Towards Performance-Aware Management of P4-based Cloud Environments

Hasanin Harkous*, Bassel Aboul Hosn†, Mu He*, Michael Jarschel*, Rastin Pries*, Wolfgang Kellerer†

**Nokia*, firstname.lastname@nokia.com
†*Technical University of Munich*, firstname.lastname@tum.de
Munich, Germany

***Abstract*—The recent trend to deploy programmable packet processors in cloud environments enhances the packet processing capability without losing the flexibility to adapt the functions at runtime. However, managing network functions, in particular deciding where to instantiate a certain function, is a non-trivial task with many decisive factors. In this paper, we propose a mathematical model for optimizing the placement of network functions implemented in P4, considering the various types of devices with different properties in terms of processing delay and supported external functions and architectures. To model the processing delay, each network function is decomposed to a set of atomic constructs, whose latency has been properly measured. The numerical evaluation considering five types of network functions shows the effectiveness of the optimization model in selecting the number of devices to be used and in minimizing the overall packet forwarding delay and costs.**

***Index Terms*—P4, Optimization, Placement, Programmable Data Plane.**

I. Introduction

Packet processors with programmable data planes enable customization of the packet forwarding pipeline, and thus, provide high flexibility in designing networks. The P4 programming language [1] is used for programming different types of programmable packet processors such as CPU, NPU, FPGA, and ASIC-based devices, where the same P4 program could be used to program different devices. In a cloud environment with heterogeneous P4-based packet processors, it is important to decide on the optimal placement of Network Functions (NFs) into different devices while ensuring an optimal performance level in terms of forwarding delay.

While most solutions in literature deal with NF placement on commodity servers, P4NFV [2] and Smartchain [3] consider P4-based devices in the problem formulation. They focus on finding the optimal NFs placement between SmartNIC and the host, but they do not consider a heterogeneous environment of different classes of P4 programmable devices or the placement on more than one device. Flightplan [4] provides a way to place P4 programs but based on coarse P4 segments.

In this work, we study the optimal NF placement in cloud environments consisting of different P4 devices. The planning targets the optimal placement decision of NFs while minimizing the overall delay and cost. The approach considers the characteristics of the state-of-the-art P4 devices at the granularity of the execution of different P4 atomic constructs. Each NF is decomposed into a set of atomic P4 constructs to find the matching P4 device that can support running that NF, and provide the best performance.

Section II describes the problem formulation. In Section III, the model parameters are surveyed, and the evaluation results are presented. Section IV concludes the work.

II. Problem Formulation

This section formulates the placement problem with the decision variables, constraints, and objective function.

A. Device Capabilities

P4's target-independence feature enables the possibility to run the same P4 program on different types of P4 devices when a compatible P4 architecture is supported. The P4 architecture defines the programmable blocks in a P4 device, as well as the supported externs, i.e., additional functionalities supported by the device which can be called within a P4 program like an encryption function. Let $\mathcal{D}$, $\mathcal{A}$, and $\mathcal{E}$ be the sets of all available types of P4 packet processors/devices, supported P4 architectures, and available P4 externs in a cloud environment respectively. The set $\mathcal{L}_d$ contains all instances of each P4 device of type $d \in \mathcal{D}$. For every P4 device of type $d \in \mathcal{D}$, the following device characteristics are considered:

- T_d represents the maximum supported throughput on device d, i.e., its line rate.
- δ_d^{Bp} represents the base processing delay on a P4 device of type d, which includes the delay of the non-programmable blocks and the header parsing operations.
- δ_d^c represents the processing delay of a P4 construct c on device d.
- δ_d^e represents the processing delay of a P4 extern function e on device d.
- ω_d represents the processing resources of P4 device d whose definition depends on the type of the processing platform. For example, while processing resources are expressed in terms of available Look-up-tables in the case of FPGA, it is expressed in terms of the number of stages in ASIC devices. As a common ground, we decide to quantify the processing resources of a P4 device as the maximum number of P4 constructs that could run simultaneously on that device.
- C_d represents the cost (CAPEX) of a P4 device d.

- $Arch_d^a$ is a Boolean variable equal to 1 if device d supports architecture $a \in \mathcal{A}$, and equal to 0 otherwise.
- Ext_d^e is a Boolean variable equal to 1 if device d supports extern $e \in \mathcal{E}$, and equal to 0 otherwise.
- T_d^e denotes the maximum supported throughput when running extern e on device d.

B. Network Function Requirements

$\mathcal{F}$ denotes the set of NFs to be placed. Note that certain functionalities, such as the Layer 3 Forwarding function, need to exist on every used packet processor to guarantee proper packet forwarding between devices. These required functions belong to $\mathcal{F}_{req}$. We denote the union of the two sets by $\mathcal{F}_{tot} = \mathcal{F} \cup \mathcal{F}_{req}$.

P4 syntax consists of a basic set of P4 constructs/operations, denoted by $\mathcal{C}$. Any NF $f \in \mathcal{F}_{tot}$ can be written as a combination of these P4 constructs. The set of P4 constructs used to build an arbitrary NF f is denoted by $\mathcal{C}_f$, where $\mathcal{C}_f \subset \mathcal{C}$. The following parameters summarize the requirements associated with any NF f:

- σ_f^c represents the number of occurrences of each construct c in $\mathcal{C}_f$ needed to build f.
- ω_f represents the total number of P4 constructs required to describe NF f, i.e., $\omega_f = \sum_{c \in C_f} \sigma_c^f$.
- T_f represents the expected throughput that should be processed by NF f.
- $Arch_f^a$ is a Boolean variable equal to 1 if NF f is compatible with P4 architecture $a \in \mathcal{A}$, and equal to 0 otherwise.
- Ext_f^e is a Boolean variable equal to 1 if NF f requires extern $e \in \mathcal{E}$, and equal to 0 otherwise.

C. Objective Function

The expected delay of running an arbitrary NF f on device d, denoted as Δ_d^f, can be calculated as the summation of the base processing delay, the delay of processing different P4 constructs, and the delay for executing extern functions:

$$\Delta_d^f = \delta_d^{BP} + \sum_{c \in C_f} \sigma_c^f \delta_d^c + \sum_{e \in E | Ext_f^e = 1} \delta_d^e \tag{1}$$

The objective is to minimize the overall packet forwarding delay $\mathbb{D}$ in the considered environment, as well as the overall costs $\mathbb{C}$, each expressed as:

$$\mathbb{D} = \sum_{f \in \mathcal{F}_{tot}} \sum_{d \in \mathcal{D}} \sum_{l \in \mathcal{L}_d} \alpha_{dl}^f \times \Delta_d^f \tag{2}$$

$$\mathbb{C} = \sum_{d \in \mathcal{D}} \sum_{l \in \mathcal{L}_d} x_{dl} \times Cost_d \tag{3}$$

where the boolean decision variable α_{dl}^f is equal to one when NF f is placed on instance l of the P4 device of type d, and the dependent variable x_{dl} indicates whether the instance l of device type d is being used:

$$x_{dl} = \begin{cases} 1 & \text{if } \sum_{f \in \mathcal{F}_{tot}} \alpha_{dl}^f \geq 1 \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

The overall objective function is to minimize the weighted sum of the two metrics $\mathbb{D}$ and $\mathbb{C}$ as shown in Eq. (5). As the two considered metrics have different ranges and units (μs and dollars), they have to be normalized according to the maximum value recorded for each metric. This is done to ensure that both objectives affect the placement decision equally without one objective overshadowing the other. The weights μ and ε are used to tune the relative importance of each metric.

$$Minimize\ (\mu\mathbb{D} + \varepsilon\mathbb{C}) \tag{5}$$

D. Constraints

First, each NF f in $\mathcal{F}$ should be placed just once on any P4 device, as expressed in Eq. (6):

$$\sum_{d \in \mathcal{D}} \sum_{l \in \mathcal{L}_d} \alpha_{dl}^f = 1 \quad \forall f \in \mathcal{F} \tag{6}$$

As mentioned earlier, certain functions (e.g. L3 Forwarding) in $\mathcal{F}_{req}$ should be placed on every used device instance. This requirement can be achieved by the constraint shown in Eq. (7):

$$\alpha_{dl}^f = x_{dl} \quad \forall d \in \mathcal{D}\ \forall l \in \mathcal{L}_d\ \forall f \in \mathcal{F}_{req} \tag{7}$$

We ensure that a NF f can be hosted on a device of type d only if the device has a compatible architecture and supports all extern functions required by f. In Eq. (8), the decision variable α_{dl}^f is forced to be zero in case the architecture required by the NF and that supported by the P4 device do not match. On the other hand, Eq. (9) makes sure that α_{dl}^f can be equal to 1 only if all externs required by NF f are available on device d. Note that N is a big number larger than the maximum number of externs required by any NF.

$$\alpha_{dl}^f \leq \sum_{a \in \mathcal{A}} Arch_f^a \times Arch_d^a \quad \forall f \in \mathcal{F}_{tot}\ \forall d \in \mathcal{D}\ \forall l \in \mathcal{L}_d \tag{8}$$

$$\sum_{e \in \mathcal{E}} Ext_f^e \leq \sum_{e \in \mathcal{E}} Ext_f^e \times Ext_d^e + N(1 - \alpha_{dl}^f) \quad \forall f \in \mathcal{F}_{tot}\ \forall d \in \mathcal{D}\ \forall l \in \mathcal{L}_d \tag{9}$$

Eq. (10) ensures that the required processing resources of all NFs to be placed on any device instance of type d never exceed the limited processing capacity of that device.

$$\sum_{f \in \mathcal{F}_{tot} | \alpha_{dl}^f = 1} \omega_f \leq \omega_d \quad \forall d \in \mathcal{D}\ \forall l \in \mathcal{L}_d \tag{10}$$

On the other hand, Eq. (11) ensures that the cumulative throughput required by different NFs to be placed on a device never exceeds the limited throughput of that device. Similarly, Eq. (12) ensures that the cumulative throughput of NFs that require offloading some functionalities to an extern on a device never exceeds the limited throughput of that extern.

$$\sum_{f \in \mathcal{F}_{tot} | \alpha_{dl}^f = 1} T_f \leq T_d \quad \forall d \in \mathcal{D}\ \forall l \in \mathcal{L}_d \tag{11}$$

$$\sum_{f \in \mathcal{F}_{tot} | \alpha_{dl}^f = 1} Ext_f^e \times T_f \leq T_d^e \quad \forall e \in \mathcal{E} \forall d \in \mathcal{D}\ \forall l \in \mathcal{L}_d \tag{12}$$

A total cost limit, denoted by $MaxCost$, is added to ensure that the total cost of used devices according to the optimal placement never exceeds the predefined limit as shown in Eq. (13):

$$\sum_{d\in\mathcal{D}}\sum_{l\in\mathcal{L}_d} x_{dl}C_d \leq MaxCost \tag{13}$$

To set the dependent variable x_{dl} to 1 when at least one NF is placed on instance l of a device of type d, the following two constraints are introduced, where M is a constant number larger than the number of NF instances in the network.

$$x_{dl} \leq \sum_{f\in\mathcal{F}_{tot}} \alpha_{dl}^{f} \quad \forall d\in\mathcal{D}\ \forall l\in\mathcal{L}_d \tag{14}$$

$$\sum_{f\in\mathcal{F}_{tot}} \alpha_{dl}^{f} \leq Mx_{dl} \quad \forall d\in\mathcal{D}\ \forall l\in\mathcal{L}_d \tag{15}$$

III. Evaluation

To evaluate the proposed model, we select a scenario with a realistic set of parameters surveyed from the literature.

A. Surveying NF and P4 Devices' Parameters

We consider five different NFs with different complexities: *(i)* Layer 2 Forwarding (L2Fwd), *(ii)* Layer 3 Forwarding (L3Fwd), *(iii)* Firewall (FW), *(iv)* VxLAN Decapsulation (Decap), *(v)* Load Balancer (LB). The L3Fwd function belongs to the set of functions required to be in every functioning device to ensure proper packet forwarding between different packet processors. Only the LB NF requires an external hashing function. All NFs are compatible with the three common P4 architectures: V1model (V1M), SimpleSumeSwitch (SUME), and Portable Switch Architecture (PSA). It is assumed that all NFs are required to process 1 Gbps traffic. Each NF is decomposed to its atomic P4 constructs with the methodology described in [5]. For example, L2Fwd NF requires parsing and modifying a single header (Ethernet), while L3Fwd requires the same operations for both Ethernet and IPv4 headers. The different parameters and requirements corresponding to the different considered NFs are summarized in Table I.

We select four types of P4 programmable packet processors: CPU, NPU, FPGA, and ASIC-based processing platforms. The different criteria related to P4 devices belonging to these classes of packet processors are shown in Table II. The throughput and per P4 construct latency is selected based on our previous work [5], [6] and another literature [7]. The performance related to the extern hash function is extracted from [8] for packet size equal to 64 Bytes as the Load Balancer function requires calculating the hash function of some headers only. The cost and the maximum number of supported constructs are based on the comparative analysis provided in [7] and [9]. Note that since no work in literature gives a quantitative evaluation of the maximum number of supported constructs on different devices, we assume some reasonable numbers that follow the order of devices in terms of available resources. The cost for CPU-based switches is assumed to be approximately equal to the price of 2 cores, as typical P4-based software switches run with this minimum requirement. As there is no evaluation in the literature that quantifies the variation of latency on ASIC devices as a function of P4 constructs and P4 externs, it is assumed that the ASIC-based devices are powerful enough to run any P4 program without latency variation and to execute hashing function with performance higher than the other device types.

Table I: Surveyed parameters of different Network Functions.

	L2Fwd	**L3Fwd**	**FW**	**Decap**	**LB**
Required	no	yes	no	no	no
Throughput	1 Gbps	1 Gbps	1 Gbps	1 Gbps	1 Gbps
Compatible P4 Arch.	V1M,PSA, SUME	V1M,PSA, SUME	V1M,PSA, SUME	V1M,PSA, SUME	V1M,PSA, SUME
Externs	none	none	none	none	SipHash-2-4
# Const.					
Parse Hdr	1	2	3	7	3
Modify Hdr	1	2	0	0	0
Copy Hdr	0	0	0	3	0
Remove Hdr	0	0	0	4	0
Add Hdr	0	0	0	0	0
Add Table	0	0	0	1	0

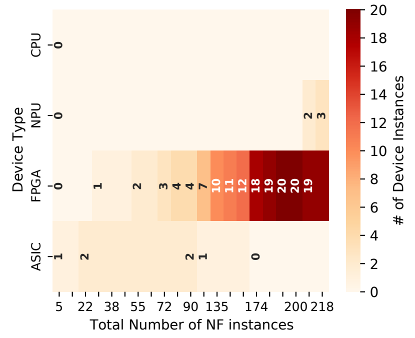
B. Evaluation Results

The objective is to find the optimal placement of NFs as well as the set of P4 devices to be used to suffice a given set of workloads as input. This workload is varied as multiples of the previously defined NFs in Table I from 1 to around 200 NFs. The optimization problem is solved using the widely used Gurobi solver [10]. Four scenarios are considered in this evaluation where μ and ε in Eq. (5) are varied: *(Scen. 1)* with $\mu = 1$ and $\varepsilon = 0$ so that the objective is to achieve the best performance disregarding cost; *(Scen. 2)* with $\mu = 0$ and $\varepsilon = 1$ so that the objective is to provide the cheapest solution where best effort performance is tolerable; *(Scen. 3)* with $\mu = \varepsilon = 0.5$ so that the objective is to find a balanced solution where both performance and costs are equally weighted; *(Scen. 4)* with $\mu = 1$ and $\varepsilon = 0$ but with predefined constraint on the costs not to exceed \$100k.

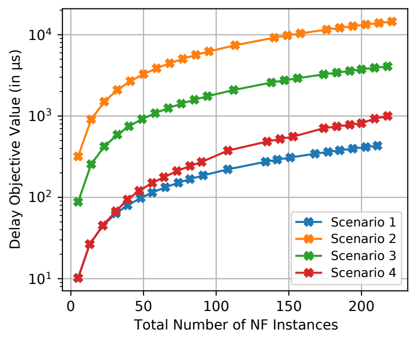
The optimal solution for *Scen. 1* is trivial where all NFs are placed on ASIC-based devices as these are the most performant devices, where 15 devices are needed in this case to place all the NFs. On the other hand, the cheapest devices, which are CPU-based, are selected in Scen. 2 to place all NFs when the objective is only minimizing costs, where 25 CPU-based devices are needed to place all NFs. In *Scen. 3*, NPU-based devices accomplish the best trade-off between

Table II: Surveyed parameters of different types of P4 devices.

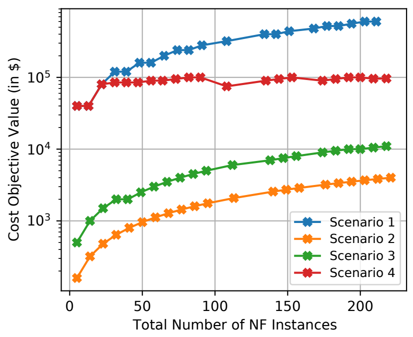
Device / **Param.**	**CPU**	**NPU**	**FPGA**	**ASIC**
P4 Arch.	V1M	V1M	SUME	PSA
Throughput (Gbps)	9	10	10	100
Max. Constructs	1000	500	250	100
Cost (\$)	160	500	5000	40000
Extern (SipHash-2-4)				
Present	yes	yes	yes	yes
Throughput (Gbps)	3.3	7.6	4.2	20
Latency (μs)	90	40	0.5	0.2
Per P4 Construct Latency (μs) [5]				
Base + Parse Header	≈ 45.9	≈ 7.8	≈ 3.9	≈ 2
Modify Header	0	0.5	0	0
Copy Header	0	0.28	0	0
Remove Header	-0.29	1.43	-0.02	0
Add Header	0.4	1.59	0.23	0
Add Table	0.08	0.37	0.13	0



(a) Devices utilization in Scenario 4.



(b) Total delay for all scenarios.



(c) Total cost for all scenarios.

Figure 1: Results corresponding to the optimal placement for different scenarios.

performance and costs, where 22 NPU devices are needed to place all NFs. More interesting is *Scen. 4*, where a cost limit of \$100k is defined. Fig. 1a shows the placement result with the number of instances of each device type needed for an increasing number of NFs. It could be observed that for low workload, one ASIC device is selected as it provides the best performance while still being affordable given the limited cost limit. Then, another ASIC device is utilized when up to 22 NFs need to be placed, where the second device is needed because the processing resources constraint is reached on the first device. After this point, no more ASIC devices could be used as the remaining \$20k budget allows only for the second-best performant device, i.e., FPGA. In this case, up to 4 FPGA devices are needed to process the additional NFs until reaching 90 NFs. After this point, one ASIC device is sacrificed to enable affording more FPGA devices that could support running the higher workload. At 174 NFs, the second ASIC device is also substituted with more FPGAs until reaching 20 FPGAs when the number of NFs reaches 200 NFs. After this point, the optimal solution further sacrifices performance by substituting FPGA devices with the next best performant device, i.e., NPUs so that it can support processing all NFs with the provided budget.

Fig. 1b and 1c show the delay and cost functions of the optimal solution for different scenarios as a function of NFs to be placed. As expected, the overall delay in *Scen. 1* is minimal, while the overall cost in *Scen. 2* is the lowest. Results corresponding to *Scen. 3* show the trade-off between the two objectives, where the overall delay and cost are equally minimized. The results of *Scen. 4* show that the delay of the system is as low as that of *Scen. 1* (when only the delay is optimized), until the budget constraint is reached after 22 NFs. After this point, the delay of the system starts increasing compared to *Scen. 1*, while the cost is always less than the preset budget of \$100k.

IV. Conclusion

In this work, the placement problem of P4 NFs in a cloud environment is studied. The placement decision targets optimizing the forwarding delay in the system as well as the cost. To guarantee the highest possible processing performance, we build the problem formulation on top of prior art that maps the atomic P4 operations to the corresponding processing delay on different P4 devices. We conduct numerical evaluations to showcase how the provided mathematical formulation can assist the planning of P4-based cloud environments with minimal forwarding delay and costs.

In future work, the model could be extended by considering service function chains embedding that could be distributed across different devices and subject to latency and cost constraints. An experimental testbed could also be built to validate the effectiveness of the proposed optimization model. Moreover, the runtime management of the P4 environment with dynamic workload requests could be further investigated.

References

[1] P4 Language Consortium, "P4 16 Language Specification," https://p4.org/p4-spec/docs/P4-16-v1.0.0-spec.html, 2016.

[2] A. Mohammadkhan, S. Panda, S. G. Kulkarni, K. K. Ramakrishnan, and L. N. Bhuyan, "P4NFV: P4 enabled NFV systems with SmartNICs," in *2019 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)*, 2019, pp. 1–7.

[3] S. Wang, Z. Meng, C. Sun, M. Wang, M. Xu, J. Bi, T. Yang, Q. Huang, and H. Hu, "SmartChain: Enabling high-performance service chain partition between SmartNIC and CPU," in *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*, 2020, pp. 1–7.

[4] S. Nik, S. John, G. Hans, P. Isaac, and H. Zhaoyang, "Flightplan: Dataplane disaggregation and placement for P4 programs," in *18th USENIX Symposium on Networked Systems Design and Implementation (NSDI 21)*. USENIX Association, Apr. 2021.

[5] H. Harkous, M. Jarschel, M. He, R. Pries, and W. Kellerer, "P8: P4 with predictable packet processing performance," *IEEE Transactions on Network and Service Management*, vol. 18, no. 3, pp. 2846–2859, 2021.

[6] H. Harkous, M. Jarschel, M. He, R. Priest, and W. Kellerer, "Towards understanding the performance of P4 programmable hardware," in *2019 ACM/IEEE Symposium on Architectures for Networking and Communications Systems (ANCS)*. IEEE, 2019, pp. 1–6.

[7] D. Scholz, H. Stubbe, S. Gallenmüller, and G. Carle, "Key properties of programmable data plane targets," in *2020 32nd International Teletraffic Congress (ITC 32)*, 2020, pp. 114–122.

[8] D. Scholz, A. Oeldemann, F. Geyer, S. Gallenmüller, H. Stubbe, T. Wild, A. Herkersdorf, and G. Carle, "Cryptographic hashing in P4 data planes," in *2019 ACM/IEEE Symposium on Architectures for Networking and Communications Systems, ANCS 2019*. IEEE, 2019, pp. 1–6.

[9] H. Harkous, M. He, M. Jarschel, R. Pries, E. Mansour, and W. Kellerer, "Performance study of P4 programmable devices: Flow scalability and rule update responsiveness," in *2021 IFIP Networking Conference (IFIP Networking)*, 2021, pp. 1–6.

[10] Gurobi Optimization, LLC, "Gurobi Optimizer Reference Manual," 2021. [Online]. Available: https://www.gurobi.com