IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 1 July 2022, accepted 17 July 2022, date of publication 20 July 2022, date of current version 26 July 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3192539

RESEARCH ARTICLE

Implementation of P4-Based Schedulers for Multipath Communication

HIROAKI MOTOHASHI[1], (Graduate Student Member, IEEE), PHI LE NGUYEN[3], (Member, IEEE), KIEN NGUYEN[1, 2], (Senior Member, IEEE), AND HIROO SEKIYA[1], (Senior Member, IEEE)
[1]Graduate School of Science and Engineering, Chiba University, Chiba 263-8522, Japan
[2]Institute for Advanced Academic Research, Chiba University, Chiba 263-8522, Japan
[3]School of Information and Communication Technology, Hanoi University of Science and Technology, Hanoi 11615, Vietnam

Corresponding author: Kien Nguyen (nguyen@chiba-u.jp)

This work was supported in part by Vingroup Joint Stock Company (Vingroup JSC), Vingroup; in part by the Vingroup Innovation Foundation (VINIF) under Project VINIF.2020.DA09; and in part by the Japan Society for the Promotion of Science (JSPS) under Grant 20H0417.

ABSTRACT Multipath communication is a well-developed technology that enhances communication effectiveness and resilience. Moreover, it can flexibly utilize network resources through load balancing among available paths. However, traditionally, deploying such load balancing functions on network devices is costly due to the required configuration changes and complicated signaling mechanisms on the devices' control planes. Programming Protocol-independent Packet Processors (P4) has recently emerged as a programming language that enables programmability on the data plane, with the potential to relieve such issues in multipath communication. This work introduces and implements three P4-based multipath schedulers that can split traffic over several paths in wireless networks. The first is P4-based Random Splitting, which distributes traffic randomly. The second is P4-based Weighted Round Robin, with path scheduling based on weights in accordance with path capability. The last is P4-based Dynamic Weighted Round Robin (DWRR), which can improve bandwidth utilization by shifting the weights following dynamic changes in the available bandwidth (i.e., when congestion occurs). We have extensively evaluated the implementation of these three P4-based schedulers in a Mininet-WiFi/P4 environment with User Datagram Protocol (UDP) traffic. The results show that these schedulers can achieve multipath communication with the designed scheduling mechanisms.

INDEX TERMS P4, multipath, scheduler, WRR, DWRR.

I. INTRODUCTION

In recent years, various research and development efforts have focused on the constructions of 5G networks and beyond [1], [2]. These networks are expected to have sufficient resources for many emerging applications, such as augmented reality (AR), virtual reality (VR), and the Internet of Things (IoT). Due to their high key performance indicator (KPI) requirements, many innovative technologies that can achieve high-throughput, low-latency communication or support a massive number of devices (e.g., 5G New Radio) have been introduced. In addition, the network infrastructure must be significantly improved to support emerging applications. However, it is challenging to deploy new network infrastructures with sufficient considerations of various issues, such as network complexity, maintenance costs, and the installation environment. Moreover, various network resources that are normally utilized with redundancy have not been efficiently used. Therefore, network softwarization technologies, including software-defined networking (SDN) and Programming Protocol-independent Packet Processors (P4), are attracting attention as an effective means to realize such a demanding network infrastructure [3], [4].

In SDN, the control plane is separated from the data plane on network devices. In addition, SDN centralizes the control functions of the network in the controller, and the data plane is

The associate editor coordinating the review of this manuscript and approving it for publication was Jad Nasreddine.

programmed (e.g., to set forwarding rules) from the controller via a control channel (OpenFlow). As a result, SDN can enable complex forwarding with commodity hardware and reduce the costs of deployment and maintenance. Despite these advantages, there are some limitations to SDN implementation. First, OpenFlow, the de facto protocol, needs to be supported by the network devices. Second, the centralized SDN architecture can lead to an overload of control traffic in the control channel. Recently, another form of network softwarization, P4, has shown the potential to complement SDN. P4 enables programmability on the data plane, relaxing the dependence on the control plane. In this work, we focus on the realization of P4-based schedulers for multipath communication that can efficiently achieve load balancing over several paths.

Multipath communication has recently become increasingly popular in mobile wireless networks. This technology can be used both on devices and in the network infrastructure for throughput improvement, fault tolerance, etc. For example, mobile devices that support the Multipath Transmission Control Protocol (MPTCP) can harness the resources of multiple wireless networks (e.g., 4G and Wi-Fi) concurrently [5]. Such multipath technology requires modification of the devices at both ends of the communication path and complicated signaling mechanisms. However, deploying multipath technology in the network infrastructure may relax these constraints while enabling efficient use of the existing network resources. In [6], the authors introduced a traffic splitter for scheduling multipath communication traffic. Several schedulers supporting different granularities of traffic splitting and route selection have been proposed in the literature. They can split multipath traffic at the packet level and/or at the flow level [7], [8]. Among them, the main route selection methods are probabilistic routing and round-robin routing [9]–[12]. However, to date, when using existing technologies, these schedulers have required configuration changes to the devices' control planes, resulting in costly deployment and maintenance. In addition, due to the lack of programmability, a fresh update is required whenever there is a change in the network configuration. These issues can be solved by leveraging the programmability offered by network softwarization. This motivates us to realize efficient schedulers for multipath communication using P4.

This paper introduces three new P4-based schedulers for multipath communication, namely, Random Splitting (RS), Weighted Round Robin (WRR), and Dynamic Weighted Round Robin (DWRR). The first scheduler randomly schedules its output traffic among several paths. The second considers predefined weights for different communication paths and then splits the traffic accordingly. The third, P4-based DWRR, dynamically adjusts the weights in accordance with the network congestion state to enable efficient load balancing. In addition to the packet forwarding function, P4-based DWRR newly includes a weight adjustment mechanism based on Explicit Congestion Notification (ECN). To show the effectiveness of these three schedulers, we evaluate them in an emulated wireless network. With P4-based RS and P4-based WRR, the packet distribution ratio does not change. Hence, we investigate these schedulers in scenarios with heterogeneous and homogeneous paths. For P4-based DWRR, we explore its ability to adjust the distribution ratio in two cases: when the available path bandwidth is fixed, and when it is varying. The results show that P4-based RS and WRR achieve throughput aggregation in all investigated cases. On the other hand, P4-based DWRR efficiently achieves load balancing by means of its weight adjustment algorithm, which tunes the packet distribution ratio following the available bandwidth.

The rest of the paper is organized as follows. In Section II, we give an overview of related works. In Section III, we describe the P4-based multipath schedulers. Section IV presents the performance evaluation. Finally, we conclude the paper in Section V.

II. RELATED WORKS

Multipath communication enables high-bandwidth, fault-tolerant communication. To date, the most successful multipath technology has been MPTCP, an extension of the Transmission Control Protocol (TCP). MPTCP has been standardized by the Internet Engineering Task Force (IETF) and has many use cases in wireless networks, such as cellular 4G/Wi-Fi [13], [14], 5G architecture [15], and WiGig/Wi-Fi networks [16]. In such networks, MPTCP improves throughput and can achieve soft handover. MPTCP has a path manager to manage paths and a congestion control mechanism to adjust the data volume for each path. In addition, MPTCP has a scheduler to schedule the next packets to be sent over multiple paths. MPTCP is independent of the network infrastructure. However, MPTCP support is required at both ends of communication, and the signaling mechanism is complicated [5]. For multipath communication, an efficient traffic splitter or scheduler is essential. In addition to using MPTCP, a scheduler can also be installed in the network devices. Doing so may reduce the dependence on the signaling mechanism (e.g., of MPTCP). Moreover, it may relax the requirements for multiple IP addresses. Figure 1 illustrates the concept of a network-based scheduler. In previous works, several proposed traffic splitters have achieved efficient load balancing in the network. Among them, the most common methods are to select the packet forwarding destinations either probabilistically or with specified patterns [9]–[12]. Other works have considered traffic at the packet level or the flow level [7], [8]. However, these previous works heavily depend on the networking devices, for which it is very costly to add a function or upgrade features. This is because the traffic splitters need to possess multiple forwarding control functions, also accounting for single-path forwarding.

SDN introduces the concept of separating the control plane on a networking device, facilitating flexible forwarding [17]. The network administrator programs the control plane, for example, for forwarding functions. On the data plane, new forwarding rules are installed in the form of a table or multiple

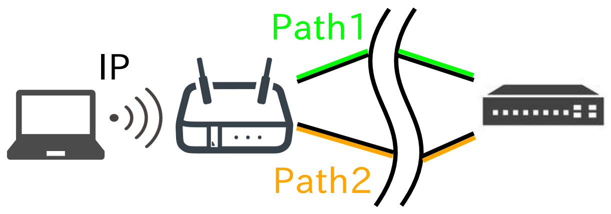


FIGURE 1. Multipath scheduler.

tables. The data plane will then perform the data forwarding function in accordance with the matching information of the associated packets. For SDN, related works have investigated how to achieve effective multipath communication [18]–[20]. However, the control functions for the entire network are usually centralized at the server (i.e., controller) in SDN. This can lead to increased communication latency (between the two planes) and failure of the control channel. To further improve the flexibility of the forwarding function, P4, which is also a high-level language for programming on the data plane, has been introduced [21]–[23]. The P4 language can be defined and compiled like other programming languages, but for the data plane of network devices.

There are several related works on P4-based multipath communication. In [24], the authors proposed a MultiPath Hop-by-hop Utilization-aware Load balancing Architecture (MP-HULA) switch capable of multipath load balancing. MP-HULA uses flowlet abstraction and a new type of message to estimate the congestion state of each port and monitor the links. The P4 implementation of MP-HULA cooperates well with MPTCP and outperforms MPTCP in data center networks. The P4-based multipath mechanism presented in [25], [26] relies on monitoring the last path used by an abstracted group of packets. The route change is calculated by computing the monitored and acquired round-trip time (RTT) values. Since the traffic distribution method is based on the flowlet concept, it is helpful for data center networks, in which the number of senders and the relative positions of the schedulers do not change. Similarly, the work in [27] presented a P4-based multipath strategy for a single stream that focuses on packets' RTTs and proposed an algorithm to prevent out-of-order packets. However, it did not consider the case in which the number of senders may change. In [28], the authors proposed Dynamic Weighted Round Robin (DWRR), in which the weights of Weighted Round Robin routing are changed in an event-driven manner to achieve efficient load balancing, where the relevant event is the occurrence of congestion during communication. The DWRR mechanism proposed in [29] calculates the required and available bandwidths and resets the weights accordingly every time a communication route is selected. However, the implementation requires the establishment of a priority queue and depends on the control plane.

In contrast to previous papers, this paper targets efficient load balancing in networks using data plane programmability. Multipath load balancing is achieved at the packet level. Moreover, we introduce three schedulers, including a P4-based DWRR scheduler, to handle the dynamic scenario.

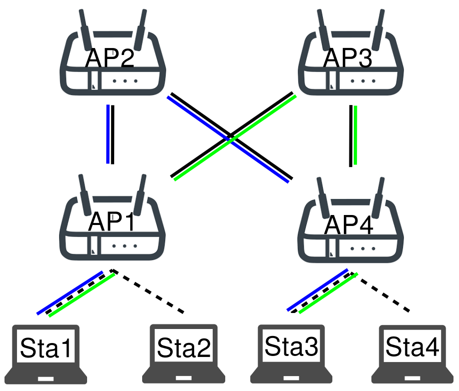


FIGURE 2. Network topology.

III. P4-BASED MULTIPATH SCHEDULERS

This section describes the proposed P4-based multipath schedulers to be deployed on the data planes of networking devices. The parameters used in P4 programs are defined in user-defined tables. These tables are stored in a JavaScript Object Notation (json) file (i.e., the format description file for the data plane). They also describe the data transfer functions in P4 and the associated information (e.g., IP address, port). In addition, they may include other elements, e.g., for updating the packet header (IP dstAddr, Egress Port, original metrics, etc.). We introduce three schedulers, each of which has different traffic splitting functions: RS, WRR, and DWRR. For ease of understanding, we describe them in reference to the network topology shown in Fig. 2. The scheduler, deployed on P4-capable access points (APs), splits end-to-end traffic into two flows following two paths (path1 and path2). In the figure, the blue and green lines represent path1 and path2, respectively. In the scope of this work, we consider User Datagram Protocol (UDP) traffic. Moreover, we simplify the multipath communication's signaling paths and focus on schedulers. Rather than introducing Connection ID as in MPQUIC [30] and Multiflow QUIC [31], we use a pair of IP addresses to determine the multipath connection.

A. P4-BASED RANDOM SPLITTING (RS)

This scheduler controls data transmission by randomly splitting arriving packets into two patterns. It either assigns a random value parameter to an arriving packet or refers to the packet's data field with the random value parameter. This parameter is used as an indicator when performing packet distribution. Theoretically, probabilistic path selection as in [9], [10] can be realized with the same method. With P4, we can use the basic function of generating random values (i.e., in a given value range) and attaching them to the packet headers. The implementation of this scheduler is simple. However, there are potential issues, such as the possibility of

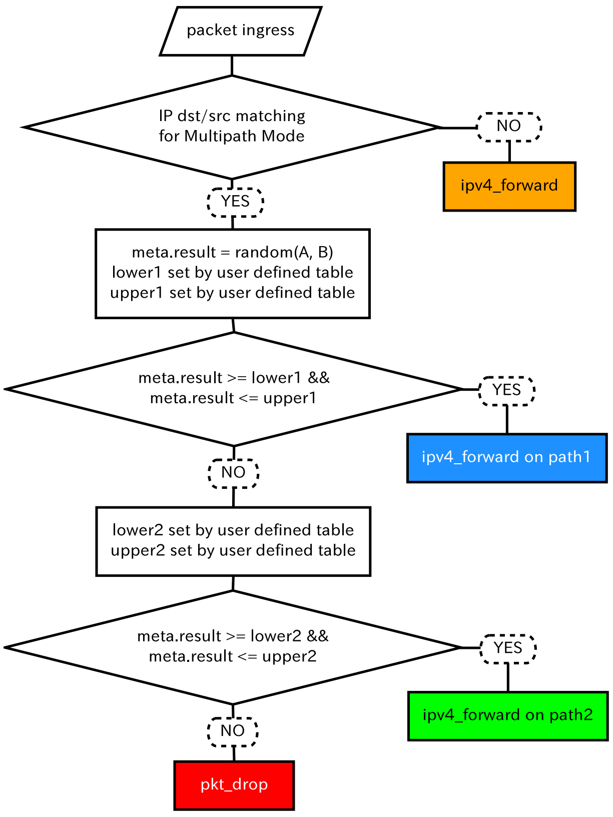

packet ingress
IP dst/src matching for Multipath Mode
NO
ipv4_forward
YES
meta.result = random(A, B)
lower1 set by user defined table
upper1 set by user defined table
meta.result >= lower1 &&
meta.result <= upper1
YES
ipv4_forward on path1
NO
lower2 set by user defined table
upper2 set by user defined table
meta.result >= lower2 &&
meta.result <= upper2
YES
ipv4_forward on path2
NO
pkt_drop

FIGURE 3. **P4-based RS.**

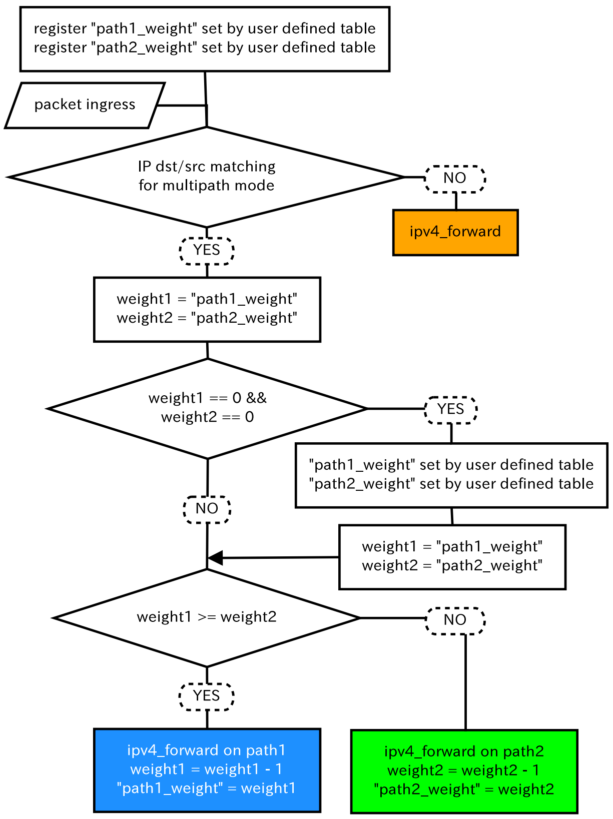


FIGURE 4. **P4-based WRR.**

packets being concentrated on one path, unstable jitter in the communication channel, and difficulty in tracking packets.

A flowchart of P4-based RS is shown in Fig. 3. The first step is to perform source and destination IP address matching of the arriving packets. If the IP address set is for traffic splitting, we input a random value between A and B into the variable "result" of the "meta" field constructed in the packet by the P4 language. In this study, we set $A = 1$ and $B = 100$. In addition, threshold values for path1 ("lower1" and "upper1") are set by a user-defined table. Next, the value of "result" is compared against the values of these thresholds to determine whether to forward on path1. If the random value ("result") assigned to the packet is within the range of the threshold values ("lower1" and "upper1") for path 1, then the packet is forwarded on path1. If the packet is not sent on path1, then the threshold values for path2 ("lower2" and "upper2") are updated and compared against the value of "result" to determine whether to forward the packet on path2. According to the flow table, if an IP address is set without traffic splitting, IPv4 forwarding is performed.

B. P4-BASED WEIGHTED ROUND ROBIN (WRR)

The WRR scheduler sets corresponding weights for paths when performing packet distribution. Traffic is split among the paths based on the defined weights, similar to the traditional WRR mechanisms [11], [12]. Thus, a traffic load distribution is achieved with the same proportions as the weight values. Since the packet distribution is more deterministic, the jitter in the communication paths is less than with the RS scheduler. Furthermore, it is relatively easy to track packets. The packet distribution ratio is guaranteed to be equal to the weight ratio between paths. However, it is not trivial to implement WRR with P4 due to the current capabilities of P4. A flowchart of our P4-based WRR algorithm is shown in Fig. 4.

The top part of the figure illustrates the method used to set the weight values in our P4. implementation. To distribute packets based on weights, we store the weight values in a register. The weight values to be held in the register should be set based on the bandwidth percentage of each communication path. Moreover, the register should be defined with a sufficient length (i.e., bit width) to store the weight values. Similar to the RS scheduler, for arriving packets, IP source and destination matching is performed to determine whether traffic splitting, IPv4 forwarding, or packet dropping should be executed. In the case of traffic splitting, the program reads the weight register and checks the packet's destination. If the weights are different, the path with the highest value is selected first. Otherwise, any preferred path can be used. In our program, we choose path1 as the preferred option.

As shown at the bottom of the flowchart, each time a packet is forwarded, the weight of the path used is decremented. For example, if the initial weight values for path1 and path2 are 2 and 1, respectively, then paths will be selected for the transmission of packets in accordance with the following pattern: the first packet will use path1, the second one will also use path1, and the third one will be sent on path2. Let us consider another case, in which the initial values of the weights for path1 and path2 are 2 and 4, respectively. Then, path selection for incoming packets will proceed as follows. The first and second packets will be assigned to path2, and the third packet will be sent on path1. Then, the fourth packet will be transmitted on path2, the fifth on path1, and the sixth on path2. We refer to the length of such a matching pattern of forwarding destinations as the scheduling period. In implementation, the scheduling period is equal to the sum of the weights: in the first example, the scheduling period is three packets, and in the second, it is six packets.

C. P4-BASED DYNAMIC WEIGHTED ROUND ROBIN (DWRR)

The P4-based DWRR scheduler is an extension of the P4-based WRR scheduler with more complicated mechanisms. In the RS and WRR schedulers, the packet distribution ratio is fixed. Therefore, their efficiency will suffer when the available bandwidth ratio between paths changes during communication. In our DWRR scheduler, we add a function that allows the weight ratios between paths to be adjusted while transmission is ongoing to solve this issue. The DWRR scheduler uses congestion detection as an indicator to drive the weight adjustment mechanism, as in [28]. For congestion detection, the lower two bits (i.e., the ECN field) of the Type of Service field included in the IP header [32], [33] are used. By adding this weight adjustment mechanism and the ECN mechanism, we can realize P4-based DWRR. Due to the varying weight characteristics, this scheduler can achieve more efficient load balancing even when the available bandwidth changes during communication.

1) IMPLEMENTATION OF P4-BASED DWRR

Figure 5 shows a flowchart of the P4-based DWRR scheduler, in which the red modules corresponding to the ECN and weight adjustment mechanisms will be described in later sections. Similar to P4-based WRR, P4-based DWRR uses a cacheable register for the path weights. In addition to the path weights, registers are also used for other entities (e.g., path's_adjust, ecn). Initially, the P4-based DWRR scheduler checks whether an incoming packet is marked with ECN. The occurrence of an ECN-marked packet indicates congestion on the communication path. This checking method is essential to guarantee efficient communication for either the scheduler or the IPv4 forwarding function. Similar to other related works [32], [33], P4-based DWRR also matches the ECN mark in arriving packets to activate the ECN mechanism. Subsequently, similar to RS and WRR, DWRR also performs IP source and destination address matching to determine where the packet belongs and needs to be distributed to for multipath communication. For traffic distribution, DWRR reads the weight of each path from the register. It then selects the communication path on which the packet is to be forwarded by comparing the magnitudes of the values. This process includes checking whether conges-

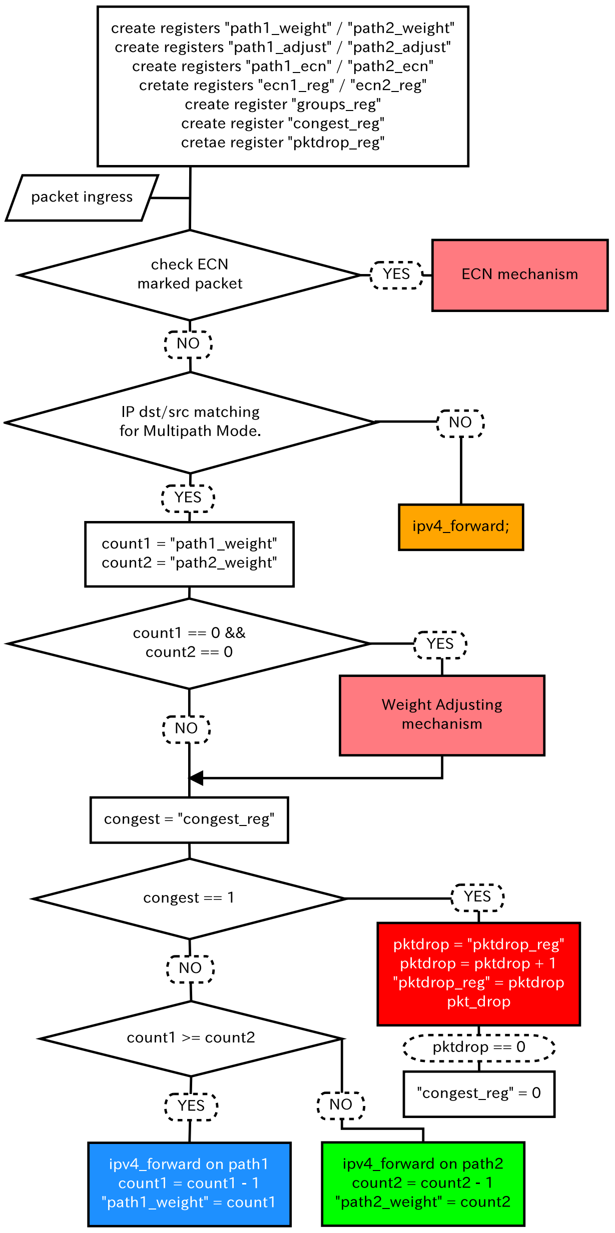


FIGURE 5. P4-based DWRR.

tion is present on the path to be used and planning congestion avoidance. Congestion avoidance is achieved by dropping a certain number of packets when congestion occurs on both paths. We set the number of intentionally dropped packets to 32 in our evaluation. The bottom part of the flowchart is similar to that for P4-based WRR. After a packet is transmitted, the weight of the path used for forwarding that packet is decremented. Moreover, the weight adjustment mechanism is checked in each scheduling period to see whether it is necessary to update the weight values.

2) ECN MARKING AND ECN HANDLING MECHANISM

DWRR relies on an ECN handling mechanism and a weight adjustment mechanism. For notification of congestion, we use P4 to mark a congested packet with ECN.

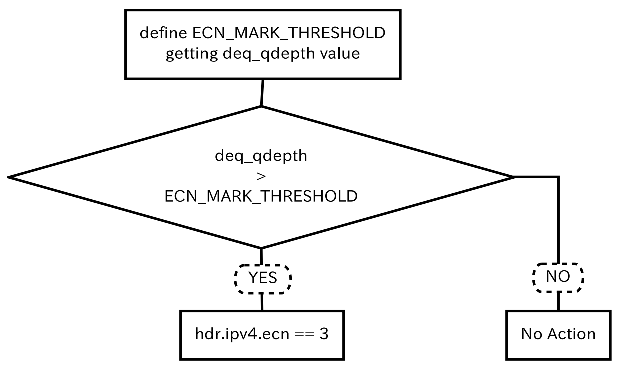


FIGURE 6. **ECN marking in P4-based DWRR.**

The ECN marking process, which rewrites the IP header's ECN field in accordance with the congestion state, is shown in Fig. 6. In P4, there are two types of data transfer control elements (tables, actions, and controls): Ingress and Egress. Between the Ingress and Egress elements, there is a P4-specific buffer in which the packets waiting to be forwarded are stored. There is also a buffer for each output port. Egress reconfirms whether to forward or not. Additionally, it refers to this information during the period when the packet is stored in the buffer. We implement the ECN marking progress on the Egress side. The "deq_qdepth" parameter indicates the buffer occupancy at the start of dequeuing from the P4-specific buffer. If congestion occurs, the packets waiting to be forwarded will be stored in this buffer. Hence, we rewrite the ECN field in the IP header once a certain number of packets have accumulated. In our implementation, we set the associated values of ECN_MARK_THRESHOLD to 60 packets.

The implementation of the ECN mechanism is shown in Fig. 7. The mechanism operates on a network device and checks whether an ECN-marked packet has arrived. Note that a network device may play different roles depending on the traffic that is the target of traffic splitting. Hence, the table entries are used to enable proper selection of the data transfer control mode. The devices in the network can serve as splitting points, passing points, and confluence points. For example, for the topology in Fig. 2, the traffic from Sta1 to Sta3 passes through devices as follows. First, when the ECN mechanism operates at a passing point, the traffic is forwarded as single-path traffic (i.e., IPv4 forwarding). A device is a confluence point if it is an uplink or a splitting point otherwise. Second, when the ECN mechanism operates at a confluence point, it checks the path used by the arriving ECN-marked packet. Since there is congestion on that path, it will send the packet to a splitting point on another path (i.e., a backup path without congestion). However, if all ECN-marked packets are sent back, then the ongoing flow may be lost; instead, we should send back one packet after the arrival of a certain number of marked ECN packets. In our implementation, we set this trigger count to 16 packets. Finally, we will discuss the ECN mechanism at a splitting point. This

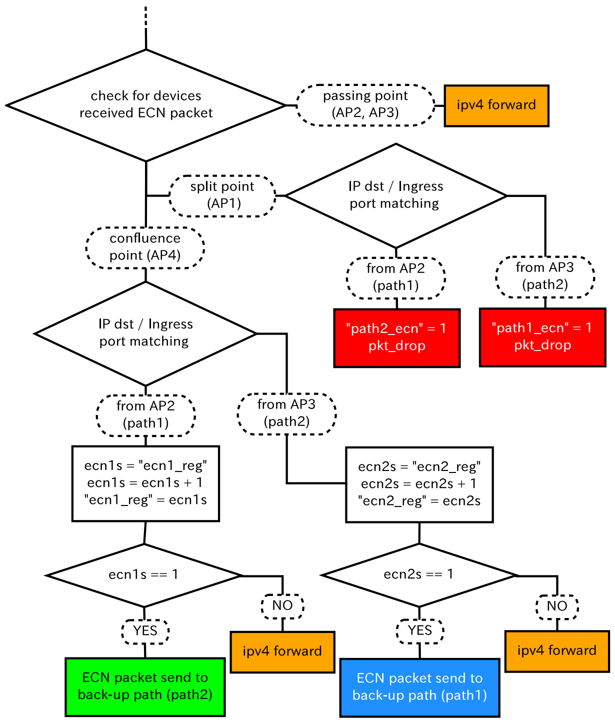


FIGURE 7. **ECN handling mechanism in P4-based DWRR.**

occurs when a confluence point returns ECN-marked packets. Then, the mechanism checks the path used for the return and uses the registers to count the ECN-marked packets received in a given period.

3) WEIGHT ADJUSTMENT MECHANISM

The weight adjustment mechanism is presented in Fig. 8. In contrast to the others, this mechanism does not involve any data transfer control, such as determining the destination or intentional packet dropping. The objective of this mechanism is to manage the parameters for selecting a path when traffic splitting is performed. Its frequency of operation is once every scheduling period. "Weight" and "Adjust" are the parameters for path selection. "Weight" is proportional to the bandwidth of the communication path used, as in P4-based WRR. "Adjust" is incremented by the number of ECN-marked packets that reach the traffic splitter within a certain period. This period is expressed as the product of the scheduling period and the initial value of the register "groups_reg". In our implementation, the initial value of "groups_reg" is 8. No action will be taken when there is no congestion within the current counting period for ECN-marked packets. However, if congestion occurs on one path, this mechanism updates the parameters for adjustment. If congestion occurs on both paths, this triggers congestion avoidance.

IV. EVALUATION

A. EXPERIMENTAL SETTINGS

This section evaluates the implementation of the three schedulers in different network scenarios. We use an Ubuntu

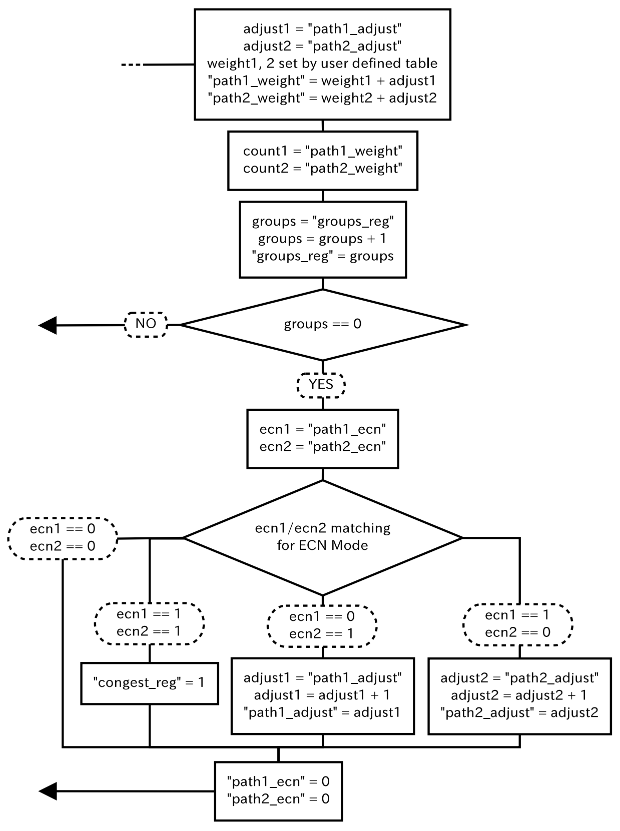


FIGURE 8. **Weight adjustment in P4-based DWRR.**

16.04 machine with P4 compiler version 1.1.0-rc1 as the environment [34]. The machine is also equipped with the network emulator Mininet-WiFi version 2.4.3 [35]. In Mininet-WiFi, we create the investigated network topology, which is similar to the one in Fig. 2. The topology is defined in a json file, where the network configurations match the content to Mininet-WiFi's syntax. Note that we can represent any custom topologies supported by mininet-wifi in the json file. The wireless links are configured to use IEEE 802.11g. During operation, the network includes two single paths (i.e., path1 and path2), of which path1 is Sta1-AP1-AP2-AP4-Sta3 (shown in blue) and path2 is Sta1-AP1-AP3-AP4-Sta3 (shown in green). We deploy our P4 schedulers on the APs and investigate the throughput performance. The three proposed P4-based multipath schedulers (i.e., P4-based RS, P4-based WRR, and P4-based DWRR) are evaluated in three scenarios (1, 2, and 3), as follows.

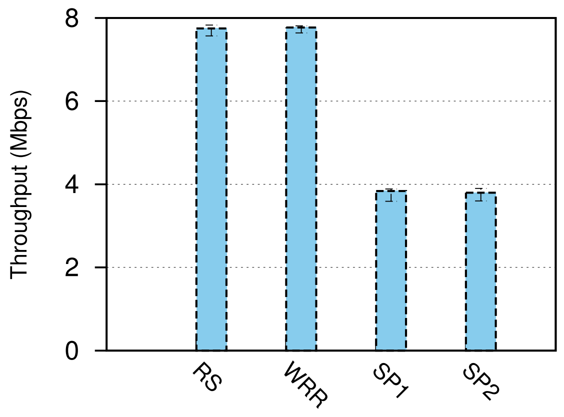
In scenario 1, we compare RS and WRR in a homogeneous network (the bandwidth of path1 is equal to that of path2, i.e., 4 Mbps). Moreover, there is 8 Mbps UDP traffic from Sta1 to Sta3. In scenario 2, we use the same experimental configuration except that the paths are heterogeneous, with different bandwidths. More specifically, the bandwidth of path1 is 6 Mbps, and that of path2 is 3 Mbps. Since WRR and DWRR perform similarly in scenarios 1 and 2, we consider only WRR in those scenarios. The UDP flow between Sta1 and Sta3 has a rate of 9 Mbps in scenario 2. Each link is assigned a specified bandwidth capacity value in the json file, which defines the topology. In P4-based WRR and DWRR, we set the initial weight values following the initial bandwidths. In scenario 3, we use the heterogeneous network settings and evaluate DWRR with two traffic patterns. First, we vary the traffic capacity from Sta1 to Sta3. Then, DWRR is evaluated in a dynamic scenario in which another flow shares the network resources.

B. EVALUATION RESULTS

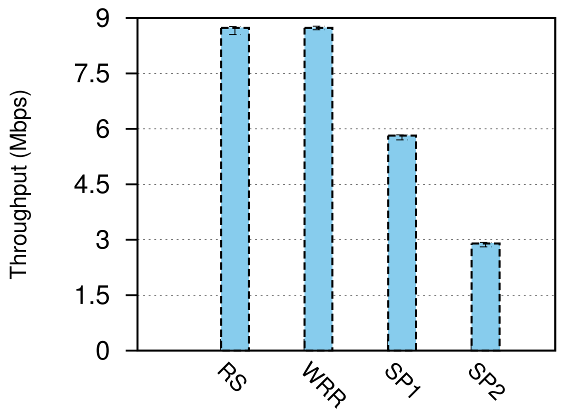
1) RS AND WRR

In scenario 1, the Sta1-Sta3 UDP traffic duration is 100 seconds. We repeat the experiment ten times. The results of RS and WRR are compared in Fig. 9a, which shows the average throughput values as a bar graph. The error bars show the minimum and maximum values of the results. The two bars on the right show the throughputs when data are transmitted on only one path. The throughput values are close to 4 Mbps for both path1 and path2 in the case of single-path communication. We observe that both P4-based RS and P4-based WRR achieve a throughput of approximately 8 Mpbs, indicating that they can both perform bandwidth aggregation for the case of multiple used paths. The throughput results of WRR are a few percent better than those of RS. Specifically, the average throughput of RS is 7.75 Mbps, 96.8% of the available bandwidth, whereas WRR achieves a throughput of 7.77 Mbps, 97.1% of the available bandwidth. The results for the network with heterogeneous paths in scenario 2 are shown in Fig. 9b. Similar to the previous results, the right side of the figure shows the single-path throughputs when the bandwidth is 6 Mbps for path1 and 3 Mbps for path2. These results indicate that UDP traffic can occupy nearly all of the available bandwidth on each path. Note that the weights for WRR are set following the ratio of the bandwidth to be used on each path. As seen from this figure, both the RS and WRR schedulers can again achieve successful path bandwidth aggregation, similar to the previous evaluation.

We can conclude from the above results that our P4-based RS and WRR schedulers can achieve throughput aggregation in the investigated networks. In addition, P4-based WRR performs load balancing slightly better than P4-based RS. This is probably due to the differences in the characteristics of the schedulers. RS selects a route with a probability proportional to the available bandwidth. Therefore, the splitting and bandwidth ratios may shift due to unstable path selection. WRR, on the other hand, always has the same splitting and bandwidth ratios, and its path selection is more stable. However, in both scenarios, the considered bandwidths on each path are fixed. Consequently, these two schedulers cannot adjust the packet distribution ratio correctly when there is another ongoing transmission process. Therefore, they may not be effective in networks where the available bandwidth changes during transmission.



(a) Scenario 1



(b) Scenario 2

FIGURE 9. Comparing RS, WRR, and two single paths (SP1: path1; SP2: path2): (a) path1 capacity 4 Mbps, path2 capacity 4 Mbps; (b) path1 capacity 6 Mbps, path2 capacity 3 Mbps.

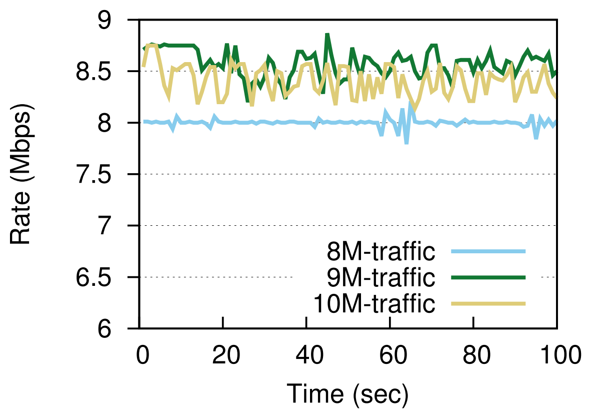


FIGURE 10. Scenario 3: Evaluation of P4-based DWRR with different input traffic.

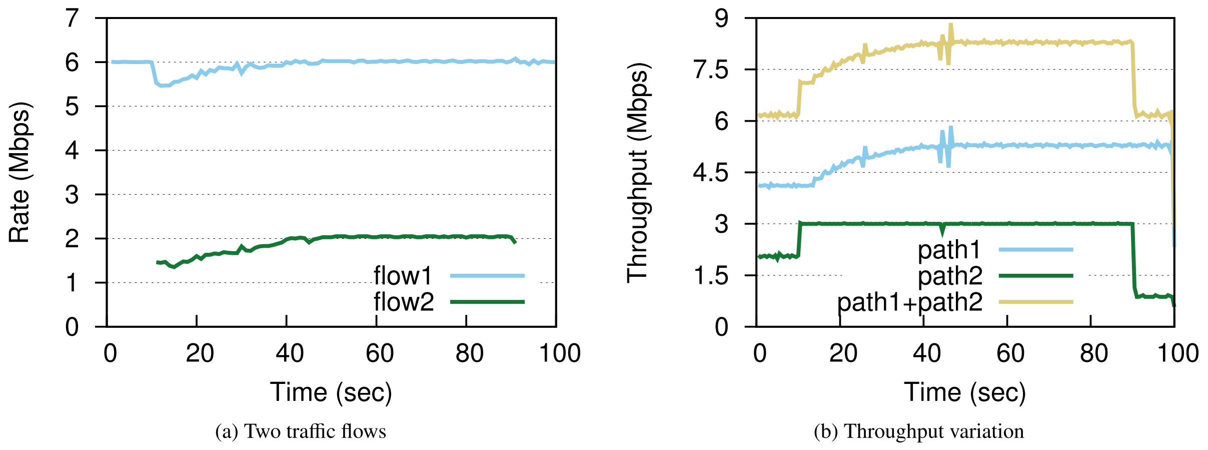
2) DWRR

This section presents the evaluation results for P4-based DWRR. Figure 10 shows the throughput variation when the UDP rate from Sta1 to Sta3 changes from 8 to 10 Mbps. Since the bandwidth on path1 is 6 Mbps and the bandwidth on path2 is 3 Mbps, a total bandwidth of 9 Mbps is available for packet distribution. With 8 Mbps traffic, DWRR can effortlessly achieve load balancing, similar to WRR. With 9 Mbps traffic, congestion occurs due to queue filling, causing the throughput value to oscillate (after approximately 10 seconds). However, the weight adjustment mechanism is not activated because all paths have 100% utilization. When the traffic exceeds the capacity of both paths (i.e., at 10 Mbps), more severe packet dropping occurs. Therefore, the weight adjustment mechanism is triggered to update the weight values. Consequently, the congestion avoidance feature in the weight adjustment mechanism mitigates some of the unnecessarily dropped packets.

Fig. 11 shows the results of the dynamic weight adjustment mechanism of DWRR. There are two UDP traffic flows in this experiment, as shown in Fig. 11a. Flow1 from Sta1 to Sta3 has a rate of 6 Mbps and uses multiple paths. Meanwhile, flow2 from Sta2 to Sta4, with a 2 Mbps rate, starts 20 seconds after the beginning of the experiment and lasts for 80 seconds. For flow2, only a single path is used (i.e., path2). The variations in throughput over the two paths are shown in Fig. 11b. To measure the throughput on each path, we use the bandwidth monitor bwm-ng [36]. Bwm-ng is run on the input interface AP4, which is the confluence point. In the first 20 seconds, flow1 (6 Mbps) is distributed over both paths. When flow2 appears, there is initial fluctuation. Subsequently, the input traffic for path2 exceeds its capacity of 3 Mbps. There is congestion on the paths, causing the ECN and weight adjustment mechanisms to be activated. As a result, the packet distribution is adjusted. There is no loss on either path. The total throughput is greater than 8 Mbps (i.e., nearly ideal throughput aggregation). From the time when flow2 stops until the end of the experiment (90 to 100th second), only flow1 is observed. At this time, the P4-based DWRR scheduler adjusts the weights to maintain a 1 Mbps flow on path2. From these results, we can conclude that P4-based DWRR achieves efficient load balancing even when the available bandwidths on the paths are dynamic.

a: DISCUSSION

In this work, we consider the multipath schedulers, whose input has one flow, to balance the traffic following usable network resources in wireless networks. It could be similar to the use case, where a client's traffic flow traverses through paths, each of which has the capability allocated by a mobile vendor. Regarding the allocation, in the current and next generation of mobile networks, network slicing technologies have become more and more popular. Hence, the resource allocation at the fine-grained Quality of Service level may be feasible. For example, a different path belongs to a slice with a different bandwidth provision. Regarding the flow identification, with the current P4, we can formalize a flow with many criteria, such as with varying fields of header or microburst characteristics, without the involvement of the control plane. For those



(a) Two traffic flows

(b) Throughput variation

FIGURE 11. **Scenario 3: Evaluation of P4-based DWRR with two traffic flows.**

reasons, the schedulers may be deployable with multiple flows. Moreover, our implementations could be available in the form of P4 source code. The interested community may reuse the P4 code in the application with little effort to modify the input parameters (e.g., flow identification or initial weight values).

V. CONCLUSION AND FUTURE WORK

In this paper, we have introduced and implemented three P4-based schedulers for multipath communication to achieve effective load balancing between paths. They include P4-based RS, P4-based WRR, and P4-based DWRR schedulers, each performing packet distribution following a predefined mechanism. While RS and WRR perform throughput aggregation with random or fixed weight values, P4-based DWRR has a weight adjustment mechanism to adapt to dynamic network conditions. We have implemented and evaluated the schedulers in an emulator environment. The experimental results show that RS and WRR can achieve bandwidth aggregation in different networks with one traffic input. Moreover, by means of its ECN handling mechanism, DWRR can dynamically schedule traffic in accordance with the network conditions.

In the future, we will extend the proposed P4-based schedulers to operate with TCP traffic. One of the feasible approaches is leveraging the MPTCP header handling and MPTCP congestion control of MP-HULA, meanwhile improving the proposed schedulers for TCP (i.e., solving the challenge of handling the reverse flow with ACK packets). Moreover, we plan to implement the schedulers on real hardware with P4 support.

REFERENCES

[1] X. Chen, D. W. K. Ng, W. Yu, E. G. Larsson, N. Al-Dhahir, and R. Schober, "Massive access for 5G and beyond," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 3, pp. 615–637, Mar. 2021.

[2] J. Kaur, M. A. Khan, M. Iftikhar, M. Imran, and Q. E. U. Haq, "Machine learning techniques for 5G and beyond," *IEEE Access*, vol. 9, pp. 23472–23488, 2021.

[3] F. Z. Yousaf, M. Bredel, S. Schaller, and F. Schneider, "NFV and SDN—Key technology enablers for 5G networks," *IEEE J. Sel. Areas Commun.*, vol. 35, no. 11, pp. 2468–2478, Nov. 2017.

[4] J. Ordonez-Lucena, P. Ameigeiras, D. Lopez, J. J. Ramos-Munoz, J. Lorca, and J. Folgueira, "Network slicing for 5G with SDN/NFV: Concepts, architectures, and challenges," *IEEE Commun. Mag.*, vol. 55, no. 5, pp. 80–87, May 2017.

[5] C. Raiciu, C. Paasch, S. Barre, A. Ford, M. Honda, F. Duchene, O. Bonaventure, and M. Handley, "How hard can it be? Designing and implementing a deployable multipath TCP," in *Proc. 9th USENIX NSDI*, Apr. 2012, pp. 399–412.

[6] K. Xinidis, I. Charitakis, S. Antonatos, K. G. Anagnostakis, and E. P. Markatos, "An active splitter architecture for intrusion detection and prevention," *IEEE Trans. Dependable Secure Computing*, vol. 3, no. 1, pp. 31–44, Jan. 2006.

[7] K.-C. Leung and V. O. K. Li, "Generalized load sharing for packet-switching networks. I. Theory and packet-based algorithm," *IEEE Trans. Parallel Distrib. Syst.*, vol. 17, no. 7, pp. 694–702, Jul. 2006.

[8] K.-C. Leung and V. O. K. Li, "Generalized load sharing for packet-switching networks. II. Flow-based algorithms," *IEEE Trans. Parallel Distrib. Syst.*, vol. 17, no. 7, pp. 703–712, Jul. 2006.

[9] H.-W. Ferng and C.-C. Peng, "Traffic splitting in a network: Split traffic models and applications," *Comput. Commun.*, vol. 27, no. 12, pp. 1152–1165, Jul. 2004.

[10] H.-W. Ferng, "Modeling of split traffic under probabilistic routing," *IEEE Commun. Lett.*, vol. 8, no. 7, pp. 470–472, Jul. 2004.

[11] H. M. Chaskar and U. Madhow, "Fair scheduling with tunable latency: A round Robin approach," in *Proc. Seamless Interconnection Universal Services, Global Telecommun. Conf.*, Dec. 1999, pp. 1328–1333.

[12] T. Balogh and M. Medvecky, "Mean bandwidth allocation model of WRR for IP networks," in *Proc. 35th Int. Conf. Telecommun. Signal Process. (TSP)*, Jul. 2012, pp. 156–160.

[13] K. Nguyen, M. G. Kibria, K. Ishizu, and F. Kojima, "Enhancing multipath TCP initialization with SYN duplication," *IEICE Trans. Commun.*, vol. 102, no. 9, pp. 1904–1913, 2019.

[14] K. Nguyen, M. G. Kibria, K. Ishizu, F. Kojima, and H. Sekiya, "An approach to reinforce multipath TCP with path-aware information," *Sensors*, vol. 19, no. 3, p. 476, Jan. 2019.

[15] M. G. Kibria, K. Nguyen, G. P. Villardi, K. Ishizu, and F. Kojima, "Next generation new radio small cell enhancement: Architectural options, functionality and performance aspects," *IEEE Wireless Commun.*, vol. 25, no. 4, pp. 120–128, Aug. 2018.

[16] K. Nguyen, M. G. Kibria, K. Ishizu, and F. Kojima, "Performance evaluation of IEEE 802.11 ad in evolving Wi-Fi networks," *Wireless Commun. Mobile Comput.*, vol. 2019, pp. 1–11, Feb. 2019.

[17] D. Kreutz, F. Ramos, P. E. Veríssimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-defined networking: A comprehensive survey," *Proc. IEEE*, vol. 103, no. 1, pp. 14–76, Jan. 2015.

[18] R. van der Pol, S. Boele, F. Dijkstra, A. Barczyk, G. van Malenstein, J. H. Chen, and J. Mambretti, "Multipathing with MPTCP and OpenFlow," in *Proc. SC Companion, High Perform. Comput., Netw. Storage Anal.*, Nov. 2012, pp. 1617–1624.
[19] J. Yan, H. Zhang, Q. Shuai, B. Liu, and X. Guo, "HiQoS: An SDN-based multipath QoS solution," *China Commun.*, vol. 12, no. 5, pp. 123–133, May 2015.
[20] Q. Wang, G. Shou, Y. Liu, Y. Hu, Z. Guo, and W. Chang, "Implementation of multipath network virtualization with SDN and NFV," *IEEE Access*, vol. 6, pp. 32460–32470, 2018.
[21] P. Bosshart, D. Daly, G. Gibb, M. Izzard, N. McKeown, J. Rexford, and C. Schlesinger, "P4: Programming protocol-independent packet processors," *SIGCOMM Comput. Commun. Rev.*, vol. 44, pp. 87–95, Jul. 2014.
[22] *P4 Language and Related Specifications*. Accessed: Dec. 2021. [Online]. Available: https://p4.org/specs/
[23] W. L. da Costa Cordeiro, J. A. Marques, and L. P. Gaspary, "Data plane programmability beyond OpenFlow: Opportunities and challenges for network and service operations and management," *J. Netw. Syst. Manage.*, vol. 25, no. 4, pp. 784–818, Oct. 2017.
[24] C. H. Benet, A. J. Kassler, T. Benson, and G. Pongracz, "MP-HULA: Multipath transport aware load balancing using programmable data planes," in *Proc. Morning Workshop-Netw. Comput.*, Aug. 2018, pp. 7–13.
[25] M. Pizzutti and A. E. Schaeffer-Filho, "An efficient multipath mechanism based on the Flowlet abstraction and P4," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2018, pp. 1–6.
[26] M. Pizzutti and A. E. Schaeffer-Filho, "Adaptive multipath routing based on hybrid data and control plane operation," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Apr. 2019, pp. 730–738.
[27] C. Zhou, W. Quan, D. Gao, Z. Liu, C. Yu, M. Liu, and Z. Xu, "AMS: Adaptive multipath scheduling mechanism against eavesdropping attacks with programmable data planes," in *Proc. IEEE 5th Adv. Inf. Technol., Electron. Autom. Control Conf. (IAEAC)*, Mar. 2021, pp. 2357–2361.
[28] S. Liu, X. Zhang, J. Yu, Y. Liu, and X. Chen, "A dynamic traffic distribution strategy for multipath routing," in *Proc. 7th Int. Conf. Inf., Commun. Signal Process. (ICICS)*, Dec. 2009, pp. 1–5.
[29] S. O. Yese, A. Abdulazeez, and A. Mohammed, "A dynamic weighted round Robin call admission (DWRR-CAC) algorithm for broadband networks," in *Proc. 15th Int. Conf. Electron., Comput. Comput. (ICECCO)*, Dec. 2019, pp. 1–6.
[30] Q. De Coninck and O. Bonaventure, "Multipath QUIC: Design and evaluation," in *Proc. 13th Int. Conf. Emerg. Netw. Exp. Technol.*, Nov. 2017, pp. 160–166.
[31] Q. De Coninck and O. Bonaventure, "Multiflow QUIC: A generic multipath transport protocol," *IEEE Commun. Mag.*, vol. 59, no. 5, pp. 108–113, May 2021.
[32] S. Wang, J. Zhang, T. Huang, T. Pan, J. Liu, and Y. Liu, "Adaptively adjusting ECN marking thresholds for datacenter networks," in *Proc. IEEE 25th Int. Conf. Netw. Protocols (ICNP)*, Oct. 2017, pp. 1–2.
[33] W. Wei, K. Xue, J. Han, D. S. L. Wei, and P. Hong, "Shared bottleneck-based congestion control and packet scheduling for multipath TCP," *IEEE/ACM Trans. Netw.*, vol. 28, no. 2, pp. 653–666, Apr. 2020.
[34] *P4 Compiler*. Accessed: Dec. 2021. [Online]. Available: https://github.com/p4lang/p4c
[35] *Mininet-WiFi*. Accessed: Dec. 2021. [Online]. Available: https://github.com/intrig-unicamp/mininet-wifi
[36] *BWM-NG*. Accessed: Dec. 2021. [Online]. Available: https://linux.die.net/man/1/bwm-ng

HIROAKI MOTOHASHI (Graduate Student Member, IEEE) was born in Saitama, Japan, in May 1997. He received the B.E. degree from Chiba University, Japan, in 2020, where he is currently pursuing the degree with the Graduate School of Science and Engineering. His research interests include SDN and multipath communication.

PHI LE NGUYEN (Member, IEEE) received the B.E. and M.S. degrees from The University of Tokyo, in 2007 and 2010, respectively, and the Ph.D. degree in informatics from The Graduate University for Advanced Studies, SOKENDAI, Tokyo, Japan, in 2019. She is currently an Assistant Professor at the School of Information and Communication, Hanoi University of Science and Technology, Vietnam. Her research interests include the Internet of Things (IoT), AI-powered networking, and data mining.

KIEN NGUYEN (Senior Member, IEEE) received the B.E. degree in electronics and telecommunication from the Hanoi University of Science and Technology (HUST), Vietnam, in 2004, and the Ph.D. degree in informatics from The Graduate University for Advanced Studies, Japan, in 2012. He was a Researcher at the National Institute of Information and Communication Technology (NICT), Japan, during 2014 to 2018. Since 2018, he has been with the Graduate School of Engineering, Chiba University, where he is currently an Associate Professor. His research achievements have been disseminated in three patents, several IETF Internet drafts, and more than 130 publications in peer-reviewed journals and conferences. His research interests include networking and distributed systems, including the Internet, the Internet of Things technologies, and distributed ledger technologies. He is a member of IEICE. He is also participates in IETF activities.

HIROO SEKIYA (Senior Member, IEEE) received the B.E., M.E., and Ph.D. degrees in electrical engineering from Keio University, Yokohama, Japan, in 1996, 1998, and 2001, respectively. Since April 2001, he has been with Chiba University, Chiba, Japan, where he is currently a Professor with the Graduate School of Science and Engineering. His research interests include high-frequency high-efficiency tuned power amplifiers, resonant dc/dc power converters, dc/ac inverters, and digital signal processing for wireless communications. He is a Senior Member of the Institute of Electronics, Information and Communication Engineers (IEICE), Japan, and a member of the Institute of Electronics, the Information Processing Society of Japan (IPSJ), and the Research Institute of Signal Processing (RISP), Japan.