Performance Analysis of General P4 Forwarding Devices with Controller Feedback

Nicolai Kröger
Technical University of Munich
Munich, Bavaria, Germany
nicolai.kroeger@tum.de

Fidan Mehmeti
Technical University of Munich
Munich, Bavaria, Germany
fidan.mehmeti@tum.de

Hasanin Harkous
Nokia
Munich, Bavaria, Germany
hasanin.harkous@nokia.com

Wolfgang Kellerer
Technical University of Munich
Munich, Bavaria, Germany
wolfgang.kellerer@tum.de

ABSTRACT

Software-Defined Networking (SDN) lays the foundation for the operation of future networking applications. The separation of the control plane from the programmable data plane increases the flexibility in network operation. One of the most used languages for describing the packet behavior in the data plane is P4. It allows protocol and hardware independent programming. With the expanding deployment of P4 programmable devices, it is of utmost importance to understand their performance behavior and limitations in order to design a network and provide Quality of Service (QoS) guarantees. One of the most important performance metrics is the packet mean sojourn time in a P4 device. While previous works already modeled the sojourn time in P4 devices with controller feedback, those models were rather simplified and could not capture the system behavior for general cases, resulting in a potential highly inaccurate performance prediction. To bridge this gap, in this paper, we consider the system behavior of P4 devices for the general case, i.e., under general assumptions. To that end, we model the behavior with a queueing network with feedback. As it is impossible to provide closed-form solutions, we consider different approximations for the mean sojourn time. We validate our results against extensive realistic simulations, capturing different behaviors in the data and control planes. Results show that the most accurate approximation in almost all cases is the one in which the queues are decoupled and considered as independent despite the fact that there are dependencies. The level of discrepancy in the worst case does not exceed 18.2% for service times distributions with a coefficient of variation not greater than 1.

CCS CONCEPTS

• **Networks** → **Network performance modeling**; Network simulations.

KEYWORDS

P4, SDN, Queueing networks with feedback.

ACM Reference Format:
Nicolai Kröger, Fidan Mehmeti, Hasanin Harkous, and Wolfgang Kellerer. 2022. Performance Analysis of General P4 Forwarding Devices with Controller Feedback. In *Proceedings of the International Conference on Modeling Analysis and Simulation of Wireless and Mobile Systems (MSWiM '22), October 24–28, 2022, Montreal, QC, Canada.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3551659.3559045

MSWiM'22, October 24–28, 2022, Montreal, Canada

ACM ISBN 978-1-4503-9479-6/22/10…$15.00
https://doi.org/10.1145/3551659.3559045

1 INTRODUCTION

Emerging applications in our digitized world, such as telemedicine or autonomous driving, impose stringent requirements on the underlying communication networks in terms of latency, throughput and more. In order to satisfy those requirements, Software-Defined Networking (SDN) evolved as a new networking paradigm which decouples the data plane from the control plane [16]. This approach allows to optimize the network management in a holistic way as one controller can define the packet processing in several forwarding devices. Furthermore, the introduction of Programmable Data Planes (PDP) enables to control the packet processing with a higher granularity. One of the most well-known concepts for data plane programming is the domain-specific P4 language concept [3], which is used for defining the packet processing in the forwarding devices in detail.

As P4 devices are becoming more and more popular in networking, it is of utmost importance to understand their performance behavior and limitations. Having this information before their deployment enables to properly design a network with regards to Quality of Service (QoS) guarantees such as low latency and high throughput. One of the most important performance metrics is the mean packet sojourn time. In a first step to model the packet sojourn time for P4 devices, the authors in [9] measured the mean sojourn time of packets for the P4-enabled devices Netronome SmartNIC [18], NetFPGA [23] and T4P4S [21], based on the complexity of the loaded P4 program. Those measurements are then further used in [10] where one P4 forwarding device with controller feedback is modeled as a Jackson network. Their results allow a first understanding of the behavior and limitations of a P4 device with controller interaction. However, as the service times are assumed to be exponentially distributed, both in the data plane and the controller, it may not reflect the behavior of real devices and

therefore the analytical performance results can vary from the real ones considerably.

Refining the system model with more general and hence realistic distributed services times is not a straightforward task though. Relaxing the assumption of exponentially distributed service times leads to a general system where the well-known exact equations for the packet mean sojourn time in the system do not hold anymore. Moreover, the real behavior can only be approximated. Another problem lies in obtaining measurement results for all possible combinations of real data planes and controllers to compare with the analytical results which is impossible. Hence, finding a way to analyse approximation techniques with results close to reality would significantly improve the understanding related to the performance of P4 devices.

In particular, several important research questions in the context of the performance of P4-devices and more specifically, the packet mean sojourn time are:

- How can the packet mean sojourn time in realistic P4 devices be obtained analytically? How well does the model fit to the actual results?
- What are the limitations of such models?
- How does the interaction with the controller impact the packet mean sojourn time?

These questions are addressed in this paper by modeling a P4 forwarding device with controller feedback using a queueing theoretic model. We assume generally distributed service times on the data plane and control plane. In a further step, we analyse approximation approaches for the packet mean sojourn time in such systems and compare the analytical results to those obtained by simulation in order to cover a broad operation region, by varying a wide amount of parameters. The evaluation results in this paper enable to obtain analytical results for the packet mean sojourn time of P4 devices and therefore to understand the behavior of our metric of interest. The main message of this work is that the most accurate approximation is the one in which the queues are decoupled and are considered as mutually independent despite the fact that in reality this is not the case.

Specifically, our main contributions are:

- We analyze different approximation methods for the mean sojourn time of a packet in a P4 forwarding device with controller feedback and generally distributed service times.
- We compare the analytical results obtained by the approximation approaches to those obtained by simulation for widely varied parameters and operation regions.

The remainder of this paper is organized as follows. In Section 2, some related work and in Section 3, some background information on the P4 programming language is provided. Then, in Section 4 the queueing model of a P4 forwarding device with controller feedback is presented. This is followed by a theoretical analysis of approximation techniques for the packet mean sojourn time of such a system in Section 5. The performance is evaluated in Section 6. Finally, Section 7 concludes the paper.

2 RELATED WORK

The introduction of SDN enabled splitting the control plane from the data plane. Since then, many works in the literature targeted modeling the performance of these devices.

To start with, OpenFlow (OF)-based switches were modeled in [12] as a feedback-oriented queueing system similar to that studied in this paper. However, in [12] each qeueing system at the control and data plane is assumed to have exponentially distributed service times. A follow up work in [15] refactors the model and presents it as a Jackson network. A model for networks made up of such models is developed and proposed in [14]. Goto *et al.* [6] improves on the model presented in [12] for a single node by incorporating processing priorities for packets going to the switch. Ansell *et al.* [1] introduce a control plane application based on queueing theory for monitoring networks and predicting their behavior.

P4 programmable data planes extended the programmability offered by the SDN paradigm to the data plane. Similar to the approach presented in [12], the model in [10] adopts a feedback-oriented queueing system to abstract the behavior of the system. Moreover, it considers that the processing delay, and thus the forwarding delay, at the data plane could vary based on the complexity of the loaded P4 data path. Accordingly, it parameterizes the average service time of the data plane's exponentially distributed service process based on the P4 pipeline's complexity. The impact of different P4 atomic operations on the processing delay of different P4 devices is evaluated in [9], where a method is proposed to estimate the packet forwarding delay on different P4 devices as a function of the complexity of the loaded P4 program. Other works in the literature evaluates and models the performance of P4 programmable devices using different approaches. For example, a benchmarking suite for P4 programmable devices is proposed and used in [4] for evaluating the performance of three different P4 devices. The authors of [20] evaluate and model the performance of hardware and software-based P4 devices according to different criteria considered relevant for each type of device. They model the packet rate that could be handled by the T4P4S software switch when the number of incoming flows increases. On the other hand, they decided to focus on modeling the usage of processing resources on TOFINO ASIC switch as they identified this metric to be more important for this type of devices.

This work reconsiders the feedback-oriented queueing network model considered in [10], by loosing the assumptions related to the exponential distribution of the service times at the control and data planes. Instead, these distributions are considered generic, and the sojourn time of the system is re-evaluated accordingly based on different theoretical approximations.

3 P4 BACKGROUND

Network programmability was first introduced with the SDN architecture. SDN separates the control plane from the data plane, introduces a centralized controller entity between the two planes, enables programmability at the control plane, and defines an interface (such as the OpenFlow protocol) to push messages to the data plane telling it how to forward packets. This approach to programming networks is found to be limited since it is bound to the

predefined protocols and actions supported by the OpenFlow architecture and protocol. To address these issues, P4 programmability was introduced to push programmability to the data plane, where the device's behavior can be customized independently from any predefined headers or actions (as in the OpenFlow case).

The latest P4 version, i.e., ($P4_{16}$), separates the syntax of the language from the details of the hosting target, which is included in the target's architecture description. This P4 architecture describes the programmable blocks on the target and the interfaces to program them. This makes the P4 language target-independent so that it can be used to program any type of packet processor with a given architecture description.

The P4 programmability is mainly concerned with defining the packet processing behavior at the data plane of a device. Although the processing behavior is described based on the match-action abstraction similar to OpenFlow, unlike the OpenFlow case, P4 allows for defining arbitrary protocols and custom actions. When a packet arrives at a P4 data plane, it goes through a sequence of processing operations defined in the P4 program. First, the headers of the packet are extracted in the parser stage, which is described as a finite state machine. Next, the ingress and egress processing stages in the P4 program transform the headers of the packet according to the defined tables in the program. Each table is defined based on a list of matching keys, and a list of possible custom actions that can be invoked upon matching. These actions are customizable and written as functions.

The P4 program defines the pipeline that a packet can go through, especially in terms of the tables that will be applied. It is the role of the control plane to populate the tables in this pipeline with the rules that achieve the intended behavior of the use case application. If a packet arriving at a P4 data plane does not find a matching rule in the visited tables, it is forwarded to the control plane, which takes care of processing this packet, sends this packet back to the data plane, and sends a rule to instruct following packets how to be processed. Accordingly, a packet can either get processed only at the data plane, or it could be processed at the data plane followed by the control plane followed by the data plane again. This lifecycle of packets in P4-based systems guides the abstraction of the behavior of these systems as a feedback-oriented system, which is evaluated and studied in the rest of this paper.

4 P4 PERFORMANCE MODEL

In this section, we describe the performance model for P4 devices with controller feedback, followed by a detailed description of data plane and control plane.

4.1 System Description

In SDN, the data plane is separated from the control plane. The former plane is responsible for the processing of packets, whereas the latter defines *how* they are processed. In case of a programmable data plane, the packet handling can be described with languages such as P4. The SDN paradigm allows one controller to control several data planes.

Modelling a network consisting of data and control planes before deployment enables to understand the network behavior and properly dimension the participating devices. One of the most important performance metrics is the *mean packet sojourn time*, i.e., the overall time a packet spends in the system. In a first step, a network consisting of only one data plane with controller feedback is considered. Fig. 1 shows a model of such a system. The data plane (lower part) and the control plane (upper part) are both represented by their own queues, each consisting of a waiting queue (buffer) and a server. We consider a data plane which is programmable with P4.

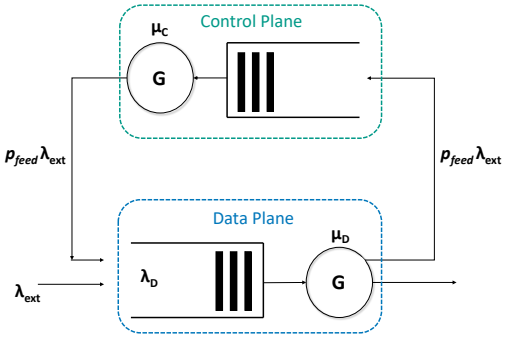


Figure 1: Queueing model for a P4 forwarding data plane with controller feedback.

The path for a packet in the system is now as follows. A packet arrives to the system following a Poisson arrival process with rate λ_{ext}. Then, it is first processed in the data plane with a service rate of μ_D. As the data plane is P4-enabled, this packet processing is defined by a P4 program. If an entry of the match-action table matches the packet information, it leaves the system. On the other hand, with a probability of p_{feed} we assume that there is no table entry and in that case the packet is sent back to the controller, which implies an arrival rate of $p_{feed}\lambda_{ext}$. The control plane processes the packet with rate μ_C, adds a table rule in the data plane for that packet and sends it back to the data plane. After being processed by the data plane (for the second time), the packet leaves the system as now there is a matching table entry. Thus, a packet can only be sent to the controller once.

4.2 Data Plane Processing

Upon arriving to the data plane, the packet is stored in the buffer. We assume that the buffer size of the data plane is infinite and the service discipline is First Come First Served (FCFS). There are two types of packets arriving to the data plane: *external packets* entering the system at that moment (with rate λ_{ext}) and *feedback packets* entering the data plane from the controller, i.e., for the second time (with rate $p_{feed}\lambda_{ext}$). Therefore, the overall arrival rate at the data plane is

$$\lambda_D = \lambda_{ext} + p_{feed} \cdot \lambda_{ext} = (1 + p_{feed})\lambda_{ext}. \quad (1)$$

Even though the external arrival process follows a Poisson process, this is not the case for the overall data plane arrival process. This stems from the fact that the controller feedback process depends on the external arrival process, violating the requirement for *independent increments*, characteristic for Poisson processes. The service

times in the data plane follow a general distribution with a mean of $\mathbb{E}[S_D]$ (the corresponding service rate is thus $\mu_D = \frac{1}{\mathbb{E}[S_D]}$) and a standard deviation of σ_D. The average total time a packet spends in the data plane (the sum of the corresponding service time and queueing delay) is denoted by $\mathbb{E}[T_D]$.

4.3 Control Plane Processing

The control plane also has an infinite buffer size and follows the FCFS order of service. The arrival process is a fraction of the departure process of the data plane and has a rate of

$$\lambda_C = p_{feed} \cdot \lambda_{ext}. \tag{2}$$

The service times in the control plane follow a general distribution with a mean of $\mathbb{E}[S_C]$ (the corresponding service rate is thus $\mu_C = \frac{1}{\mathbb{E}[S_C]}$) and a standard deviation of σ_C. The average total time a packet spends in the control plane (the sum of the corresponding service time and queueing delay) is denoted by $\mathbb{E}[T_C]$.

Both queues together form a network, known as *queueing networks with feedback* [22].

4.4 Packet Sojourn Time

The mean packet sojourn time is an important performance metric. Packets in the system either directly leave the system after being processed in the data plane or experience also controller involvement. Hence, the mean packet sojourn time $\mathbb{E}[T]$ consists of the time a packet spends at the data plane $\mathbb{E}[T_D]$ and the control plane $\mathbb{E}[T_C]$. This is expressed as

$$\mathbb{E}[T] = \mathbb{E}[T_D] + p_{feed}\left(\mathbb{E}[T_D] + \mathbb{E}[T_C]\right), \tag{3}$$

or equivalently,

$$\mathbb{E}[T] = (1 + p_{feed})\mathbb{E}[T_D] + p_{feed}\mathbb{E}[T_c]. \tag{4}$$

Analyzing queueing networks and obtaining closed-form solutions is possible only for Jackson networks [7], i.e., when all the processes in the system are characterized by the memoryless property. Given that we assume non-exponential service times at the data plane and general distribution at the controller, the queueing network at hand is not a Jackson network. Therefore, we cannot exploit the results from there.

The aforementioned assumptions and requirements make the derivation of exact analytical solutions in closed-form infeasible. Consequently, in this paper, we focus on comparing the prediction accuracy three approximation approaches offer (see Section 5).

In order to cover a wide range of possible distributions, we analyze the approximations for three different distribution types on the data plane and control plane service times based on their *coefficient of variation* c_V, which is defined as the ratio of the standard deviation of a random variable and its mean, and is a measure of how dispersed the samples from the mean of a random variable are.

The first considered distribution is the Erlang distribution, which represents a distribution with low variability, for which it holds $c_V < 1$. The second distribution is the exponential (memoryless) distribution, for which $c_V = 1$. Finally, the hyper-exponential distribution (for which $c_V > 1$) represents the scenario with a heavy-tailed service time.

In the next section, we derive the theoretical equations for all approximation approaches. Then, in Section 6 we use simulations to compare the theoretical results to the simulation for the different approaches.

5 APPROXIMATION APPROACHES

In this section, we first present two state-of-the-art approaches to approximate the mean packet sojourn time. Then, we use the equations corresponding to two independent M/G/1 queues.

5.1 Diffusion Approximation

The first approximation has been introduced in [13] and [19]. They use the diffusion approximation technique to approximate the number of packets $\mathbb{E}[N]$ in a queueing system with generally distributed service times. With this approach, the probability of finding k packets in such a system with arrival rate λ, service rate μ, utilization $\rho = \frac{\lambda}{\mu}$ and coefficients of variation c_A for the arrival and c_S for the service process can be approximated as:

$$\hat{\pi}(k) = \begin{cases} 1 - \rho, & k = 0 \\ \rho(1-\hat{\rho})\hat{\rho}^k - 1, & k > 0 \end{cases} \tag{5}$$

where

$$\hat{\rho} = \exp\left(\frac{-2(1-\rho)}{\rho c_A^2 + c_S^2}\right). \tag{6}$$

As a stable queue is required, i.e., $\rho < 1$, for the mean number of packets we have

$$\mathbb{E}[N] = \sum_{k=1}^{\infty} k\hat{\pi}(k) = \frac{\rho}{1-\hat{\rho}}. \tag{7}$$

These results are now applied to our model (following the steps like in [2]) to find an approximation for the mean sojourn time assuming known mean and standard deviations of the service times. First, we need to introduce the parameters e_D and e_C, which denote the frequency of visits to the data plane and controller by each packet, respectively. Hence, $e_D = 1 + p_{feed}$ and $e_C = p_{feed}$.

The utilization at the data plane is

$$\rho_D = \frac{e_D \lambda_{ext}}{\mu_D}, \tag{8}$$

whereas the control plane

$$\rho_C = \frac{e_C \lambda_{ext}}{\mu_C}. \tag{9}$$

With the coefficients of variations of the data plane service process $c_{D,S}$ and $c_{C,S}$ for the controller, the coefficients of variations for the arrival processes at the data plane $c_{D,A}$ and the controller $c_{C,A}$ can be obtained from the following expression [13]:

$$c_{i,A}^2 = 1 + \sum_{j=0}^{N} (c_{j,S}^2 - 1) \cdot p_{j,i}^2 \cdot \frac{e_j}{e_i}, \tag{10}$$

where N is set to 2 with $i \in \{D, C\}$ and

$$j = \begin{cases} j = 0, & \text{representing external arrival stream} \\ j = 1, & \text{representing data plane} \\ j = 2, & \text{representing control plane.} \end{cases} \tag{11}$$

Using $p_{0,D} = p_{2,D} = 1$ (every external packet and every packet from the controller has to go to the data plane), $p_{0,C} = p_{1,D} = p_{2,C} = 0$ (external packets cannot go directly to the control plane, a

packet from the data plane cannot re-enter immediately the data plane, and a packet from the control plane cannot re-enter the control plane again), and $p_{1,C} = \frac{p_{feed}}{1+p_{feed}}$ (the probability that a packet in the data plane is a feedback packet) as well as exploiting the fact that the coefficient of variation of the external Poisson arrival process is 1 ($c_{0,S} = 1$), $c^2_{D,A}$ can be calculated as

$$c^2_{D,A} = 1 + (c_{C,S} - 1) \cdot \frac{e_C}{e_D}. \tag{12}$$

Similarly, for the control plane we have

$$c^2_{C,A} = 1 + (c_{D,S} - 1) \cdot \frac{p^2_{feed}}{(1+p_{feed})^2} \cdot \frac{e_D}{e_C}. \tag{13}$$

Further,

$$\hat{\rho}_D = \exp\left(\frac{-2(1-\rho_D)}{\rho_D c^2_{D,A} + c^2_{D,S}}\right), \tag{14}$$

and

$$\hat{\rho}_C = \exp\left(\frac{-2(1-\rho_C)}{\rho_C c^2_{C,A} + c^2_{C,S}}\right). \tag{15}$$

The mean number of jobs at the data plane $\mathbb{E}[N_D]$ and at the control plane $\mathbb{E}[N_C]$ are

$$\mathbb{E}[N_D] = \frac{\rho_D}{1-\hat{\rho}_D}, \tag{16}$$

and

$$\mathbb{E}[N_C] = \frac{\rho_C}{1-\hat{\rho}_C}. \tag{17}$$

Using Little's law [8], Eq. (1) and Eq. (2), the mean sojourn time for the data plane $\mathbb{E}[T_D]$ and for the control plane $\mathbb{E}[T_C]$ can be calculated as

$$\mathbb{E}[T_D] = \frac{\mathbb{E}[N_D]}{(1+p_{feed})\lambda_{ext}}, \tag{18}$$

and

$$\mathbb{E}[T_C] = \frac{\mathbb{E}[N_C]}{p_{feed}\lambda_{ext}}. \tag{19}$$

Replacing Eq.(18) and Eq.(19) into Eq.(4), we obtain the approximated mean sojourn time using this approach.

The coefficient of variation for exponentially distributed service times is 1. For the special case of Erlang distributed service times with a service rate of μ with k stages, the squared coefficient of variation reduces to $\frac{1}{k}$ [8]. The squared coefficient of variation for a hyper-exponential distribution with two branches, i.e., consisting of 2 parallel exponential distributions each with probability p and $1-p$ and rates μ_1 and μ_2, with mean $\frac{p}{\mu_1} + \frac{1-p}{\mu_2}$, and squared coefficient of variation of [8]

$$c^2_{v,hyper} = \frac{2\left(\frac{p}{\mu_1^2} + \frac{1-p}{\mu_2^2}\right)}{\left(\frac{p}{\mu_1} + \frac{1-p}{\mu_2}\right)^2} - 1. \tag{20}$$

From Eq.(20), it can be shown that $c_{v,hyper} > 1$.

For each of these special cases, we can obtain the corresponding mean sojourn times, by substituting the corresponding coefficients of variation in Eqs.(12)-(15).

5.2 Modified Diffusion Approximation

The Diffusion approximation presented in Section 5.1 is further modified in [5] and [17]. In particular, by adjusting the calculation of the mean number of packets

$$\mathbb{E}[N] = \rho\left(1 + \frac{\rho c^2_A + c^2_S}{2(1-\rho)}\right). \tag{21}$$

the authors claims to achieve more precise results for regions of higher utilization. The calculation approach follows the same scheme as in Section 5.1, with $\mathbb{E}[N_D]$ and $\mathbb{E}[T_C]$ now obtained as

$$\mathbb{E}[N_D] = \rho_D\left(1 + \frac{\rho c^2_{D,A} + c^2_{D,S}}{2(1-\rho_D)}\right), \tag{22}$$

and

$$\mathbb{E}[N_C] = \rho_C\left(1 + \frac{\rho c^2_{C,A} + c^2_{C,S}}{2(1-\rho_C)}\right). \tag{23}$$

Depending on the distribution of the service time, the actual values for $\mathbb{E}[N_D]$ and $\mathbb{E}[N_C]$ can be obtained from Eq.(22) and Eq.(23), respectively.

Finally, substituting Eq.(22) and Eq.(23) into Eq.(4), we obtain the approximate mean sojourn time with this approach.

5.3 M/G/1 Approximation

This approximation is based on the assumption that both queues are considered as mutually independent with corresponding Poisson arrival processes. Looking at each queue separately, the mean sojourn time $\mathbb{E}[T]$ can be calculated with the well known expression for the average sojourn time in an M/G/1 queue [2]:

$$\mathbb{E}[T] = \frac{\rho}{1-\rho} \cdot \frac{\mathbb{E}[S]}{2} \cdot (c^2_S + 1) + \mathbb{E}[S], \tag{24}$$

where $\mathbb{E}[S]$ represents the mean service time and c_S is the coefficient of variation of the service time. Thus, $\mathbb{E}[T_D]$ and $\mathbb{E}[T_C]$ used for the calculation of the mean sojourn time of the P4 device system are

$$\mathbb{E}[T_D] = \frac{\rho_D}{1-\rho_D} \cdot \frac{\mathbb{E}[S_D]}{2} \cdot (C^2_{D,B} + 1) + E[S_D], \tag{25}$$

and

$$\mathbb{E}[T_C] = \frac{\rho_C}{1-\rho_C} \cdot \frac{\mathbb{E}[S_D]}{2} \cdot (C^2_{C,B} + 1) + \mathbb{E}[S_C]. \tag{26}$$

Finally, replacing Eq.(25) and Eq.(26) into Eq.(4), we obtain the mean sojourn time approximation with this approach.

6 EVALUATION

In this section, the different approximation techniques for the packet mean sojourn time are evaluated by comparison to simulation result. First, the simulation setup and varied parameters are described. Then, the results for the different combinations of data plane and control plane service time distributions are presented. Finally, the analytical results of all approaches are compared in order to find a good model for the different operation regions.

6.1 Simulation Setup

In order to evaluate the goodness of the analytical results for different regions of operations, the P4 device model with feedback is implemented in a packet-based MATLAB simulation and various parameters are varied. First, the data plane and control plane service times each are either exponentially, Erlang or hyper-exponentially distributed, representing distributions with lower and higher variance. The mean service time of the data plane $\mathbb{E}[S_D] = 45.9\mu s$ is taken from [10] as the mean forwarding time for T4P4S running VxLAN. Additionally, the controller mean service times are also taken from [10]: $\mathbb{E}[S_C] = 31\mu s$ represents a controller which is faster, $\mathbb{E}[S_C] = 240\mu s$ which is medium, and $\mathbb{E}[S_C] = 10$ ms which is much slower compared to the data plane, i.e., comparable to an ONOS controller [11]. Whenever the Erlang distribution is used, the shape parameter is arbitrarily set to $k = 100$ in order to ensure a $c_{v,erl} = 0.1$. The hyper-exponential distribution consists of two exponential distributions with rates μ_1 and μ_2. The probability to choose the first exponential distribution is $p = 0.9$, and $1 - p = 0.1$ otherwise. In order to ensure a $c_{v,hyper} > 1$, the following relation is set:

$$\mu_2 = 100\mu_1. \tag{27}$$

From Eq.(20) it can be shown that $c_{v,hyper} = 15.8488$, independently of the used service time averages.

Another varied parameter is the probability for packets being sent to the controller, i.e., $p_{feed} \in \{0, 0.1, 0.5, 1\}$, showing the impact of the controller on the overall sojourn time. Each simulation is run for 100000 packets. For each of those cases, the controller utilization is increased to a maximum of $\rho_c = 0.95$, always ensuring a stable controller queue. Also, a stable data plane is always ensured. The results obtained by those simulations are compared to the approximation results by taking the average error over the increasing load cases of the controller for one value of p_{feed}.

6.2 Error Evaluation

In the following, the simulation results are compared to those obtained by the three approximation approaches for all combinations of service times distributions in the data plane and control plane.

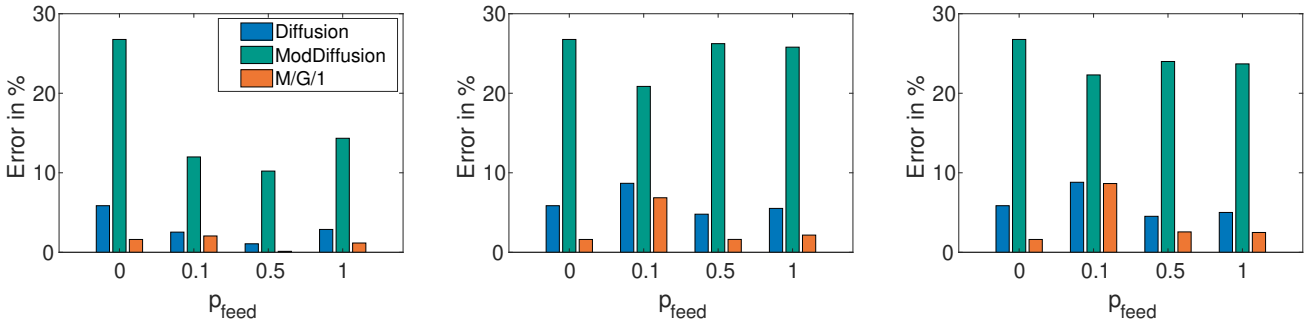
6.2.1 Exponential-Exponential. Fig. 2 shows the errors of the different approximations for all controller processing speeds and a data plane as well as a control plane with exponentially distributed service times, i.e., both distributions have a $c_V = 1$. In the case of the fast controller (see Fig. 2a) and a $p_{feed} = 0$, i.e., no controller interaction, the data plane can be modelled as an *M/G/1* queue and therefore the error of the closed-form equations for the *M/G/1* approach is very small. Additionally, the error results are independent of the controller speed. This also holds for other data plane distributions than the exponential one and when $p_{feed} = 0$. Hence, this case is not explained in the rest of this paper again. While the Modified Diffusion approach has high errors, the Diffusion approach is closer to the simulation and the *M/G/1* approximation is the best for all values of p_{feed}. This behavior can also be observed for the medium (see Fig. 2b) and the fast controller (see Fig. 2c). In both cases, the Modified Diffusion approach approximates the simulation significantly worse than the other two approaches. Note, that for the *M/G/1* and the Diffusion approximation the highest error appears for a low value of p_{feed}.

6.2.2 Exponential-Erlang. In the second case, the service times of the data plane are exponentially and those of the control plane are Erlang distributed. The approximation errors are shown in Fig. 3. For the fast controller case (see Fig. 3a), the Modified Diffusion approach has a high error compared to the other ones. However, for higher values of p_{feed} the error is slightly smaller than for the Diffusion approach. The *M/G/1* approximation lead in all case to the lowest errors. For the medium controller (see Fig. 3b), the behavior of the first two approximations change. While for low and medium values of p_{feed} the Diffusion approximation is better than the Modified Diffusion one, this changes for $p_{feed} = 1$. In the slow controller case (see Fig. 3c), the Diffusion approach has a much higher error compared to the other two approximations. Note that for the case of $p_{feed} = 0.1$ the Modified Diffusion is better than the *M/G/1* approach.

6.2.3 Exponential-Hyper-exponential. In this case, the data plane service times are exponentially and those of the controller are hyper-exponentially distributed, i.e., the control plane shows a high variance in its service times compared to the data plane. This impacts the errors of the Diffusion and the Modified Diffusion approach significantly (see Fig. 4). The errors for the fast (see Fig. 4a), the medium (see Fig. 4b) and the slow controller (see Fig. 4c) behave similarly. Even though the Modified Diffusion approach has the highest error for all values of p_{feed}, the Diffusion approximation is only slightly better. On the other side, the *M/G/1* approach produces very small errors in comparison to the other two approaches.

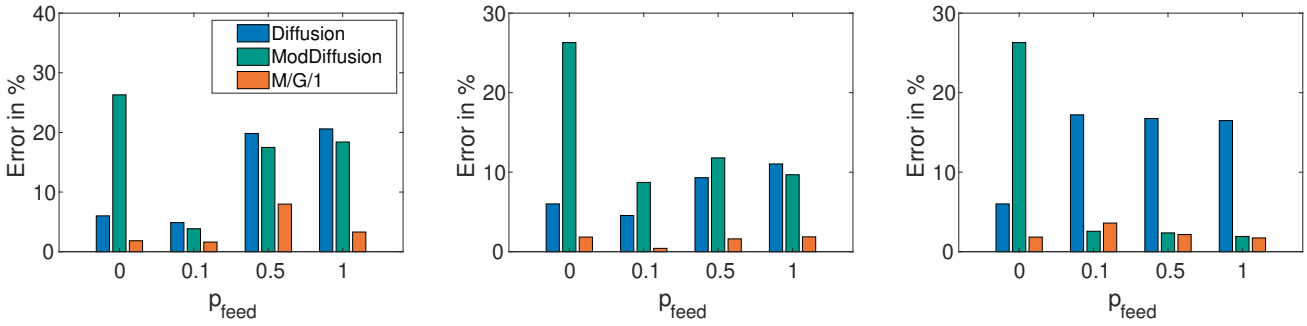
6.2.4 Erlang-Exponential. Fig. 5 shows the approximation errors for a data plane with Erlang and a controller with exponentially distributed service times. In the fast controller case (see Fig. 5a), the Diffusion approach error increases for higher values of p_{feed} and is the worst approximation. As opposed to most of the other cases, the *M/G/1* approach performs slightly worse as the Modified Diffusion one. All errors increase with higher values of p_{feed}. The *M/G/1* approach is the best approximation again for all values of p_{feed} of the medium controller (see Fig. 5b). While the Diffusion approach performs worst for lower values of p_{feed}, the errors for higher values of p_{feed} are smaller than those of the Modified Diffusion approach. For the slow controller and a $p_{feed} > 0$, the error for the Diffusion approach increases for higher values of p_{feed}, but is still smaller than the error of the Modified Diffusion approach. The *M/G/1* performs the best, with decreasing errors for increasing values of p_{feed}.

6.2.5 Erlang-Erlang. Fig. 6 presents the approximation errors for the case, where the service times of both, data plane and control plane, follow the Erlang distribution. For the fast controller case (see Fig. 6a), all errors increase with higher values of p_{feed}. The best performing approximation is the *M/G/1* approach, the worst the Diffusion one. That is the same for the medium controller (see Fig. 6b). However, as the error of the Diffusion approach is only slightly changing for higher values of p_{feed}, that of the *M/G/1* approximation decrease. The slow controller case (see Fig. 6c) shows the same behavior as the fast controller case with the exception



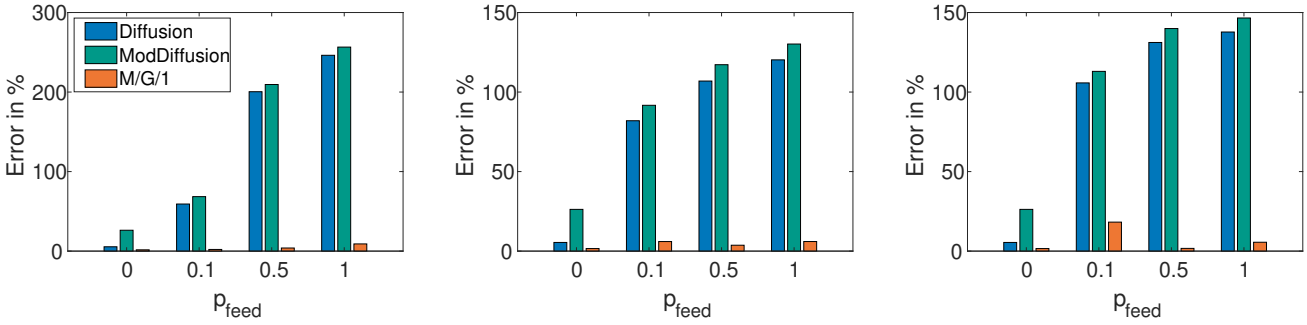
(a) Fast **(b) Medium** **(c) Slow**

Figure 2: Approximation errors for the three different controller speeds and the combination Exponential-Exponential.



(a) Fast **(b) Medium** **(c) Slow**

Figure 3: Approximation errors for the three different controller speeds and the combination Exponential-Erlang.

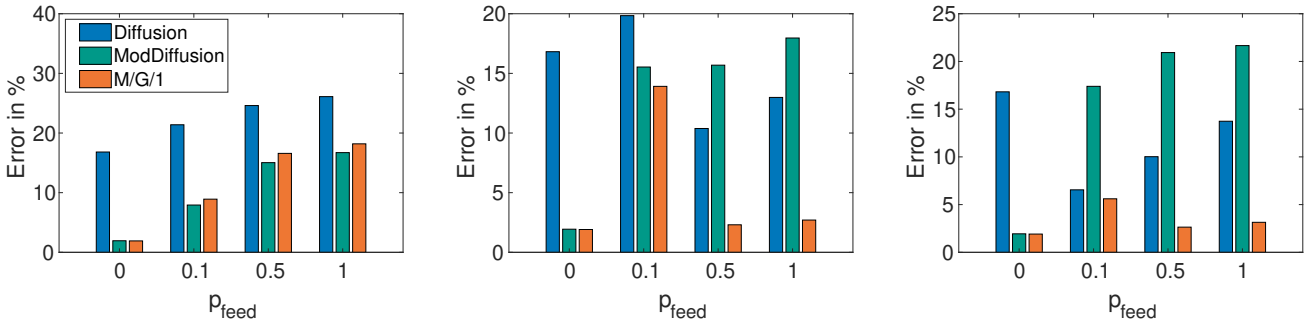


(a) Fast **(b) Medium** **(c) Slow**

Figure 4: Approximation errors for the three different controller speeds and the combination Exponential-Hyper-exponential.

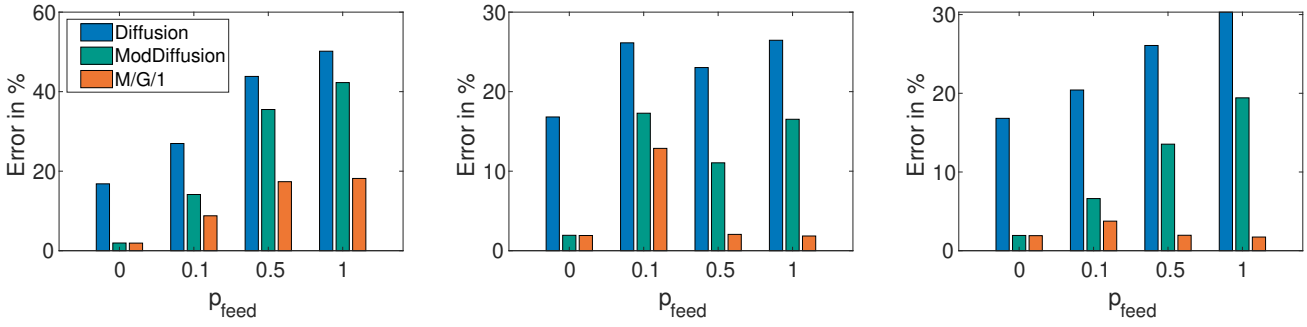
of decreasing errors for increasing values of p_{feed} with the *M/G/1* approximation.

6.2.6 Erlang-Hyper-exponential. In this case, the service times of the data plane are Erlang and those of the controller are hyper-exponentially distributed. The high variance of the control plane service times have a high impact on the errors for the Diffusion and the Modified Diffusion approach (see Fig. 7). The cases of the fast (see Fig. 7a), the medium (see Fig.7b) and the slow controller (see Fig.7c) show the same behavior for the approximation errors. All errors are increasing for higher values of p_{feed} for all approaches except the *M/G/1* one and a medium and slow controller, where the error first decreases for medium values of p_{feed} and then increases again. The Modified Diffusion approach performs slightly worse than the Diffusion approach, but significantly worse than the *M/G/1* one.



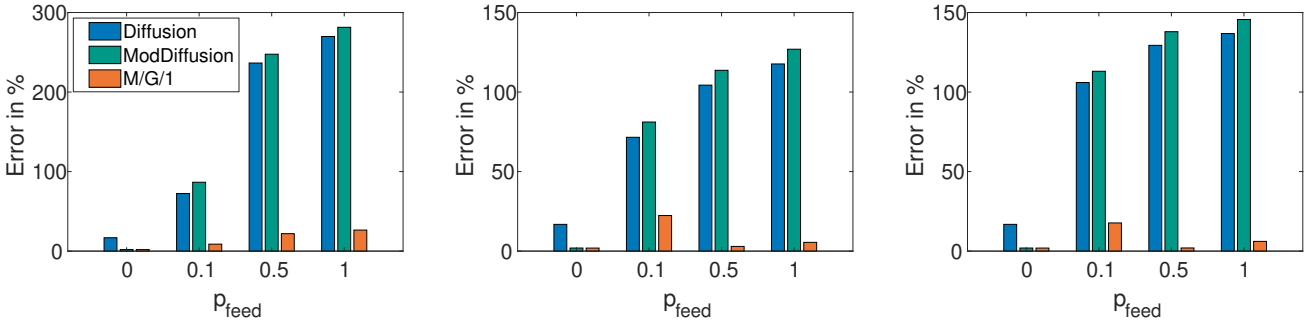
(a) Fast (b) Medium (c) Slow

Figure 5: Approximation errors for the three different controller speeds and the combination Erlang-Exponential.



(a) Fast (b) Medium (c) Slow

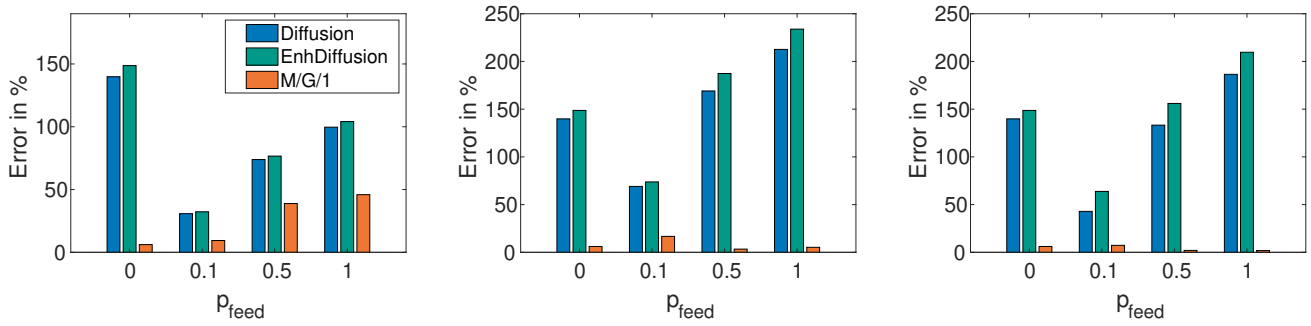
Figure 6: Approximation errors for the three different controller speeds and the combination Erlang-Erlang.



(a) Fast (b) Medium (c) Slow

Figure 7: Approximation errors for the three different controller speeds and the combination Erlang-Hyper-exponential.

6.2.7 Hyper-exponential-Exponential. Fig. 8 shows the approximation errors for a data plane with hyper-exponentially and a control plane with exponentially distributed service times. The high impact of the variance of the data plane service times can be observed for the fast controller (see Fig. 8). Without controller involvement, i.e., $p_{feed} = 0$, the errors for the Diffusion and the Modified Diffusion approach is very high, whereas the *M/G/1* approximation is very close to simulation results. For higher values of p_{feed}, the errors also increase with the Diffusion approach slightly performing better than the Modified Diffusion approach. For all values of p_{feed}, the *M/G/1* approximation performs the best. Note that the controller involvement decreases the error for the Diffusion and Modified Diffusion approach. A similar behavior can be observed for the medium (see Fig. 8b) and slow controller (see Fig. 8c) with the exception of decreasing errors for the *M/G/1* approximation for higher values of p_{feed}, almost tending to 0. Additionally, the errors



(a) Fast **(b) Medium** **(c) Slow**

Figure 8: Approximation errors for the three different controller speeds and the combination Hyper-exponential-Exponential.

for higher values of p_{feed} for the Diffusion and Modified Diffusion approach grow larger than for $p_{feed} = 0$.

6.2.8 Hyper-exponential-Erlang. In this case, the service times of the data plane are hyper-exponentially and those of the control plane are Erlang distributed. The approximation errors are shown in Fig. 9. The behavior of the errors for the fast (see Fig. 9a), the medium (see Fig. 9b) and the slow controller (see Fig. 9c) is similar to to the case of a data plane with hyper-exponentially and a control plane with exponentially distributed service times and is described there.

6.2.9 Hyper-exponential-Hyper-exponential. Fig. 10 shows the approximation errors for the last case, where the service times of both, the data plane and control plane, are hyper-exponentially distributed. Again, the behavior for the fast (see Fig. 10a), the medium (see Fig. 10b) and the slow controller (see Fig. 10c) is similar to the Hyper-exponential-Exponential case and described there.

6.3 Comparison of the Approaches

The *M/G/1* approximation approach outperforms the other two approaches in almost every case significantly. Moreover, the errors for Diffusion and Modified Diffusion approach increase significantly, if a distribution with a coefficient of variation $c_V > 1$ is used for the service times in one or both planes. Using the *M/G/1* approximation, the errors for such cases is much smaller compared to the other approaches. Specifically, the maximal mean error obtained by the *M/G/1* approach for using distributions with coefficients of variation close to 1 or lower for the service times in the P4-forwarding model with a medium or slow controller feedback is 13.9%. Considering a fast controller increases the error up to 18.2%. For a data plane with a service times distribution with c_D close to 1 and a control plane service times distribution with a $c_C > 1$, the error goes up to 26.5% for all controller speeds. Considering a data plane which service times distribution has a $c_D > 1$, the error for a fast controller with any service times distribution is 46.2%. For the medium and slow controller, this error reduces to 18.2%.

7 CONCLUSION

In this paper, we analyzed several approximation approaches for the mean sojourn time of a general P4-forwarding device with controller feedback. Hereby, the data plane and control plane are modeled by using queueing theory. Additionally, the analysis focuses on different regions of controller utilization. We showed that the best approximation, among the compared ones, is the one in which the queue behavior in the data plane is considered independent from the queue behavior in the control plane. In future work, we plan to find an exact expressions for the packet mean sojourn time of such a general P4-forwarding device with feedback. Also, analyzing the behavior beyond the first moment and describing it with closed-form equations is a further objective.

ACKNOWLEDGMENTS

This work was supported by the Federal Ministry of Education and Research of Germany (BMBF) in part under the project "6G-Life", with project identification number 16KISK002, and in part under the project "6G-ANNA", with project identification number 16KISK107.

REFERENCES

[1] J. Ansell, W. K. G. Seah, B. Ng, and S. Marshall. 2016. Making queueing theory more palatable to SDN/OpenFlow-based network practitioners. In *NOMS 2016 - 2016 IEEE/IFIP Network Operations and Management Symposium*, 1119–1124.

[2] Gunter Bolch, Stefan Greiner, Hermann de Meer, and Kishor S. Trivedi. 1998. *Queueing Networks and Markov Chains: Modeling and Performance Evaluation with Computer Science Applications*. Wiley-Interscience, USA. ISBN: 0471193666.

[3] Pat Bosshart et al. 2014. P4: programming protocol-independent packet processors. *ACM SIGCOMM Computer Communication Review*, 44, 3.

[4] Huynh Tu Dang, Han Wang, Theo Jepsen, Gordon Brebner, Changhoon Kim, Jennifer Rexford, Robert Soulé, and Hakim Weatherspoon. 2017. Whippersnapper: a p4 language benchmark suite. In *Proceedings of the Symposium on SDN Research*, 95–101.

[5] Erol Gelenbe. 1975. On approximate computer system models. *J. ACM*, 22, 2, (Apr. 1975), 261–269. DOI: 10.1145/321879.321888.

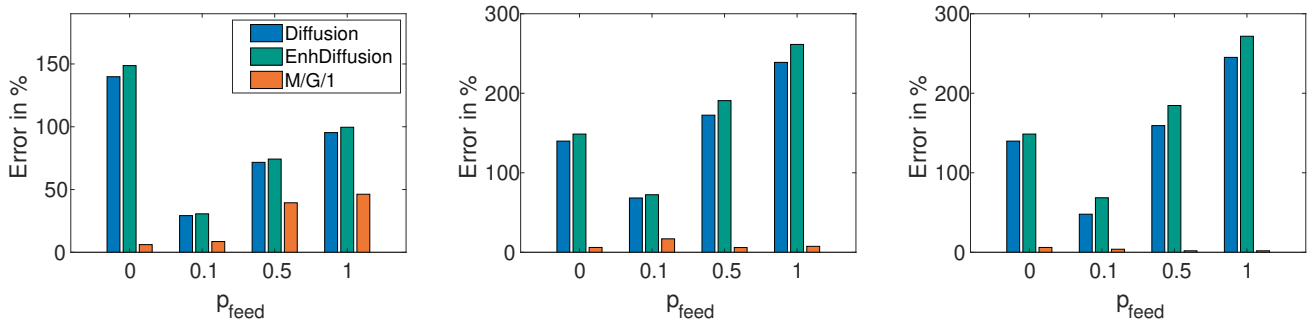
[6] Y. Goto, H. Masuyama, B. Ng, W. K. G. Seah, and Y. Takahashi. 2016. Queueing analysis of Software Defined Network with realistic OpenFlow–based switch model. In *2016 IEEE 24th International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS)*, 301–306.

[7] Donald Gross, John F. Shortle, James M. Thompson, and Carl M. Harris. 2008. *Fundamentals of Queueing Theory*. (4th ed.). Wiley-Interscience.

[8] Mor Harchol-Balter. 2013. *Performance Modeling and Design of Computer Systems: Queueing Theory in Action*. (1st ed.). Cambridge University Press, USA. ISBN: 1107027500.

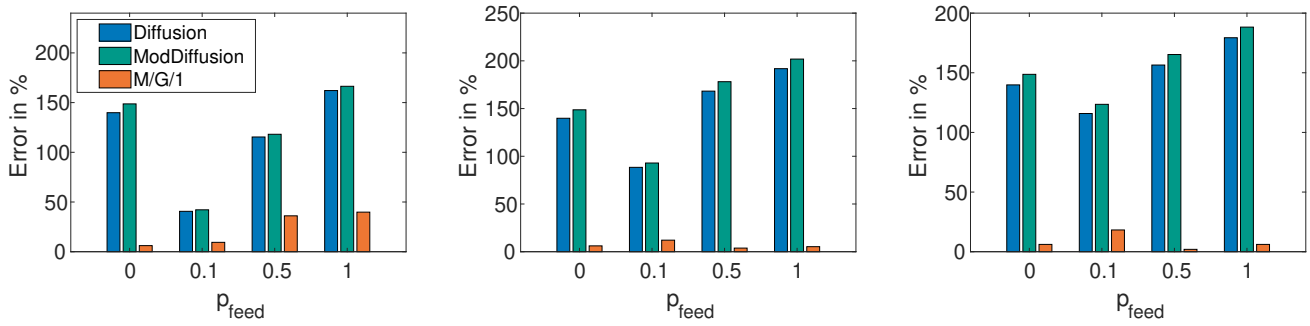
[9] Hasanin Harkous, Michael Jarschel, Mu He, Rastin Pries, and Wolfgang Kellerer. 2021. P8: P4 with predictable packet processing performance. *IEEE Transactions on Network and Service Management*, 18, 3.

[10] Hasanin Harkous, Nicolai Kröger, Michael Jarschel, Rastin Pries, and Wolfgang Keller. 2021. Modeling and performance analysis of p4 programmable devices. In *2021 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)*, 67–73. DOI: 10.1109/NFV-SDN53031.2021.9665141.



(a) Fast **(b) Medium** **(c) Slow**

Figure 9: Approximation errors for the three different controller speeds and the combination Hyper-exponential-Erlang.



(a) Fast **(b) Medium** **(c) Slow**

Figure 10: Approximation errors for the three different controller speeds and the combination Hyper-exponential-Hyper-exponential.

[11] Hasanin Harkous, Khaled Sherkawi, Michael Jarschel, Rastin Pries, Mu He, and Wolfgang Kellerer. 2021. P4RCProbe for evaluating the performance of P4Runtime-based controllers. In *Proc. of IEEE NFV-SDN*.

[12] Michael Jarschel, Simon Oechsner, Daniel Schlosser, Rastin Pries, Sebastian Goll, and Phuoc Tran-Gia. 2011. Modeling and performance evaluation of an openflow architecture. In *2011 23rd International Teletraffic Congress (ITC)*. IEEE, 1–7.

[13] Hisashi Kobayashi. 1973. Application of the diffusion approximation to queuing networks: part i equilibrium queue distributions. In *Proceedings of the 1973 ACM SIGME Symposium* (SIGME '73). Association for Computing Machinery, New York, NY, USA, 54–62. ISBN: 9781450379427. DOI: 10.1145/800268.809336.

[14] Kashif Mahmood, Ameen Chilwan, Olav Østerbø, and Michael Jarschel. 2015. Modelling of OpenFlow-based software-defined networks: the multiple node case. *IET Networks*, 4, 5, 278–284.

[15] Kashif Mahmood, Ameen Chilwan, Olav N Sterb, and Michael Jarschel. 2014. On the modeling of OpenFlow-based SDNs: The single node case. *Computer Science & Information Technology*.

[16] Nick McKeown, Tom Anderson, Hari Balakrishnan, Guru Parulkar, Larry Peterson, Jennifer Rexford, Scott Shenker, and Jonathan Turner. 2008. OpenFlow: Enabling innovation in campus networks. *ACM SIGCOMM Computer Communication Review*, 38, 2.

[17] U. Mitzlaff. 1997. *Diffusionsapproximationen von Warteschlangensystemen*. Ph.D. Dissertation. Technical University Clausthal.

[18] Netronome. 2022. Netronome agilio smartnics. Retrieved June 11, 2022 from https://www.netronome.com/products/smartnic/overview/.

[19] M. Reiser and H. Kobayashi. 1974. Accuracy of the diffusion approximation for some queuing systems. *IBM Journal of Research and Development*, 18, 2, 110–124. DOI: 10.1147/rd.182.0110.

[20] Dominik Scholz, Henning Stubbe, Sebastian Gallenmüller, and Georg Carle. 2020. Key Properties of Programmable Data Plane Targets. In *Teletraffic Congress (ITC 32), 2020 32nd International*. Osaka, Japan.

[21] Péter Vörös, Dániel Horpácsi, Róbert Kitlei, Dániel Leskó, Máté Tejfel, and Sándor Laki. 2018. T4P4S: A target-independent compiler for protocol-independent packet processors. In *Proc. of IEEE HPSR*.

[22] W. Whitt. 1983. The queueing network analyzer. *The Bell System Technical Journal*, 62, 9.

[23] Noa Zilberman, Yury Audzevich, G. Adam Covington, and Andrew W. Moore. 2014. Netfpga sume: toward 100 Gbps as research commodity. *IEEE Micro*, 34, 5.