Received: 29 May 2023 | Revised: 20 September 2023 | Accepted: 18 November 2023

DOI: 10.1002/dac.5687

RESEARCH ARTICLE

WILEY

P4toNFV: Offloading from P4 switches to NFV in programmable data planes

Adrian Pekar[1,2] | **Laszlo A. Makara**[1] | **Yuan-Cheng Lai**[3] | **Ying-Dar Lin**[4] | **Winston Seah**[5]

[1]Department of Networked Systems and Services, Faculty of Electrical Engineering and Informatics, Budapest University of Technology and Economics, Budapest, Hungary

[2]HUN-REN-BME Information Systems Research Group, Budapest University of Technology and Economics, Budapest, Hungary

[3]Department of Information Management, National Taiwan University of Science and Technology, Taipei, Taiwan

[4]Department of Computer Science, National Yang Ming Chiao Tung University, Hsinchu, Taiwan

[5]School of Engineering and Computer Science, Victoria University of Wellington, Wellington, New Zealand

Correspondence
Adrian Pekar, Department of Networked Systems and Services, Faculty of Electrical Engineering and Informatics, Budapest University of Technology and Economics, Műegyetem rkp. 3., H-1111 Budapest, Hungary.
Email: apekar@hit.bme.hu

Funding information
János Bolyai Research Scholarship of the Hungarian Academy of Sciences; National Research, Development and Innovation Fund of the Ministry of Culture and Innovation of Hungary, Grant/Award Numbers: ÚNKP-23-5-BME-461, TKP2021-NVA-02

Summary

P4 combines the benefits of hardware-based networking with the adaptability of software-based network operations. However, when faced with intricate network functions, P4 switches reveal constraints in memory and processing primitives. To address these, we advocate offloading traffic demanding intricate processing from the programmable data plane to network function virtualization (NFV). By leveraging this approach, P4 switches handle the core data plane, ensuring maximum performance, whereas virtualized network functions (VNF) cater to the intricate processing. Central to our research is the optimization of this offloading process, specifically considering delay constraints. We developed an analytical model that examines a P4 switch overseen by an SDN controller, integrating an offloading capability to NFV. The principal objective was to determine an offloading rate that minimizes packet processing delay. To this end, we employed a Bounded method, an advancement from Brent's method, to determine this optimal rate. The findings indicate that offloading approximately 66% of packets to the VNF achieves the lowest total delay, registering at 0.1505 μs. This strategy of optimal offloading can notably reduce the system's average delay as the demand for network functions increases. The optimization technique we adopted exhibited rapid convergence in our experiments, reflecting the method's efficacy. Furthermore, a rigorous parametric sensitivity analysis spanning no offloading, full offloading, and optimal offloading strategies underscores that optimal offloading to NFV consistently augments system performance, particularly in terms of delay reduction. Conclusively, our study furnishes valuable insights into offloading strategies, augmenting the narrative on resource allocation in both PNFs and VNFs.

KEYWORDS
network function virtualization, P4 switch, software-defined networking

1 | INTRODUCTION

Traditional networks have faced significant scalability, operational, and management challenges.[1] Even though these limitations have been mitigated by softwarization[2] and virtualization,[3] which enabled network operators to implement, manage, and scale their networks more efficiently, the fixed behavior of network devices that was directly burned into the chips by vendors remained a major constraint. The main drawback of fixed-function switching is the lack of ability to add new network protocols/functions and change the forwarding logic.

The introduction of programmable switch application-specific integrated circuits (ASICs) has eventually brought significant flexibility, ushering in a new era of innovation in networking. Programming Protocol-Independent Packet Processors (P4)[4] has been at the forefront of innovation, enabling the expression of forwarding behaviors, also known as network functions (NFs), using a domain-specific language. This can then be compiled into a program that can be loaded and run on target network devices with programmable data planes. As a result of this advancement, P4-based switch design has led to the widespread production of programmable data plane switches,[5] commonly referred to as P4 switches. They offer service providers, particularly in the cloud[6–10] and telecommunications[11,12] industries, a range of benefits, including ultrafast and flexible packet handling structures and the ability to fully customize previously vendor-specific functions.

Despite technological advances in network programmability, certain obstacles persist in deploying P4 switches, including limited memory capacity and support for processing primitives. The available P4 switches in the market have match table capacities that range from a few megabytes (TCAM) to a couple of hundred megabytes (SRAM) per pipeline.[13] However, certain NFs, such as cloud gateway applications (e.g., load-balancer, firewall, tunnel, and translation functions) and telecommunications gateway applications (e.g., broadband network gateway, access gateway function, and user plane function), exceed the boundaries of the silicon to manage frequently changing per-flow information for a massive number of concurrent flows.[14–16] This constrains P4 switches from efficiently supporting NFs that are critical for meeting complex demands of networks, traffic, and services.

The limited support of P4 switches for complex operations also poses a challenge. Each stage in the P4 switch pipeline can only execute one ALU instruction per packet field, with supported operations being limited to signed addition, bitwise logic, and hashing primitives.[17] The lack of support for operations such as multiplication, division, and floating-point calculations, as well as control flow mechanisms,[18,19] makes it infeasible to deploy complex NFs, particularly those requiring intelligent packet processing for applications such as machine learning/artificial intelligence and distributed processing workloads, enhanced security, and traffic management (e.g., monitoring, steering, scheduling, and shaping) on current P4 switches.

Several techniques have been recently proposed to overcome the limitations of P4 switches, such as using FPGAs to complement their capabilities,[13] modifying or extending the streaming-pipeline architecture,[18,19] performing computation on end servers by converting floating-point numbers to integers,[20,21] or utilizing local switch CPUs.[22] One effective solution is to offload traffic that requires advanced processing from P4 switches to network function virtualization (NFV). P4 switches, typically implemented in hardware, have higher performance but limited resources than virtualized network functions (VNFs), which are commonly implemented in software and operated on virtualized infrastructures. However, the hardware typically running underneath VNFs is not constrained in terms resource capacity and support for arithmetical operations, making it more suitable for implementing and operating complex NFs. By using P4 switches in combination with NFV, the main data plane path can stay on P4 switches to ensure the best performance switching capability, while VNFs can handle the complex processing not supported by P4 switches. This way, offloading plays a vital role in maximizing the capabilities of both P4 switches and NFV.

Despite the potential benefits of offloading traffic from P4 switches to NFV, there is currently a limited understanding of the efficacy it can quantitatively offer. Only by understanding the offloading behavior can proper performance-optimal measures be taken. To address this, this work focuses on optimizing offloading traffic with delay constraints in programmable data planes using an M/M/1 queue as an analytical model. Specifically, it examines the characteristics of a P4 switch managed by an SDN controller with packet processing offloading capability (i.e., offloading packets that the switch cannot process) to NFV. Offloading is performed with respect to the probability of packets being forwarded to NFV, while the goal of optimization is to determine the offloading rate for the analytical model that minimizes total packet processing delay. Furthermore, we perform parametric sensitivity impact analysis across a range of various parameters to gain insights into offloading strategies. The main contributions of this work are summarized as follows:

(i) We design an analytical model to examining the delay characteristics of a P4 switch governed by a logically centralized controller, endowed with offloading capabilities to VNF. To the best of our knowledge, no existing model addresses situations where NFV augments the operation of a programmable data plane. Our contribution stands as the pioneering effort in modeling this unique scenario.

(ii) We leverage the Bounded version of Brent's method to tackle the offloading challenge, with a goal to reduce latency while observing NF property constraints. Our analysis reveals that, for the chosen parameter configuration, an optimal offloading of approximately 66% of packets to the VNF results in the minimal total delay of 0.1505 μs. Furthermore, the optimization method showcases low computational complexity, emphasizing the method's efficiency.

(iii) We provide parametric sensitivity analysis to assist researchers and professionals in identifying critical factors that may impact network performance. This not only accentuates the academic and practical merit of our offloading solution but also elucidates the potential advantages and compromises in the collaborative deployment of P4 switches and NFV. Such findings can be instrumental in guiding superior network design, fostering informed architectural decisions, and facilitating strategic deployment choices.

(iv) We introduce a versatile methodology that offers bidirectional applicability. Our solution evaluates offloading from P4 switches to NFV by considering the probability of packets being forwarded to VNF. If analysis in the reverse direction—from NFV to P4 switches—is required, offloading can be assessed by inverting our formula. This focuses on the probability of packets being directed to physical network functions (PNFs), showcasing the adaptability of our approach.

The remainder of this paper is organized as follows. Section 2 delves into the background and related work. Section 3 introduces our model, detailing the interplay between a P4 switch and NFV, and delves into the delay analysis, laying the groundwork for optimization. Section 4 focuses on the optimization problem, elucidating the methodology adopted for its resolution. Section 5 offers a parameter sensitivity impact analysis, uncovering key findings. Section 6 offers a discussion underscoring the significance and application of our work, while also highlighting its limitations and suggesting potential future studies. Finally, Section 7 wraps up our discussions and conclusions.

2 | BACKGROUND

Scalability and flexibility have been constant challenges for traditional networks.[1] These limitations have been alleviated by introducing softwarization, virtualization, and programmability into networking, yielding SDN, NFV, and P4, respectively. They have driven many innovations in the field and fuelled numerous research works since their introduction. In what follows, we briefly overview these paradigms in a context relevant to this work and discuss related work.

2.1 | Software-defined networking (SDN)

SDN[2] decouples network control functions and forwarding functions by abstracting physical networking resources such as routers and switches. Consequently, decision-making is moved to a (logically centralized) virtual network control plane. The control plane determines what path(s) the traffic must take, while the network devices become simple forwarders that deliver the traffic.

SDN has become a favored alternative to traditional networking[23] for several reasons. It allows network architecture to be reconfigured as needed, bringing flexibility and dynamism to networking as opposed to traditional networking. It also helps optimize the efficiency of the upscaling process, facilitate network resources, reduce the maintenance workload, and develop more efficient data centers.[24]

2.2 | NFV

In contrast to SDN, NFV[3] decouples NFs from (proprietary) hardware appliances (e.g., routers and firewalls) and delivers equivalent network functionality without the need for specialized hardware. Thus, SDN and NFV can be seen as paradigms where SDN softwarizes the control plane and NFV virtualizes the data plane. Certainly, NFV and SDN are often used in tandem with one another.

NFV increases flexibility, agility, and scalability while reducing costs by enabling NFs to operate on commodity hardware. It also allows better isolation in terms of network security and production/test environments. However, the NFs are designed to be deployed on top of hypervisors (in virtual machines) and operated on commodity hardware with general-purpose CPUs. These are not explicitly designed for compute-intense tasks typical for NFs, and in doing so, it hinders scalability and performance challenges.[25,26]

2.3 | P4 switch

P4 switches are paramount to modern networking, promising several advantages. They enable managing NFs operating simultaneously in hardware and software. In addition, since P4 is protocol-independent, P4 switches enable implementing custom switching pipelines and provide an API for SDN controllers to populate tables. Furthermore, the runtime reconfigurability of P4 switches assists the flexible operation of NFs in the face of network dynamics.[27]

Figure 1 illustrates a reference model of a P4 switch. The architecture closely follows the Very Simple Switch reference model.[28] The white blocks denote programmable components whose behavior must be specified by a corresponding P4 program. At an architectural level, the programmable pipeline (i.e., PISA) includes three major components[29]: a Parser that defines what header fields are recognized and matched by later stages, a sequence of Match-Action Units programmed to match one or more header fields, and a Deparser that re-serializes the packet metadata into the packet before it is transmitted on the output link.

Figure 1 also shows the input arbiter, parser runtime, and output queue, colored orange, which are fixed-function components. The flow of user-defined data is denoted by dotted-red lines, while data plane interfaces conveying information between the programmable and fixed-function components are denoted by dashed-green lines. It must be noted that although P4 programs partly define an interface for communication between the control and data planes, as shown in Figure 1, P4 was designed to specify only the data plane functionality of the target.[28]

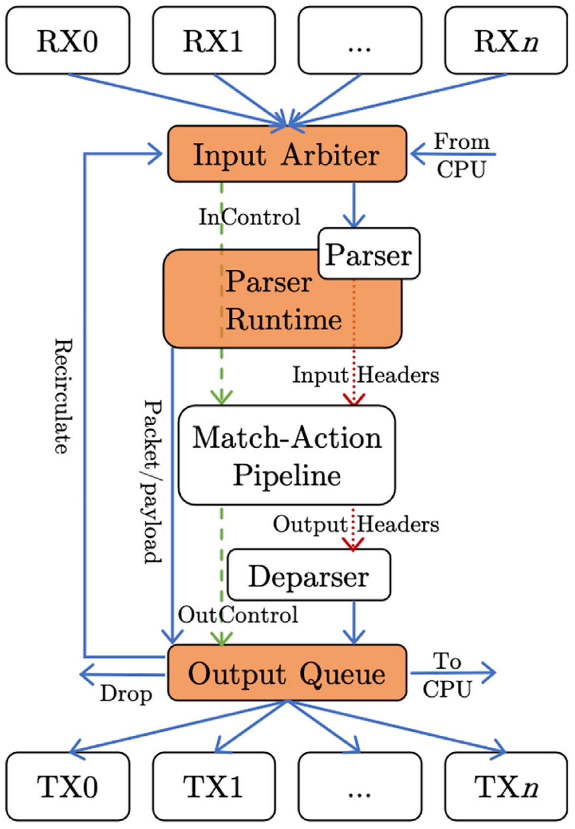

FIGURE 1 High-level schematic of a P4 switch.

The reference model in Figure 1 receives the packets via either one of the ingress ports, a port connected directly to the CPU, or a recirculation channel. Then, the packet follows the route through a single parser that feeds into a single match-action pipeline, which subsequently feeds into a single deparser.[28] Packet parsing is implemented as a state machine; the match-action table defines the packet processing pipeline, while the table is populated via a control plane interface. The behavior of a P4 program that processes a packet can be described through vectors of bits to vectors of bits mapping. When exiting the deparser, the packets are sent forth through either one of the egress ports or one of the three particular ports, which are (i) a CPU port that forwards the packets to the control plane, (ii) a drop port that discards the packets, and (iii) a recirculate port that re-injects the packets in the switch via a particular ingress port.

2.4 | Related work

Queuing models are extensively employed to analyze network performance under various conditions. We have compiled a comprehensive overview of related work in analytical modeling in Table 1. This table provides an easy comparison, summarizing the year of publication, the utilized model, whether the approach incorporates VNF and PNF, and unique characteristics of each study.

Several works[30,33–36] tackle the performance of OpenFlow switches. Jarschel et al[30] were the first to derive a basic model for the forwarding speed and blocking probability of an OpenFlow switch combined with an OpenFlow controller, where the OpenFlow controller was modeled by an M/M/1-S feedback queuing system. Although the model proposed in Jarschel et al[30] provides valuable insight, it is accurate only when the probability of expecting a new flow is small. Furthermore, extending the model to more than one forwarding element in the data plane is complex. Motivated by these limitations, Mahmood et al[33] proposed a model that addresses both of these challenges. Their model is based on the Jackson assumption,[45,46] but with corrections tailored to the OpenFlow-based SDN network. Jarschel et al[34] also reports on the performance analysis of an OpenFlow-based SDN. Compared to Jarschel et al[30] that examined the

TABLE 1 Related work aimed at analytical modeling.

	Year	SDN switch	VNF	PNF	Characteristics
Jarschel et al.[30]	2011	M/M/1	No	No	Fundamental modeling
Azodolmolky et al.[31]	2013	Network calculus	No	No	Boundary performance of packet delay and buffer sizing of controller
Bozakov and Rizk[32]	2013	Network calculus	No	No	Control message processing
Mahmood et al.[33]	2014	M/M/1	No	No	Adjusted traffic arrival rate at switch
Jarschel et al.[34]	2015	M/M/1	No	No	Multiple switches
Miao et al.[35]	2015	HPQ:MMPP/M/1 LPQ:MMPP/M/1/k	No	No	Priority queues
Shang and Wolter[36]	2016	M/H_2/1	No	No	Separation of packet-in messages
Sood et al.[37]	2016	M/Geo/1	No	No	Geometric distribution for service
Miao et al.[38]	2016	HPQ:MMPP/M/1 LPQ:MMPP/M/1/k	No	No	MMPP for multimedia traffic arrivals
Xiong et al.[39]	2016	M^x/M/1	No	No	Batch traffic arrival rate at switch
Goto et al.[40]	2016	HPQ:MMPP/M/1 LPQ:MMPP/M/1/k	No	No	Exact solution with Markov chain
Fahmin et al.[41]	2018	M/M/1	M/M/1	No	Combination of SDN and VNF
Singh et al.[42]	2018	2D Markov chain (HPQ, LPQ)	No	No	Software versus hardware switches
Zhao et al.[43]	2020	M/G/1	No	No	Software-defined WAN
Singh et al.[44]	2020	4D MC (internal buffer, HPQ, LPQ, hardware)	No	No	Encapsulation versus internal buffer
This work		M/M/1	M/M/1	M/M/1	P4 switch with NF

network performance in a single-controller setup, Jarschel et al[34] consider a multi-node setting. Hence, the proposed model of Jarschel et al[34] claimed to be suitable for analyzing network performance under more realistic topologies. Table 1 provides a summary of these works.

Research in Jarschel et al[30,33,34] has mainly focused on exploring the capacity of logically centralized controllers in the control plane while falling short of quantitatively investigating the performance of the underlying data plane. Miao et al[35] attempted to fill this gap and presented a preemption-based packet-scheduling scheme to improve global fairness and reduce the packet loss rate in the data plane. They evaluated the proposed scheduling scheme quantitatively using a system where the controller was modeled by an M/M/1 queue, while the switch was implemented using a combination of a high-priority (MMPP/M/1) and a low-priority (MMPP/M/1/k) queue. They then pinpointed the performance bottleneck of the SDN architecture. As a step forward, Miao et al[38] later adopted a priority-queue (PQ) system to model the data plane. The proposal was motivated by the realistic nature of multimedia traffic and the likely occurrence of packet arrival bursts which was modeled using a Markov-modulated Poisson process. Goto et al[40] also considered switches with two high- and low-priority queues. They proposed a queuing model of an OpenFlow-based SDN that considers the classful treatment of packets arriving at a switch. They also provided a detailed analysis of the proposed model. Their work focused on two critical aspects of analytic modeling: (i) the different treatment of packets arriving at a switch from a controller and (ii) the validation of analytical models. These approaches are summarized in Table 1.

Not all existing models are based on M/M/1 queues. Shang and Wolter[36] assessed the packet processing time of OpenFlow switches and controllers, however, using an M/H_2/1 queue. Sood et al[37] modeled an SDN switch as an M/Geo/1 system, where the incoming packets obeyed a Poisson distribution, and the service rate (rule-based match-action packet processing of SDN switch) obeyed a geometric distribution. Xiong et al[39] modeled the packet forwarding of OpenFlow switches and the packet-in message processing of SDN controller, respectively, as the M^x/M/1 and M/G/1 queuing systems. Finally, Zhao et al[43] developed a queuing system of packet forwarding performance in software-defined WAN with its controller modeled as an M/M/n queue, while the packet processing characterized as an M/G/1 queue. Table 1 lists these studies.

Network calculus[47] is also commonly used for performance modeling.[31,32] Azodolmolky et al[31] used network calculus to describe the functionality of an SDN switch and controller, aiming to model delay and queue length boundaries and analyzing the buffer length of the SDN controller and SDN switch. Bozakov and Rizk[32] focused on the concurrent operation of switches with diverse capacities for control message processing, which leads to unpredictable delays in SDN applications. They addressed this issue using a queuing model to characterize a switch's control interface service and used network calculus to derive the corresponding parameters. Details about these studies are cataloged in Table 1.

Only a handful of works are aimed at the analytical modeling of SDN and NFV. The pioneering work by Fahmin et al. modeled and analyzed SDN and NFV architectures using an M/M/1 queue.[41] These authors considered two architectures: one, in which the controller interacts with NFV, and another, where the switch interacts with NFV. Related works,[42,44] but generalized to hardware and software switches, were presented by Singh et al. These works also use queuing theory to model various performance characteristics (e.g., delay, packet loss, and throughput) of hardware switches and software switches in SDN. The main takeaway from them is that SDN and NFV are complementary technologies. Together they can help improve the flexibility and simplicity of networks and service delivery. These works are summarized in Table 1.

Lastly, existing performance evaluations for P4 switches are limited to focusing only on common metrics such as throughput and packet processing latency.[48–50] A variety of architectures have also been proposed to tackle resource management aspects (e.g., dynamic resource scheduling and NF provisioning and latency reduction) in NFV.[51–53] Furthermore, the reconfiguration capability of P4 switches has also been leveraged to achieve data plane virtualization.[27,54–56] Nonetheless, no work has been carried out on the analytical modeling of P4 switches with NFV. To the best of our knowledge, this paper is the first to attempt to close this gap.

3 | ANALYTICAL MODEL

In light of the inherent constraints of P4 switches, we propose a model where NFV works in tandem with a P4 switch to mitigate scalability issues, overcome hardware limitations, and reduce performance bottlenecks, specifically aiming to better cater to NFs that surpass the processing capabilities of conventional P4 switches. The offloading mechanism is shaped by the probability of packets being directed towards the NFV. The primary optimization objective is to determine a parametric configuration of components to minimize packet processing delay.

For a nuanced understanding, we dissect the internal architecture of the P4 switch into distinct components. The switch processing (SP) component undertakes the initial processing of incoming packets. The PNF manages the execution of intrinsic network operations within the P4 switch. The VNF caters to the packets offloaded from the P4 switch for further processing in the NFV environment. The switch communication (SC) component is dedicated to the transmission of packets that are lined up in the egress ports of the P4 switch. The controller (C) provides dynamic programmability and decision-making capabilities, especially for packets that do not match any predefined switch rules.

3.1 | Notations and assumptions

Table 2 provides a list of notations utilized in our analysis. The specific context of each notation, when used in individual components, is highlighted by its associated superscripts. Packet arrival rates across the different components are denoted by λ. Computing capacities for individual components are signified by C. Delays within the system are represented by the symbol D.

Our analytical model is based on the following assumptions:

1. The data arrival process at the switch conforms to a Poisson process.
2. Packets that have been processed by NF are treated as newly observed packets in the input queue without any differentiation.

TABLE 2 Notations used in the analysis.

Symbol	Description	Attribute
VNF	Virtualized network function	Abbreviation
PNF	Physical network function	Abbreviation
SC	Switch communication	Abbreviation
SP	Switch processing	Abbreviation
λ	Packet arrival rate	Input
λ^{PNF}	Packet arrival rate (PNF)	Variable
λ^{VNF}	Packet arrival rate (VNF)	Variable
λ^{SC}	Packet arrival rate (SC)	Variable
p^{C}	Table-miss probability (controller bound)	Input
p^{N}	Prob. requiring network function	Input
p^{VNF}	Prob. directed to VNF	Output
C^{VNF}	VNF capacity	Input
C^{PNF}	PNF capacity	Input
C^{SP}	SP capacity	Input
C^{SC}	SC capacity	Input
C^{C}	Controller capacity	Input
D^{SV}	Fixed propagation delay (P4 switch-VNF)	Input
D^{SC}	Fixed propagation delay (P4 switch-controller)	Input
$\overline{D}^{Total}$	Average total packet delay	Output
$\overline{D}^{SP}$	Average delay at SP	Variable
$\overline{D}^{C}$	Average delay at controller	Variable
$\overline{D}^{VNF}$	Average delay at VNF	Variable
$\overline{D}^{PNF}$	Average delay at PNF	Variable
$\overline{D}^{SC}$	Average delay at SC	Variable

3. NF processing occurs only once for each packet.
4. No partial flow offloads are supported. That is, if a flow is determined to be offloaded, all its packets will be offloaded.
5. High capacity on the NFV side is achieved through a server farm consisting of multiple highly scalable physical machines hosting VNF.
6. The analytical model employs an independent M/M/1 queue for VNF, PNF, SP, SC, and the controller.

3.2 | Model architecture

Figure 2 presents the structure of the employed queuing model. This model distinctly portrays the P4 switch, VNF, and the controller in separate segments. Packets, as depicted by the incoming arrow on the left, arrive following a Poisson process with a rate denoted by λ.

Upon arrival, every packet is initially processed by the P4 switch. During the ingress operation execution sequence within the P4 switch, the SP component evaluates if the packet requires NF processing. Should NF be necessary, the SP component then ascertains whether to process this packet locally using the P4 switch (PNF) or to offload it to the VNF, guided by the offloading ratio p^{VNF}. The selection of this parameter is crucial, as suboptimal values might adversely impact the average system latency. It is essential to emphasize that packets are physically transmitted out of the P4 switch when offloaded to the VNF, which precludes the presence of feedback loops within our model.

In situations where the switch is indecisive about the requisite operation for a particular packet, the SP component seeks guidance from the controller. This packet is then appropriately tagged and directed to the controller during the system's egress step. Similarly, during this operation, packets physically exit the P4 switch, ensuring that no scenario arises where a packet simultaneously demands both the controller and NF interventions. Once the controller introduces the necessary modifications at the packet-level, the modified packet is reintroduced into the P4 switch via its ingress port. Subsequently, the SP component processes this modified packet and relays it to the SC component, which then transmits the packet via the designated port.

Elucidating the color-coding in Figure 2:

- *Black* denotes packets with an arrival rate λ, initially directed to the SP component for processing.
- *Green* signifies the path followed by packets necessitating NF, serviced within the P4 switch with capacity C^{PNF}.

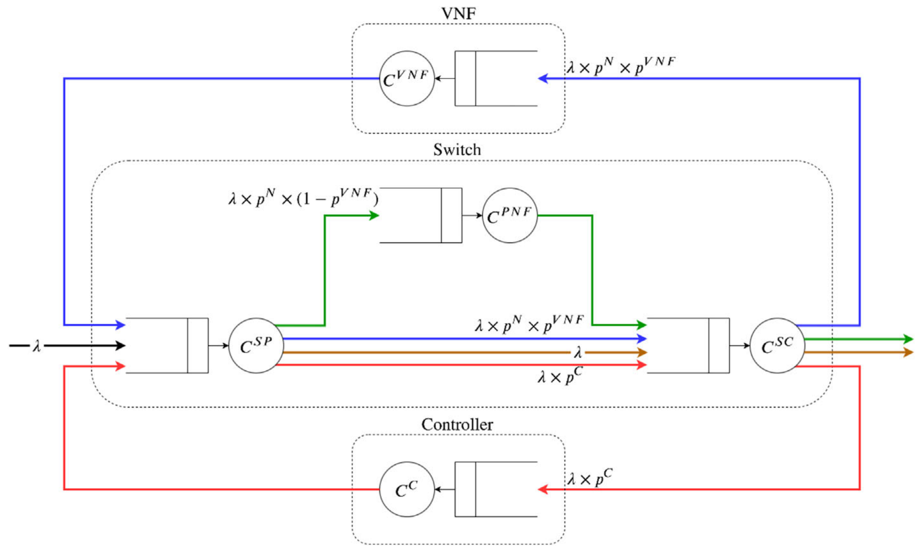

FIGURE 2 Analytical model structure illustrating the distinct segments for P4 switch, VNF, and controller. Each segment operates on an independent M/M/1 queue. The packet paths are color-coded: black for packets directed to SP; green for those requiring NF within the P4 switch; blue for packets offloaded to VNF, which, after processing, are returned to the SP, then passed to SC for egress; red for packets that the SP component cannot resolve, prompting the SC to forward them to the controller for instructions.

- *Blue* represents the route of packets directed to the VNF for load balancing. After being processed at node C^{VNF}, these packets are reintroduced to the SP component, then swiftly passed to the SC component without further modifications and finally dispatched through the relevant switch output port.
- *Red* highlights the route of packets that the SP component struggles to process due to a lack of a corresponding match + action table entry. These packets are directed by the SC component to the controller for further guidance.
- *Brown* delineates the route of packets that either do not require NF or have already undergone NF and possess a valid table hit. Such packets are initially processed by the SP component and subsequently transmitted through the SC component.

3.3 | Delay analysis

To analyze the delay in the system, we focus on the delay characteristics of our model, particularly within a P4 switch that utilizes VNF offloading. Each component of our system, including the VNF, PNF, SP, SC, and the SDN controller, is modeled as an M/M/1 queue. The delay at each node is determined by its capacity and the arrival rate given by

$$D = \frac{1}{\mu - \lambda}, \tag{1}$$

where μ denotes the *service rate* (or capacity), the maximum rate at which a queue can process packets, and λ represents the *arrival rate*, indicating the rate at which packets arrive at the component.

The system's *load* is defined by the traffic intensity ρ, which is the ratio of the arrival rate λ to the service rate μ, that is, $\rho = \lambda/\mu$. When the traffic intensity ρ approaches 1 (meaning λ is nearing μ), the delay in the M/M/1 queue system increases significantly. This situation is indicative of an M/M/1 queue nearing its capacity, implying that the system is approaching saturation and the processing rate is struggling to match the packet arrival rate. With this understanding, we proceed to discuss the delay components for each system node.

3.3.1 | Delay at the SP node

The packet delay, represented as $\overline{D}^{SP}$, considers interactions among the SP, VNF, and the SDN controller. It offers insights into packet processing at the SP node and is described by the equation

$$\overline{D}^{SP} = \frac{1}{C^{SP} - \lambda \times (1 + p^N \times p^{VNF} + p^C)}. \tag{2}$$

This delay is influenced by the SP node's capacity (or service rate) C^{SP} and the total packet arrival rate at SP, which consists of three components:

(i) *Inter-Switch Traffic*: Packets arriving from another switch to the SP at a rate λ.
(ii) *VNF-Routed Traffic*: Packets that return to the SP after processing by the VNF at a rate of $\lambda \times p^N \times p^{VNF}$. Here, p^N represents the chance that packets require NF processing, and p^{VNF} is the probability they are processed by the VNF.
(iii) *Controller-Routed Traffic*: Packets returning to the SP after consultation with the SDN controller at a rate $\lambda \times p^C$. Here, p^C is the table-miss probability, signifying the frequency at which the SP seeks guidance from the controller for routing decisions.

Summing these components, the total packet arrival rate at the SP is $\lambda \times (1 + p^N \times p^{VNF} + p^C)$. This rate includes the direct packet inflow, packets coming from the VNF, and packets returning from the SDN controller.

3.3.2 | Delay at the controller node

The controller is crucial for the operation of the P4 switch. If the switch encounters a packet without the required instructions, the packet is sent to the controller for further processing. This interaction introduces delays, determined as

$$\overline{D}^{C} = \frac{1}{C^{C} - \lambda \times p^{C}} + 2 \times D^{SC}, \tag{3}$$

where C^{C} is the controller's service rate, indicating its processing capacity. The term $\lambda \times p^{C}$ represents the packet arrival rate at the controller, where λ is the average packet rate and p^{C} is the table-miss probability, reflecting how often the P4 switch sends a packet to the controller due to missing instructions.

The term $2 \times D^{SC}$ represents the communication delay between the P4 switch and the controller. Specifically, D^{SC} denotes the one-way delay, and since sending a packet involves round-trip communication, the delay is effectively doubled.

3.3.3 | Delay at the VNF node

The VNF serves as an alternative processing unit when the PNF is unable to handle packets due to excessive load or hardware constraints. The average delay at the VNF node, $\overline{D}^{\mathrm{VNF}}$, is calculated using

$$\overline{D}^{\mathrm{VNF}} = \frac{1}{C^{\mathrm{VNF}} - \lambda \times p^{N} \times p^{\mathrm{VNF}}} + 2 \times D^{\mathrm{SV}}, \tag{4}$$

where C^{VNF} is the service rate of the VNF, which represents its maximum packet processing capability. The term $\lambda \times p^{N} \times p^{\mathrm{VNF}}$ denotes the effective packet arrival rate at the VNF, with λ as the average packet arrival rate, p^{N} as the probability that packets require NF processing, and p^{VNF} as the fraction of those packets offloaded to the VNF. Additionally, $2 \times D^{\mathrm{SV}}$ accounts for the round-trip communication delay between the P4 switch and the VNF.

3.3.4 | Delay at the PNF node

The PNF serves as the primary processing unit within the P4 switch. When a packet requires NF processing and is not offloaded to the VNF, it is processed by the PNF. The average delay at the PNF node, $\overline{D}^{\mathrm{PNF}}$, is then

$$\overline{D}^{\mathrm{PNF}} = \frac{1}{C^{\mathrm{PNF}} - \lambda \times p^{N} \times (1 - p^{\mathrm{VNF}})}, \tag{5}$$

where C^{PNF} represents the PNF's service rate. The term $\lambda \times p^{N} \times (1 - p^{\mathrm{VNF}})$ indicates the effective packet arrival rate at the PNF, with λ as the average packet arrival rate, p^{N} as the probability that packets require NF processing, and $(1 - p^{VNF})$ as the subset of those packets processed by the PNF, as opposed to being offloaded to the VNF.

Importantly, the PNF, being an integral component of the P4 switch, is optimized for efficiency. Thus, unlike the scenario described in Equation 4, the PNF does not introduce any additional communication delays into the system. This ensures that no additional communication delays are introduced for packets processed by the PNF, optimizing the overall system performance.

3.3.5 | Delay at the SC node

The SC component serves as the egress point of the system, directing packets as they exit the switch. The average delay at the SC node, $\overline{D}^{\mathrm{SC}}$, is given by

$$\overline{D}^{\mathrm{SC}} = \frac{1}{C^{\mathrm{SC}} - \lambda \times (1 + p^{N} \times p^{\mathrm{VNF}} + p^{C})}, \tag{6}$$

where C^{SC} indicates the SC node's service rate for managing and directing outgoing packets. Analogous to Equation (2), the effective packet departure rate at the SC node consists of three components: *Traffic Routed Out of the Switch*, *Traffic Routed to VNF*, and the *Traffic Routed to Controller*.

3.3.6 | Individual route delays

The total packet delay within the system accumulates as packets traverse various components from ingress to egress. The exact delay depends on the designated processing route of each packet, whether it be through the VNF, the controller, or the PNF.

For packets that are routed through the VNF, the total delay $\overline{D}^{\mathrm{VNF}}_{\mathrm{Total}}$ is the sum of the delays experienced at the SP and SC nodes, both of which are traversed twice, as well as the delay at the VNF itself. Mathematically, this relationship is given by

$$\overline{D}^{\mathrm{VNF}}_{\mathrm{Total}} = 2\times\overline{D}^{\mathrm{SP}} + 2\times\overline{D}^{\mathrm{SC}} + \overline{D}^{\mathrm{VNF}}. \tag{7}$$

In the case of packets directed to the controller, the total delay $\overline{D}^{C}_{\mathrm{Total}}$ includes twice the delays at the SP and SC nodes and the delay incurred at the controller. This can be expressed as

$$\overline{D}^{C}_{\mathrm{Total}} = 2\times\overline{D}^{\mathrm{SP}} + 2\times\overline{D}^{\mathrm{SC}} + \overline{D}^{C}. \tag{8}$$

For packets processed by the PNF, the total delay $\overline{D}^{\mathrm{PNF}}_{\mathrm{Total}}$ is simply the sum of the delays at the SP, PNF, and SC nodes. This sum can be described as

$$\overline{D}^{\mathrm{PNF}}_{\mathrm{Total}} = \overline{D}^{\mathrm{SP}} + \overline{D}^{\mathrm{PNF}} + \overline{D}^{\mathrm{SC}}. \tag{9}$$

3.3.7 | Total system delay

The primary aim of this analysis is to quantify the average packet delay throughout the system. This overall delay, denoted as $\overline{D}^{\mathrm{Total}}$, is an aggregate measure that incorporates delays from the various routes a packet may take, weighted by the probability of each route.

In light of the individual delay components previously calculated, the total system delay is a weighted sum of the delays experienced by packets routed through the VNF, the controller, and the PNF, in addition to the delays experienced at the SP and SC nodes. This system delay is expressed as

$$\overline{D}^{\mathrm{Total}} = p^N\times p^{VNF}\times\overline{D}^{VNF}_{\mathrm{Total}} + p^N\times(1-p^{VNF})\times\overline{D}^{PNF}_{\mathrm{Total}} + p^C\times\overline{D}^{C}_{\mathrm{Total}} + (1-p^N-p^C)\times(\overline{D}^{SP}+\overline{D}^{SC}), \tag{10}$$

where

$p^N\times p^{VNF}\times\overline{D}^{VNF}_{\mathrm{Total}}$	denotes the cumulative delay for packets needing NF processing and routed through the VNF.
$p^N\times(1-p^{VNF})\times\overline{D}^{PNF}_{\mathrm{Total}}$	signifies the cumulative delay for packets requiring NF processing but handled by the PNF.
$p^C\times\overline{D}^{C}_{\mathrm{Total}}$	represents the cumulative delay for packets routed to the controller.
$(1-p^N-p^C)\times(\overline{D}^{SP}+\overline{D}^{SC})$	corresponds to the probability of a packet bypassing NF processing and the controller, thus only incurring delays at the SP and SC nodes.

4 | OFFLOADING OPTIMIZATION

In modern networks, offloading specific tasks to VNF offers flexibility and scalability advantages. However, the decision to offload—and to what extent—often manifests as a trade-off between computational efficiency and system latency. By optimizing the offloading probability, we can strike a balance that minimizes the overall system delay, thereby enhancing performance.

We commence by presenting a formal problem statement aimed at determining the optimal offloading probability, denoted as p^{VNF}. Subsequently, we establish the convexity of the optimization problem in question and employ a computational technique for efficiently finding its solution.

4.1 | Problem statement

The optimization problem at hand can be formalized as
Given:

Packet arrival rate, denoted as λ;
Table-miss probability, denoted as p^{C};
Probability of requiring NF processing, denoted as p^{N};
Capacities of PNF, VNF, SP, SC, and the controller, denoted as $C^{PNF}, C^{VNF}, C^{SP}, C^{SC}$, and C^{C}, respectively;
Propagation delay between the P4 switch and the VNF, denoted as D^{SV};
Propagation delay between the P4 switch and the controller, denoted as D^{SC}.

Determine:

The optimal offloading probability, p^{*VNF}.

Objective:

Minimize the average total delay, $\overline{D}^{Total}$, by optimizing p^{VNF}.

To summarize, the crux of the optimization problem is to determine the value of p^{VNF} that minimizes the total delay, $\overline{D}^{Total}$, while adhering to the natural constraints of a probability measure, $0 \leq p^{VNF} \leq 1$. The optimal offloading probability is represented as p^{*VNF} and is formally expressed

$$p^{*VNF} := \min_{0 \leq p^{VNF} \leq 1} \overline{D}^{Total}(p^{VNF}). \tag{11}$$

4.2 | Convexity of the problem

The convexity of the objective function is a critical factor in determining the feasibility and computational efficiency of solving the optimization problem. In the context of our offloading optimization problem, it is crucial to establish whether the objective function $\overline{D}^{Total}$ is convex within the domain of p^{VNF}.

To rigorously examine this, the function $\overline{D}^{Total}$ would be considered convex if its second derivative with respect to p^{VNF} is non-negative across the entire range of p^{VNF} values, specifically between 0 and 1. Mathematically, this criterion can be formalized as

$$\frac{\partial^2 \overline{D}^{Total}}{\partial p^{VNF^2}} \geq 0, \ \forall p^{VNF} \in [0,1]. \tag{12}$$

Evaluating Equation (12) serves as the basis for determining whether standard convex optimization techniques can be applied efficiently to our problem. If the function is found to be convex, this would affirm that a unique global minimum exists and that it can be found efficiently. On the other hand, if the function is not convex, then alternative optimization approaches might be necessary, involving more complex and potentially computationally intensive methods.

4.3 | Baseline parameters

To set a solid foundation for our convexity verification (and subsequent impact analysis), we delineate a specific set of parameters. While certain specifications of P4 switches might be publicly available, translating CPU, storage, and other capacities into the domain of queuing model capacities presents an intricate challenge. Our analytical model, firmly anchored in the M/M/1 queuing framework, is designed to abstractly represent specific resources such as CPU, storage, and others into more generalized capacity metrics. The inherent nature of the M/M/1 model, while being influenced by these resources, is not equipped to delineate them in detailed granularity without introducing complex intricacies and potential deviations from practicality.

Consequently, the essence of our study leans towards a higher tier abstraction, where the capacities of the nodes are based on generalized observations from analogous hardware systems and well-established networking norms. This level of abstraction ensures a broader applicability while encapsulating the fundamental dynamics of system performance influenced by underlying resource constraints.

Our hypothetical P4 switch's modeling, with a line rate of up to 4.8 Tbps, aims to be representative of real-world scenarios. For context, real-world P4 switches like the Tofino series feature bandwidths ranging from 1.8 to 6.4 Tbps, with the Tofino 2 series extending this range from 4.8 to 12.8 Tbps.[13] With a packet size of 1500 Bytes in mind, a rate of 4.8 Tbps corresponds to a potential packet processing capacity of 400 pkts/μs for both C^{SP} and C^{SC}.

The packet arrival rate λ is fixed at 25 pkts/μs. This value is representative of a moderate-to-high traffic scenario, offering a good balance for assessing system performance under non-extreme conditions while still presenting challenges worth optimizing.

For the capacities of PNF, VNF, and the controller (C^{PNF}, C^{VNF}, and C^{C}), our assumptions parallel the expectations of their computational prowess. PNF, typically dedicated hardware, is set at 20 pkts/μs. VNF, visualized as a more scalable server farm configuration, naturally assumes a higher capacity—hence the 80 pkts/μs. Controllers, mainly tasked with management and not high-speed processing, are conservatively pegged at 10 pkts/μs.

Propagation delays, D^{SV} and D^{SC}, factor in the physical distance and transmission medium between the P4 switch and other components. Assuming short distances typical of data center setups—spanning a few kilometers or less—and fiber-optic channels, D^{SV} is set at 0.04 μs. D^{SC}, at 0.19 μs, reflects the potential for more varied and potentially slower transmission paths to the controller.

Finally, we outline network-specific probabilities p^{C} and p^{N}. A flow arrival probability of 56% and a 76% share for NFV services represent anticipated typical service distributions in contemporary networks. While the exact values are approximations, they reflect patterns observed in high-performance network settings.

Table 3 tabulates the values allocated to each parameter. For accuracy in our experiments, propagation delays and network probabilities remain unrounded, as indicated in Table 3, ensuring the preservation of nuanced effects these parameters introduce.

4.4 | Verification of convexity

To verify the convexity of $\overline{D}^{\text{Total}}$ over the interval $[0,1]$, we investigate its second derivative with respect to p^{VNF}. The first derivative of $\overline{D}^{Total}$ with respect to p^{VNF} is

TABLE 3 Baseline parameters.

Parameter	Value	Description
λ	25 pkts/μs	Packet arrival rate at C^{SP}
C^{SP}	400 pkts/μs	Service rate at SP
C^{SC}	400 pkts/μs	Service rate at SC
C^{PNF}	20 pkts/μs	Service rate at PNF
C^{VNF}	80 pkts/μs	Service rate at VNF
C^{C}	10 pkts/μs	Service rate at controller
D^{SV}	0.0422μs	Propagation delay at VNF
D^{SC}	0.1876μs	Propagation delay at controller
p^{C}	0.5602	Probability of redirecting to controller
p^{N}	0.7553	Probability of redirecting to NF

$$\begin{aligned}\frac{\partial \overline{D}^{Total}}{\partial p^{VNF}} &= p^C\left(\frac{2\times\lambda\times p^N}{(C^{SP}-\lambda\times(p^C+p^N\times p^{\mathrm{VNF}}+1))^2}+\frac{2\times\lambda\times p^N}{(C^{SC}-\lambda\times(p^C+p^N\times p^{\mathrm{VNF}}+1))^2}\right)\\
&+p^N\times p^{\mathrm{VNF}}\left(\frac{\lambda\times p^N}{(C^{VNF}-\lambda\times p^N\times p^{\mathrm{VNF}})^2}+\frac{2\times\lambda\times p^N}{(C^{SP}-\lambda\times(p^C+p^N\times p^{\mathrm{VNF}}+1))^2}+\frac{2\times\lambda\times p^N}{(C^{SC}-\lambda\times(p^C+p^N\times p^{\mathrm{VNF}}+1))^2}\right)\\
&+p^N\times(1-p^{\mathrm{VNF}})\left(\frac{\lambda\times p^N}{(C^{SP}-\lambda\times(p^C+p^N\times p^{\mathrm{VNF}}+1))^2}+\frac{\lambda\times p^N}{(C^{SC}-\lambda\times(p^C+p^N\times p^{\mathrm{VNF}}+1))^2}-\frac{\lambda\times p^N}{(C^{PNF}-\lambda\times p^N\times(1-p^{\mathrm{VNF}}))^2}\right)\\
&-p^N\left(\frac{1}{C^{SP}-\lambda\times(p^C+p^N\times p^{\mathrm{VNF}}+1)}+\frac{1}{C^{SC}-\lambda\times(p^C+p^N\times p^{\mathrm{VNF}}+1)}+\frac{1}{C^{PNF}-\lambda\times p^N\times(1-p^{\mathrm{VNF}})}\right)\\
&+p^N\left(2\times D^{SV}+\frac{1}{C^{VNF}-\lambda\times p^N\times p^{\mathrm{VNF}}}+\frac{2}{C^{SP}-\lambda\times(p^C+p^N\times p^{\mathrm{VNF}}+1)}+\frac{2}{C^{SC}-\lambda\times(p^C+p^N\times p^{\mathrm{VNF}}+1)}\right)\\
&+\left(\frac{\lambda\times p^N}{(C^{SP}-\lambda\times(p^C+p^N\times p^{\mathrm{VNF}}+1))^2}+\frac{\lambda\times p^N}{(C^{SC}-\lambda\times(p^C+p^N\times p^{\mathrm{VNF}}+1))^2}\right)\times(-p^C-p^N+1).\end{aligned}\tag{13}$$

Proceeding further, the second derivative of $\overline{D}^{Total}$ is given by

$$\begin{aligned}\frac{\partial^2 \overline{D}^{Total}}{\partial p^{VNF2}} &= p^C\times\left(\frac{4\times\lambda^2\times(p^N)^2}{(C^{SP}-\lambda\times(p^C+p^N\times p^{\mathrm{VNF}}+1))^3}+\frac{4\times\lambda^2\times(p^N)^2}{(C^{SC}-\lambda\times(p^C+p^N\times p^{\mathrm{VNF}}+1))^3}\right)\\
&+p^N\times p^{\mathrm{VNF}}\times\left(\frac{2\times\lambda^2\times(p^N)^2}{(C^{VNF}-\lambda\times p^N\times p^{\mathrm{VNF}})^3}+\frac{4\times\lambda^2\times(p^N)^2}{(C^{SP}-\lambda\times(p^C+p^N\times p^{\mathrm{VNF}}+1))^3}+\frac{4\times\lambda^2\times(p^N)^2}{(C^{SC}-\lambda\times(p^C+p^N\times p^{\mathrm{VNF}}+1))^3}\right)\\
&+p^N\times(1-p^{\mathrm{VNF}})\times\left(\frac{2\times\lambda^2\times(p^N)^2}{(C^{SP}-\lambda\times(p^C+p^N\times p^{\mathrm{VNF}}+1))^3}+\frac{2\times\lambda^2\times(p^N)^2}{(C^{SC}-\lambda\times(p^C+p^N\times p^{\mathrm{VNF}}+1))^3}+\frac{2\times\lambda^2\times(p^N)^2}{(C^{PNF}-\lambda\times p^N\times(1-p^{\mathrm{VNF}}))^3}\right)\\
&-2\times p^N\times\left(\frac{\lambda\times p^N}{(C^{SP}-\lambda\times(p^C+p^N\times p^{\mathrm{VNF}}+1))^2}+\frac{\lambda\times p^N}{(C^{SC}-\lambda\times(p^C+p^N\times p^{\mathrm{VNF}}+1))^2}-\frac{\lambda\times p^N}{(C^{PNF}-\lambda\times p^N\times(1-p^{\mathrm{VNF}}))^2}\right)\\
&+2\times p^N\times\left(\frac{\lambda\times p^N}{(C^{VNF}-\lambda\times p^N\times p^{\mathrm{VNF}})^2}+2\times\frac{\lambda\times p^N}{(C^{SP}-\lambda\times(p^C+p^N\times p^{\mathrm{VNF}}+1))^2}+2\times\frac{\lambda\times p^N}{(C^{SC}-\lambda\times(p^C+p^N\times p^{\mathrm{VNF}}+1))^2}\right)\\
&+\left(\frac{2\times\lambda^2\times(p^N)^2}{(C^{SP}-\lambda\times(p^C+p^N\times p^{\mathrm{VNF}}+1))^3}+\frac{2\times\lambda^2\times(p^N)^2}{(C^{SC}-\lambda\times(p^C+p^N\times p^{\mathrm{VNF}}+1))^3}\right)\times(-p^C-p^N+1).\end{aligned}\tag{14}$$

Owing to the intricate nature of the second derivative in Equation (14), undertaking a formal verification of convexity would be excessively arduous. Therefore, we opt for numerical verification as a more feasible approach to validate the convexity condition. To this end, we plot the function of the second derivative over the interval [0,1] as a means of visual confirmation.

Figure 3 delineates the behavior of the second derivative of $\overline{D}^{\mathrm{Total}}$ in relation to p^{VNF} within the interval [0,1]. The parameter settings for this representation are detailed in Table 3. As can be observed from Figure 3, the second derivative consistently assumes positive values throughout the specified interval. This positive trend substantiates the convex nature of $\overline{D}^{\mathrm{Total}}$ across its entire domain. Consequently, the enduring positivity of the second derivative within the interval [0,1] validates the suitability of employing convex optimization techniques for problem-solving in this context.

4.5 | Convex optimization

For our global minimization task, we employ the Bounded version of Brent's method.[57] Brent's method is a sophisticated root-finding algorithm, integrating the techniques of the bisection method, the secant method, and inverse quadratic interpolation.

One inherent drawback of the traditional Brent's method is its bracketing interval. Although it allows for the stipulation of an initial interval, it does not affirmatively confine the solution within this span. Such a limitation is pronouncedly significant in situations where domain-specific knowledge dictates a solution's boundary, or when

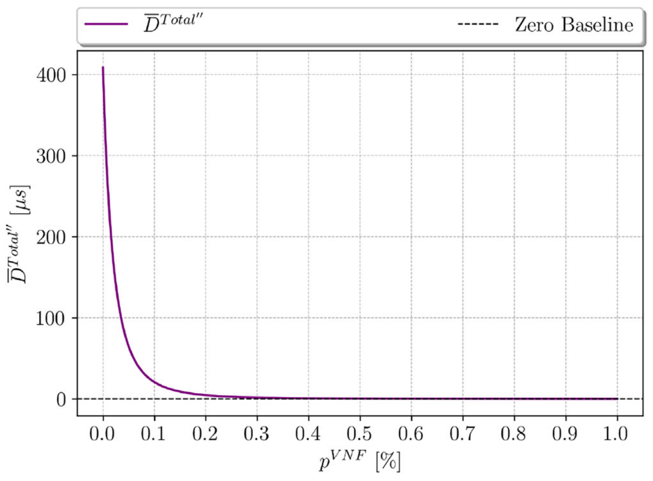

FIGURE 3 The second derivative of $\overline{D}^{\text{Total}}$ with respect to p^{VNF} over the interval [0,1]

surmounting certain limits is impermissible, due to stringent physical constraints or essential safety protocols. In addressing these challenges, the Bounded variant stands out. While echoing the core tenets of the original, this adaptation ensures that, during its execution, the deduced solutions remain rigorously within the boundaries set by the user.

Given our function $\frac{\partial \overline{D}^{Total}}{\partial p^{VNF}}$ representing the first derivative of the total delay with respect to p^{VNF}, we aim to find the value of p^{VNF} that makes this derivative zero, which in turn will give us the optimal value for p^{VNF} that minimizes $\overline{D}^{Total}$. The pseudocode for our adapted Bounded method, based on Brent's algorithm, is shown in Algorithm 1.

Algorithm 1 Bounded Method (Based on Brent) for Minimizing $\overline{D}^{Total}$

```
1:  procedure BoundedBrent(f, a, b)   ▷ Here, $f$ is the first derivative of $\overline{D}^{Total}$, $a$ and $b$ are the endpoints of the interval
2:      Tolerance ← $1e-5$
3:      if $|f(a)| < |f(b)|$ then
4:          Swap $a$ and $b$
5:      end if
6:      $c \leftarrow b$
7:      while $|b-a| \geq$ Tolerance do
8:          if $f(b) \neq f(a)$ and $f(b) \neq f(c)$ and $|b-a| <$ Tolerance $\times |b|$ then
9:              $s \leftarrow$ Inverse Quadratic Interpolation of $a, b,$ and $c$
10:         else
11:             $s \leftarrow \frac{a\times f(b) - b\times f(a)}{f(b)-f(a)}$   ▷ Secant method
12:         end if
13:         if $s$ is not between $(3a+b)/4$ and $b$ or $s<0$ or $s>1$ then   ▷ Enforce bounds
14:             $s \leftarrow \frac{a+b}{2}$   ▷ Bisection method
15:             $c \leftarrow b$
16:         end if
17:         if $f(s) < f(b)$ then
18:             $a \leftarrow b$
19:         else
20:             $f(a) \leftarrow f(a)/2$
21:             $b \leftarrow s$
22:         end if
23:     end while
24:     return $b$   ▷ Approximate optimal $p^{VNF}$
25: end procedure
```

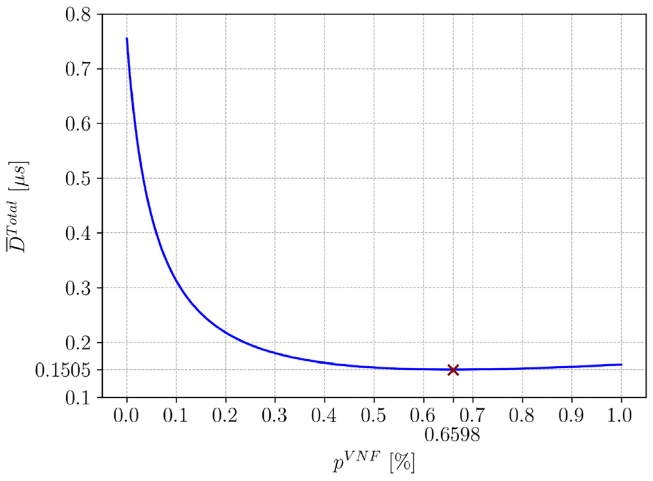

FIGURE 4 Optimal p^{VNF} and total system delay.

The NFV Probability Optimization with the Bounded Brent's method (NPOB) in Algorithm 1 efficiently finds the optimal p^{VNF} that minimizes the total average delay $\overline{D}^{Total}$ in our system. The method starts with a predetermined interval $[a,b]$ where $a=0$ and $b=1$. The specific steps of the Bounded method, built on Brent's approach, ensure fast convergence while maintaining reliability within the specified bounds. It iteratively refines this interval by estimating the root using either the secant method, inverse quadratic interpolation, or the bisection method. The variable c stores the previous value of b from the last iteration, aiding in convergence and in determining the method of root estimation for the current iteration. A pivotal aspect of the procedure is the tolerance value which dictates the accuracy of the approximation. For our research, we maintained its value at the default $1e-5$. The algorithm concludes when the size of the interval $[a,b]$ shrinks below this tolerance, signifying that the root has been approximated to the targeted precision. This root identifies the optimal value of p^{VNF} we aim to determine.

Given the parameter configuration detailed in Table 3, we conducted a convex optimization to determine the optimal p^{VNF} that minimizes the overall average delay $\overline{D}^{Total}$ within our system. As illustrated in Figure 4, the graph depicts the behavior of the objective function across the interval [0,1], encompassing the optimal p^{VNF} value and its corresponding total system delay $\overline{D}^{Total}$.

The information derived from Figure 4 verifies the success of the optimization process. The outcomes of the optimization reveal the following insights:

- The optimal p^{VNF} value that leads to minimal total delay is 0.6598. This suggests that approximately 66% of the packets should be offloaded to the VNF in order to minimize the overall delay.
- The attained minimum total delay at this optimal p^{VNF} value is 0.1505 μs.

This finding implies that according to the established system model and the pragmatic parameter boundaries, offloading a proportion of the network traffic to the VNF offers justifiable benefits from a delay reduction perspective.

To foster reproducibility, transparency, and adherence to open science principles, we have made the scripts used in our experiments publicly accessible.[*] We believe this will facilitate a thorough comprehension of our methodologies and encourage additional investigation in this domain.

4.6 | Computational complexity

Our approach is characterized by its computational efficiency, which stands as one of its primary merits. We address the optimization challenge using the Bounded method, which is built upon Brent's root-finding algorithm. This method

*https://github.com/FlowFrontiers/P4toNFV/.

excels with continuous, non-differentiable functions. The Bounded method ensures convergence within the specified bounds, benefiting from the combination of the bisection method's robustness and the faster-converging methods, such as the secant method and inverse quadratic interpolation.

The worst-case complexity for the Bounded method, akin to Brent's, is $O(\log n)$. Here, n can be interpreted as the inverse of the smallest distinguishable difference within the interval $[0,1]$, essentially representing the precision at which the solution is sought. This complexity predominantly arises from the bisection steps. However, when conditions favor the use of inverse quadratic interpolation or the secant method, the algorithm can achieve super-linear convergence, often resulting in fewer iterations compared to a pure bisection approach.

For our unique problem, the function $\overline{D}^{Total}$ displays favorable behavior within the relevant range for p^{VNF}, enabling efficient convergence by the Bounded method. Notably, each iteration necessitates the evaluation of $\overline{D}^{Total}$. Given that our function encompasses basic arithmetic operations and rational functions, its evaluation possesses a complexity of $O(1)$ per iteration.

In our experiments, with a precision n set to 100, the Bounded method consistently exhibited rapid convergence. Specifically, it converged to the optimal solution in just 10 iterations, with the objective function undergoing minimization merely 10 times throughout the optimization process. Evaluations extended to parameter ranges differing from those in Section 4.3 yielded consistent observations. These outcomes suggest that the associated computational effort can be characterized by an $O(k)$ complexity. In the context of our challenge, this complexity essentially mirrors constant-time behavior.

Factoring in both the time dedicated to function evaluations and the inherent complexity of the method, the overall time complexity of our procedure remains $O(k)$. In practical terms, this can be considered virtually constant-time, underscoring the method's superior efficiency for real-time computations in dynamic environments.

The proficiency of our approach indicates that real-time re-optimization of p^{VNF} or adjustments in response to fluctuating network conditions is feasible. This adaptability is crucial in contemporary networks, especially in programmable data planes like P4 switches, which might encounter varying loads.

5 | IMPACT ANALYSIS

This section delineates the results of our impact analysis. Using the baseline parameters enumerated in Table 3, we delved into the interplay between offloading strategies and system performance, specifically the optimal probability p^{*VNF} and the ensuing total delay $\overline{D}^{Total}(p^{*VNF})$.

5.1 | Analysis overview

We begin by overviewing the methodology for the analysis, ensuring that we present our approach in a structured manner. Our discourse seeks to furnish readers with a lucid, methodical insight, allowing for a nuanced appreciation of the system's behavior across various conditions.

5.1.1 | Parameters of interest

We examine offloading from P4 switches to NFV across five pivotal dimensions:

- Packet arrival rate (λ)
- Probability of redirecting to NF (p^N)
- Service rate at PNF (C^{PNF})
- Service rate at VNF (C^{VNF})
- Propagation delay at VNF (D^{SV})

5.1.2 | Parameter-centric analysis

To rigorously assess the robustness and validity of our model, each parameter of interest was independently varied within a predefined range. During this variation, all other parameters were consistently held at their baseline values. This range was selected to encompass values proximal to our baseline parameters, as detailed in Section 4.3. In this regard, our proposed system is examined with a combination of static and dynamic behaviors, depending on the context. By constraining our analysis in this manner, we guard against potential confounding variables, ensuring that observed alterations in system performance are solely a function of the varied parameter. This investigative methodology, rooted deeply in the characteristics of our chosen hardware benchmark, not only provides precision but also accentuates the real-world relevance and applicability of our conclusions.

5.1.3 | Strategies as case studies

Our analysis is segmented into three distinct strategies: no offloading, full offloading, and optimal offloading. These strategies can be envisioned as individual case studies that illuminate unique facets of the system's behavior. Each strategy offers insights into how the system responds under specific conditions. By amalgamating these strategies into a single figure, our goal is to streamline the narrative, enhancing the reader's interpretability and comprehension. The juxtaposition of these strategies furnishes a panoramic view, granting the reader a holistic understanding of the system's multifaceted dynamics.

5.1.4 | Figure structure and interpretation

The resulting figures embody the following conventions:

- *Axes*:
  - *Left y axis*: Denotes the average delay, expressed in μs.
  - *Right y axis*: Reflects the optimal p^{VNF}.
  - *x axis*: Specifies the scrutinized parameter.
- *Curves*: System behavior is depicted via four distinguishable curves:
  - *Blue curve*: Captures the delay without offloading ($\overline{D}^{Total}(0)$).
  - *Green curve*: Chronicles the delay under full offloading ($\overline{D}^{Total}(1)$).
  - *Red curve*: Mirrors the delay given the optimal offloading p^{VNF} ($\overline{D}^{Total}(p^{*VNF})$).
  - *Orange curve*: Illustrates the optimal p^{VNF} value (p^{*VNF}).
- *Values*: The analysis utilizes values derived from the Python "np.linspace" function, which divides the ranges into evenly spaced samples. As a result, some specific values referenced in the discussion might be approximate rather than exact.

5.2 | Impact of λ

In our analysis concerning the packet arrival rate, λ, we specifically assess its range from 22 to 26 pkts/μs. The implications of varying λ on the average system delay are depicted in Figure 5.

The 22 pkts/μs rate is distant from any saturation point, as deduced from the consistently low average delays across all offloading strategies. Interestingly, such a modest packet arrival rate already underscores the pronounced benefits of offloading. In particular, optimal offloading emerges as a superior strategy, providing a delay reduction that markedly outperforms both no offloading and full offloading. At λ of 22 pkts/μs, the average delay under optimal offloading can be slashed by around 76.21% (from approximately 0.201 μs down to approximately 0.048 μs) when juxtaposed with P4 switches under no offloading. Comparatively, it offers a reduction of about 17.73% (from approximately 0.058 μs down to approximately 0.048 μs) against the full offloading scenario. This underlines the assertion that offloading to VNF, even at moderate rates, can propel system performance from a delay reduction perspective.

As we venture into higher packet arrival rates, due to the increased load on the PNF as compared to the VNF, the delay observed under no offloading spikes more rapidly than that under full offloading as the packet arrival rate rises.

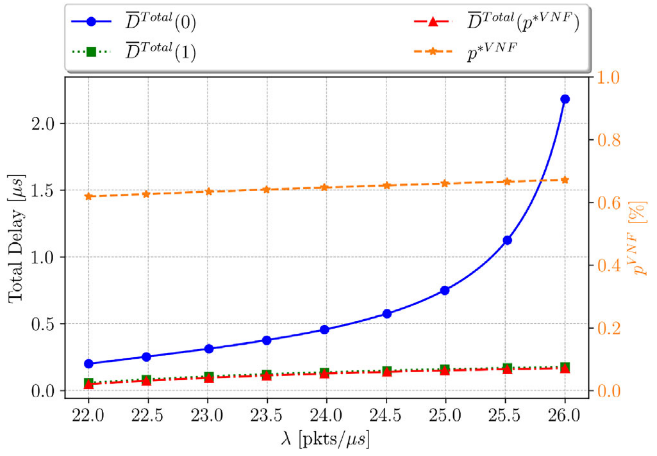

FIGURE 5 Impact of λ on average system delay.

This is understandable, considering the PNF's maximum service rate is capped at 20 pkts/μs, while the VNF can handle up to 80 pkts/μs. This also drives an increment in p^{*VNF} with a higher packet arrival rate. The implication is that an increased number of packets warrant offloading to the VNF to minimize average delay. As a corollary, the average delay under optimal offloading undergoes a slight rise. For a packet arrival rate λ of 25 pkts/μs, which aligns with our baseline parameter, employing optimal offloading can result in a delay reduction of approximately 79.96% (from around 0.750 μs to around 0.150 μs) vis-à-vis no offloading. In comparison to full offloading, the reduction stands at approximately 5.93% (from around 0.160 to 0.150 μs).

As the packet arrival rate further elevates to 26 pkts/μs, the benefits rendered by optimal offloading become even more pronounced. The delay reduction swells to an impressive 92.29% (from around 2.181 to around 0.168 μs) as an increased number of packets are channeled to the VNF. In scenarios employing full offloading, optimal offloading can carve down the delay by approximately 5.21% (from around 0.177 to 0.168 μs).

Interestingly, across the entire range of interest, from 22 to 26 pkts/μs, the average delays for both full offloading ($\overline{D}^{\text{Total}}(1)$) and optimal offloading ($\overline{D}^{\text{Total}}(p^{*VNF})$) are comparable. However, optimal offloading is slightly more advantageous. This is primarily attributed to the added propagation delay between the switch and VNF, a requisite for full offloading.

5.2.1 | Key insights from packet arrival rate analysis

From the examination of the system's response to varying packet arrival rates, several salient observations emerge:

- *Transition in system behavior*: The system displays a marked transition as the packet arrival rate, λ, shifts from 22 to 26 pkts/μs. Initially, while the system operates comfortably below saturation, the benefits of offloading become strikingly apparent as congestion intensifies towards the upper end of the examined range.
- *Consistent efficacy of optimal offloading*: Regardless of the specific packet arrival rate within the studied interval, the optimal offloading strategy invariably showcases its potential to boost system performance. This consistent efficacy underscores the strategy's inherent adaptability and relevance.
- *System saturation dynamics*: At a relatively relaxed packet arrival rate of $\lambda = 22$ pkts/μs, the system components function well beneath their saturation thresholds. Contrastingly, as the rate climbs to $\lambda = 26$ pkts/μs, the PNF displays evident strain, approaching its operational limits and thereby accentuating the necessity for judicious offloading.
- *Diverse benefits of offloading strategy*: The advantages proffered by the offloading strategy are multifaceted and evolve across the range. At the outset, with $\lambda = 22$ pkts/μs, the emphasis lies predominantly on realizing reduced average delays. However, as λ escalates to 26 pkts/μs, the strategy's role pivots, becoming indispensable for averting potential system overburdening.
- *User experience across the spectrum*: The choice of offloading strategy can profoundly influence user experience, especially for latency-critical applications. Whether operating at the modest end of the range with a primary focus on

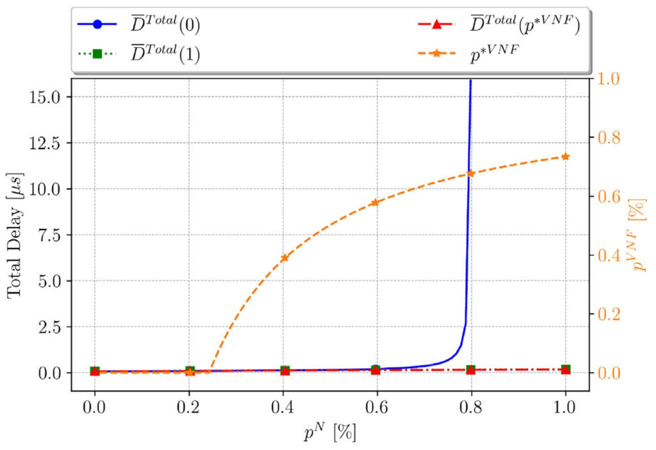

FIGURE 6 Impact of p^N on average system delay.

curtailing delays or navigating the busier segments where ensuring sustained system responsiveness becomes vital, the judicious selection of offloading strategy emerges as a cornerstone.

5.3 | Impact of p^N

Figure 6 delves into the relationship between the parameter p^N, spanning the valid range between 0 and 1, and the average delay. Intuitively, as p^N ascends, the average delay follows suit, primarily because a greater fraction of packets demands NF processing. With the PNF's capacity capped at 20 pkts/μs—a stark contrast to the VNF's superior 80 pkts/μs—the delay under the no offloading strategy surges more precipitously than the delay under full offloading as p^N elevates. This dynamic sets in motion a noticeable rise in the optimal p^{*VNF}, which transitions from 0 to a value of around 0.734. Such a shift underscores the premise that channeling more packets towards the VNF could serve to curb the average delay.

In the specific instance where $p^N = 0.79$, the delay observed with no offloading stands at approximately 2.679 μs. Introducing optimal offloading into this equation slashes the delay to a mere 0.154 μs—a dramatic dip by roughly 94.25%. This pronounced decrement is attributed to the optimal offloading strategy's adept utilization of the PNF's capacity, in sharp contrast to the no offloading approach that essentially sidesteps it. Conversely, juxtaposed against full offloading—registering a delay of about 0.164 μs—the delay under optimal offloading is pared down by around 5.88%.

Upon reaching $p^N = 0.8$, without offloading the system is thrust into a state of saturation. This situation is starkly delineated by the equation $\overline{D}^{PNF} = 1/(20 - 25 \times 0.8 \times 1) = \infty$, pointing to an overwhelming load on the PNF that surpasses its processing bandwidth. While one might be instinctively inclined to route all packets to the VNF to alleviate this strain, initiating the offloading process has a mitigating effect on the PNF's load. This staged offloading uncovers the PNF's latent capacity. By astutely offloading only a subset of the packets, the PNF can be maintained at a manageable load level, ensuring it furnishes acceptable delays. This interplay culminates in an optimal p^{*VNF} value hovering at around 0.676, debunking the notion that total offloading is invariably the superior choice.

An intriguing case emerges when $p^N = 0$, signifying that none of the packets mandates NF processing. In such a scenario, both $\overline{D}^{Total}(0)$ and $\overline{D}^{Total}(1)$ align perfectly at approximately 0.079 μs, rendering the choice of p^{*VNF} inconsequential since no packet is earmarked for NF processing.

5.3.1 | Key insights from analyzing the proportion of packets needing NF processing

From the examination of the system's response to varying probabilities of packets requiring NF processing, several observations can be derived:

- *Dichotomy between no offloading and full offloading*: As projected, a rising p^N sees the delay associated with no offloading ($\overline{D}^{Total}(0)$) amplify at a steeper trajectory than the delay under full offloading ($\overline{D}^{Total}(1)$). This trend underscores the PNF's capacity constraints vis-à-vis the VNF. The inflection point around $p^N = 0.8$ is particularly illuminating, where the no offloading paradigm registers a pronounced surge in delay, signaling a system saturation.
- *Strategic merits of optimal offloading*: The delay curve representative of optimal offloading ($\overline{D}^{Total}(p^{*VNF})$) elucidates the advantages inherent in judiciously calibrating offloading decisions predicated on packet-specific NF requirements. Around the pivotal $p^N = 0.8$ saturation threshold, a nuanced diversion of even a minor packet share to the VNF can substantially curtail delay magnitudes. This observation amplifies the essence of optimal load distribution between the PNF and VNF.
- *Unique behavior at $p^N = 0$*: With p^N grounded at zero, an absence of NF requisites for any packet emerges. This scenario bestows a uniform behavioral characteristic upon the system, rendering offloading decisions inconsequential. This homogeneity is captured by the congruence of delay curves for both no offloading and full offloading paradigms.
- *Deciphering optimal offloading fractions p^{*VNF}*: The trajectory of the p^{*VNF} curve provides a window into the optimal packet fractions warranting offloading to the VNF. As p^N forges ahead, a corresponding ascent in p^{*VNF} materializes, flagging an intensifying imperative for VNF-centric offloading. Yet, the observed inflection in proximity to $p^N = 0.8$ posits that a saturated PNF does not intuitively mandate full packet offloading to the VNF. Instead, offloading an astutely identified fraction maximizes efficiency, emphasizing the cardinality of dynamic decisions anchored in real-time system dynamics.

5.4 | Impact of C^{PNF}

For the parameter C^{PNF}, we assessed it over a range spanning from 20 to 40 pkts/μs. Our examination, as portrayed in Figure 7, yielded several noteworthy observations.

The average delay with no offloading, $\overline{D}^{Total}(0)$, exhibits a decline as C^{PNF} increases. This is attributable to the larger capacity of the PNF, enabling it to serve packets more efficiently. In contrast, the average delay with full offloading, $\overline{D}^{Total}(1)$, remains constant, underscoring the point that when all packets necessitating NF are sent to the VNF, the PNF capacity becomes inconsequential.

An intriguing dynamic is observed regarding offloading. As the capacity of the PNF escalates, the impetus for offloading to the VNF diminishes. In particular, as C^{PNF} heightens, a reduced inclination towards VNF offloading is observed in order to achieve minimal average delay.

When examining specific capacities, for $C^{PNF} = 20$ pkts/μs, about 65.98% of the packets are directed to the VNF. The average delay with optimal offloading is roughly 0.151 μs. This denotes a sizeable reduction of 80.06% from the 0.755 μs delay observed with no offloading. Furthermore, this also implies a reduction of 5.92% when compared to the 0.160 μs delay with full offloading.

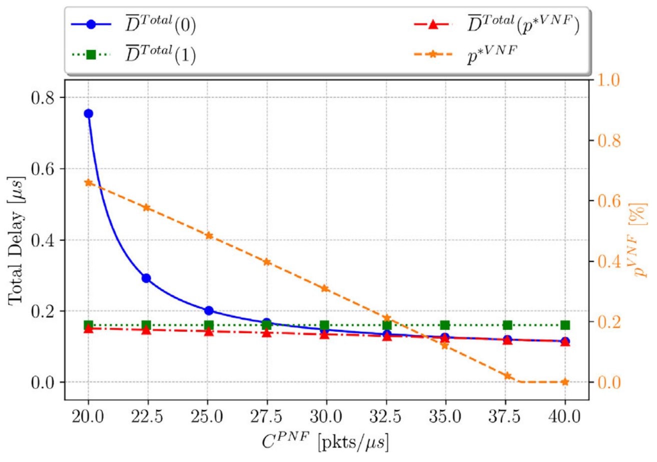

FIGURE 7 Impact of C^{PNF} on average system delay.

For a capacity of $C^{PNF} = 30$ pkts/μs, p^{*VNF} is estimated to be around 30.90%. With optimal offloading, the average delay diminishes to 0.134 μs, resulting in a decline of about 9.01% from the 0.148 μs delay without offloading. This is also 16.08% lower than the 0.160 μs delay experienced with full offloading.

A significant inflection point emerges as C^{PNF} nears 40 pkts/μs, pinpointing specifically at 37.98 pkts/μs. Here, p^{*VNF} is nearly zero at an estimated 0.005%. Delays with both optimal offloading and no offloading converge around 0.119 μs, presenting only a minuscule reduction of 0.002% for the optimal offloading scenario. In the meantime, the delay with full offloading consistently stands at 0.160 μs, revealing an optimal offloading advantage of about 34.98%.

Across the spectrum, the pronounced efficiency of optimal offloading remains unmistakable. As depicted in Figure 7, $\overline{D}^{Total}(p^{*VNF})$ consistently showcases the minimal delay for every evaluated C^{PNF} value. In scenarios, when C^{PNF} capacity dips below 20 pkts/μs, the benefits of optimal offloading become more pronounced. For instance, at $C^{PNF} = 19$ pkts/μs—a value not directly represented in Figure 7—the delay with optimal offloading stands at 0.152 μs. In relative terms, this translates to an astounding reduction of 97.67% from the 6.507 μs delay seen without offloading, and it surpasses the 0.160 μs delay from full offloading by an appreciable margin of approximately 5.07%.

5.4.1 | Key insights from analyzing the variation of PNF capacities

Drawing from the observations, there are several takeaways:

- *Significant dependency at lower capacities*: When C^{PNF} is below 20 pkts/μs, the system experiences pronounced constraints. The distinctions in delay between no offloading, full offloading, and optimal offloading become markedly evident. This underlines the significance of judicious offloading strategies. Notably, the system's inclination towards VNF peaks, achieving approximately 66% at $C^{PNF} = 20$ pkts/μs and intensifying further for reduced capacities.
- *System vulnerabilities at reduced capacities*: Endeavoring with C^{PNF} values under 20 pkts/μs presents substantial operational hurdles. There exists an inherent risk of the system reaching its limit, culminating in compromised service quality. Ensuring robust surveillance and fluid resource distribution is of essence. Additionally, the economic consequences of a heightened dependency on, possibly, cloud-integrated VNFs warrant thorough evaluation.
- *Unwavering full offloading delay*: Irrespective of the C^{PNF} value, the delay associated with full offloading remains invariant. This accentuates the notion that the efficacy of full offloading remains unaffected by C^{PNF} since all NF-requisite packets are directed towards NFV.
- *Inflection point for marginal offloading*: On the verge of C^{PNF} touching 38 pkts/μs, the utility of offloading starts to wane, with p^{*VNF} verging on nullity. Beyond this threshold, the PNF's processing prowess appears to suffice, diminishing the need for notable offloading.
- *Convergence of performance at elevated capacities*: With ascending C^{PNF} values, especially post the inflection point, the disparity in delay between the absence of offloading and its optimal state begins to contract. This insinuates that the system can uphold its operational efficacy with a diminishing dependency on VNFs.
- *Operational implications*: Navigating through the spectrum, comprehending these dynamics becomes instrumental for network administrators. In scenarios where PNF augmentation proves to be economically or logistically taxing, discerning an optimal offloading methodology gains precedence. Conversely, if elevating C^{PNF} beyond the inflection point is viable, the advantages of offloading start to recede, rendering it less pivotal.

5.5 | Impact of C^{VNF}

In our exploration, we assessed the parameter C^{VNF} over a range spanning from 20 to 140 pkts/μs, as depicted in Figure 8.

From Figure 8, we observe that employing solely PNF (thus no offloading) yields an average delay that remains invariant. This constancy arises because the system's average delay is impervious to variations in C^{VNF} under this configuration.

A scenario in which all packets are consistently directed to the VNF (full offloading) unveils an intriguing behavior. As one augments the capacity of the VNF, a threshold is eventually encountered. Beyond this juncture, additional

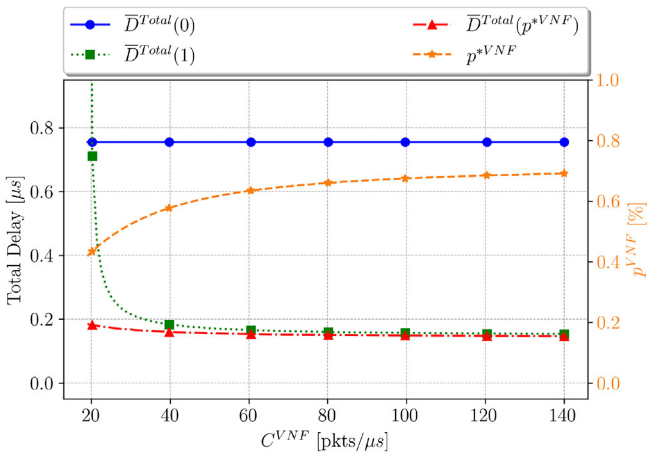

FIGURE 8 Impact of C^{VNF} on average system delay.

capacity enhancements cease to offer substantial reductions in the average delay, signaling that the system has reached a saturation point in terms of capacity utility.

An upward adjustment in the value of C^{VNF} naturally steers the system towards favoring packet direction to the VNF. A case in point is the notable escalation in the optimal offloading percentage, p^{*VNF}, from 43.13% at $C^{VNF} = 20$ pkts/μs to 63.33% at $C^{VNF} = 60$ pkts/μs. This substantial increment of approximately 20.2% accentuates the merits of a bolstered VNF capacity.

Beyond a C^{VNF} threshold of 40 pkts/μs, the system, when leveraging optimal offloading, manifests a commendable performance uptick. Specifically, the average delay experiences a contraction of approximately 78.79% when compared to no offloading (dwindling from 0.755 to 0.160 μs) and around 12.70% when compared to full offloading (from 0.183 to 0.160 μs).

The findings underscore the nuanced interplay between VNF capacity and offloading decisions, highlighting the pivotal role of capacity planning in ensuring optimal system performance.

5.5.1 | Key insights from analyzing VNF capacities

From the ensuing observations, several salient points emerge:

- *Operational efficiency*: There is a discernible decline in delay as C^{VNF} increases. This suggests that enhancing VNF capacity up to a certain threshold can impart considerable operational efficiency. This observation is pivotal for network operators contemplating capacity augmentation or investment in VNF technologies.
- *Threshold behavior and cost–benefit analysis*: As one augments the capacity of the VNF, a saturation point is eventually reached. Beyond this threshold, the marginal gains in terms of delay reduction become less pronounced. This inflection point is vital for financial evaluations, indicating a juncture where further capacity investments might not yield proportionate benefits.
- *Strategic offloading decisions*: The optimal offloading paradigm, evidenced by its consistent outperformance against both no offloading and full offloading strategies, accentuates the value of making judicious offloading decisions. This revelation underscores the necessity for intelligent traffic management systems in contemporary networks.
- *Scalability implications*: The observed performance enhancements with increased C^{VNF} imply that as network traffic intensifies, bolstering VNF capacity will be indispensable to preserving service quality and minimal latency.
- *Dynamic workload management*: The fluctuations in optimal p^{VNF} underscore the importance of adaptive strategies, especially in scenarios characterized by mutable workloads and variable traffic patterns. Such adaptability can be crucial to consistently achieving optimal performance.

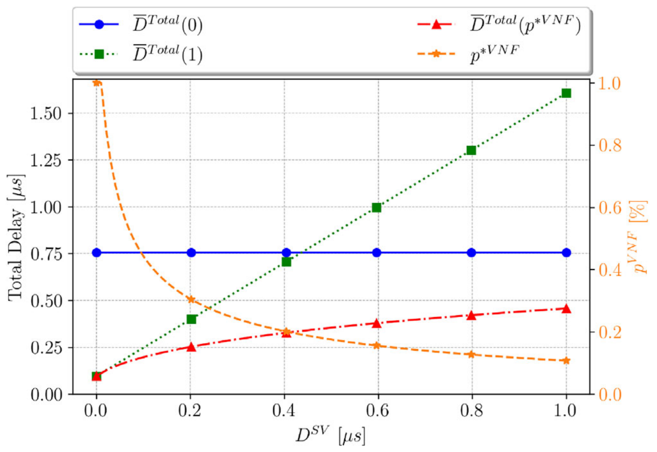

FIGURE 9 Impact of D^{SV} on average system delay.

5.6 | Impact of D^{SV}

We evaluate the influence of D^{SV} across a span from 0 to 1 μs. As illustrated in Figure 9, when the system relies exclusively on PNF ($\overline{D}^{Total}(0)$), the delay average remains consistent irrespective of variations in D^{SV}. This consistency stems from the fact that D^{SV}, representing the propagation delay between the P4 switch and VNF, lacks any consequential impact on the packets processed within PNF. Conversely, employing only VNF ($\overline{D}^{Total}(1)$) results in a direct linear increase in average delay as D^{SV} elevates. This trend underscores a declining inclination towards VNF usage as D^{SV} rises, culminating in a logarithmic decrement in p^{*VNF}.

For an instance where $D^{SV} = 0.2$ μs, a significant 30.42% of packets are routed towards VNF. With optimal offloading strategies in place, the average delay undergoes a stark reduction—about 66.50% when juxtaposed with no offloading (diminishing from approximately 0.755 to around 0.253 μs) and roughly 36.99% in contrast to full offloading (reducing from roughly 0.755 to close to 0.401 μs).

Embracing optimal offloading remains beneficial, even at elevated VNF propagation delays. Taking the case of $D^{SV} = 1.0$ μs, about 10.74% of packets are steered towards the VNF. Yet, having 89.26% of packets processed in the P4 switch alleviates its load, transitioning from a state of high congestion (where 100% of packets are handled) to a more moderate congestion. This shift induces a decrease in the average delay from 0.755 ($\overline{D}^{Total}(0)$) to 0.457 μs ($\overline{D}^{Total}(p^{*VNF})$), marking a decline by 39.45%. Contrarily, when set against full offloading, the average delay witnesses a substantial reduction of 71.55% (from 1.607 down to 0.457 μs).

5.6.1 | Key insights from analyzing VNF propagation delays

From the examination of the system's response to varying VNF propagation delays, several observations can be derived:

- *Threshold sensitivity*: As D^{SV} increases, there appears to be a threshold beyond which using VNF becomes less advantageous. Identifying and optimizing this threshold could be pivotal for system designers deciding on the balance between PNF and VNF.
- *Resource allocation*: The marked reduction in delay when utilizing optimal offloading underscores the significance of adaptive resource allocation. Systems capable of dynamically adjusting the packet distribution between PNF and VNF, especially in light of varying parameters like D^{SV}, will inherently exhibit enhanced efficiency.
- *Implications for scalability*: With network expansion and the addition of more devices or services, the propagation delay, D^{SV}, might naturally rise. This study accentuates the importance of accounting for such delays when strategizing for scalability, particularly when there's a heavy reliance on VNF.
- *Impact on user experience*: The pronounced disparities in delays, notably between no offloading and optimal offloading, could considerably influence user experiences, especially in applications demanding real-time

interactions. A delay reduction exceeding 66% (at $D^{SV} = 0.2\,\mu s$) can drastically transform a user's experience, potentially differentiating between a seamless video call and a disrupted one.

6 | DISCUSSION

The primary goal of our study was to enhance offloading from a P4 switch to NFV, given that VNF is equipped to handle intricate processing not may not be directly feasible with P4 switches. By finessing the offloading probability, our approach offers a more balanced system, leading to minimized delays and improved overall performance. To further grasp the dynamics of offloading decisions, a parametric sensitivity analysis was conducted across parameters such as $\lambda, p^N, C^{PNF}, C^{VNF}$, and D^{SV}. In the ensuing section, we encapsulate our findings, elucidating the merits and constraints of the showcased offloading technique.

6.1 | Summary of findings

Our baseline configuration hinted at an optimal offloading probability of 0.6598, implying that around 66% of packets should be directed towards VNF to strike the least delay, clocking in at a brisk 0.1505 μs. The subsequent parametric sensitivity analysis reiterated the robustness of the optimal offloading approach, with the synergy between PNF and VNF offloading proving beneficial across use cases.

While a cursory glance might suggest full packet offloading as a promising avenue to manage intricate NFs, a deeper dive reveals it might not be the most efficient strategy, especially when scaling to meet burgeoning network demands. On the opposite end, eschewing offloading entirely allows P4 switches to rapidly process packets but compromises the ability to deploy intricate NFs. Thus, an optimal offloading ratio stands out as a potent solution, offering a blend of rapid in-switch processing and sophisticated NF capabilities via NFV offloading.

Choosing this ideal offloading ratio is paramount to ensure system reliability and comprehensive performance. From our evaluations, a direct correlation emerges: as the delay of non-offloaded processes escalates, offloading to VNF should increase; inversely, as delays from complete offloading mount, offloading should be curtailed.

In line with fostering reproducibility and championing open science, our experimental scripts have been made publicly available.[†] We posit that such transparency not only deepens understanding but also sparks further exploration in the field.

6.2 | Significance and applications

Our research situates itself within the intricate landscape of network architectures, focusing specifically on delay analysis as it pertains to the offloading of traffic from P4 switches to VNF—a domain that remains uncharted in extant literature. The hallmark of our contribution lies in the unveiling of delay dynamics uniquely associated with this offloading process, set within the contours of our proposed system. While the fundamentals of delay analysis are based on established principles, the novelty lies in its application and in-depth exploration within the specific context of our proposed system. This positions our research as a novel addition to network management studies.

Beyond the scope of this present work, the presented offloading solution from programmable data planes to NFV can be helpful in various additional applications. Offloading P4-NF to VNF can be advantageous to keep operational costs low, for example, by selecting a low-end switch instead of a high-end, pricy model and compensating the computing disparity by offloading to NFV. Offloading can also be useful because of other missing resources, such as limited SRAM, TCAM, or other internal data structures.

Additionally, offloading from P4 to NFV can also be advantageous in addressing dependency challenges. If a complex NF (e.g., payload encryption) is forbidden to be run on a P4 switch,[8] execution can be moved to NFV, ensuring platform independence. Furthermore, in the sense that one P4 switch might be connected to an NFV server farm with

[†]https://github.com/FlowFrontiers/P4toNFV/.

tens or hundreds of servers, when there is too much traffic to be offloaded to NFV, scalability can be achieved by adding additional servers to maintain QoS and SLA.

Lastly, offloading can also help implement fault tolerance strategies to address switch failure problems[58] and network outages they eventually often cause.[59] Measures to ensure no traffic and service are compromised are critical to network availability strategies, where offloading can provide efficient resolution.[9]

6.3 | NFV to P4

This work presents a methodology and analytical evaluation focused on the probability of packets being offloaded to VNF, denoted as p^{VNF}, in the direction from the P4 switch to NFV. To examine offloading characteristics in the opposite direction, from NFV to the P4 switch, the formula can be inverted to focus on the probability of packets being offloaded to PNFs instead, which is represented as $p^{PNF} = 1 - p^{VNF}$. Therefore, our method provides a practical solution for analyzing offloading characteristics in both directions. Examining offloading from NFV to P4 can benefit architectures that improve VNF operation via a programmable data plane. This approach is particularly relevant in cloud and telecommunication networks, the primary application domains for NFV, which rely heavily on cost-effectiveness and service agility.[8]

6.4 | Limitations and future work

Our work, while rigorous and comprehensive, inevitably has areas that can be explored further. The following subsections shed light on the potential extensions and future avenues of our research, as well as pinpoint certain limitations inherent to the scope of our study.

6.4.1 | SDN controller

Our primary focus in the current manuscript is on the programmable data plane paradigm and its specific applications. Particularly, our research seeks to determine the optimal balance of offloading packets for NF processing between PNF and VNF, aiming to minimize latency. The role of the SDN controller, while instrumental, was not the focal point of this study.

In our system, every packet is initially processed by the P4 switch. It is then determined whether the packet requires NF processing and, if so, whether to process it locally (PNF) or offload it to the VNF. The decision to offload is guided by the offloading ratio p^{VNF}. Once this offloading decision is made, the packet physically leaves the P4 switch.

Conversely, when the switch is uncertain about the packet's required operation, it seeks guidance from the controller. In this situation, the packet also physically exits the P4 switch. However, this consultation is an autonomous process and remains distinct from the offloading decision to the VNF.

Given these operational intricacies, the offloading choice concerning the VNF p^{VNF} operates independently of the table-miss probability p^{C}. Notably, our analytical model is based on MM1 queuing system, it does not utilize a feedback mechanism to adjust p^{VNF} based on p^{C} conditions or outcomes.

However, investigating the role of the SDN controller is essential as we move forward. SDN controllers offer a centralized view of the network, enabling more informed and dynamic decision-making processes regarding traffic routing and resource allocation. This centralized approach can further augment the efficacy of our offloading strategies, potentially leading to even more optimized outcomes. Additionally, the SDN controller's capability to provide a programmable, adaptive, and responsive network infrastructure can complement the advantages of the programmable data plane, thus presenting a synergistic opportunity for future research.

The insights and methodologies developed in this paper lay the groundwork for more in-depth exploration into the integration and centrality of the SDN controller in subsequent studies. Our future endeavors will revolve around harnessing the combined power of both paradigms, programmable data planes and SDN, to achieve even more streamlined and efficient NF offloading strategies.

6.4.2 | Optimization decision dynamics in offloading

The primary emphasis of our research has been on modeling the offloading decisions, especially regarding the parameter p^{VNF}, in a static context. For each node and its associated variables, once the system parameters have been set, it remains fixed, not adapting to real-time fluctuations in network conditions. While this static approach ensures predictable and consistent system behavior, which is desirable under stable network conditions, it might not leverage the benefits of real-time adjustments in volatile environments.

In our impact analysis, we introduced controlled dynamism by altering individual parameters while keeping others unchanged. Although this does not fully emulate real-time adaptive mechanisms, it provides a deeper understanding of how the system responds to specific parameter variations. This analysis is pivotal in pinpointing potential performance bottlenecks and identifying avenues for enhancements.

Our current findings and methodologies pave the way for incorporating more dynamic adjustments in future models. By adopting a dynamic framework where parameters adjust in real-time to system states, we can harness benefits like increased responsiveness and system resilience.

Future endeavors will explore algorithms and mechanisms that accommodate such dynamism, factoring in the constraints of our M/M/1 queuing system which inherently lacks feedback mechanisms for real-time adjustments. While the promise of real-time adaptability is enticing, it's accompanied by challenges such as potential decision oscillations, elevated computational overhead, and the imperative for sophisticated monitoring infrastructures.

6.4.3 | Parameter selection and generalization

While the parameters selected for this research were meticulously chosen based on well-established networking norms and analogous hardware systems, it's important to note that the findings derived might not be universally generalizable. The parameters act as the foundation of the modeling and influence the overall behavior of the system. They are reflective of specific scenarios and are tailored to cater to the objectives of this study. As the networking ecosystem is vast and continually evolving, different setups, configurations, or use cases may yield diverse results that deviate from our findings.

One primary limitation to performing verification across an expansive range of parameter settings is the sheer complexity and variability within real-world networking systems. Each parameter modification can introduce a multitude of potential system states, exponentially increasing the analytical overhead. Ensuring that each of these states is adequately addressed and examined is a monumental task, often straining human analytical resources.

That said, addressing this challenge remains an important goal for our future endeavors. We're currently conceptualizing the development of a dedicated simulator. This tool will not only provide a graphical interface for easy manipulation of parameters but will also incorporate automated mechanisms, such as sliders, to streamline and expand the range of testing scenarios. Such advancements aim to enhance the comprehensiveness of our analysis and move closer to universally applicable findings.

6.4.4 | Integration of multiple P4 switches

Our current research primarily revolves around the integration of a single P4 switch with a single NFV. In subsequent investigations, we aspire to expand this model to incorporate systems that integrate multiple P4 switches with a single NFV. This expansion will allow for a comparative analysis against our current single-switch-single-NFV configuration, possibly revealing new dynamics and optimizations.

6.4.5 | Inclusion of heterogeneous traffic

The current study makes assumptions of homogeneous traffic, wherein only one type of traffic is considered. This simplification, while aiding in our initial modeling, might not accurately capture real-world complexities. In realistic SDN networks, traffic is a rich tapestry of diverse workloads with varying packet sizes. We are particularly interested in understanding the dynamics in an environment characterized by heterogeneous traffic, spanning both computation-bound and communication-bound varieties. This would provide a more encompassing representation of real-world traffic scenarios.

6.4.6 | Empirical validation through data plane measurements

While our research heavily leans on theoretical modeling, we recognize the irreplaceable value of empirical validation. In light of this, we plan on conducting actual data plane measurements. This empirical approach would juxtapose our theoretical findings against real-world observations, potentially refining our understanding and predictions.

6.4.7 | Technological examination of offloading

Beyond the purely analytical and empirical perspectives, we are setting our sights on the deeper technological intricacies of offloading. This exploration aims to shed light on the actual hardware and software mechanisms that underpin the offloading process, enhancing our holistic comprehension of the entire offloading paradigm.

7 | CONCLUSION

This research delved into the integration of a P4 switch with NFV, with an overarching objective to offset the inherent limitations of P4 switches by leveraging NFV for packet processing. To this end, we formulated a queuing model that scrutinizes the nuances of offloading packets from a P4 switch to NFV. Alongside, we articulated a formal optimization problem centered on the offloading ratio. The Bounded version of Brent's method was then employed to ascertain the optimal value with utmost efficiency. A comprehensive parametric sensitivity analysis was also undertaken, which provided invaluable insights into the influence of various parameters on the optimal offloading ratio and the associated average delay.

Drawing from our findings, we assert that the strategic offloading from a P4 switch to NFV presents pronounced benefits, chiefly manifested as a notable reduction in the system's average delay when juxtaposed with both non-offloading and full offloading scenarios. The methodology showcased in this study paves the way for discerning the optimal parameter configuration, effectively minimizing the packet processing delay during the offloading process from the P4 switch to NFV.

AUTHOR CONTRIBUTIONS

All authors contributed equally to each aspect of the work, including conceptualization, methodology, data collection, analysis, interpretation, writing, and revision of the manuscript. They all actively participated in discussions and decision-making throughout the research process.

ACKNOWLEDGMENTS

This work was supported by the János Bolyai Research Scholarship of the Hungarian Academy of Sciences and supported by the ÚNKP-23-5-BME-461 New National Excellence Program of the Ministry for Culture and Innovation from the source of the National Research, Development and Innovation Fund. The work presented in this paper was supported by project No. TKP2021-NVA-02. Project No. TKP2021-NVA-02 has been implemented with the support provided by the Ministry of Culture and Innovation of Hungary from the National Research, Development and Innovation Fund, financed under the TKP2021-NVA funding scheme.

CONFLICT OF INTEREST STATEMENT

The authors declare no potential conflict of interests.

DATA AVAILABILITY STATEMENT

Data sharing is not applicable to this article as no datasets were generated or analyzed during the current study.

ORCID

Adrian Pekar https://orcid.org/0000-0003-4511-8267
Yuan-Cheng Lai https://orcid.org/0000-0003-3695-5784
Winston Seah https://orcid.org/0000-0001-7147-5167

REFERENCES

1. Benson T, Akella A, Maltz D. Unraveling the complexity of network management. In: Proceedings of the 6th Usenix Symposium on Networked Systems Design and Implementation, NSDI'09. USENIX Association; 2009:335-348.
2. Nunes BAA, Mendonca M, Nguyen X-N, Obraczka K, Turletti T. A survey of software-defined networking: past, present, and future of programmable networks. *IEEE Commun Surv Tutor*. 2014;16(3):1617-1634.
3. Mijumbi R, Serrat J, Gorricho J-L, Bouten N, De Turck F, Boutaba R. Network function virtualization: state-of-the-art and research challenges. *IEEE Commun Surv Tutor*. 2016;18(1):236-262.
4. Bosshart P, Daly D, Gibb G, et al. P4: programming protocol-independent packet processors. *SIGCOMM Comput Commun Rev*. 2014; 44(3):87-95.
5. Kfoury EF, Crichigno J, Bou-Harb E. An exhaustive survey on P4 programmable data plane switches: taxonomy, applications, challenges, and future trends. *IEEE Access*. 2021;9:87094-87155.
6. Kfoury EF, Crichigno J, Bou-Harb E. Offloading media traffic to programmable data plane switches. In: ICC 2020—2020 IEEE International Conference on Communications (ICC). IEEE; 2020:1-7.
7. Pan T, Yu N, Jia C, et al. Sailfish: accelerating cloud-scale multi-tenant multi-service gateways with programmable switches. In: Proceedings of the 2021 ACM Sigcomm 2021 Conference, SIGCOMM '21. Association for Computing Machinery; 2021:194-206.
8. Chen X, Liu H, Zhang D, et al. Automatic performance-optimal offloading of network functions on programmable switches. *IEEE Trans Cloud Comput*. 2023;11(2):1591-1607.
9. Huang H, Wu W. HyperSFP: fault-tolerant service function chain provision on programmable switches in data centers. In: NOMS 2022-2022 IEEE/IFIP Network Operations and Management Symposium. IEEE; 2022:1-9.
10. Kianpisheh S, Taleb T. A survey on in-network computing: programmable data plane and technology specific applications. *IEEE Commun Surv Tutor*. 2023;25(1):701-761.
11. Cong Z, Baokang Z, Baosheng W, Yulei Y. CeUPF: offloading 5G user plane function to programmable hardware base on co-existence architecture. In: Proceedings of the 2021 ACM International Conference on Intelligent Computing and its Emerging Applications, ACM ICEA '21. Association for Computing Machinery; 2022:34-39.
12. Paolucci F, Scano D, Cugini F, et al. User plane function offloading in P4 switches for enhanced 5G mobile edge computing. In: 2021 17th International Conference on the Design of Reliable Communication Networks (DRCN). IEEE; 2021:1-3.
13. Intel. Intel Tofino Product Family Brochure. https://www.intel.com/content/dam/www/central-libraries/us/en/documents/tofino-product-family-brochure.pdf. Accessed: 2023-01-12; 2021.
14. Kim D, Liu Z, Zhu Y, Kim C, Lee J, Sekar V, Seshan S. TEA: enabling state-intensive network functions on programmable switches. In: Proceedings of the Annual Conference of the ACM Special Interest Group on Data Communication on the Applications, Technologies, Architectures, and Protocols for Computer Communication, SIGCOMM '20. Association for Computing Machinery; 2020:90-106.
15. Gebara N, Lerner A, Yang M, Yu M, Costa P, Ghobadi M. Challenging the stateless quo of programmable switches. In: Proceedings of the 19th ACM Workshop on Hot Topics in Networks, HotNets '20. Association for Computing Machinery; 2020:153-159.
16. Zeng C, Luo L, Zhang T, et al. Tiara: a scalable and efficient hardware acceleration architecture for stateful layer-4 load balancing. In: 19th Usenix Symposium on Networked Systems Design and Implementation (NSDI 22). USENIX Association; 2022:1345-1358.
17. Sharma NK, Kaufmann A, Anderson T, Kim C, Krishnamurthy A, Nelson J, Peter S. Evaluating the power of flexible packet processing for network resource allocation. In: Proceedings of the 14th Usenix Conference on Networked Systems Design and Implementation, NSDI'17. USENIX Association; 2017:67-82.
18. Yuan Y, Alama O, Fei J, et al. Unlocking the power of inline floating-point operations on programmable switches. In: 19th Usenix Symposium on Networked Systems Design and Implementation (NSDI 22). USENIX Association; 2022:683-700.
19. Cui P, Pan H, Li Z, et al. NetFC: enabling accurate floating-point arithmetic on programmable switches. In: 2021 IEEE 29th International Conference on Network Protocols (ICNP). IEEE; 2021:1-11.
20. Lao C, Le Y, Mahajan K, Chen Y, Wu W, Akella A, Swift M. ATP: in-network aggregation for multi-tenant learning. In: 18th Usenix Symposium on Networked Systems Design and Implementation (NSDI 21). USENIX Association; 2021:741-761.
21. Sapio A, Canini M, Ho C-Y, et al. Scaling distributed machine learning with in-network aggregation. In: 18th Usenix Symposium on Networked Systems Design and Implementation (NSDI 21). USENIX Association; 2021:785-808.
22. Gupta A, Harrison R, Canini M, Feamster N, Rexford J, Willinger W. Sonata: query-driven streaming network telemetry. In: Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication, SIGCOMM '18. Association for Computing Machinery; 2018:357-371.
23. Anan M, Al-Fuqaha A, Nasser N, Mu T-Y, Bustam H. Empowering networking research and experimentation through software-defined networking. *J Netw Comput Appl*. 2016;70:140-155.
24. Hamdan M, Hassan E, Abdelaziz A, et al. A comprehensive survey of load balancing techniques in software-defined network. *J Netw Comput Appl*. 2021;174:102856.

25. Bronstein Z, Roch E, Xia J, Molkho A. Uniform handling and abstraction of NFV hardware accelerators. *IEEE Netw*. 2015;29(3):22-29.
26. Niu Z, Xu H, Liu L, Tian Y, Wang P, Li Z. Unveiling performance of NFV software dataplanes. In: Proceedings of the 2nd Workshop on Cloud-Assisted Networking, CAN '17. Association for Computing Machinery; 2017:13-18.
27. He M, Basta A, Blenk A, Deric N, Kellerer W. P4NFV: an NFV architecture with flexible data plane reconfiguration. In: 2018 14th International Conference on Network and Service Management (CNSM). IEEE; 2018:90-98.
28. The P4 Language Consortium. P4$_{16}$ Language Specification v1.2.3. https://p4.org/p4-spec/docs/P4-16-v-1.2.3.html. Accessed: 2022-10-18; 2022.
29. Peterson LL, Cascone C, O'Connor B, Vachuska T, Davie B. *Software-Defined Networks: A Systems Approach*: Systems Approach LLC; 2020;154.
30. Jarschel M, Oechsner S, Schlosser D, Pries R, Goll S, Tran-Gia P. Modeling and performance evaluation of an OpenFlow architecture. In: 2011 23rd International Teletraffic Congress (ITC). IEEE; 2011:1-7.
31. Azodolmolky S, Wieder P, Yahyapour R. Performance evaluation of a scalable software-defined networking deployment. In: 2013 Second European Workshop on Software Defined Networks. IEEE; 2013:68-74.
32. Bozakov Z, Rizk A. Taming SDN controllers in heterogeneous hardware environments. In: 2013 Second European Workshop on Software Defined Networks. IEEE; 2013:50-55.
33. Mahmood K, Chilwan A, Østerbø ON, Jarschel M. On the modeling of OpenFlow-based SDNs: the single node case. In: Proceedings of Computer Science and Information Technology (CS & IT), Vol. 4; 2014:207-214.
34. Jarschel M, Østerbø O, Chilwan A, Mahmood K. Modelling of OpenFlow-based software-defined networks: the multiple node case. *IET Netw*. 2015;4(5):278-284.
35. Miao W, Min G, Wu Y, Wang H. Performance modelling of preemption-based packet scheduling for data plane in software defined networks. In: 2015 IEEE International Conference on Smart City/Socialcom/Sustaincom (Smartcity). IEEE; 2015:60-65.
36. Shang Z, Wolter K. Delay evaluation of OpenFlow network based on queueing model. arXiv:1608.06491v1; 2016.
37. Sood K, Yu S, Xiang Y. Performance analysis of software-defined network switch using $M/Geo/1$ model. *IEEE Commun Lett*. 2016; 20(12):2522-2525.
38. Miao W, Min G, Wu Y, Wang H, Hu J. Performance modelling and analysis of software-defined networking under bursty multimedia traffic. *ACM Trans Mult Comput Commun Appl*. 2016;12(5s):77.
39. 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.
40. Goto Y, Masuyama H, Ng B, Seah WKG, Takahashi Y. 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). IEEE; 2016:301-306.
41. Fahmin A, Lai Y-C, Hossain MS, Lin Y-D. Performance modeling and comparison of NFV integrated with SDN: under or aside? *J Netw Comput Appl*. 2018;113:119-129.
42. Singh D, Ng B, Lai Y-C, Lin Y-D, Seah WKG. Modelling software-defined networking: software and hardware switches. *J Netw Comput Appl*. 2018;122:24-36.
43. Zhao J, Hu Z, Xiong B, Yang L, Li K. Modeling and optimization of packet forwarding performance in software-defined WAN. *Future Gener Comput Syst*. 2020;106:412-425.
44. Singh D, Ng B, Lai Y-C, Lin Y-D, Seah WKG. Full encapsulation or internal buffering in OpenFlow based hardware switches? *Comput Netw*. 2020;167:107033.
45. Walrand J, Varaiya P. Sojourn times and the overtaking condition in Jacksonian networks. *Adv Appl Probab*. 1980;12(4):1000-1018.
46. Goodman JB, Massey WA. The non-ergodic Jackson network. *J Appl Probab*. 1984;21(4):860-869.
47. Boudec J-YL, Thiran P. *Network Calculus: A Theory of Deterministic Queuing Systems for the Internet*, Lecture Notes in Computer Science, vol. 2050: Springer; 2001.
48. Shahbaz M, Choi S, Pfaff B, Kim C, Feamster N, McKeown N, Rexford J. PISCES: a programmable, protocol-independent software switch. In: Proceedings of the 2016 ACM Sigcomm Conference, SIGCOMM '16. Association for Computing Machinery; 2016:525-538.
49. Wang H, Soulé R, Dang HT, Lee KS, Shrivastav V, Foster N, Weatherspoon H. P4FPGA: a rapid prototyping framework for P4. In: Proceedings of the Symposium on SDN Research, SOSR '17. Association for Computing Machinery; 2017:122-135.
50. Dang HT, Wang H, Jepsen T, et al. Whippersnapper: a P4 language benchmark suite. In: Proceedings of the Symposium on SDN Research, SOSR '17. Association for Computing Machinery; 2017:95-101.
51. Shen W, Yoshida M, Minato K, Imajuku W. vConductor: an enabler for achieving virtual network integration as a service. *IEEE Commun Mag*. 2015;53(2):116-124.
52. Lombardo A, Manzalini A, Schembra G, Faraci G, Rametta C, Riccobene V. An open framework to enable NetFATE (Network Functions at the edge). In: Proceedings of the 2015 1st IEEE Conference on Network Softwarization (NETSOFT). IEEE; 2015:1-6.
53. Callegati F, Cerroni W, Contoli C, Santandrea G. Dynamic chaining of virtual network functions in cloud-based edge networks. In: Proceedings of the 2015 1st IEEE Conference on Network Softwarization (NETSOFT). IEEE; 2015:1-5.
54. Hancock D, van der Merwe J. HyPer4: Using P4 to virtualize the programmable data plane. In: Proceedings of the 12th international on conference on emerging networking experiments and technologies, CoNEXT '16. Association for Computing Machinery; 2016:35-49.
55. Zhang C, Bi J, Zhou Y, Dogar AB, Wu J. HyperV: a high performance hypervisor for virtualization of the programmable data plane. In: 2017 26th International Conference on Computer Communication and Networks (ICCCN). IEEE; 2017:1-9.

56. Saquetti M, Bueno G, Cordeiro W, Azambuja JR. P4VBox: enabling P4-based switch virtualization. *IEEE Commun Lett*. 2020;24(1): 146-149.
57. Press WH, Teukolsky SA, Vetterling WT, Flannery BP. Van Wijngaarden–Dekker–Brent method. *Numerical Recipes in FORTRAN: The Art of Scientific Computing*. 2nd ed.: Cambridge University Press; 1992:352-355.
58. Singh R, Mukhtar M, Krishna A, Parkhi A, Padhye J, Maltz D. Surviving switch failures in cloud datacenters. *SIGCOMM Comput Commun Rev*. 2021;51(2):2-9.
59. Meza J, Xu T, Veeraraghavan K, Mutlu O. A large scale study of data center network reliability. In: Proceedings of the Internet Measurement Conference 2018, IMC '18. Association for Computing Machinery; 2018:393-407.

How to cite this article: Pekar A, Makara LA, Lai Y-C, Lin Y-D, Seah W. P4toNFV: Offloading from P4 switches to NFV in programmable data planes. *Int J Commun Syst*. 2023;e5687. doi:10.1002/dac.5687