\lambda, & j = 0, \ k = 0, \\ -\mu_1 - \lambda, & j = 0, \ 0 < k < K_1, \\ -\mu_1, & 0 \leq j \leq K_1, \ k = K_1 - j, \\ -\mu_1 - \lambda, & 0 < j \leq K_1, \ 0 \leq k < K_1 - j. \end{cases}$$

4) Stability condition of the Markov chain: The Markov chain is irreducible. If it is positive recurrent, we have the unique state distribution vector $\boldsymbol{\pi} = (\pi_{i,j,k})_{(i,j,k) \in \mathbb{F}}$ where

$$\pi_{i,j,k} = \lim_{t \to \infty} \Pr(N_{\mathrm{C}}(t) = i, N_{\mathrm{F}}(t) = j, N_{\mathrm{S}}(t) = k).$$

We assume $\beta\lambda - \mu_2 < 0$, and thus the Markov chain is positive recurrent.

B. Numerical solution

As the transition rate matrix $\boldsymbol{Q}$ is block tridiagonal, we have the following equation for the stationary state distribution $\boldsymbol{\pi} = (\boldsymbol{\pi}_0, \boldsymbol{\pi}_1, \ldots)$ where $\boldsymbol{\pi}_i := (\pi_{i,j,k})_{(i,j,k) \in \mathbb{F}_i}$:

$$\boldsymbol{\pi}_{k-1}\boldsymbol{A}_0 + \boldsymbol{\pi}_k\boldsymbol{A}_1 + \boldsymbol{\pi}_{k+1}\boldsymbol{A}_2 = 0, \quad k = 2, 3, \ldots. \quad (4)$$

The Markov chain is the quasi-birth-and-death process, hence, the stationary state distribution is a matrix-geometric solution and there exists a matrix $\boldsymbol{R}$ such that $\boldsymbol{\pi}_k = \boldsymbol{\pi}_{k-1}\boldsymbol{R}$ [10], where $\boldsymbol{R}$ is the least nonnegative solution of

$$\boldsymbol{A}_0 + \boldsymbol{R}\boldsymbol{A}_1 + \boldsymbol{R}^2\boldsymbol{A}_2 = \boldsymbol{0}.$$

Substituting $\boldsymbol{\pi}_k = \boldsymbol{\pi}_{k-1}\boldsymbol{R}$ in (4), we obtain $\boldsymbol{\pi}_k = \boldsymbol{\pi}_1\boldsymbol{R}^{k-1}$. Writing down the boundary elements of $\boldsymbol{\pi Q} = \boldsymbol{0}$, we have:

$$\boldsymbol{\pi}_0\boldsymbol{B}_1 + \boldsymbol{\pi}_1\boldsymbol{A}_2 = \boldsymbol{0},$$
$$\boldsymbol{\pi}_0\boldsymbol{A}_0 + \boldsymbol{\pi}_1\boldsymbol{A}_1 + \boldsymbol{\pi}_2\boldsymbol{A}_2 = \boldsymbol{0}.$$

Substituting $\boldsymbol{\pi}_2 = \boldsymbol{\pi}_1\boldsymbol{R}$ for the above equation, we have the following equation:

$$(\boldsymbol{\pi}_0, \boldsymbol{\pi}_1)\begin{pmatrix} \boldsymbol{A}_1^{(0)} & \boldsymbol{A}_0 \\ \boldsymbol{A}_2 & \boldsymbol{A}_1 + \boldsymbol{RC} \end{pmatrix} = \boldsymbol{0}. \tag{5}$$

The sum of all probabilities is equal to 1, and we obtain

$$1 = \boldsymbol{\pi}_0\boldsymbol{e} + \sum_{k=1}^{\infty}\boldsymbol{\pi}_k\boldsymbol{e} = \boldsymbol{\pi}_0\boldsymbol{e} + \boldsymbol{\pi}_1(\boldsymbol{I} - \boldsymbol{R})^{-1}\boldsymbol{e}. \tag{6}$$

With $\boldsymbol{R}$, we can calculate $\boldsymbol{\pi}$ from (5) and (6). Moreover, quasi-birth-and-death process is a special case of the Markov chains of M/G/1-type. Therefore, the probability matrix $\boldsymbol{G}$, showing the first passage probability until $N_{\mathrm{C}}(t)$ decreases by 1, exists and fulfills the following equation [10]:

$$\boldsymbol{A}_2 + \boldsymbol{A}_1\boldsymbol{G} + \boldsymbol{A}_0\boldsymbol{G}^2 = 0.$$

Now, $\boldsymbol{R}$ is expressed by using $\boldsymbol{G}$ as follows:

$$\boldsymbol{R} = \boldsymbol{A}_0(-\boldsymbol{A}_1 - \boldsymbol{A}_0\boldsymbol{G})^{-1}.$$

To calculate $\boldsymbol{G}$, we use the Logarithmic Reduction algorithm described in [11].

C. Performance measures

We consider three performance measures, namely, packet loss probabilities in class S and F, and average packet transfer delay through the system. The packet loss probabilities of class S and F are denoted as $P_{\mathrm{loss}}^{(\mathrm{S})}$ and $P_{\mathrm{loss}}^{(\mathrm{F})}$, respectively. In the following, we analyze the packet trajectory and derive packet loss probabilities.

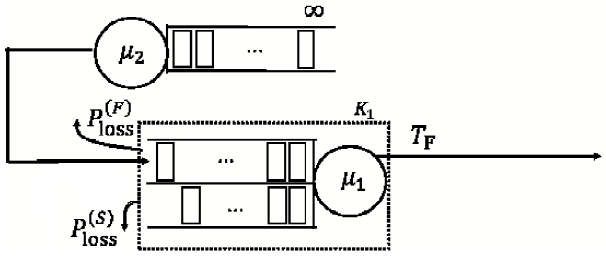

Fig. 2. Modelling class S and class F packets in the network.

First, we trace the packet trajectory of a class S packet as shown in Figure 2. Because the switch forwards a packet with rate μ_1, the throughput of class S packets, T_{S}, is expressed as:

$$T_{\mathrm{S}} = \mu_1\sum_{i=0}^{\infty}\sum_{k=1}^{K_1}\pi_{i,0,k}.$$

In stationary state, the packet arrival rate and the packet departure rates are equal and this yields $\lambda(1 - P_{\mathrm{loss}}^{(\mathrm{S})}) = T_{\mathrm{S}}$. Therefore $P_{\mathrm{loss}}^{(\mathrm{S})}$ is expressed as:

$$P_{\mathrm{loss}}^{(\mathrm{S})} = 1 - \frac{\mu_1\sum_{k=1}^{K_1}\sum_{i=0}^{K_2}\pi_{i,0,k}}{\lambda}.$$

Next, we focus on packet trajectory of a class C packet, which is shown in Figure 2. As the controller finishes forwarding a packet with rate μ_2, the throughput of class C, T_{C}, is given by:

$$T_{\mathrm{C}} = \mu_2\sum_{i=1}^{\infty}\sum_{k=0}^{K_1}\sum_{j=0}^{K_1-k}\pi_{i,j,k}.$$

Similarly, the probability of $N_{\mathrm{F}}(t) = j$ $(j \geq 1)$ is given by $\sum_{i=0}^{\infty}\sum_{k=0}^{K_1-j}\pi_{i,j,k}$ and the switch finishes forwarding a packet with rate μ_1. Thus, we obtain the throughput of class F, T_F, as:

$$T_{\mathrm{F}} = \mu_1\sum_{i=0}^{\infty}\sum_{j=1}^{K_1}\sum_{k=0}^{K_1-j}\pi_{i,j,k}.$$

As for $P_{\mathrm{loss}}^{(\mathrm{F})}$, we have the equation $T_{\mathrm{C}}(1 - P_{\mathrm{loss}}^{(\mathrm{F})}) = T_{\mathrm{F}}$, thus, $P_{\mathrm{loss}}^{(\mathrm{F})}$ is given as:

$$P_{\mathrm{loss}}^{(\mathrm{F})} = 1 - \frac{\mu_1\sum_{j=1}^{K_1}\sum_{k=0}^{K_1-j}\sum_{i=0}^{K_2}\pi_{i,j,k}}{\mu_2\sum_{i=1}^{K_2}\sum_{k=0}^{K_1}\sum_{j=0}^{K_1-k}\pi_{i,j,k}}.$$

Finally, we derive the average packet transfer delay in the system. The average total number of packets in the system is $\sum_{i=0}^{\infty}\sum_{k=0}^{K_1}\sum_{j=0}^{K_1-k}(i+j+k)\pi_{i,j,k}$ and the throughput of the system, T, is obtained as,

$$T = (1-\beta)T_{\mathrm{S}} + T_{\mathrm{F}},$$

and, from Little's formula, we obtain the average packet transfer delay as:

$$E[W] = \frac{\sum_{i=0}^{\infty}\sum_{k=0}^{K_1}\sum_{j=0}^{K_1-k}(i+j+k)\pi_{i,j,k}}{(1-\beta)\mu_1\sum_{i=0}^{\infty}\sum_{k=1}^{K_1}\pi_{i,0,k} + \mu_1\sum_{i=0}^{\infty}\sum_{j=1}^{K_1}\sum_{k=0}^{K_1-j}\pi_{i,j,k}}.$$

V. Numerical example & Validation

In this section, we show numerical results of three performance measures namely: (i) packet loss probability for class S ($P_{\mathrm{loss}}^{(\mathrm{S})}$), (ii) packet loss probability for class F ($P_{\mathrm{loss}}^{(\mathrm{F})}$) and (iii) average packet transfer delay derived earlier in Section IV. The purpose of this evaluation is to validate the proposed queueing model and to better understand dynamics of increased traffic on the switch and controller. Table I lists the values of the parameters used in the numerical evaluation.

We first investigate how performance measures change as the external packet arrival rate λ increases from 20 to 40 in Section V-A. Next, in Section V-B, we consider the effect of increasing flow size on each performance measure. Simulation results shown in Figure 3 to Figure 8 with the corresponding 95% confidence interval.

TABLE I
PARAMETER SETTING

Parameter	Value
Buffer capacity of switch K_1 [packets]	10
Forwarding rate of switch μ_1 [packets/ms]	50
Forwarding rate of controller μ_2 [packets/ms]	4
External packet arrival rate λ [packets/ms]	20, 40
Average flow size $1/\beta$ [packets]	20, 100, 500

A. Effect of external packet arrival rate

In this subsection, we investigate the effect of external packet arrival rate for the case of $1/\beta = 20, 100,$ and 500. The curves in Figure 3 show that a higher external packet arrival rate induces a higher packet loss probability in class S. This is intuitively explained through observation of λ. If λ is large, there is increased likelihood that packets are buffered in the switch and this contributes to higher probability of packet drops. A similar explanation holds for packet loss probabilities for class F shown in Figure 4 with the difference that the loss probabilities are slightly lower for identical arrival rates.

Figure 5 illustrates the average packet transfer delay where we observe that it increases when external packet arrival rate increases. This is explained by a larger value of λ inducing a larger number of packets in the switch and the cumulative effect results in packets having to wait for longer durations until the switch forwards the packets. The average packet transfer delay decreases with increasing flow size because the proportion of packets passing through class F with lower delay is significantly higher for larger flow sizes. Again, we see that the average delay obtained by the queueing analysis matches the simulation results very well thus validating the analysis presented in this paper.

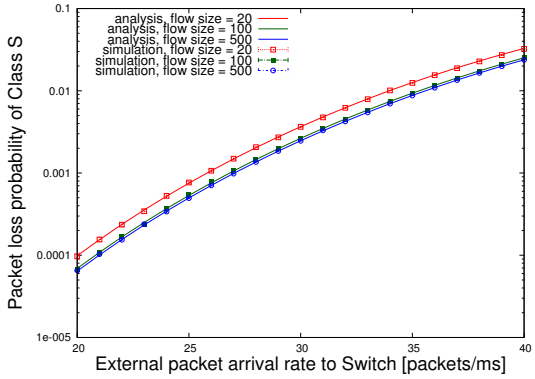

Fig. 3. Effect of external packet arrival rate on packet loss probability of class S.

B. Effect of flow size

In this subsection, we discuss the influence of average flow size on network performance for $\lambda = 20$ and 40, and $\mu_1 = 50$ and 100. Both Figure 6 and Figure 7 exhibit a trend of packet loss probability slowly decreasing as a function of increasing flow size. When average flow size gets larger while keeping external arrival rate unchanged, the arrival rate of class F packets becomes smaller, and then the packet loss probability decreases and gradually gets less sensitive to the average flow size.

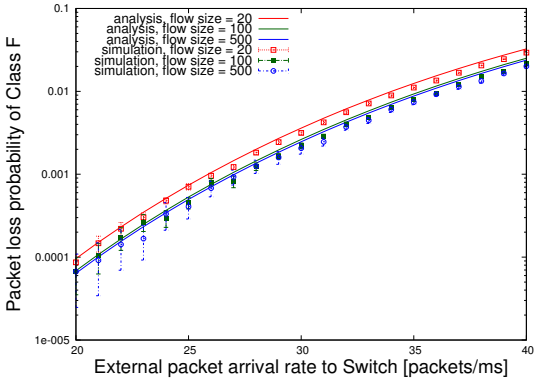

Fig. 4. Effect of external packet arrival rate on packet loss probability of class F.

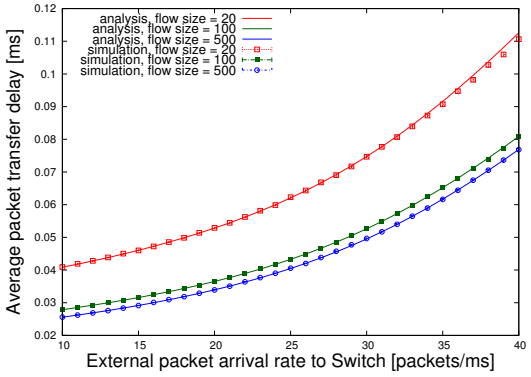

Fig. 5. Average packet transfer delay in the system.

Figure 8 shows the relationship between the average packet transfer delay and average flow size. Recall that only the first packet of a new flow is sent to the controller and thus most of the packets in a large sized flow are directed to the output port without recourse to the controller. Hence, for long flows,

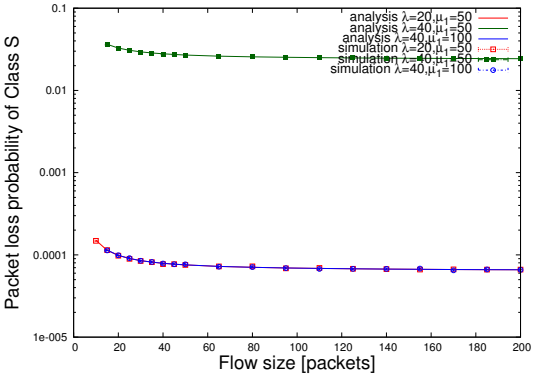

Fig. 6. Effect of average flow size on packet loss probability of class S.

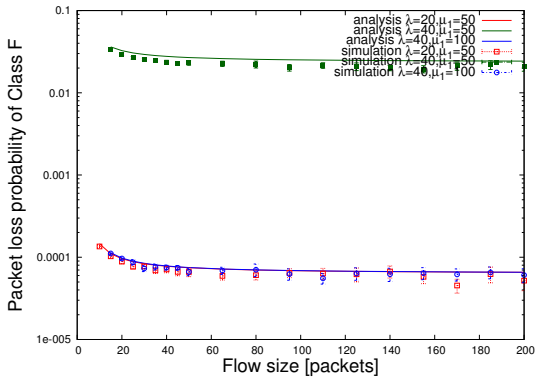

Fig. 7. Effect of average flow size on packet loss probability of class F.

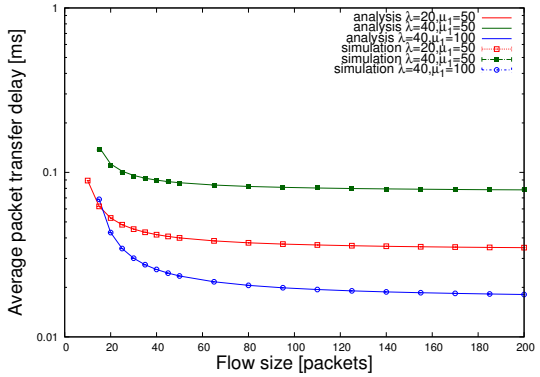

Fig. 8. Packet transfer delay in the system.

the average packet transfer delay is insensitive to flow sizes, since the number of packets transferred to the controller is small, which alleviates queueing delays at both the switch and controller.

VI. Conclusion

In this paper, we have proposed a queueing model to analyze an OpenFlow–based software defined network as realistically and accurately as possible. We constructed a continuous-time Markov chain whose states represent the number of class C, class F and class S packets, and derived average packet transfer delay in the whole system and packet loss probabilities in class S and class F.

In our ongoing research, we are extending the queueing model to consider a network comprising a single controller with multiple switches while future work includes modelling the performance of various controller architectures. Concurrently, we are applying queueing analysis to evaluate the performance of test bed and then production networks. After validating our analytical results in comparison with both experimental results from the test bed and simulation results, we will then use these queueing models for future network design and dimensioning.

References

[1] N. Feamster, J. Rexford, and E. Zegura, "The Road to SDN – An intellectual history of programmable networks," *ACM Queue*, vol. 11, no. 12, December 2013.

[2] M. Kobayashi, S. Seetharaman, G. Parulkar, G. Appenzeller, J. Little, J. Reijendam, P. Weissmann, and N. Mckeown, "Maturing of openflow and software-defined networking through deployments," *Computer Networks*, 2013.

[3] B. Nunes, M. Mendonca, X. Nguren, K. Obraczka, and T. Turletti, "A Survey of Software-Defined Networking: Past, Present, and Future of Programmable Networks," *IEEE Communications Surveys & Tutorials*, 2014.

[4] D. Kreutz, F. M. V. Ramos, P. E. V. issimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-Defined Networking: A Comprehensive Survey," *Proceedings of the IEEE*, vol. 103, no. 1, Jan 2015.

[5] M. Jarschel, S. Oechsner, D. Schlosser, R. Pries, S. Goll, and P. Tran-Gia, "Modeling and Performance Evaluation of an OpenFlow Architecture," in *Proceedings of the 23rd ITC*, San Francisco, CA, USA, Sep 2011, pp. 1–7.

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

[7] S. H. Yeganeh, A. Tootoonchian, and Y. Ganjali, "On Scalability of Software-Defined Networking," *IEEE Communications Magazine*, vol. 51, no. 2, pp. 136–141, Feb 2013.

[8] J. Hu, C. Lin, X. Li, and J. Huang, "Scalability of Control Planes for Software Defined Networks: Modeling and Evaluation," in *Proceedings of the IEEE/ACM International Symposium on Quality of Service (IWQoS)*, Hong Kong, China., 26-27 May 2014.

[9] L. Yao, P. Hong, and W. Zhou, "Evaluating the controller capacity in software defined networking," in *Proceedings of the 23rd International Conference on Computer Communication and Networks (ICCCN)*, Shanghai, China, 4-7 Aug 2014.

[10] N. Makimoto, *Queueing Algorithm – Matrix Analytic Approach*. Asakura Publishing Co. Ltd., 2001, in Japanese.

[11] G. Latouche and V. Ramaswami, *Introduction to Matrix Analytic Methods in Stochastic Modeling*. ASA-SIAM, 1999.