applied sciences

Article

Time-Dependent Performance of a Multi-Hop Software Defined Network

Tadeusz Czachórski [1,*], Erol Gelenbe [1], Godlove Suila Kuaban [1] and Dariusz Marek [2]

[1] Institute of Theoretical and Applied Informatics, Polish Academy of Sciences, Bałtycka 5, 44-100 Gliwice, Poland; seg@iitis.pl (E.G.); gskuaban@iitis.pl (G.S.K.)
[2] Department of Distributed Systems and Informatic Devices, Faculty of Automatic Control, Electronics and Computer Science, Silesian University of Technology, Akademicka 16, 44-100 Gliwice, Poland; Dariusz.Marek@polsl.pl
* Correspondence: tadek@iitis.pl

Abstract: It has been recently observed that Software Defined Networks (SDN) can change the paths of different connections in the network at a relatively frequent pace to improve the overall network performance, including delay and packet loss, or to respond to other needs such as security. These changes mean that a network that SDN controls will seldom operate in steady state; rather, the network may often be in transient mode, especially when the network is heavily loaded and path changes are critically important. Hence, we propose a transient analysis of such networks to better understand how frequent changes in paths and the switches' workloads may affect multi-hop networks' performance. Since conventional queueing models are difficult to solve for transient behaviour and simulations take excessive computation time due to the need for statistical accuracy, we use a diffusion approximation to study a multi-hop network controlled by SDN. The results show that network optimization should consider the transient effects of SDN and that transients need to be included in the design of algorithms for SDN controllers that optimize network performance.

Keywords: SDN switch; internet traffic; quality of service (QoS); diffusion approximation

check for updates

Citation: Czachórski, T.; Gelenbe, E.; Kuaban, G.S.; Marek, D. Time-Dependent Performance of a Multi-Hop Software Defined Network. *Appl. Sci.* **2021**, *11*, 2469. https://doi.org/10.3390/app11062469

Academic Editor: Norbert Herencsar

Received: 4 December 2020
Accepted: 28 February 2021
Published: 10 March 2021

1. Introduction

Software defined networking (SDN) is a dynamic, adaptable, and manageable paradigm that facilitates innovations in computer networks [1], together with the prototyping and deployment of flexible routing mechanisms [2,3]. Traditional networking is based on manual configurations of distributed proprietary network devices, a cumbersome and error-prone process that can underutilize network resources [4]. SDN offers a programmable architecture where routing decisions are moved to centralized controllers. SDN data plane switches are simple forwarding devices that forward the data traffic depending on the controller's flow forwarding rules. Routing algorithms are implemented and communicated by each SDN controller to the SDN switches, which follow its instructions. Metrics such as hub count, delay, packet loss, bandwidth, jitter, and power consumption can be measured by SDN switches and sent to the controllers, which may use these metrics to determine the best routing paths and then install the flow forwarding rules in the data plane SDN switches. Indeed, the IoT [5] interacting Cloud Services [6] for the decision and control of the cyber-physical world create challenges for networks that achieve a better quality of service (QoS) and security and less energy consumption, and can exploit the opportunities offered by machine learning [7–9]. These challenges can be met by SDN networks [9,10], which offer greater flexibility and ease of implementation [2,11].

These developments suggest that SDN is likely to become the preferred networking approach not only in core networks because of the centralized network intelligence and management that it enables but also for sub-networks of IoT devices and edge devices with specific QoS needs that can benefit from SDN programmability and flexibility. Thus, in [12], conventional routing protocols such as RIP, OSPF, EIGRP, and BGP, are compared with SDN with respect to convergence times after link failures, showing that SDN routing is better than conventional IP networks. Considerable work has also shown that SDN can select routing paths based on criteria such as quality of service (QoS) [13–16], while energy-aware SDN routing has also been discussed in several papers [17–19].

The scalability of SDN routers that conduct QoS routing has been studied in [20], where the authors propose an SDN-based scalable QoS routing scheme between autonomous systems. In [21], a survey of the scalability issues that arise when SDN's centralized scheme deals with relatively frequent path updates is conducted. Both hierarchical and concurrent (distributed) approaches are investigated to alleviate SDN controllers' additional workload. In recent work, [22] SDN is discussed as a means to choose the best paths based on a function of time-varying traffic in order to optimize the QoS metrics of interest. Other work [23] examines a broad class of QoS-based algorithms to assign paths to flows in SDN and analyzes the resulting performance. In addition, the work in [24] discusses the implementation of SDN based real-time QoS in industrial settings with mobile robots or palets, where motion and reliability requirements impose changes in paths to constantly meet real-time requirements. In [25], the use of AI-driven dynamic QoS routing in SDN is used to optimize QoS, reduce energy consumption, and improve security based on Autonomic Communications [7] and the Cognitive Packet Network algorithm [26].

However, in addition to scalability issues, QoS-driven SDN routing can create traffic and time-dependent changes in network topology and in the load and paths that are serviced by SDN switches. SDN network performance has been analyzed using queueing theory [27–30] and network calculus [31–33], but these performance evaluations are based on the assumption that the network is in steady state—i.e., after a sufficiently long time—so that network metrics such as queueing delays, the length of packet queue buffers of SDN switches, and packet losses become stable (or time-independent). On the other hand, it is important to understand the time-dependent behaviour of SDN switches affected by changes in paths notified by the SDN controller. The controller can suddenly change the flows that an SDN switch receives, changing its input traffic. Furthermore, for a given switch some flows may be moved from one output port to another to comply with the new path that they must follow. These sudden changes will have performance consequences, including queueing delays and packet losses, which can only be understood via time-dependent transient analysis.

Unfortunately, conventional queueing network models are difficult to use in the transient regime because of the computational burden associated with their analysis; indeed, analyzing transients even in a simple single-server system with Poisson arrivals and exponential service times leads to the use of Bessel function expansions, and interconnected systems are quite hard to analyze in the transient case [34–36]. The analytical solution is known only in the case of single queues with Poisson input stream and exponentially distributed service times; see [37] for infinite and [38,39] for finite queues. The models use Markov chains and solve Chapman–Kolmogorov equations (first-order linear differential equations), defining the state probabilities of n customers present in the system at time t. The equations are solved analytically in the Laplace domain, and then the original functions in the time domain are found. Even in these relatively simple models, the solutions are quite complex—e.g., in the case of the infinite queue, the state probabilities are given in the form of the infinite series of modified Bessel functions of the first type and various order; the Bessel functions are themselves the infinite series. Some simplifications were proposed—e.g., the generating function of the distribution in the Laplace domain may be replaced by expressions with simpler original functions in the time domain [40], or Bessel functions may be replaced by easier-to-compute functions [41].

These analytical results do not fit well with the problem of modelling computer networks, where the streams incoming to switches are not Poisson and the sizes of packets—and therefore also the service times—are not exponentially distributed. We may introduce to Markov models interarrival and service times distributions composed of exponentially distributed phases—e.g., Cox distributions or hyper-Erlang distributions; the state definition is extended to include the current phase. There are numerous tools—e.g., [42]—that can match a phase-type distribution to any empirical histogram. However, the initial number of states should be multiplied—in this case, by the number of phases. This substantially increases the number of equations to be solved numerically. The solution is usually obtained using an existing tool—e.g., [43,44]. We have applied this approach in modelling the transient states of an IP router, ref. [45]; to represent the service time distribution, we needed a hyper-Erlang distribution with three parallel Erlang distributions with 21, 1387, and 2 phases. This could be conducted in the case of a single queue, as we are able to solve systems of millions of equations numerically, but it is hard to use this approach in modelling a network of switches. The typical method is to use either fluid flow approximation [46] or diffusion approximation. The fluid flow approximation is much simpler, as it considers only the time-dependent mean values of flows, queues, and delays. However, its errors are much larger than those of diffusion approximation; see a comparison in [47]. On the other hand, discrete event simulations of transients require many hundreds of independent repetitions of simulation runs to achieve a sufficient statistical accuracy, making the computation times of such simulations prohibitive [48].

Thus, in this paper we extend the approach we developed in [49] to the time-dependent analysis of multiple SDN switches using diffusion approximations [50,51], which are very convenient to analyze in a time-dependent regime. The accuracy of diffusion approximations has been validated in industry-based research over many decades [52–55], including for patented techniques [26], and also validated in many academic papers [56–58]. Their advantage includes a more accurate representation of interarrival and service processes, the ease of obtaining delay predictions from traffic measurements, and much faster numerics for transients than discrete queueing models [59] or simulations. Thus, we compute the transient behaviour of each SDN switch after changes occur in its input traffic rate. Packet loss probabilities can also be computed even when they are "tiny" and impossible to estimate by conventional means. The analysis we undertake considers both single SDN switch and multiple interconnected SDN switches controlled by an SDN controller.

The rest of the article is organized as follows. Section 2 presents the queueing model of an SDN switch, justifies its simplifications, and introduces the diffusion approach's principles. In Section 2.1, we give a detailed steady-state diffusion model of a single queueing station representing the SDN switch, and in Section 2.2 we extend this to the transient-state model. Section 3 presents a numerical analysis of the model from Section 2.2. We choose various parameters of the traffic and determine a scenario for the traffic changes. The obtained results include queue distributions, mean queues, and loss probabilities at the switch as a function of time. Section 4 presents the network model; the network may have any number of switches and any topology. Section 5 studies the transient behaviour of a chosen network of several nodes. It also gives a simple example of how the model may predict queue evolution and minimize the total backlog at nodes during a time interval. Conclusions are given in Section 6.

2. Diffusion Model of an SDN Switch

Figure 1 describes the basic system architecture of an SDN switch proposed in [1]. Arriving packets are temporarily queued at the input buffers and are then removed by the Arbiter and placed scheduled into the Packet Buffer. A copy of the packet header is forwarded to the Parser. The Parser parses the packet header to extract the header fields and then creates a tuple with the extracted information and forwards it to the Flow Match Unit. In the Flow Match Unit, the tuple is matched against existing flow rules stored in Flow Tables' flow entries. The flow entries in the Flow Tables are maintained under the controller's guidance and are updated when the controller installs new flow rules. The Flow Match Unit determines whether the packet is associated with a known flow and hence a known path. In case of success, the packet is then forwarded via the backplane. In case of failure (no flow table entry matches the packet header), a packet-in message will be sent by the SDN switch to the corresponding SDN controller [60] to notify the controller about the absence of a flow rule for the packet. The packet-in message contains either the packet or the packet's ID. The controller decides the correct action for the packet and then installs appropriate data in the flow table of the switch so that packets belonging to that particular flow can be forwarded subsequently. If there is no corresponding response from the controller, the packet will be dropped.

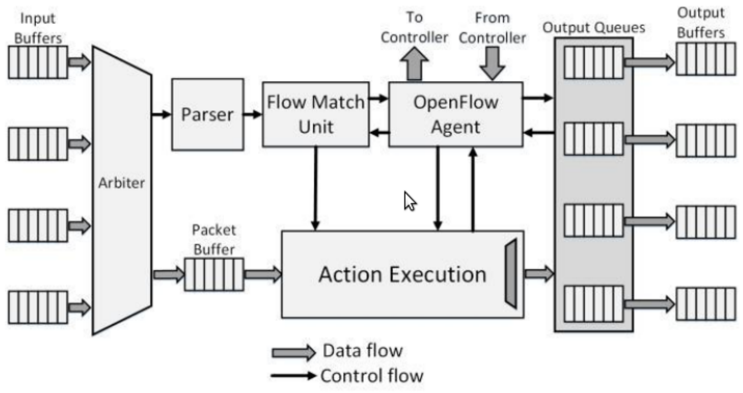

Figure 1. The architecture of an SDNswitch [1].

The delay that can be experienced by packets consists of the queueing delay in the input buffer, the service delay in the input buffer, and the delay at the output buffers. When the output ports' processing and line speeds are significantly greater than the Openflow processing time, which includes the time required to parse the packet, check the flow tables to find matching entries, and execute the flow rule actions, the switch can be represented by a single server queueing as in several recent papers [27–30,61–63]. Since the size of the input buffer is limited, we represent the SDN switch as a single server queueing model with finite capacity N. The packet interarrival times depend on the type of traffic, which the switch carries so that we allow for a generally distributed interarrival time distribution. The service time distribution will also be general to represent the manner in which the packets are handled in the entry buffer.

The flow table matching process involves searching the flow tables to find the entries containing corresponding action sets—i.e., flow rules that match the header of the packets of the input flows that pass through the Parser. If there is more than one flow table, the flow match process starts from the first flow table, searches all of its entries, and jumps to the next flow table. The search process continues till an entry that matches the packet header is found. Otherwise, a "packet-in" message is generated. For hardware switches whose flow tables are implemented using Ternary Content Addressable Memory (TCAM) modules, the flow match process can be performed within one clock cycle, with parallel access to all the entries of the flow table, resulting in a constant access time. Because of the limited TCAM flow table size and the power-hungry nature of TCAM in hardware switches, software-based switches are an attractive alternative. However, they are limited by a slow processing speed when flow tables are searched sequentially. A majority of previous papers model the matching time as an exponentially distributed random variable [27–29,61–63]. The use of a diffusion process allows us to model the flow matching process with a service time distribution that includes the flow tables' sequential search.

Denote by p the probability that the flow rule of the arriving packet is not installed. The switch knows it after the examination of all K entries stored in the Flow Match Unit—i.e., time KT—where the examination of each entry requires time T. As a consequence, p is the probability that the service time is a constant KT, representing the case when all the flow entries in the table are examined without success, while with probability $(1-p)$ the packet's flow match is found in a time that is uniformly distributed in $[T, KT]$—i.e., on average in time $(K+1)T/2$ and variance $(K^2-1)T^2/12$. Sophisticated hardware means may also be designed to improve this performance but are not considered in this paper.

To analyze this system, we use a continuous state space and continuous time diffusion process $\{X(t), t \geq 0\}$ to replace the discrete state-space buffer queue, where the increments $dX(t) = X(t+dt) - X(t)$ are normally distributed, with mean βdt and variance αdt, which appear in the diffusion Equation (1).

Assuming an arrival rate λ and average service time μ^{-1}, the changes in a small time interval ΔT tend to a normal distribution with mean $(\lambda - \mu)\Delta T$ and variance $(\lambda^3\sigma_A^2 + \mu^3\sigma_B^2)\Delta T = (\lambda C_A^2 + \mu C_B^2)\Delta T$, where σ_A^2 and σ_B^2 are the variances of the interarrival and service times, and C_A^2 and C_B^2 are the corresponding squared coefficients of variation. Therefore, for the diffusion process we have $\beta = \lambda - \mu$ and $\alpha = \lambda C_A^2 + \mu C_B^2$ [64].

The buffer's size is limited to N packets, therefore the diffusion process resides in the interval $[0, N]$, and we use a diffusion process with returns from the barriers at $x = 0$ and $x = N$ to represent the jumps that occur when the buffer queue is empty and a packet arrives, and when the queue is full and a service occurs as in [65], leading to the equations:

$$\begin{aligned}
\frac{\partial f(x,t;x_0)}{\partial t} &= \frac{\alpha}{2}\frac{\partial^2 f(x,t;x_0)}{\partial x^2} - \beta\frac{\partial f(x,t;x_0)}{\partial x} + \lambda p_0(t)\delta(x-1) + \mu p_N(t)\delta(x-N+1), \\
\frac{dp_0(t)}{dt} &= \lim_{x\to 0}\left[\frac{\alpha}{2}\frac{\partial f(x,t;x_0)}{\partial x} - \beta f(x,t;x_0)\right] - \lambda p_0(t) \\
\frac{dp_N(t)}{dt} &= \lim_{x\to N}\left[\frac{\alpha}{2}\frac{\partial f(x,t;x_0)}{\partial x} - \beta f(x,t;x_0)\right] - \mu p_N(t),
\end{aligned} \tag{1}$$

where $f(x, t; x_0)$ is the probability density function (pdf) of the diffusion process; $p_0(t)$ and $p_N(t)$ are, respectively, the probabilities that the process is at the barrier at $x = 0$ or $x = N$ at time t, corresponding to probabilities that the system is empty or saturated; and $\delta(x)$ is the Dirac delta function.

The first of the above equations defines the pdf of the diffusion process with jumps from $x = 0$ to $x = 1$ (arrival of the first customer after the idle period) with intensity λ and from $x = N$ to $x = N - 1$ (departure of a customer ending the saturation period) with intensity μ. The next two equations represent the probability balance of the barriers.

2.1. Steady-State Diffusion Model with General Interarrival and Service Processes

In steady state, when $\lim_{t\to\infty} p_0(t) = p_0$, $\lim_{t\to\infty} p_N(t) = p_N$, $\lim_{t\to\infty} f(x, t; x_0) = f(x)$, Equation (1) becomes an ordinary differential one and its solution, for $\varrho = \lambda/\mu$, $\rho < 1$, can be expressed as [64]:

$$f(x) = \begin{cases} \dfrac{\lambda p_0}{-\beta}(1 - e^{zx}) & \text{for} \quad 0 < x \leq 1\,, \\ \dfrac{\lambda p_0}{-\beta}(e^{-z} - 1)e^{zx} & \text{for} \quad 1 \leq x \leq N-1\,, \\ \dfrac{\mu p_N}{-\beta}(e^{z(x-N)} - 1) & \text{for} \quad N-1 \leq x < N\,, \end{cases} \tag{2}$$

where $z = \frac{2\beta}{\alpha}$, and due to normalization:

$$\begin{aligned}
p_0 &= \{1 + \varrho e^{z(N-1)} + \frac{\varrho}{1-\varrho}[1 - e^{z(N-1)}]\}^{-1}\,, \\
p_N &= \varrho p_0 e^{z(N-1)}\,.
\end{aligned} \tag{3}$$

In this way, $f(n)$, given by Equations (2) and (4), approximates the steady-state distribution $p(n)$ in the Packet Buffer queue. A few examples of the curve $f(x)$ depending on $\varrho = \lambda/\mu$—i.e., the utilization of the system—are presented in Figure 2. The next figure presents $p_N(\varrho)$, which is the loss probability due to the buffer overflow.

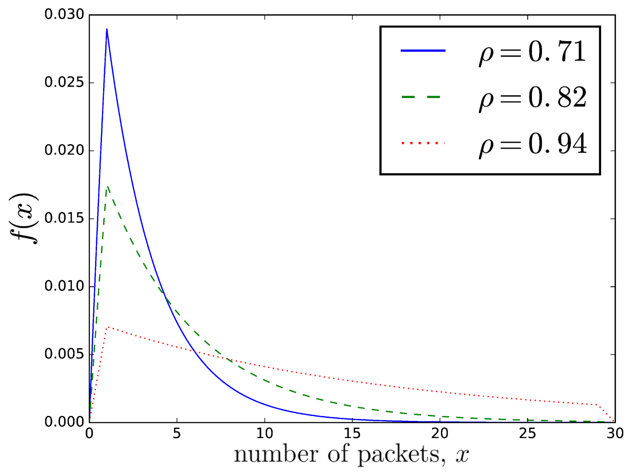

Figure 2. Steady-state pdf $f(x)$ of the diffusion process as an approximation of the queue distribution: illustration of the solution in Equation (2).

The steady-state queueing delay can be modelled by the time it takes the diffusion process to drift from the point $x = x_0$, corresponding to the queue length at the moment of the packet arrival, to $x = 0$ when the packet is already on the head of the queue (its distance to the transmitter is equal to zero), and is removed to be forwarded. The density of the diffusion process $f(x)$ given by Equation (2) determines the queue distribution and, at the same time, the density of the initial point x_0 at Equation (8).

The density $\phi(x,t;x_0)$ of the diffusion process starting at x_0 and ending at the absorbing barrier at the origin is given in [66]. The method of images, usually applied to heat conduction problems, is used. We may imagine the barrier as a mirror with an image source placed at $x = 2x_0$, and the solution is a superposition of a source of unit strength, placed at the origin and a source of strength $-\exp(\frac{2\mu x_0}{\alpha^2})$ placed at $x = 2x_0$:

$$\phi(x,t;x_0) = \frac{e^{\frac{\beta}{\alpha}(x-x_0)-\frac{\beta^2}{2\alpha}t}}{\sqrt{2\beta\alpha t}}\left[e^{-\frac{(x-x_0)^2}{2\alpha t}} - e^{-\frac{(x+x_0)^2}{2\alpha t}}\right]. \tag{4}$$

The density function $\gamma_{x_0,0}(t)$ of the first passage time from $x = x_0$ to $x = 0$, i.e., probability density that the process enters the barrier at time t, is equal to the probability density that the process is leaving the diffusion interval ($x > 0$):

$$\gamma_{x_0,0}(t) \quad = \quad \frac{\partial}{\partial t}\int_{0+}^{\infty}\phi(s,t;x_0)dx = \lim_{x\to 0}\left[\frac{\alpha}{2}\frac{\partial}{\partial x}\phi(x,t;x_0) - \beta\phi(x,t;x_0)\right] = \frac{x_0}{\sqrt{2\Pi\alpha t^3}}e^{-\frac{(x_0+\beta t)^2}{2\alpha t}}. \tag{5}$$

This density should be normalized to include only the cases when the process ends at the barrier, which is certain for $\beta < 0$. Therefore:

$$\int_0^{\infty}\gamma_{x_0,0}(t)dt = e^{\frac{2x_0\beta}{\alpha}}. \tag{6}$$

The first passage time of the diffusion process from the point $x = x_0$ to the barrier at $x = 0$ becomes:

$$\gamma_{x_0,0}(t) = \frac{x_0}{\sqrt{2\Pi\alpha t^3}}e^{-\left[\frac{2x_0\beta}{\alpha}+\frac{(x_0-\beta)^2}{2\alpha t}\right]}. \tag{7}$$

Suppose that a newly arrived packet joins the queue when the switch already contains x packets. Assuming the first-in-first-out service, the packet will be forwarded out from the switch after all the packets that arrived earlier have been forwarded, so that if the queue length probability density function is $f(x)$, the probability density function of the packet's queueing delay is:

$$f_R(t) = \int_0^{\infty} \left[\frac{x}{\sqrt{2\Pi\alpha t^3}} e^{-\left[\frac{2x\beta}{\alpha} + \frac{(x-\beta)^2}{2\alpha t}\right]} \right] f(x)dx. \tag{8}$$

Figure 3 illustrates this result with a few curves of $f_R(t)$ for different values of the traffic intensity ρ, and with the parameters; $C_A^2 = C_B^2 = 1$, which have been used in all the examples of Figures 2–4.

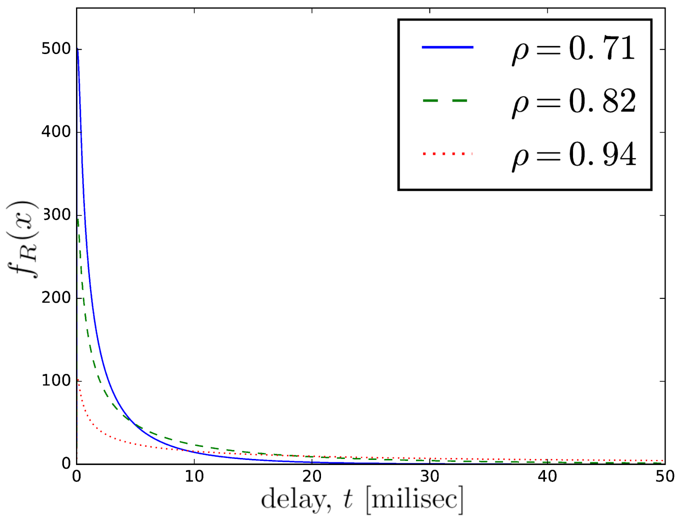

Figure 3. Density $f_R(t)$ of the queueing time; see Equation (8).

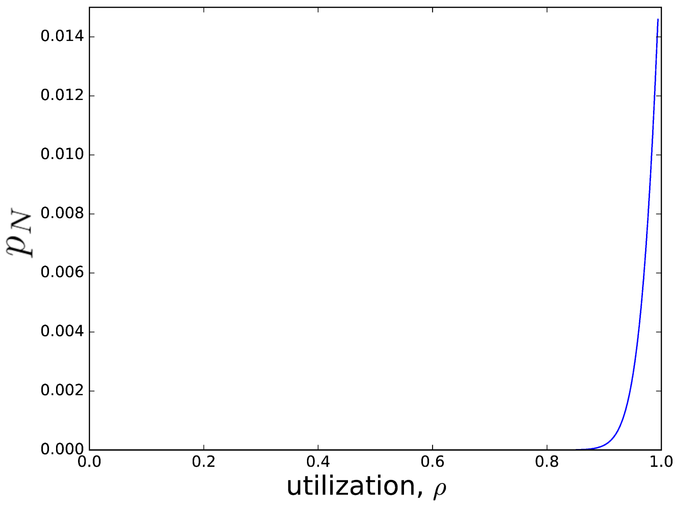

Figure 4. Probability p_N of the buffer overflow; illustration of the solution given by Equation (4).

The mean delay experienced by a packet whose flow rule is contained in the flow table will be the sum of the queueing delay and processing time. If the mean queueing delay is D_q and the processing time is t_p, then the mean packet delay at the switch D_s is:

$$D_s = D_q + t_p = \int_0^{\infty} t f_R(t)dt + \frac{1}{\mu}. \tag{9}$$

However, if a flow entry for an arriving packet is not found, then the packet is encapsulated and sent to the controller, which determines the flow rules, installs the flow entry for the packet, and sends back the packet in the packet-out message. In that case, the delay experienced by the packet, D is the sum of the delay in the switch and the delay at the controller D_c [27–29,61–63]:

$$D = D_s + D_c. \tag{10}$$

The modelling of the controller's working mechanism to determine D_c is beyond the scope of this work and will be considered in future works, and in our numerical examples all the flows are known, i.e., $p = 0$.

2.2. Transient Diffusion Model with General Interarrival and Service Processes

We express the time-dependent solution $f(x, t; x_0)$ of Equation (1) for the diffusion process having barriers with jumps with the use of the pdf $\phi(x, t; x_0)$ of another diffusion process which has absorbing barriers at $x = 0$ and $x = N$. There is no jumps, the process ends once it reaches any of the barriers. The density of such a process is known [66]:

$$\phi(x, t; x_0) = \begin{cases} \delta(x - x_0) & \text{for} \quad t = 0\,, \\ \dfrac{1}{\sqrt{2\Pi\alpha t}} \displaystyle\sum_{n=-\infty}^{\infty} \{a(t) + b(t)\} & \text{for} \quad t > 0\,, \end{cases} \tag{11}$$

where:

$$a(t) = \exp\left[\frac{\beta x'_n}{\alpha} - \frac{(x - x_0 - x'_n - \beta t)^2}{2\alpha t}\right],$$

$$b(t) = \exp\left[\frac{\beta x''_n}{\alpha} - \frac{(x - x_0 - x''_n - \beta t)^2}{2\alpha t}\right],$$

and $x'_n = 2nN$, $x''_n = -2x_0 - x'_n$.

If the initial condition is not given by a single point x_0 but by a density $\psi(\xi)$, $\xi \in (0, N)$, $\lim_{\xi\to 0} \psi(\xi) = \lim_{\xi\to N} \psi(\xi) = 0$, then the pdf of the process with absorbing barriers has the form:

$$\phi(x, t; \psi) = \int_0^N \phi(x, t; \xi)\psi(\xi)d\xi. \tag{12}$$

The Laplace transform of $\phi(x, t; x_0)$ is

$$\bar{\phi}(x, s; x_0) = \frac{\exp[\frac{\beta(x - x_0)}{\alpha}]}{A(s)} \sum_{n=-\infty}^{\infty} \left\{\exp\left[-\frac{|x - x_0 - x'_n|}{\alpha} A(s)\right] - \exp\left[-\frac{|x - x_0 - x''_n|}{\alpha} A(s)\right]\right\}, \tag{13}$$

with $A(s) = \sqrt{\beta^2 + 2\alpha s}$.

The process with jumps from barriers is a sequence of processes started by jumps at $x = 1$ and $x = N - 1$, ending at the barriers, and then after a time spent there starting again due to jumps, therefore the pdf $f(x, t; \psi)$ of the diffusion processs with jumps may be represented by functions $\phi(x, t; \psi)$ as follows: [51,67]:

$$f(x, t; \psi) = \phi(x, t; \psi) + \int_0^t g_1(\tau)\phi(x, t - \tau; 1)d\tau + \int_0^t g_{N-1}(\tau)\phi(x, t - \tau; N - 1)d\tau\,. \tag{14}$$

where $g_1(t)$ and $g_N(t)$ are the densities of jumps and are related to the pdfs $l_0(x)$, $l_N(x)$, the densities of sojourn times at $x = 0$ and $x = N$ respectively, and to $\gamma_0(t)$ and $\gamma_N(t)$, the probability densities that at time t the process enters $x = 0$ or $x = N$, in the following way:

$$\begin{aligned} g_1(\tau) &= \int_0^\tau \gamma_0(t) l_0(\tau - t)dt\,, \\ g_{N-1}(\tau) &= \int_0^\tau \gamma_N(t) l_N(\tau - t)dt\,, \end{aligned} \tag{15}$$

where:

$$\begin{aligned}\gamma_0(t) &= p_0(0)\delta(t) + [1 - p_0(0) - p_N(0)]\gamma_{\psi,0}(t) \\ &\quad + \int_0^t g_1(\tau)\gamma_{1,0}(t-\tau)d\tau + \int_0^t g_{N-1}(\tau)\gamma_{N-1,0}(t-\tau)d\tau\,, \\ \gamma_N(t) &= p_N(0)\delta(t) + [1 - p_0(0) - p_N(0)]\gamma_{\psi,N}(t) \\ &\quad + \int_0^t g_1(\tau)\gamma_{1,N}(t-\tau)d\tau + \int_0^t g_{N-1}(\tau)\gamma_{N-1,N}(t-\tau)d\tau\,, \end{aligned} \tag{16}$$

and $\gamma_{1,0}(t)$, $\gamma_{1,N}(t)$, $\gamma_{N-1,0}(t)$, $\gamma_{N-1,N}(t)$ are the densities of the first passage time between corresponding points—e.g.,:

$$\gamma_{1,0}(t) = \lim_{x\to 0}[\frac{\alpha}{2}\frac{\partial\phi(x,t;1)}{\partial x} - \beta\phi(x,t;1)]\,. \tag{17}$$

For absorbing barriers:

$$\lim_{x\to 0}\phi(x,t;x_0) = \lim_{x\to N}\phi(x,t;x_0) = 0\,,$$

Hence: $\gamma_{1,0}(t) = \lim_{x\to 0}\frac{\alpha}{2}\frac{\partial\phi(x,t;1)}{\partial x}$. The functions $\gamma_{\psi,0}(t)$, $\gamma_{\psi,N}(t)$ are the pdfs of first passage time when the initial condition is given by the density $\psi(\xi)$.

The delay introduced by the queue length distribution with density $f(\xi,t;\psi)$ is

$$: f_R(x,t) = \int_0^N \gamma_{\xi,0}(x)f(\xi,t;\psi)d\xi. \tag{18}$$

The convolutions of functions present in Equations (14)–(16) are more tractable when we consider the equations and solve them in the Laplace domain, receiving:

$$\bar{f}(x,s;\psi) = \bar{\phi}(x,s;\psi) + \bar{g}_1(s)\bar{\phi}(x,s;1) + \bar{g}_{N-1}(s)\bar{\phi}(x,s;N-1). \tag{19}$$

The process is in the barriers with probabilities

$$\bar{p}_0(s) = \frac{1}{s}[\bar{\gamma}_0(s) - \bar{g}_1(s)], \qquad \bar{p}_N(s) = \frac{1}{s}[\bar{\gamma}_N(s) - \bar{g}_{N-1}(s)]. \tag{20}$$

The original of $\bar{f}(x,s;\psi)$ is obtained via the numerical inversion of the solution (19); in the examples, below Stehfest's algorithm [68] was used.

Note that the above transient solution assumes constant diffusion parameters. In the case of time-dependent traffic, and therefore time-dependent diffusion parameters, we should apply it in short consecutive intervals of time where we may treat the parameters as constant.

3. Single SDN Switch Model

We use the above model to examine the behaviour of a single SDN switch. Assume the length of the packet buffer $N = 100$, $T = 8\cdot 10^{-7}$ s, $K = 950$, and $p = 0$—i.e., all connections have their entries in the Flow Match Unit; in consequence, the service time is uniformly distributed with the mean $1/\mu = 0.038$ msec and the squared coefficient of variation is $C_B^2 = 0.33$.

Figure 5 presents an example of the distribution of interarrival times between IP packets taken from CAIDA measurements; on this basis, we determined $C_A^2 = 1.02$ for the input flow. Additionally, we considered the greater variability of interarrival times expressed by $C_A^2 = 4.08$ and $C_A^2 = 8.16$ to investigate its influence on the performance of the switch.

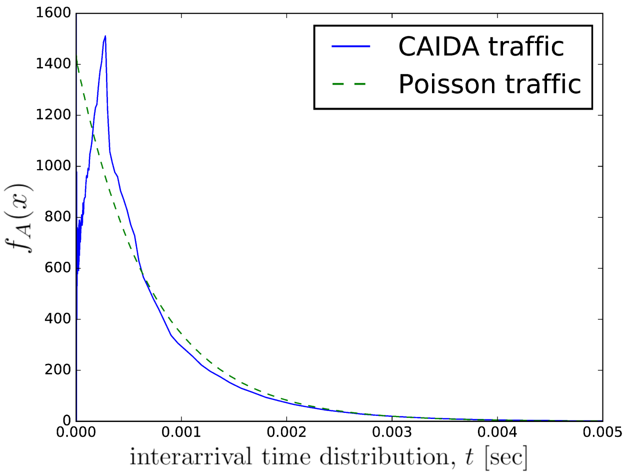

Figure 5. Packet interarrival times of IP traffic and CAIDAmeasurements versus Poisson distribution.

The changes in the input rate λ during the considered interval of 1 s are plotted in Figures 6 and 7. Figure 8 displays an example of the probability density function in Equation (14) for a chosen moment $t = 0.750$ s. when the empty queue is empty at the beginning of the interval under consideration, i.e., $x_0 = 0$ at $t = 0$; it shows the impact of the interarrival time variability on the density function. It is worth to note the logarithmic scale of the plot and the possibility to determine by the model very small numerical values of the function.

Figure 6 shows the changes in $p_N(t)$—i.e., probability of packet loss due to the buffer congestion—following the input traffic pattern, and also as a function of the variability of interarrival times, defined by the squared coefficient of variation C_A^2. Again, even very small numerical values are given by the model, and the computations are stable. Figure 7 presents the time-dependent mean queue. When traffic intensity λ is small, transient periods are short, and the changes in the mean queue follow the changes of λ closely. However, for greater values of λ, i.e., for higher utilization ϱ, the transient states are visibly longer than the time between the traffic changes, and the queue is permanently in a transient state. The increase in C_A^2 also increases the duration of transient states and the size of the queue.

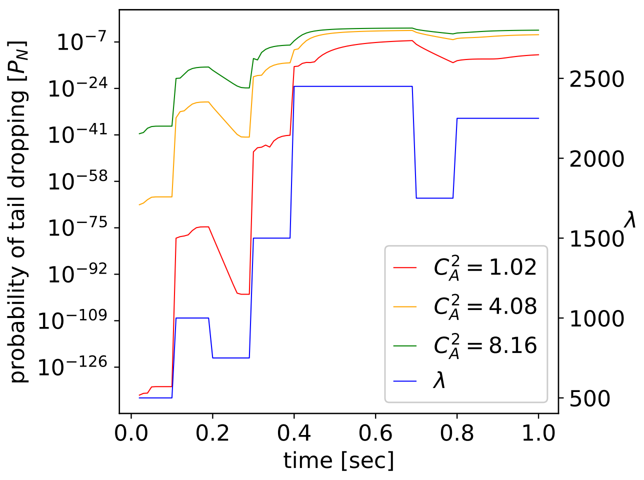

Figure 6. $p_N(t)$ for different C_A^2.

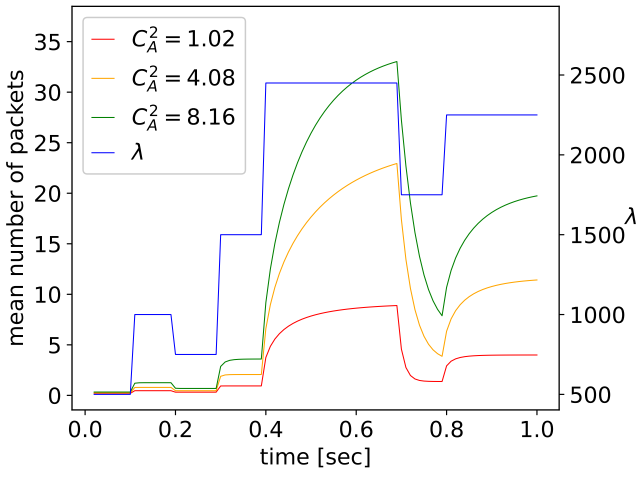

Figure 7. Mean queue for different C_A^2.

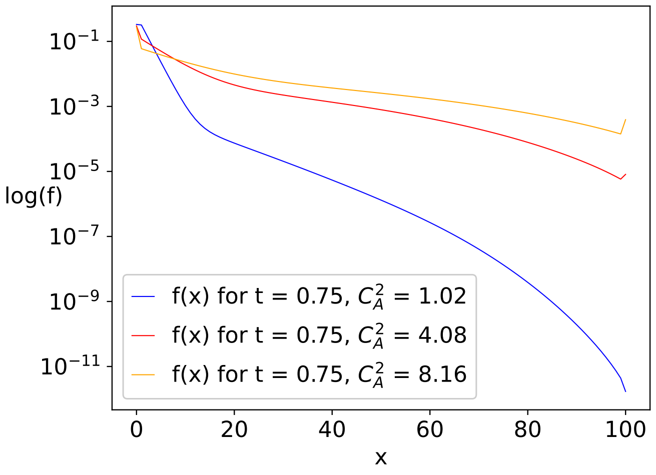

Figure 8. $f(x,t;0)$, $t = 0.750$ s.

4. Transient Analysis of Multiple SDN Switches

Consider a network of M stations with an arbitrary topology with routing probabilities $r_{ij}(t)$. We follow the approach of [69] developed for the steady-state network model then adapted to transient analysis in [70]. Additionally, we introduce time-dependent routing to model an SDN network.

The first step to solve the network model is to decompose the network—i.e., to determine the input traffic parameters λ_i, C_{Ai}^2 at every station i and then apply the single server model of the previous section to each station separately.

In the transient state, we should distinguish at any station i the input traffic $\lambda_{i-in}(t)$ and the output traffic intensities $\lambda_{i-out}(t)$:

$$\lambda_{i-out}(t) = [1 - p_{0i}(t)]\mu_i$$

which are different; $p_{0i}(t)$ denotes the probability that the station i is idle at time t, i.e., the diffusion process related to this station is inside the barrier at $x = 0$. The term $1 - p_{0i}(t) = \varrho_i(t)$ presents probability that the station i is busy and customers are leaving it with the rate μ_i.

The traffic equations balancing the flows of stations are:

$$\lambda_{i-in}(t) = \lambda_{0i}(t) + \sum_{j=1}^{M} \lambda_{j-out}(t) r_{ji}(t)\,, \qquad i = 1, \ldots, M, \tag{21}$$

where the first term λ_{0i} represents the traffic flows coming from the outside of the network directly to station i.

The routing probabilities $r_{ji}(t)$ change each interval Δ following the decisions of the controler, remaining constant inside the interval, and the flow parameters may change every interval $\delta < \Delta$; we assume $\Delta = n\delta$, in numerical examples below $n = 10$. This way all model parameters are constant witin intervals δ when the solution (14) is computed.

Denote by $f_{Aj}(x,t)$ and $f_{Bj}(x,t)$ the density functions of the interarrival and service times distributions at station j at time t. The pdf $f_{Dj}(x,t)$ of the interdeparture times from this node at time t may be expressed as:

$$f_{Dj}(x,t) = \varrho_j(t) f_{Bj}(x,t) + [1 - \varrho_j(t)] f_{Aj}(x,t) * f_{Bj}(x,t)\,, \qquad j = 1, \ldots, M, \tag{22}$$

where $*$ denotes the convolution with respect to x. The first term of the right side in (22) represents the interdepature times of packets when the node j is working, and the second term gives the interdeparture times when it is idle. The formula (22), known as Burke's theorem [71], is exact for Poisson input (the pdf of the idle period distribution that should be used in the second term of (22) is the same as $f_{Aj}(x,t)$) and approximate in other cases. From (22), we receive:

$$C^2_{Dj}(t) = \varrho_j^2(t) C^2_{Bj}(t) + C^2_{Aj}(t)(1 - \varrho_j(t)) + \varrho_j(t)[1 - \varrho_j(t)]\,. \tag{23}$$

where $C^2_{Dj}(t)$, $C^2_{Bj}(t)$, and $C^2_{Aj}(t)$ are time-dependent square coefficients of the variation in interdeparture, service, and interarrival times, respectively. Packets leaving the node j according to the distribution $f_{Dj}(x,t)$ choose any node i with probability $r_{ji}(t)$ and the times between two packets routed from node j to i has pdf $f_{ji}(x,t)$

$$\begin{aligned} f_{ji}(x,t) &= f_{Dj}(x,t) r_{ji}(t) + f_{Dj}(x,t) * f_{Dj}(x,t)[1 - r_{ji}(t)] r_{ji}(t) + \\ &\quad f_{Dj}(x,t) * f_{Dj}(x,t) * f_{Dj}(x,t)[1 - r_{ji}(t)]^2 r_{ji} + \cdots \end{aligned} \tag{24}$$

For example, a packet leaving station j goes to station i with probability $r_{ji}(t)$ or with probability $1 - r_{ji}(t)$ it goes elswhere but the second packet goes to i with probability $r_{ji}(t)$, hence the gap has has pdf $f_{Dj}(x,t) * f_{Dj}(x,t)$ with probability $[1 - r_{ji}(t)] r_{ji}(t)$, etc., or, after Laplace transform:

$$\begin{aligned} \bar{f}_{ji}(s,t) &= \bar{f}_{Dj}(s,t) r_{ji}(t) + \bar{f}_{Dj}(s,t)^2 [1 - r_{ji}(t)] r_{ji} + \bar{f}_{Dj}(s,t)^3 (1 - r_{ji}(t))^2 r_{ji} + \cdots \\ &= \frac{r_{ji}(t) \bar{f}_i(s,t)}{1 - [1 - r_{ji}(t)] \bar{f}_i(s,t)}\,, \end{aligned}$$

Then we compute the squared coefficient of variation:

$$C^2_{ji}(t) \quad = \quad r_{ji}(t)[C^2_{Dj}(t) - 1] + 1\,,$$

Hence:

$$C^2_{Ai}(t) = \frac{1}{\lambda_{i-in}(t)} \sum_{j=1}^{M} r_{ji}(t) \lambda_{i-out}(t) [(C^2_{Di}(t) - 1) r_{ji}(t) + 1] + \frac{C^2_{0i}(t) \lambda_{0i}(t)}{\lambda_{i-in}(t)}\,, \tag{25}$$

where the parameters λ_{0i} and C^2_{0i} refer to the flows coming to station i from outside of the network.

The parameters of the input flow at station i are given by (21) and (25). Equations (23) and (25) form a system of linear equations yielding $C^2_{Ai}(t)$ and also the diffusion parameters $\beta_i(t)$, $\alpha_i(t)$ for every node i. At each interval δ, the functions $f_i(x,t;\psi_i)$ providing the queue length distributions at every station i for $t \in \delta$ are computed. Their values at the end of the interval yield, among others, the current utilizations ϱ_i used to determine the flow parameters and diffusion parameters for the next interval δ.

The pdf $f_{Ri}(x,t)$ of the time-dependant response time (waiting time plus service) is determined using the first passage time from the end of the queue to zero, as defined by Equation (18). If $f_{Ri}(x,t)$ is the response time pdf at node i, then the response time pdf $f_R(x,t)$ for the path $1,\dots,n$ of n stations is:

$$f_R(x,t) = f_{R1}(x,t) * f_{R2}(x,t) * f_{R3}(x,t) * \cdots * f_{Rn}(x,t),$$

or:

$$\bar{f}_R(x,s) = \prod_{i=1}^{n} \bar{f}_{Ri}(x,s).$$

The loss probability $p_{loss}(t)$ for same entire path may be computed from:

$$1 - p_{loss}(t) = (1 - p_{N1}(t))(1 - p_{N2}(t))(1 - p_{N3}(t))\dots(1 - p_{Nn}(t)) \tag{26}$$

where $p_{Ni}(t)$ is the probability that the queue at station i is saturated at time t—i.e., the diffusion process for this station is at time t at the barrier $x = N$.

5. Network Model

Consider a network composed of four SDN switches, $S1$–$S4$; see Figure 9. Their parameters are the same as the switch in Example 1, except for the SDN switch $S4$, which is twice as fast. Therefore $\mu_1 = \mu_2 = \mu_3 = 2628.8$ packets/sec while $\mu_4 = 5257.6$ packets/s. At all the switches, the squared coefficient of variation of service time is identical to the value $C^2_{Bi} = 0.33$.

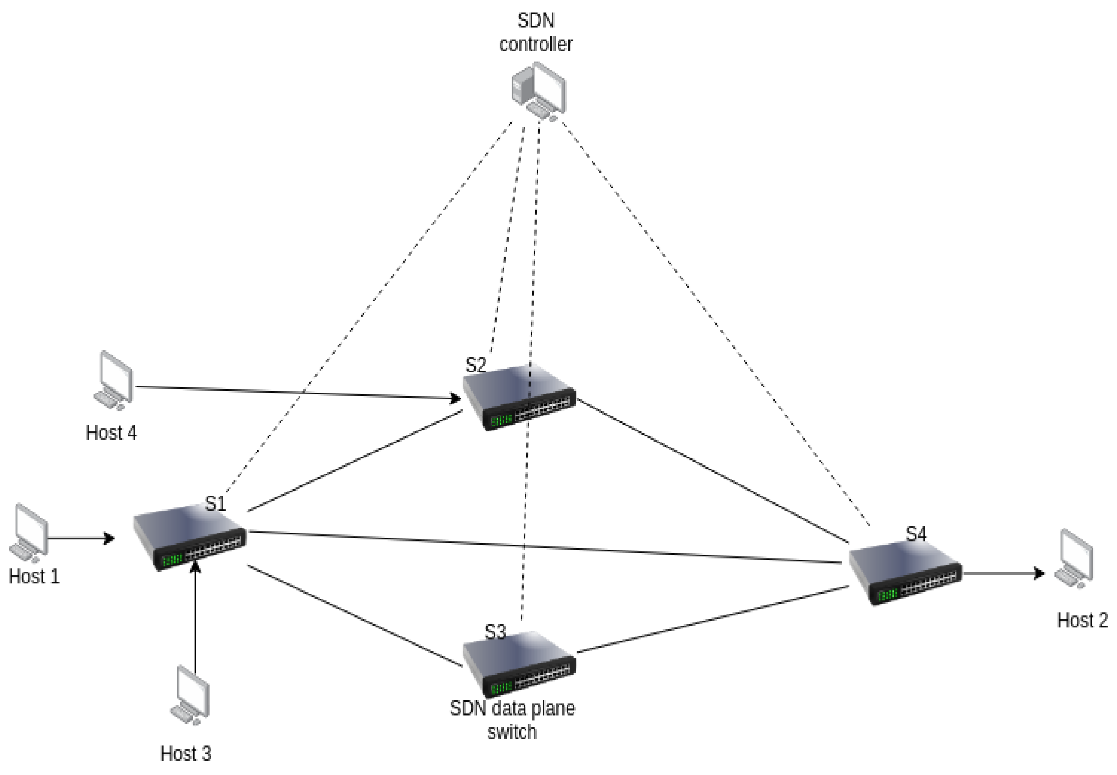

Figure 9. The example network being considered.

Similarly to Example 1, the network's performance is investigated during 1 s. Host 1 is sending packet flows of intensity λ_{01} to Host 2, and the traffic rate is changing in the range 500–2500 packets/sec, as shown in Figure 10. Host 4 generates traffic at rate λ_{02}, as shown in Figure 10, which is forwarded to Host 2 via the SDN switches $S2$ and $S4$.

As in Example 1, the squared coefficient of variation of interarrival times in the flows λ_{01} is $C_{A1}^2 = C_{01}^2 = 1.02$, or $C_{A1}^2 = 4.08$ or $C_{A1}^2 = 8.16$. The second input traffic λ_{02} at $S2$ has only one parameter $C_{A2}^2 = C_{02}^2 = 1.02$.

The SDN controller alters—if needed—the routing to balance the load of nodes every 100 msec; in this example, it refers to the routing probabilities r_{12} and r_{13}; see Figure 11.

Figures 12 and 13 illustrate the decomposition of the network model. They present the flows $\lambda_i(t)$ given by Equation (21) and the squared coefficients of variation $C_{Ai}^2(t)$ received from Equations (23) and (25). The service times at the stations have relatively small squared coefficients of variation $C_{Bi}^2 = 0.33$. Therefore, the important variability of the first flow entering the network is reduced at the network interior, as defined by Equation (23); see Figure 13.

The transient solution of diffusion equations is computed in intervals of the length $\delta = 10$ msec—i.e., we have 100 intervals with fixed diffusion parameters; at the end of each δ, the Equations (21) and (25) are solved to determine the new parameters of flow for the single-station models in the next interval. The diffusion density function obtained for any station i at the end of an interval gives the initial conditions for the diffusion equation at the next one.

The curves in Figure 14 compare the loss probability (note here the minimal values computed by the model), and, in Figure 15, the mean queues for all four stations, in case of $C_{A1}^2 = 1.04$. We may observe the changes in mean queues in $S2$ and $S3$ due to load balancing after the second flow becomes active. Observing the mean queues at $S1$ and $S2$, we can see that the transient periods may be longer than the time between the controller's decisions. As noted earlier, the length of the transient time increases with a load of a station and the variability of the input flow. For greater variabilities of the first flow, the path $S1 - S3 - S4$ becomes saturated; see Figure 16. This happens due to saturation in $S4$, as shown in Figure 17.

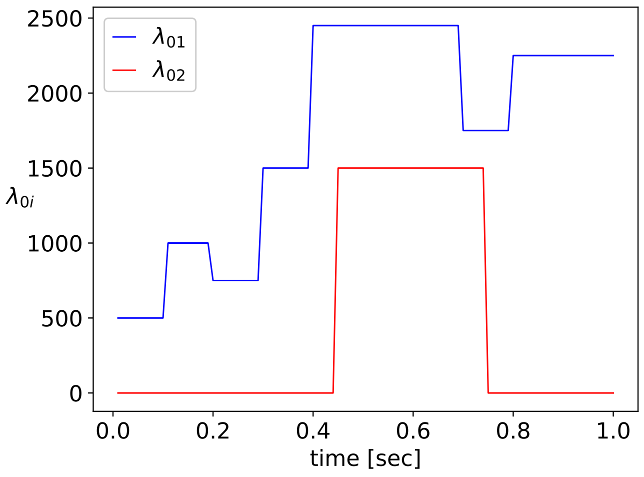

Figure 10. Input flows $\lambda_{01}(t)$, $\lambda_{02}(t)$, time in seconds.

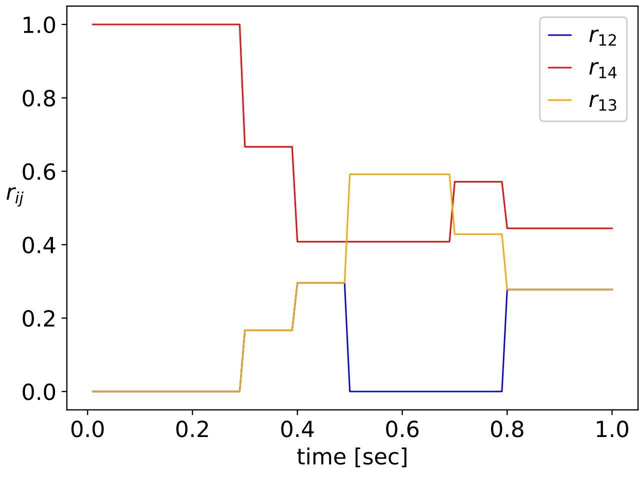

Figure 11. Routing probabilities $r_{12}(t)$, $r_{13}(t)$, $r_{14}(t)$.

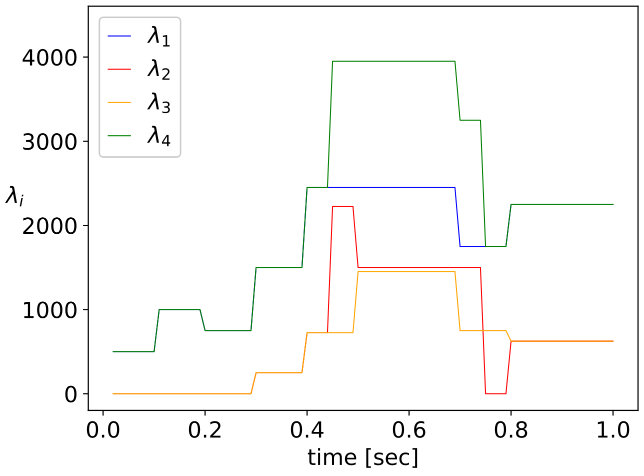

Figure 12. Input flows $\lambda_i(t)$ for stations $S1 \ldots S4$.

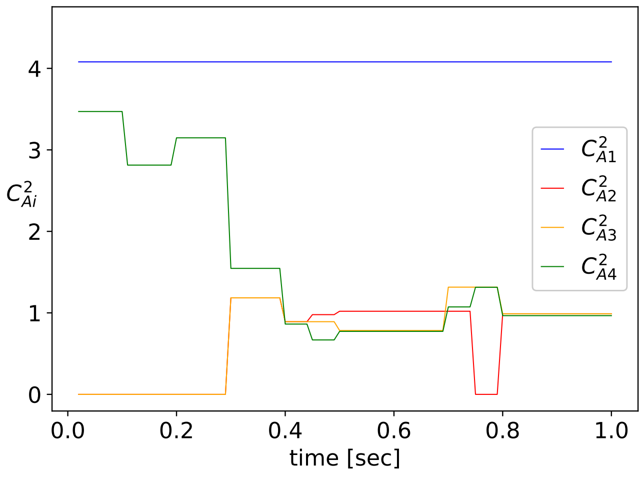

Figure 13. Squared coefficients of variation $C^2_{Ai}(t)$ for stations $S1 \ldots S4$.

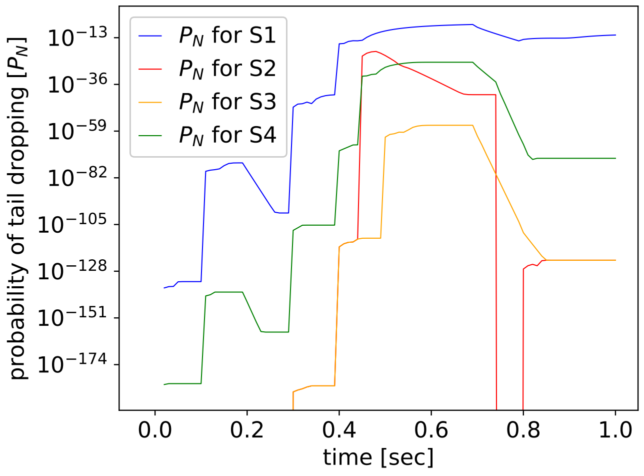

Figure 14. Stations $S1$, $S2$, $S3$, $S4$: $p_N(t)$, $C^2_{A1} = 1.02$.

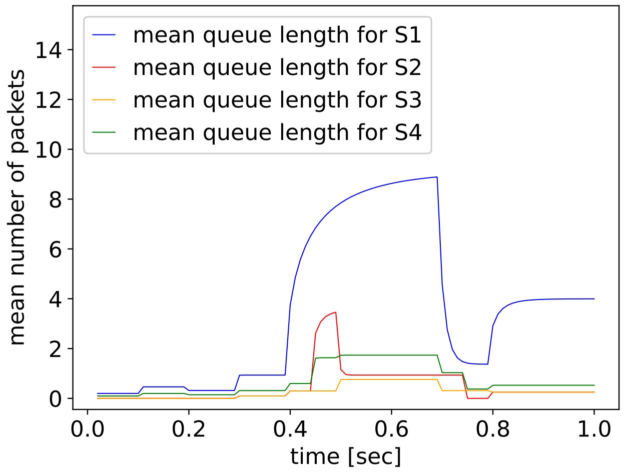

Figure 15. Stations $S1$, $S2$, $S3$, $S4$: mean queue, $C^2_{A1} = 1.02$.

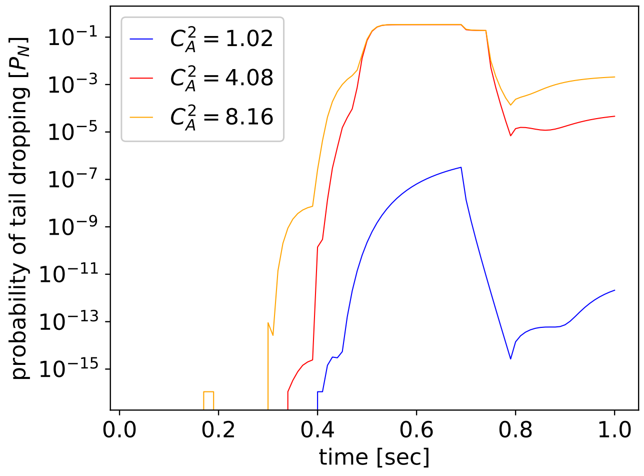

Figure 16. Total loss probability for path $S1 - S3 - S4$ for different C^2_{A1}.

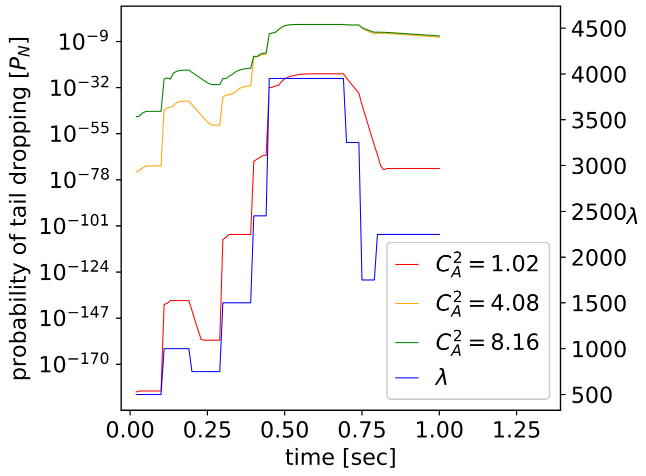

Figure 17. Station $S4$: $p_N(t)$ for different C^2_{A1}.

Let us also consider a **simple example of optimization**. Suppose, as previously, that station $S1$ is forwarding a flow λ_{01} packets to nodes $S2$ and $S3$. Station $S2$ is additionally receiving from Host 4 a flow of $\lambda_{02}^{(loc)}$ packets. The controller is changing routing every $\Delta = 100$ msec and needs to determine the routing probabilities for the nearest Δ, knowing the current parameters of flows at the beginning of the interval, as well as the current queue distributions at $S1$, $S2$, and $S3$ representing previous behaviour of the network. The goal is to minimize the mean backlog Ψ at $S2$ and $S3$ during Δ:

$$\min_{r_{12}, r_{13}} \left\{ \Psi = \frac{1}{\Delta} \int_0^{\Delta} [E[N_2(t)] + E[N_3(t)]] dt \right\}.$$

We compute $E[N_2(t)]$, $E[N_3(t)]$ for $t \in \Delta$ and minimize Ψ by the choice of r_{12}, $r_{13} = 1 - r_{12}$; see Figure 18.

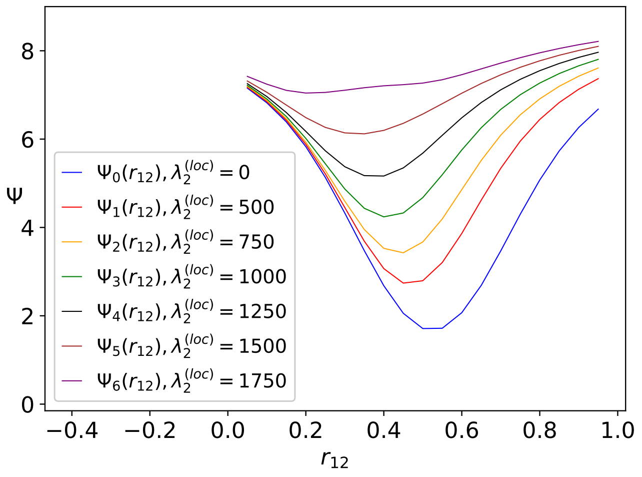

Figure 18. Mean backlog Ψ during Δ as a function of routing probabilities r_{12}, $r_{13} = 1 - r_{12}$, each curve corresponds to a different flow coming from Host 4 to $S2$.

6. Conclusions

The advent of SDN allows the implementation of smart adaptive routing, which changes network paths so that new connections may be established and inactive may be removed, as well as to deal with changes in traffic loads and incidents that affect network security. This leads to an interesting paradigm shift in network modelling, which has traditionally addressed "long term" behaviours and computational methods which are appropriate for steady-state analysis. However, when SDN intervenes dynamically to change paths and traffic levels, the network is seldom at a steady state, and optimization must take transients into account.

Therefore in this paper we have used diffusion approximation modelling for the performance evaluation of a network of SDN switches, that considers both steady-state and transient analysis. We have shown how changes in routing or forwarding decisions by the SDN controller can influence performance parameters such as delay, queue size, and packet loss probability in the transient state. Our results indicate that this method is computationally operational and can provide useful quantitative results for models with realistic parameter values.

Our analysis captures the interactions among the main parameters of the network, and numerical examples display the dependence of the queue lengths, queueing delays and their dynamics as a function of the changing flow intensity and variance of interarrival times. Our approach also confirms that transient periods play a significant role in the performance of SDN networks, and that they will be useful to analyze much larger networks in future work.

While the performance evaluations performed in this paper are purely numerical, and based on diffusion approximations models that have been widely validated by simulations [47,67,72], in future work we intend to use network emulation tools such as Mininet with real traffic, as well as experiments on a SDN test-bed, to study the influence of time dependent forwarding decisions on the main performance metrics of large SDN networks.

Author Contributions: E.G. developed the methodology and T.C. wrote the original draft of the article. D.M. and G.S.K. developed the software to solve the models, conducted the computations, and obtained the numerical results. E.G. then edited the paper. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the SerIoT Research and Innovation Action, funded by the European Commission (EC) under the H2020-IOT-2017 Program, G.A. 780139. The EC's support does not constitute an endorsement of this paper, which reflects the views only of the authors.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Wijeratne, S.; Ekanayake, A.; Jayaweera, S.; Ravishan, D.; Pasqual, A. Scalable High Performance router Architecture on FPGA for Core Networks. In Proceedings of the 2019 ACM/SIGDA International Symposium, Seaside, CA, USA, 24–26 February 2019. [CrossRef]
2. Francois, F.; Gelenbe, E. Towards a cognitive routing engine for software defined networks. In Proceedings of the 2016 IEEE International Conference on Communications, Kuala Lumpur, Malaysia, 22–27 May 2016; pp. 1–6. [CrossRef]
3. Du, J.; Gelenbe, E.; Jiang, C.; Zhang, H.; Ren, Y. Contract design for traffic offloading and resource allocation in heterogeneous ultra-dense networks. *IEEE J. Sel. Areas Commun.* **2017**, *35*, 2457–2467. [CrossRef]
4. Xia, W.; Wen, Y.; Foh, C.H.; Niyato, D.; Xie, H. A Survey on Software-Defined Networking. *IEEE Commun. Surv. Tutor.* **2015**, *17*, 27–51. [CrossRef]
5. Lee, S.K.; Bae, M.; Kim, H. Future of IoT Networks: A Survey. *Appl. Sci.* **2017**, *7*, 1072. [CrossRef]
6. Buyya, R.; Srirama, S.N.; Casale, G.; Calheiros, R.; Simmhan, Y.; Varghese, B.; Gelenbe, E.; Llorente, I.M.; Milojicic, D.; Jin, H.; et al. A manifesto for future generation cloud computing: Research directions for the next decade. *ACM Comput. Surv. (CSUR)* **2019**, *51*, 1–38. [CrossRef]
7. Dobson, S.; Denazis, S.; Fernandez, A.; Gaiti, D.; Gelenbe, E.; Massacci, F.; Saffre, F.; Nixon, P.; Zambonelli, F. A survey of autonomic communications. *ACM Trans. Auton. Adapt. Syst. (TAAS)* **2006**, *1*, 223–259. [CrossRef]
8. Gelenbe, E.; Liu, P.; Lainé, J. Genetic algorithms for route discovery. *IEEE Trans. Syst. Man Cybern. Part B (Cybern.)* **2006**, *36*, 1247–1254. [CrossRef]
9. Gelenbe, E.; Lent, R.; Nunez, A. Self-aware networks and QoS. *Proc. IEEE* **2004**, *92*, 1478–1489. [CrossRef]
10. McKeown, N.; Anderson, T.; Balakrishnan, H.; Parulkar, G.; Peterson, L.; Rexford, J.; Shenker, S.; Turner, J. OpenFlow: Enabling Innovation in Campus Networks. *ACM SIGCOMM Comput. Commun. Rev.* **2008**, *38*, 69–74. [CrossRef]
11. Goto, Y.; Ng, B.; Seah, W.K.; Takahashi, Y. Queueing analysis of software defined network with realistic OpenFlow-based switch model. *Comput. Netw.* **2019**, *164*, 106892. [CrossRef]

12. Gopi, D.; Cheng, S.; Huck, R. Comparative Analysis of SDN and Conventional Networks using Routing Protocols. In Proceedings of the 2017 International Conference on Computer, Information and Telecommunication Systems (CITS), Dalian, China, 21–23 July 2017; pp. 108–112. [CrossRef]
13. Lin, C.; Wang, K.; Deng, G. A QoS-aware routing in SDN hybrid networks. *Procedia Comput. Sci.* **2017**, *110*, 242–249. [CrossRef]
14. Jiawei, W.; Xiuquan, Q.; Guoshun, N. Dynamic and adaptive multi-path routing algorithm based on software-defined network. *Int. J. Distrib. Sens. Netw.* **2018**, *14*, 1–10. [CrossRef]
15. Dutra, D.L.C.; Bagaa, M.; Taleb, T.; Samdanis, K. Ensuring End-to-End QoS Based on Multi-Paths Routing Using SDN Technology. In Proceedings of the GLOBECOM 2017–2017 IEEE Global Communications Conference, Singapore, 4–8 December 2017; pp. 1–6. [CrossRef]
16. Lin, S.C.; Akyildiz, I.F.; Wang, P.; Luo, M. QoS-Aware Adaptive Routing in Multi-layer Hierarchical Software Defined Networks: A Reinforcement Learning Approach. In Proceedings of the 2016 IEEE International Conference on Services Computing (SCC), San Francisco, CA, USA, 27 June–2 July 2016; pp. 25–33. [CrossRef]
17. Maaloul, R.; Taktak, R.; Chaari, L.; Cousin, B. Energy-Aware Routing in Carrier-Grade Ethernet Using SDN Approach. *IEEE Trans. Green Commun. Netw.* **2018**, *2*, 844–858. [CrossRef]
18. Nam, T.M.; Thanh, N.H.; Thu, N.Q.; Hieu, H.T.; Covaci, S. Energy-aware routing based on power profile of devices in data center networks using SDN. In Proceedings of the 2015 12th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON), Hua Hin, Thailand, 27 June 2015; pp. 1–6. [CrossRef]
19. Al-Hubaishi, M.; Ceken, C.; Al-Shaikhli, A. A novel energy-aware routing mechanism for SDN-enabled WSAN. *IEEE Trans. Green Commun. Netw.* **2018**, *32*, e3724. [CrossRef]
20. Karakus, M.; Durresi, A. A Scalable Inter-AS QoS Routing Architecture in Software Defined Network (SDN). In Proceedings of the 2015 IEEE 29th International Conference on Advanced Information Networking and Applications, Gwangju, Korea, 24–27 March 2015; pp. 148–154. [CrossRef]
21. Karakus, M.; Durresi, A. A survey: Control plane scalability issues and approaches in Software-Defined Networking (SDN). *Comput. Netw.* **2017**, *112*, 279–293. [CrossRef]
22. Shakthipriya, P.; Bevi, A.R. Network Protocol-Based QoS Routing Using Software Defined Networking. In *Artificial Intelligence and Evolutionary Computations in Engineering Systems*; Dash, S.S., Vijayakumar, K., Panigrahi, B.K., Das, S., Eds.; Springer Singapore: Singapore, 2017; pp. 363–374.
23. Guck, J.W.; Bemten, A.V.; Reisslein, M.; Kellerer, W. Unicast QoS Routing Algorithms for SDN: A Comprehensive Survey and Performance Evaluation. *IEEE Commun. Surv. Tutor.* **2018**, *20*, 388–415. [CrossRef]
24. Jhaveri, R.H.; Tan, R.; Ramani, S.V. Real-time QoS-aware Routing Scheme in SDN-based Robotic Cyber-Physical Systems. In Proceedings of the IEEE 5th International Conference on Mechatronics System and Robots (ICMSR), Singapore, 3–5 May 2019. [CrossRef]
25. Gelenbe, E.; Domanska, J.; Frohlich, P.; Nowak, M.; Nowak, S. Self-Aware Networks that Optimize Security, QoS and Energy. *Proc. IEEE* **2020**, *108*, 1150–1167. [CrossRef]
26. Gelenbe, S.E. Cognitive Packet Network. U.S. Patent No. US6804201B1, 12 October 2004.
27. Mahmood, K.; Chilwan, A.; Østerbø, O.; Jarschel, M. Modelling of OpenFlow-based software-defined networks: the multiple node case. *IET Netw.* **2015**, *4*, 278–284. [CrossRef]
28. Ansell, J.; Seah, W.K.G.; Ng, B.; Marshall, S. Making Queueing Theory More Palatable to SDN/OpenFlow-based Network Practitioners. In Proceedings of the 2016 IEEE/IFIP Network Operations and Management Symposium, Istanbul, Turkey, 25–29 April 2016; pp. 1119–1124. [CrossRef]
29. Miao, W.; Min, G.; Wu, Y.; Wang, H.; Hu, J. Performance Modelling and Analysis of Software Defined Networking under Bursty Multimedia Traffic. *ACM Trans. Multimed. Comput. Commun. Appl.* **2018**, *12*, 24–36. [CrossRef]
30. Sood, K.; Yi, S.; Xiang, Y. Performance Analysis of Software-Defined Network router using M/Geo/1. *IEEE Commun. Lett.* **2016**, *20*, 27–51. [CrossRef]
31. Azodolmolky, S.; Wieder, P.; Yahyapour, R. Performance evaluation of a scalable software-defined networking deployment. In Proceedings of the 2013 Second European Workshop on Software Defined Networks, Berlin, Germany, 10–11 October 2013; pp. 68–74. [CrossRef]
32. Azodolmolky, S.; Nejabati, R.; Pazouki, M.; Wieder, P.; Yahyapour, R.; Simeonidou, D. An analytical model for software defined networking: A network calculus-based approach. In Proceedings of the IEEE Global Communications Conference, Atlanta, GA, USA, 9–13 December 2013; pp. 1397–1402. [CrossRef]
33. Bozakov, Z.; Rizk, A. Taming SDN controllers in heterogeneous hardware environments. In Proceedings of the 2013 Second European Workshop on Software Defined Networks, Berlin, Germany, 10–11 October 2013; pp. 50–55. [CrossRef]
34. Parthasarathy, P.; Selvaraju, N. Transient analysis of a queue where potential customers are discouraged by queue length. *Math. Probl. Eng.* **2001**, *7*, 433–454. [CrossRef]
35. Parthasarathy, P.; Sudhesh, R. Exact transient solution of a discrete time queue with state-dependent rates. *Am. J. Math. Manag. Sci.* **2006**, *26*, 253–276. [CrossRef]
36. Sudhesh, R. Transient analysis of a queue with system disasters and customer impatience. *Queueing Syst.* **2010**, *66*, 95–105. [CrossRef]

37. Champernowne, D.C. An elementary method of solution of the queueing problem with a single server and constant parameters. *J. R. Statist. Soc.* **2000**, *3*, 263–266. [CrossRef]
38. Takâcs, L. *Introduction to the Theory of Queues*; Oxford University Press: Oxford, UK, 1960.
39. Tarabia, A.M.K. Transient Analysis of M/M/1/N Queue-An Alternative Approach. *Tamkang J. Sci. Eng.* **2000**, *3*, 263–266.
40. Kotiah, T.C.T. Approximate transient analysis of some queueing systems. *Oper. Res.* **1998**, *26*, 334–346.
41. Jones, S.K.; Cavin, R.K.; Johnston, D.A. An Efficient Computational Procedure for the Evaluation of the $M/M/1$ Transient State Occupancy Probabilities. *IEEE Trans. Commun.* **1980**, *28*, 2019–2020. [CrossRef]
42. Reinecke, P.; Krauß, T.; Wolter, K. HyperStar: Phase-Type Fitting Made Easy. In Proceedings of the 9th International Conference on the Quantitative Evaluation of Systems (QEST 2012), London, UK, 17–20 September 2012; pp. 201–202.
43. PRISM—Probabilistic Model Checker. Available online: http://www.prismmodelchecker.org/ (accessed on 9 March 2020).
44. Kwiatkowska, M.; Norman, G.; Parker, D. PRISM 4.0: Verification of Probabilistic Real-time Systems. In Proceedings of the 23rd International Conference on Computer Aided Verification (CAV'11), Snowbird, UT, USA, 14–20 July 2011; pp. 585–591.
45. Czachórski, T.; Domańska, A.; Domańska, J.; Rataj, A. A Study of IP Router Queues with the Use of Markov Models. In Proceedings of the 2016 International Conference on Computer Networks (CN2016), Brunów, Poland, 14–17 June 2016; pp. 294–305. [CrossRef]
46. Misra, V.; Gongnad, W.B.; Towsley, D. A Fluid-based Analysis of a Network of AQM Routers Supporting TCP Flows with an Application to RED. In Proceedings of the Conference on Applications, Technologies, Architectures and Protocols for Computer Communication (SIGCOMM 2000), Stockholm, Sweden, 28 August–1 September 2000; pp. 152–160. [CrossRef]
47. Czachórski, T.; Nycz, M.; Nycz, T. Modelling transient states in queueing models of computer networks: A few practical Issues. In *Distributed Computer and Communication Networks*; Springer: Moscow, Russia, 2013; Volume 279, pp. 58–72. [CrossRef]
48. Czachórski, T.; Fourneau, J.M.; Jędruś, S.; Pekergin, F. Transient State Analysis in Cellular Networks: The Use of Diffusion Approximation. In Proceedings of the Fourth International Workshop on Queueing Networks with Finite Capacity QNETs 2000, West Yorkshire, UK, 20–21 July 2000.
49. Czachórski, T.; Gelenbe, E.; Suila, K.; Marek, D. Transient behaviour of a network router. In Proceedings of the 43th International Conference on Telecommunications and Signal Processing (TSP), Milan, Italy, 7–9 July 2020; pp. 246–252.
50. Kobayashi, H. Application of the diffusion approximation to queueing networks, Part 1: Equilibrium queue distributions. *J. ACM* **1974**, *21*, 316–328. [CrossRef]
51. Czachórski, T. A method to solve diffusion equation with instantaneous return processes acting as boundary conditions. *Bull. Pol. Acad. Sci. Tech. Sci.* **1993**, *41*, 417–451.
52. Reiser, M.; Kobayashi, H. Accuracy of the Diffusion Approximation for Some Queuing Systems. *IBM J. Res. Dev.* **1974**, *18*, 110–124. [CrossRef]
53. Lu, Y. *Diffusion Approximation for Random Walk Reflected Away from Boundary*; IBM Research Report; RC22866 (W0308-012); IBM: Yorktown Heights, NY, USA, 2003. Available online: https://dominoweb.draco.res.ibm.com/0467ca3209c3fe1b85256d88006c7667.html (accessed on 9 March 2020).
54. Marin, G.A.; Mang, X.; Gelenbe, E.; Onvural, R.O. Statistical Call Admission Control. International Business Machines Corporation. U.S. Patent No. 6222824, 24 April 2001.
55. Gelenbe, E.; Mang, X.; Onvural, R.O. Diffusion based statistical admission control in ATM. *Perform. Eval.* **1996**, *27*, 411–436. [CrossRef]
56. Woodside, C.; Pagurek, B.; Newell, G. A diffusion approximation for correlation in queues. *J. Appl. Probab.* **1980**, *17*, 1033–1047. [CrossRef]
57. Kimura, T. Diffusion Approximation for an M/G/m Queue. *Oper. Res.* **1983**, *31*, 304–321. [CrossRef]
58. Konakov, V.; Mammen, E. Accuracy of Diffusion Approximations for High Frequency Markov Data. *arXiv* **2006**, arXiv:math/0602430.
59. Bisnik, N.; Abouzeid, A.A. Queuing network models for delay analysis of multihop wireless ad hoc networks. *Ad Hoc Netw.* **2009**, *7*, 79–97. [CrossRef]
60. Xiong, B.; Yang, K.; Zhao, J.; Li, W.; Li, K. Performance evaluation of OpenFlow-based software-defined networks based on queueing model. *Comput. Netw.* **2016**, *102*, 172–185. [CrossRef]
61. Singh, D.; Ng, B.; Lai, Y.C.; Lin, Y.D.; Seah, W.K.G. Modelling Software-Defined Networking: Software and Hardware Switches. *J. Comput. Netw. Comput. Appl.* **2018**, *122*, 24–36. [CrossRef]
62. Lai, Y.C.; Ali, A.; Hassan, M.; Hossain, S.; Lin, Y.D. Performance Modeling and Analysis of TCP Connections over Software Defined Networks. In Proceedings of the 2017 IEEE Global Communications Conference, Singapore, 4–8 December 2017; pp. 1–6. [CrossRef]
63. Fahmin, A.; Lai, Y.C.; Hossain, S.; Lin, Y.D.; Saha, D. Performance Modeling of SDN with NFV under or aside the Controller. In Proceedings of the 5th International Conference on Future Internet of Things and Cloud Workshops, Prague, Czech Republic, 21–23 August 2017; pp. 211–216. [CrossRef]
64. Gelenbe, E. On approximate computer system models. *J. ACM* **1975**, *22*, 261–269. [CrossRef]
65. Gelenbe, E. Probabilistic models of computer systems Part II: Diffusion approximations, waiting times and batch arrivals. *Acta Inform.* **1979**, *12*, 285–303. [CrossRef]
66. Cox, R.P.; Miller, H.D. *The Theory of Stochastic Processes*; Chapman and Hall: London, UK, 1965.

67. Czachórski, T. Queuing models for performance evaluation of computer networks: Transient state analysis. In *Analytic Methods in Interdisciplinary Applications*; Mityushev, V., Ed.; Springer: Berlin/Heidelberg, Germany, 2014; Volume 116, pp. 51–80. [CrossRef]
68. Stehfest, H. Numeric inversion of Laplace transform. *Commun. ACM* **1970**, *13*, 47–49. [CrossRef]
69. Gelenbe, E.; Pujolle, G. The behaviour of a single queue in a general queueing network. *Acta Inform.* **1976**, *7*, 123–136. [CrossRef]
70. Duda, A. Diffusion approximations for time-dependent queueing systems. *IEEE J. Sel. Areas Commun.* **1986**, *4*, 905–918. [CrossRef]
71. Burke, P.J. The Output of a Queuing System. *Oper. Res.* **1956**, *4*, 699–704. [CrossRef]
72. Czachórski, T.; Pekergin, F. Diffusion approximation as a modelling tool. In *Network Performance Engineering*; Kouvatsos, D.D., Ed.; Springer: Berlin/Heidelberg, Germany, 2011; Volume 5233, pp. 447–476. [CrossRef]