Learning-Based Proactive Resource Allocation for Delay-Sensitive Packet Transmission

Jiayin Chen, Peng Yang, *Member, IEEE*, Qiang Ye, *Member, IEEE*, Weihua Zhuang, *Fellow, IEEE*, Xuemin Shen, *Fellow, IEEE*, and Xu Li

***Abstract*—In this article, a learning-based proactive resource sharing scheme is proposed for the next-generation core communication networks, where the available forwarding resources at a switch are proactively allocated to the traffic flows in order to maximize the efficiency of resource utilization with delay satisfaction. The resource sharing scheme consists of two joint modules, estimation of resource demands and allocation of available resources. For service provisioning, resource demand of each traffic flow is estimated based on the predicted packet arrival rate. Considering the distinct features of each traffic flow, a linear regression algorithm is developed for resource demand estimation, utilizing the mapping relation between traffic flow status and required resources, upon which a network switch makes decision on allocating available resources for delay satisfaction and efficient resource utilization. To learn the implicit relation between the allocated resources and delay, a multi-armed bandit learning-based resource allocation scheme is proposed, which enables fast resource allocation adjustment to traffic arrival dynamics. The proposed algorithm is proved to be asymptotically approaching the optimal strategy, with polynomial time complexity. Extensive simulation results are presented to demonstrate the effectiveness of the proposed resource sharing scheme in terms of delay satisfaction, traffic adaptiveness, and resource allocation gain.**

***Index Terms*—Delay requirements, efficient resource allocation, proactive resource sharing, traffic dynamics, multi-armed bandit learning.**

I. Introduction

THE PROLIFERATION of new applications has placed significant pressure on service provisioning in future core communication networks beyond the fifth generation (5G). Some applications are delay-sensitive with strict requirements that are challenging to satisfy. For instance, video conferencing requires low latency and high reliability to ensure interactivity and video quality. To meet these requirements, the utilization of buffer spaces at each network switch needs to be properly controlled. Specifically, the dominant contributing factor for end-to-end (E2E) packet transmission delay is the delay for packet queuing at network switches, and the packet loss is mainly caused by buffer overflow during network congestion [1].

Recently, some congestion control methods are proposed to meet the stringent quality-of-service (QoS) requirements, e.g., performance-oriented congestion control (PCC) [2] and BBR [3], both of which are implemented at the packet source node, to adjust sending rate based on the observed E2E performance (e.g., achieved goodput, packet loss rate, and average latency). On the other hand, in-network congestion control schemes (e.g., active queue management (AQM)) adjust the queue lengths at switches by dropping or marking packets [4]. In [5], the interplay of end congestion control and in-network AQM is investigated to enable real-time Web browsing. A variation of BBR is proposed for a multi-path transmission scenario [6]. Congestion control schemes in general assume a fixed amount of forwarding resources for the traffic flow of each service (i.e., an aggregation of packets of the same service type being transmitted from a source node to a destination node). To guarantee QoS satisfaction for different applications, resource over-provisioning is usually the case to support the peak traffic volume, while resource multiplexing is exploited among traffic flows of different services to improve QoS with efficient resource utilization.

To support delay-sensitive services, packet transmission delay in core networks should be minimized, in order to meet the E2E delay requirement. The delivery delay of each service flow in core networks is determined by the routing path and the allocated forwarding resources at each switch on route. Software-defined networking (SDN) is an emerging technology to enhance the routing performance in the next-generation core networks, in which the routing path for packet transmission can be customized and optimized for different services [7], [8]. Specifically, the SDN controller calculates the routing path for each traffic flow and distributes the flow tables to all the related switches, thereby guiding the packet forwarding. In addition, the SDN controller can calculate the average delay of packets traversing each switch [9], [10], which can be utilized to configure the per-hop delay requirement. Given the routing path, the amount of allocated forwarding resources

Manuscript received May 12, 2020; revised August 25, 2020; accepted August 31, 2020. Date of publication September 8, 2020; date of current version June 9, 2021. This work was supported by research grants from the Natural Sciences and Engineering Research Council of Canada (NSERC) and from Huawei Canada. The associate editor coordinating the review of this article and approving it for publication was F. Gao. *(Corresponding author: Peng Yang.)*

Jiayin Chen, Weihua Zhuang, and Xuemin Shen are with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON N2L 3G1, Canada (e-mail: j648chen@uwaterloo.ca; wzhuang@uwaterloo.ca; sshen@uwaterloo.ca).

Peng Yang is with the School of Electronic Information and Communications, Huazhong University of Science and Technology, Wuhan 430074, China (e-mail: yangpeng@hust.edu.cn).

Qiang Ye is with the Department of Electrical and Computer Engineering and Technology, Minnesota State University, Mankato, MN 56001 USA (e-mail: qiang.ye@mnsu.edu).

Xu Li is with Ottawa Wireless Advanced System Competency Center, Huawei Technologies Canada Inc., Ottawa, ON K2K 3J1, Canada (e-mail: xu.lica@huawei.com).

Digital Object Identifier 10.1109/TCCN.2020.3022671

at each passing switch collectively determines the E2E delay of a traffic flow. The decomposition of E2E delay requirement enables the development of a per-hop resource allocation solution. Compared with an E2E solution, a per-hop resource allocation solution can be more flexible when dealing with varying network conditions.

To satisfy the decomposed per-hop delay requirement with efficient resource utilization, each switch makes decision on resource sharing among traffic flows. Delay requirements for per-hop packet transmission are taken into account by wait time priority (WTP) [11] and earliest due date (EDD) [12] algorithms. In using the algorithms, the switch makes decision each time that a packet is forwarded. Considering the high forwarding rate of core network switches, flow-level resource sharing schemes with lower computational complexity are investigated for fairness and delay requirements, such as weighted fair queuing (WFQ) [13] and deficit round robin (DRR) [14], [15]. To adapt to varying traffic and network conditions, dynamic WFQ updates the allocation of resources at the beginning of each resource sharing interval [16], where the resource allocation decisions are made based on the estimation of average packet queuing delay in the upcoming interval. For supporting applications with stringent delay requirements, resource allocation is expected to be conducted for delay satisfaction of individual packets. Furthermore, if the resource sharing is conducted in line with packet arrival instants, the QoS can be degraded due to burst of traffic in the upcoming resource sharing interval. Hence, a proactive resource sharing and packet scheduling solution potentially can help timely QoS enhancement based on traffic prediction [17], [18]. Different from the packet-level decision made based on traffic characteristics, estimating resource demand is required for flow-level resource sharing decisions. Thus, to accommodate the large data volume and traffic fluctuations in future core communication networks while achieving efficient resource utilization, we resort to generalized traffic prediction and resource demand estimation to develop a proactive resource sharing scheme.

To support delay-sensitive applications in a dynamic networking environment, learning-based scheduling algorithms are investigated [19], [20]. To be adaptive to traffic dynamics, the resource scheduler applies machine learning techniques to categorize traffic flows with different characteristics [21]. Based on instantaneous system states (e.g., number of arrived packets and resource occupancy), different decisions on allocating resources are made by the learning module, such as reinforcement learning (RL) [22], deep Q network (DQN) [23], and stacked auto-encoder (SAE) deep learning [24], through which the implicit relation between the allocated resources and the achieved QoS satisfaction can be learned. Due to the distinct delay requirements of applications, different models (e.g., parameters of neural networks) are trained to capture the relations between resource allocation and QoS performance separately, which leads to increased computational cost. To incorporate a large number of flows in the core communication networks, a resource sharing algorithm implemented at in-network switches needs to be effective in achieving satisfactory QoS performance with low computational complexity.

In this work, we aim at developing a lightweight online resource demand estimation model for adaptive resource sharing. The resources are for packet forwarding at the egress port of a network switch. On the basis of resource demand estimation to accommodate differentiated delay requirements of different flows, a learning module is developed to learn the relation between allocated resources and achieved QoS for distinct applications. Based on the information of flows (including estimated resource demand, predicted traffic load and resource occupancy), a resource allocation module facilitates resource sharing among flows to achieve maximal overall QoS satisfaction, with the consideration of tradeoff between exploitation and exploration. The multi-armed bandit (MAB) framework has a potential to balance the exploration-exploitation tradeoff, in resource allocation [25], data offloading decision [26], and beam alignment [27] in wireless networks. In the MAB framework, the player makes sequential decisions, choosing one from the arm set each time to maximize the obtained reward. The player focuses on exploiting the most rewarding arms based on historical performance or pulling new arms for exploration. Upper confidence bound (UCB) [28] is used to balance the exploitation-exploration tradeoff for bandit problems, which provides theoretical confidence bound of regret. In the resource sharing problem, the amount of resources allocated to each flow can be formulated as an arm, while the obtained QoS satisfaction is the reward for pulling the arm. The resource demand can also be considered as the guide for resource sharing, which provides context information of each arm. Thus, the resource sharing is a feature-based exploration-exploitation problem, which can be formulated as a contextual bandit problem [29].

In this article, we aim at maximizing the efficiency of resource sharing at a switch (i.e., the ratio of delay satisfaction to the allocated resources), and propose a bandit learning-based proactive forwarding resource sharing scheme, which maps the delay requirement and packet arrivals of each traffic flow to forwarding resource demand. Then, the available resources are allocated to those traffic flows accordingly. The resource allocation is formulated as a bandit learning problem and a regret bounded allocation strategy is developed, which is shown to be efficient in resource utilization. We consider the resource sharing at a software-programmable switch, which is capable of supporting virtual network functions and packet processing functions, in addition to packet forwarding [30], [31]. The main contributions of this article are summarized in the following:

(1) *Resource sharing framework* – We develop a learning-based framework to make resource sharing decisions at each switch. In this framework, the resource demand is firstly extracted as a service feature, by considering both traffic arrivals and delay requirements. Then, the switch allocates an optimal amount of available resources to different traffic flows based on their features;
(2) *Resource demand estimation* – We propose an online resource demand estimation module in the resource sharing framework, which combines a linear regression model with an online gradient descent method. Utilizing the linear mapping relation between traffic loads and

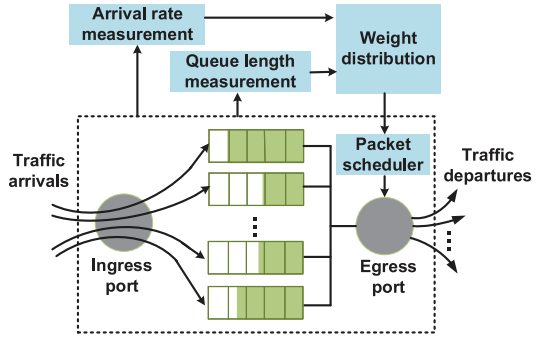


Fig. 1. Illustration of packet transmission at a switch.

the required forwarding resources, the resource demand for each flow at a switch is estimated based on traffic prediction and its per-hop delay requirement;

(3) *Allocation of available resources* – We design a MAB-based allocation of available resources scheme in the resource sharing framework. Based on the estimated resource demand, the available resources are allocated to the flows accordingly. Using the measured delay as feedback, the parameters of the proposed learning module are updated.

The remainder of this article is organized as follows. The system model is described and the research problem is formulated in Section II. In Section III, the learning-based available resource sharing solution is presented. The performance evaluation and comparison are given in Section IV. Finally, conclusions are drawn in Section V.

II. System Model and Problem Formulation

A. Network Model

Traffic flows of different services traverse a sequence of network switches in core networks before reaching their destinations for E2E service delivery. A network switch refers to a software switch, e.g., an openflow switch, centrally managed by an SDN control module in the core networks [32], [33]. An openflow switch is capable of both layer 2 and layer 3 network functionalities, depending on whether the device is located within a local area network or is interconnecting two public network segments. The E2E delay requirement for each traffic flow is considered, which is composed of the per-hop packet delays at all passing switches along the routing path from the source to the destination. This per-hop delay consists of packet queuing delay and transmission delay at a switch.

At each switch, packets from different flows enter corresponding transmission queues of the packet scheduler, as shown in Fig. 1. The packet scheduler makes packet forwarding decisions based on the resource allocation for traffic flows. If more forwarding resources are allocated to a flow, the forwarding rate becomes higher, leading to a reduced queuing delay and transmission delay for this flow. Hence, delay requirements of the traffic flows can be satisfied through adjusting the amount of allocated resources. Consider a set, $\mathcal{N}$, of in-network switches. The set of flows traversing an intermediate switch $n \in \mathcal{N}$ is denoted by $\mathcal{F}_n$, and the set of switches on the routing path of flow $f \in \mathcal{F}_n$ is denoted by $\mathcal{N}_f \subseteq \mathcal{N}$. We assume the flow set, $\mathcal{F}_n$, and the switch set, $\mathcal{N}_f$, remain stable in the course of scheduling. The amount of forwarding resources of switch n is denoted by $C^{(n)}$, and the amount of pre-allocated forwarding resources for flow f is denoted by $\bar{C}_f^{(n)}$, which is determined based on the long-term QoS satisfaction and is assumed to be always guaranteed. The amount of available resources of switch n is $C_I^{(n)} = C^{(n)} - \sum_{f\in\mathcal{F}_n} \bar{C}_f^{(n)}$.

Time is partitioned into resource sharing intervals of constant duration. At the beginning of the mth interval, switch n makes decision on the amount of available resources allocated to flow f, denoted by $\Delta C_{f,m}^{(n)} (\geq 0)$. Then, the allocated forwarding resource for flow f is $C_{f,m}^{(n)} = \bar{C}_f^{(n)} + \Delta C_{f,m}^{(n)}$, and $[C_{f,m}^{(n)}]_{f\in\mathcal{F}_n}$ are the allocated forwarding resources for all the flows at the mth interval. The weighted round-robin scheme is adopted for packet forwarding, where the weights of flow f, denoted by $v_{f,m}^{(n)}$, is proportional to its allocated forwarding resources. That is,

$$v_{f_1,m}^{(n)} : v_{f_2,m}^{(n)} = C_{f_1,m}^{(n)} : C_{f_2,m}^{(n)}, \ \forall f_1, f_2 \in \mathcal{F}_n \tag{1}$$

where $\sum_{f\in\mathcal{F}_n} v_{f,m}^{(n)} = 1$.

B. Per-Hop Delay Requirements

We consider traffic flows with differentiated delay requirements and packet arrival patterns. Suppose the E2E packet delay for flow f is decomposed to per-hop delay requirements based on the pre-allocated forwarding resources at each switch. To support the applications with strict delay requirements, we consider SDN enabled core networks, in which an SDN controller has global network information. Upon service requests, the SDN controller configures routing paths and distributes the flow tables to the passing switches for packet forwarding [34], [35]. In addition, the controller can calculate the average queuing delay and average transmission delay for packets traversing each switch [9], [10], the summation of which can be utilized to configure the per-hop delay requirement for the switch. Specifically, the delay requirement for flow $f \in \mathcal{F}_n$ at switch $n \in \mathcal{N}$ is denoted by $D_f^{(n)}$.

We denote the overall delay satisfaction ratio of switch n in interval m as $\rho_m^{(n)} = \sum_{f\in\mathcal{F}_n} \rho_{f,m}^{(n)}$, where $\rho_{f,m}^{(n)}$ is the delay satisfaction ratio for flow f, i.e., the ratio of number of packets with experienced delay smaller than $D_f^{(n)}$ over total number of transmitted packets belonging to flow f in the mth interval. Let ε be the tolerance of per-hop delay violation ratio, i.e., $\rho_{f,m}^{(n)} \geq 1 - \varepsilon$.

C. Problem Formulation

We consider the resource allocation ratio when making the resource sharing decision, which is denoted by $\eta_m^{(n)}$ for switch n in interval m, $\eta_m^{(n)} = \sum_{f\in\mathcal{F}_n} C_{f,m}^{(n)} / C^{(n)}$. As shown in (1), the weighted resource sharing is conducted proportionally to the assigned weight for each flow.

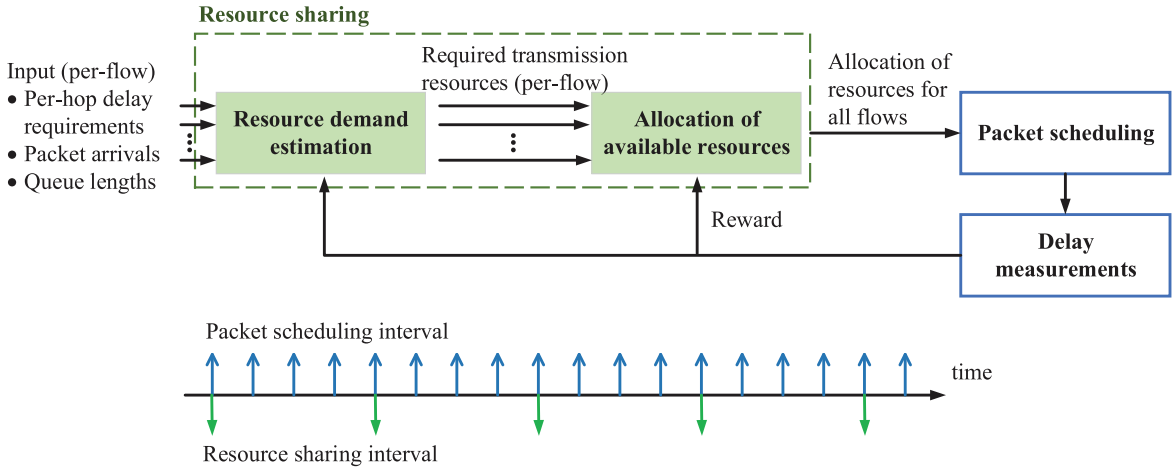


Fig. 2. Illustration of resource sharing for packet transmission at a switch.

Considering resource allocation for delay satisfaction, we describe the resource sharing efficiency as the delay satisfaction ratio achieved by per unit of allocated resources, denoted by $\rho_m^{(n)}/\eta_m^{(n)}$. The resource sharing optimization problem is formulated as (**P1**), in which our objective is to maximize the resource sharing efficiency at switch n under the resource constraints to determine the optimal decision of allocated available resources ($\{\Delta C_{f,m}^{(n)}\}_{f\in\mathcal{F}_n}$).

$$(\mathbf{P1}): \max_{\{\Delta C_{f,m}^{(n)}\}_{f\in\mathcal{F}_n}} \rho_m^{(n)}/\eta_m^{(n)}$$

$$\text{s.t.} \begin{cases} \Delta C_{f,m}^{(n)} \geq 0, f \in \mathcal{F}_n & \text{(2a)} \\ \sum_{f\in\mathcal{F}_n} \Delta C_{f,m}^{(n)} \leq C_I^{(n)}. & \text{(2b)} \end{cases}$$

Delay satisfaction ratio $\rho_m^{(n)}$ is determined based on the forwarding resources and the packet-level traffic information. Due to the uncertainty of traffic arrival patterns, it is difficult to establish an analytical model for the ratio. Thus, a model-free learning-based method is expected to solve the optimization problem, in which the delay satisfaction ratio can be measured as feedback to the algorithm.

III. Learning-Based Proactive Resource Sharing

In this section, we present a bandit learning-based proactive resource sharing framework to improve delay satisfaction of different services while achieving efficient utilization of network resources.

A. Proactive Resource Sharing Framework

For proactive resource sharing among all flows at each switch, we propose two modules, the resource demand estimation and allocation of available resources, as shown in Fig. 2. Within each resource sharing interval, there are integer multiple packet scheduling intervals. The resource sharing framework for packet transmission at a switch consists of the following four functionalities:

- *Resource demand estimation* – To make the resource allocation decisions, the resource demands from different flows are estimated, based on their delay requirements for per-hop packet transmission, the queue lengths for each flow, and the predicted numbers of arrived packets;
- *Allocation of available resources* – At the beginning of each resource sharing interval, the allocation of resources for all flows passing through the switch is updated to maximize the efficiency of resource sharing at the switch, based on the estimated resource demands;
- *Packet scheduling* – For packet transmission, the switch schedules packets from different flows at the egress port according to their allocated resource shares. The decision on packet scheduling is made at each scheduling interval, which is shorter than the resource sharing interval;
- *Delay measurements* – When a packet is being transmitted at the egress port, its delay at the switch is measured by comparing the time difference between the packet transmission instant and the packet arrival instant [32]. Based on the delay requirements, the overall delay satisfaction ratio at the switch is calculated at the end of each resource sharing interval. With the resource allocation ratio, the resource sharing efficiency is evaluated, which is fed back to the resource demand estimation module and the resource allocation module for updating the resource sharing decisions in the following intervals.

For each flow, the more allocated forwarding resources, the higher delay satisfaction ratio. However, the improvement of delay satisfaction achieved by allocating the same amount of resources to different flows may be different. Considering a flow with sufficient allocated resources, which already has a high delay satisfaction ratio, its performance improvement upon additional allocated resources is limited. On the contrary, allocating resources to a flow with low delay satisfaction ratio will lead to more significant performance improvement. Thus, when allocating resources to different flows, their resource demands should be considered as a feature of the flows, such that a higher resource demand indicates a larger potential improvement of overall delay satisfaction ratio.

The resource demands of flows in the resource sharing problem is analogous to user preferences in an advertisement click problem, referred to as context information. This type of context-based decision making problems can be formulated as contextual bandit problem, and solved through UCB methods [36]. Hence, we propose a MAB formulation with context information to describe the resource sharing problem in (**P1**). In our MAB-based resource sharing problem, an arm represents a potential resource allocation decision for all flows in each resource sharing interval. To maximize the resource sharing efficiency, the reward of the MAB problem is represented by the delay satisfaction ratio, and the optimal arm represents the resource allocation decision which achieves the highest ratio of the reward over the allocated resources. Before arm selection, the rewards of pulling different arms are estimated, considering the context information (i.e., the estimated resource demands in our problem). The achieved reward depends on both arm selection and context information, and the resource sharing among flows is iteratively converged through the learning process with reward feedback.

At the beginning of each resource sharing interval, the learning module makes resource sharing decision. Decision variables $\Delta C_{f,m}^{(n)}$ in (**P1**) is continuous, leading to an infinite number of arms. Since the reward distribution knowledge is learned through pulling different arms, it is necessary to make the arm set finite, i.e., a finite number of arms in the set. Thus, we discretize the available resources at switch *n* into $I^{(n)} = \lfloor C_I^{(n)}/B \rfloor$ resource blocks (RBs), each with the same amount of forwarding resources of *B* (in unit of bits per second). The decision variables $\Delta C_{f,m}^{(n)}$ are also discretized correspondingly as $\Delta I_{f,m}^{(n)}$. Hence, (**P1**) is transformed to (**P2**).

$$(\mathbf{P2}): \max_{\left\{\Delta I_{f,m}^{(n)}\right\}_{f\in\mathcal{F}_n}} \rho_m^{(n)}/\eta_m^{(n)}$$

$$\text{s.t.} \begin{cases} \Delta I_{f,m}^{(n)} \in \mathbb{N}, f \in \mathcal{F}_n & \text{(3a)} \\ \Delta C_{f,m}^{(n)} = \Delta I_{f,m}^{(n)} \times B & \text{(3b)} \\ \sum_{f\in\mathcal{F}_n} \Delta C_{f,m}^{(n)} \le C_I^{(n)}. & \text{(3c)} \end{cases}$$

A contextual-bandit algorithm proceeds in the discrete resource sharing intervals. At the beginning of the *m*th interval, switch *n* estimates its current resource demand, $[\hat{C}_{f,m}^{(n)}]_{f\in\mathcal{F}_n}$, based on reward information from the previous $(m-1)$ intervals $[\rho_i^{(n)}]_{i=1,2,\ldots,m-1}$. The arm is a vector of $|\mathcal{F}_n|$ integers, each element representing the number of allocated available resource blocks to a flow. Thus, each selected arm $[\Delta I_{f,m}^{(n)}]_{f\in\mathcal{F}_n}$ determines the allocation of available resources. To avoid redundant resource allocation for a flow, its resource demand is considered. Thus, the number of RBs that can be allocated to flow f at the *m*th interval should be less than $\lceil \frac{\hat{C}_{f,m}^{(n)} - \bar{C}_{f,m}^{(n)}}{B} \rceil$. As the total amount of available resources at switch *n* is $I^{(n)}$, for each flow, we obtain the upper bound of the number of allocated resource blocks $\Delta I_{f,m}^{(n)}$, as $U_{f,m}^{(n)} = \min[I^{(n)}, \max(\lceil \frac{\hat{C}_{f,m}^{(n)} - \bar{C}_{f,m}^{(n)}}{B} \rceil, 0)]$. Since the arm describes the allocation of available resources for all flows traversing the switch, the set of potential arms is denoted by $\mathcal{A}_{f,m}^{(n)}|_{f\in\mathcal{F}_n}$, where $\mathcal{A}_{f,m}^{(n)} = \{0, 1, 2, \ldots, U_{f,m}^{(n)}\}$, and the size of the arm set is $\prod_{f\in\mathcal{F}_n}(U_{f,m}^{(n)} + 1)$. In each interval, an arm is selected from $\mathcal{A}_{f,m}^{(n)}|_{f\in\mathcal{F}_n}$ to achieve the maximal potential reward $\hat{\rho}_m^{(n)}$, i.e., the estimated value of overall delay satisfaction ratio, $\rho_m^{(n)}$.

We propose the resource sharing framework to solve this contextual-bandit problem, with the resource demand estimation module for context information extraction and allocation module of available resource blocks for arm selection. Fig. 3 shows the operation procedure of allocating available resource blocks to different flows based on their estimated resource demands. The switch executes the following steps in the *m*th interval:

- For flow $f \in \mathcal{F}_n$, switch *n* estimates its current resource demand $\hat{C}_{f,m}^{(n)}$, and determines its arm set $\mathcal{A}_{f,m}^{(n)}$ accordingly, as discussed in Section III-B. Only the flows with $U_{f,m}^{(n)} > 0$ are processed in the following steps, to reduce computing complexity in a large-scale network scenario;
- Based on rewards $[\rho_i^{(n)}]_{i=1,2,\ldots,m-1}$ in the previous $(m-1)$ intervals and $\hat{C}_{f,m}^{(n)}$, switch *n* estimates the potential value of reward, $\hat{\rho}_m^{(n)}$, for each arm from $\mathcal{A}_{f,m}^{(n)}|_{f\in\mathcal{F}_n}$, as discussed in Section III-C. Then, the arm with the maximal potential reward is selected and the available resources are allocated;
- At the end of the interval, $\rho_m^{(n)}$ is observed and used as the feedback reward. With this new observation, the tuple, $([\hat{C}_{f,m}^{(n)}]_{f\in\mathcal{F}_n}, [\Delta I_{f,m}^{(n)}]_{f\in\mathcal{F}_n}, \rho_m^{(n)})$, including the context information, the selected arm, and reward, can be used to improve the arm-selection strategy. Note that only the selected arm has the feedback of reward.

B. Resource Demand Estimation

At the end of the *m*th resource sharing interval, to obtain delay satisfaction ratio $\rho_{f,m}^{(n)}$ for flow *f*, switch *n* measures the delay of staying at the switch for all x_R packets from flow f arriving within interval *m*. Denote the delay of each packet staying at the switch as d_i, $i = 1, 2, \ldots, x_R$. The delay satisfaction ratio $\rho_{f,m}^{(n)}$ is calculated as

$$\rho_{f,m}^{(n)} = \frac{1}{x_R} \sum_{i=1}^{x_R} \mathbf{1}\left(D_f^{(n)} - d_i\right) \tag{4}$$

where $\mathbf{1}(x) = 1$ if $x \ge 0$, otherwise $\mathbf{1}(x) = 0$. The delay measurement can be accomplished through pipelined packet processing, which includes the functionalities of reading and writing packet headers [32]. At the end of interval *m*, current queue length (i.e., initial queue length of interval $(m+1)$) and the number of arrived packets during interval *m*, denoted by $b_{f,m+1}^{(n)}$ and $\lambda_{f,m}^{(n)}$ respectively, are measured by the switch.

At switch *n*, for packets from flow *f* arriving within interval *m*, the required forwarding resources for delay satisfaction is

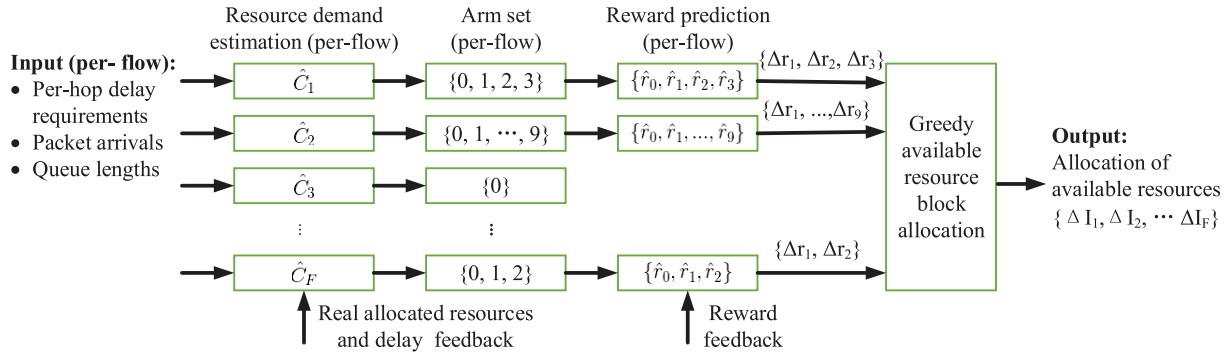


Fig. 3. Details of resource sharing framework.

determined by a linear combination of $\lambda_{f,m}^{(n)}/l$ and $b_{f,m}^{(n)}/l$ as in [16], where l is the average per-hop delay requirement. For supporting applications with stringent delay requirements, resource allocation decisions are made by considering the delay requirements, $D_f^{(n)}$, for individual packet transmission. At switch n, the resource demand for flow f in interval m, $\hat{C}_{f,m}^{(n)}$, is defined as the minimal amount of forwarding resources to satisfy $\rho_{f,m}^{(n)} \geq 1-\varepsilon$. Since both new packet arrivals and the packets waiting in the queue are to be processed in the next resource sharing interval, the resource demand is dependent on the predicted number of arrived packets $\hat{\lambda}_{f,m}^{(n)}$, $b_{f,m}^{(n)}$, and $D_f^{(n)}$. To estimate $\hat{C}_{f,m}^{(n)}$, we use a linear regression (LR) model to approximate the mapping relation between $(D_f^{(n)}, \hat{\lambda}_{f,m}^{(n)}, b_{f,m}^{(n)})$ and $\hat{C}_{f,m}^{(n)}$ [37]. The LR model is widely used in engineering areas, such as signal processing and financial engineering [38]. To approximate the mapping relation, the number of arrived packets, the observed queue length, and the measured delay bound to satisfy the per-hop delay violation ratio are collected at each resource sharing interval to train the model parameters. For better approximation accuracy, we combine an online weight update method (e.g., gradient decent algorithm in [39]) with linear regression. The detailed description of the linear regression based resource demand estimation is given in Algorithm 1, in which the initial model parameters are obtained in the training stage.

The resource demand estimation module takes $(D_f^{(n)}, \hat{\lambda}_{f,m}^{(n)}, b_{f,m}^{(n)})$ as input, and estimates corresponding $\hat{C}_{f,m}^{(n)}$ as output. We omit n and f from the symbols used in Algorithm 1 for clarity. Based on the LR model, $\hat{C}_m$ is estimated as

$$\hat{C}_m = w_m^1 \hat{\lambda}_m + w_m^2 \frac{\hat{\lambda}_m}{D} + w_m^3 b_m + w_m^4 \frac{b_m}{D} + w_m^5 \quad (5)$$

where the weight vector in the mth resource sharing interval is denoted as $\mathbf{W}_m = [w_m^1, \ldots, w_m^5]$, and the input vector is $\mathbf{I}_m = [\hat{\lambda}_m, \frac{\hat{\lambda}_m}{D}, b_m, \frac{b_m}{D}, 1]$.

As shown in Fig. 3, at the beginning of each resource sharing interval, say interval m, the amount of required resources is estimated and used as contextual information for the following allocation of available resources module. At the end of the interval, the switch can observe the actual number of arrived packets, λ_m, and the utilized forwarding resources, C_m. In addition, based on the measurement for the delay of each packet staying at the switch, d_i, we can determine D_m^R that satisfies $\frac{1}{x_R}\sum_{i=1}^{x_R} \mathbf{1}(D_m^R - d_i) = 1-\varepsilon$, with the allocated forwarding resources C_m in interval m. Hence, (D_m^R, λ_m, b_m) and C_m are used to iteratively refine the weight parameters of the LR model in (5). According to [39], $\mathbf{W}_m$ is updated as

$$\mathbf{W}_{m+1} = \mathbf{W}_m - \eta_{m+1}\Big(\mathbf{W}_m^T \mathbf{I}_m^R - C_m\Big)\mathbf{I}_m^R \quad (6)$$

where $\mathbf{I}_m^R = [\lambda_m, \frac{\lambda_m}{D_m^R}, b_m, \frac{b_m}{D_m^R}, 1]$, $\eta_{m+1} = \frac{1}{m+1}$.

Algorithm 1: Online Linear Regression Based Resource Demand Estimation Algorithm

```
Initialize W_1
for m = 1, 2, 3, ... do
    I_m ← [λ̂_m, λ̂_m/D, b_m, b_m/D, 1]
    Estimate Ĉ_m = W_m^T I_m
    Give Ĉ_m to allocation of available resources module
    At the end of interval, observe λ_m, C_m, and D_m^R
    I_m^R ← [λ_m, λ_m/D_m^R, b_m, b_m/D_m^R, 1]
    Update W_{m+1} ← W_m − η_{m+1}(W_m^T I_m^R − C_m) I_m^R
```

C. Allocation of Available Resource Blocks

To determine the amount of allocated available resources, switch n first estimates the potential reward $(\hat{\rho}_m^{(n)})$ by selecting each possible arm in $\mathcal{A}_{f,m}^{(n)}|_{f\in\mathcal{F}_n}$, which has $\prod_{f\in\mathcal{F}_n}(U_{f,m}^{(n)}+1)$ arms. However, the increasing number of flows leads to a growth of arm set with high computational complexity for reward estimation. To make the algorithm scalable in a large scale network, we estimate the reward on a per-flow basis. As illustrated in Fig. 3, switch n estimates the potential per-flow reward $(\hat{\rho}_{f,m}^{(n)})$ with the corresponding number of allocated available resource blocks ($a \in \mathcal{A}_{f,m}^{(n)}$), i.e., $\hat{\rho}_{f,m}^{(n)}$ is estimated under the action of $\Delta I_{f,m}^{(n)} = a$. After that, a greedy method is used to select the arm that achieves the highest ratio of potential reward $\hat{\rho}_m^{(n)}$ over the allocated resources.

In the following, we explain in detail how the per-flow performance is estimated. This is similar to the reward estimation in a contextual bandit problem, which observes the user

feature and potentially selected arm as prior knowledge. The problem is well solved by LinUCB, a variation of UCB method proposed as a generic contextual bandit algorithm in [36]. It is proved that a closed-form confidence interval (i.e., the deviation of reward estimation) can be computed efficiently when the reward model is linear. For each flow f, switch n runs one LinUCB to estimate per-flow reward. For clarity, we present the symbols used in the LinUCB based reward estimation algorithm, where indexes n and f are omitted. In the mth interval, we have

1) *Two-dimensional feature:* $\mathbf{x}_m = [1, \hat{C}_{f,m}^{(n)}]$, $\mathbf{x}_m \in \mathbb{R}^2$;
2) *Arm:* $a = \Delta I_{f,m}^{(n)}$, $a \in \mathcal{A}_{f,m}^{(n)}$;
3) *Reward:* $r_{m,a} = \hat{\rho}_{f,m}^{(n)}$, with the action of choosing arm a.

Under the assumption that the potential reward of a given arm is linear versus its observed feature $\mathbf{x}_m$ with unknown parameter vector θ_a^*, we have

$$\mathbf{E}\big[r_{m,a}|\mathbf{x}_m\big] = \mathbf{x}_m^T \theta_a^*. \tag{7}$$

If arm a was selected m' ($m' < m$) times in the previous ($m-1$) time intervals, the associated reward feedback can be used to improve the estimation of θ_a^*. Let $\mathbf{X}_{m,a}$ and $\mathbf{R}_{m,a}$ denote the previous m' feature vectors and observed rewards, where we have $\mathbf{X}_{m,a} \in \mathbb{R}^{m' \times 2}$ and $\mathbf{R}_{m,a} \in \mathbb{R}^{m'}$. With training data $\mathbf{X}_{m,a}$ and $\mathbf{R}_{m,a}$, parameter vector θ_a^* is estimated as

$$\hat{\theta}_{m,a} = \left(\mathbf{X}_{m,a}^T \mathbf{X}_{m,a} + \mathbf{I}_2\right)^{-1} \mathbf{X}_{m,a}^T \mathbf{R}_{m,a} \tag{8}$$

where $\mathbf{I}_2$ is the 2×2 identity matrix. For clarity, we denote $\mathbf{A}_{m,a} = \mathbf{X}_{m,a}^T \mathbf{X}_{m,a} + \mathbf{I}_2$, and $\mathbf{b}_{m,a} = \mathbf{X}_{m,a}^T \mathbf{R}_{m,a}$. Based on the confidence interval given in [36], with the probability of at least $(1-\delta)$, we have

$$\left|\mathbf{x}_m^T \hat{\theta}_{m,a} - \mathbf{E}\big[r_{m,a}|\mathbf{x}_m\big]\right| \le \alpha \sqrt{\mathbf{x}_m^T \mathbf{A}_{m,a}^{-1} \mathbf{x}_m} \tag{9}$$

for $\forall \delta > 0$, where $\alpha = 1 + \sqrt{\ln(2/\delta)/2}$ is a constant. Based on the analysis in [36], we set $\alpha = 1.5$. The inequality in (9) gives an upper bound of the reward estimated by (7)-(8). In the mth interval, the potential reward for arm a is

$$\hat{r}_{m,a} = \mathbf{x}_m^T \hat{\theta}_{m,a} + \alpha \sqrt{\mathbf{x}_m^T \mathbf{A}_{m,a}^{-1} \mathbf{x}_m}. \tag{10}$$

We define the marginal performance gain of allocating one more available RB to flow f as $\Delta r_{m,a}^f = \hat{r}_{m,a} - \hat{r}_{m,a-1}$, $a = 1, 2, \ldots, U_{f,m}^{(n)}$, where $\hat{r}_{m,a}$ is obtained by the LinUCB for flow f. Since a better delay satisfaction ratio should be achieved with more allocated forwarding resources, we have $\Delta r_{m,a}^f > 0$. Due to the constant size of RB, a greater $\Delta r_{m,a}^f$ indicates a larger delay satisfaction improvement achieved by the allocated RB. Thus, based on $\Delta r_{m,a}^f$, switch n allocates available RBs greedily for all flows in $\mathcal{F}_n$. Algorithm 2 gives a detailed description of how to allocate available RBs at switch n, and the details of greedy RB allocation are given in Algorithm 3.

To demonstrate how to allocate available RBs, we use the resource sharing among 5 flows as an example shown in Fig. 4. The thick red line indicates the upper bound of the amount of

Algorithm 2: Proposed Allocation Algorithm for Available Resource Blocks

for $m = 1, 2, 3, \ldots$ **do**
  Estimate resource demand (by Algorithm 1) for all flows $f \in \mathcal{F}_n$, $\hat{C}_{f,m}^{(n)}$
  for $f = 1, 2, 3, \ldots |\mathcal{F}_n|$ **do**
    Determine $\mathbf{x}_m = \left[1, \hat{C}_{f,m}^{(n)}\right]$ and $U_{f,m}^{(n)}$
    for $a = 0, 1, 2, 3, \ldots U_{f,m}^{(n)}$ **do**
      if *a is new* **then**
        $\mathbf{A}_{m,a} \leftarrow \mathbf{I}_2$
        $\mathbf{b}_{m,a} \leftarrow \mathbf{0}_{2\times 1}$ (zero vector)
      $\hat{\theta}_{m,a} \leftarrow \mathbf{A}_{m,a}^{-1} \mathbf{b}_{m,a}$
      $\hat{r}_{m,a} \leftarrow \mathbf{x}_m^T \hat{\theta}_{m,a} + \alpha \sqrt{\mathbf{x}_m^T \mathbf{A}_{m,a}^{-1} \mathbf{x}_m}$
      if $a = 0$ **then**
        $\Delta r_{m,a}^f = \hat{r}_{m,a}$
      else
        $\Delta r_{m,a}^f = \hat{r}_{m,a} - \hat{r}_{m,a-1}$
  Execute Algorithm 3
  Implement the resource sharing decision $\left[C_{f,m}^{(n)}\right]_{f \in \mathcal{F}_n}$ through packet scheduling module
  At the end of interval, $\rho_{f,m}^{(n)}$ is observed as the feedback to all flows $f \in \mathcal{F}_n$ as real-valued reward $r_{f,m}$
  for $f = 1, 2, 3, \ldots |\mathcal{F}_n|$ **do**
    $a = a_{f,m}$, $r = r_{f,m}$
    $\mathbf{A}_{m,a} \leftarrow \mathbf{A}_{m,a} + \mathbf{x}_m \mathbf{x}_m^T$
    $\mathbf{b}_{m,a} \leftarrow \mathbf{b}_{m,a} + r\mathbf{x}_m$

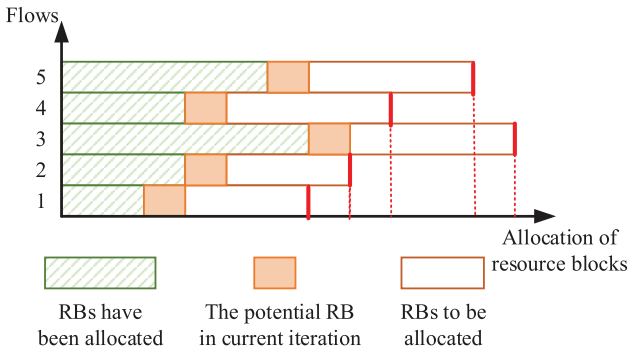


Fig. 4. Illustration of the greedy available resource block allocation.

RBs that can be allocated to each flow, and the green blocks indicate the amount of RBs already allocated to the flows. The blocks highlighted in orange indicate the potential RB allocation to the flows in each algorithm iteration, if the highest marginal gain (e.g., $\Delta r_{m,a}^f$ for flow f) is observed. In Algorithm 3, one RB is allocated in each iteration, until all the available RBs at switch n have been allocated or all the flows get sufficient RBs (i.e., their upper bounds are reached).

D. Discussion

1) Regret Analysis: As shown in (7), the potential reward is linear with respect to the observed feature $\mathbf{x}_m$, and the true coefficient vector is θ_a^*. Without loss of generality, we assume $\|\mathbf{x}_m\| \le L$, where $\|\cdot\|$ represents the l_2-norm.

Algorithm 3: Greedy Allocation of Available Resource Blocks

Input: $\Delta r^{f}_{m,a}$
Initialization: $\mathcal{F}_t \leftarrow \{\}$
for $f = 1, 2, 3, \ldots |\mathcal{F}_n|$ **do**
  $a_{f,m} \leftarrow 0$
  if $U^{(n)}_{f,m} > 0$ **then**
    $\mathcal{F}_t \leftarrow \mathcal{F}_t + \{f\}$
for $i = 1, 2, 3, \ldots I^{(n)}$ **do**
  if $\mathcal{F}_t$ *is not empty* **then**
    $f_i = \arg\max_{f \in \mathcal{F}_t} \Delta r^{f}_{m,a}$ where $a = a_{f,m} + 1$. Randomly select f_i if multiple flows have the maximal $\Delta r_{m,a}$
    $a_{f_i,m} \leftarrow a_{f_i,m} + 1$
    if $a_{f_i,m} = A^{(n)}_{f_i,m}$ **then**
      $\mathcal{F}_t \leftarrow \mathcal{F}_t - \{f_i\}$
for $f = 1, 2, 3, \ldots |\mathcal{F}_n|$ **do**
  $\Delta I^{(n)}_{f,m} \leftarrow a_{f,m}$
Based on $\left[\Delta I^{(n)}_{f,m}\right]_{f\in\mathcal{F}_n}$, obtain the corresponding resource sharing decision $\left[C^{(n)}_{f,m}\right]_{f\in\mathcal{F}_n}$
Output: Resource sharing decision $\left[C^{(n)}_{f,m}\right]_{f\in\mathcal{F}_n}$ and the selected arm $\left[a_{f,m}\right]_{f\in\mathcal{F}_n}$

In the mth interval, denote the best arm $[a^*_{f,m}]_{f\in\mathcal{F}_n}$ that satisfies

$$\left[a^*_{f,m}\right]_{f\in\mathcal{F}_n} = \underset{[a_f]_{f\in\mathcal{F}_n}}{\arg\max} \sum_{f\in\mathcal{F}_n} \mathbf{x}_m^T \theta^*_{a_f} \tag{11}$$

where $a_f \in \mathcal{A}^{(n)}_{f,m}$. Then, comparing with the reward achieved by the selected arm $[a_{f,m}]_{f\in\mathcal{F}_n}$, the M-trail regret of this resource allocation algorithm is calculated by

$$L_M = \sum_{m=1}^{M} \left(\sum_{f\in\mathcal{F}_n} \mathbf{x}_m^T \theta^*_{a^*_{f,m}} - \sum_{f\in\mathcal{F}_n} \mathbf{x}_m^T \theta^*_{a_{f,m}} \right). \tag{12}$$

Considering the confidence interval in (9), with probability at least $1-\delta$, we have

$$\sum_{f\in\mathcal{F}_n} \mathbf{x}_m^T \theta^*_{a_{f,m}} \geq \sum_{f\in\mathcal{F}_n} \left(\mathbf{x}_m^T \hat{\theta}_{m,a_{f,m}} - \alpha \sqrt{\mathbf{x}_m^T \mathbf{A}^{-1}_{m,a_{f,m}} \mathbf{x}_m} \right). \tag{13}$$

According to the proposed available resource block allocation algorithm and (9), we obtain

$$\begin{aligned} \sum_{f\in\mathcal{F}_n} \mathbf{x}_m^T \theta^*_{a^*_{f,m}} &\leq \sum_{f\in\mathcal{F}_n} \left(\mathbf{x}_m^T \hat{\theta}_{m,a^*_{f,m}} + \alpha \sqrt{\mathbf{x}_m^T \mathbf{A}^{-1}_{m,a^*_{f,m}} \mathbf{x}_m} \right) \\ &\leq \sum_{f\in\mathcal{F}_n} \left(\mathbf{x}_m^T \hat{\theta}_{m,a_{f,m}} + \alpha \sqrt{\mathbf{x}_m^T \mathbf{A}^{-1}_{m,a_{f,m}} \mathbf{x}_m} \right). \end{aligned} \tag{14}$$

Thus,

$$L_M \leq \sum_{m=1}^{M} \left(\sum_{f\in\mathcal{F}_n} 2\alpha \sqrt{\mathbf{x}_m^T \mathbf{A}^{-1}_{m,a_{f,m}} \mathbf{x}_m} \right). \tag{15}$$

To simplify (15), we apply [40, Lemma 4.4 and Lemma 4.5], as

$$\sum_{m=1}^{M} \mathbf{x}_m^T \mathbf{A}^{-1}_{m,a_{f,m}} \mathbf{x}_m \leq 2 \log \frac{\left|\mathbf{A}_{m,a_{f,m}}\right|}{\beta} \tag{16}$$

$$\left|\mathbf{A}_{m,a_{f,m}}\right| \leq \left(\beta + mL^2/d\right)^d \tag{17}$$

where $d \geq 1$ and $\beta \geq \max(1, L^2)$. Apply the lemmas in (15), we have

$$\begin{aligned} L_M &\leq 2\alpha \sum_{f\in\mathcal{F}_n} \left(\sum_{m=1}^{M} \sqrt{\mathbf{x}_m^T \mathbf{A}^{-1}_{m,a_{f,m}} \mathbf{x}_m} \right) \\ &\leq 2\alpha \sum_{f\in\mathcal{F}_n} \sqrt{M \left(\sum_{m=1}^{M} \mathbf{x}_m^T \mathbf{A}^{-1}_{m,a_{f,m}} \mathbf{x}_m \right)} \\ &\leq 2\alpha \sum_{f\in\mathcal{F}_n} \sqrt{2M \log \frac{\left|\mathbf{A}_{m,a_{f,m}}\right|}{\beta}} \\ &\leq 2\alpha \sum_{f\in\mathcal{F}_n} \sqrt{2Md \cdot \log\left(1 + \frac{ML^2}{\beta d}\right)} \\ &\leq 2\alpha F \sqrt{2Md} \cdot \sqrt{\log(1 + M/d)}. \end{aligned} \tag{18}$$

In (18), the M-trail regret bound, $O(\sqrt{Md \cdot \log(1+M/d)})$, indicates the zero-regret feature,[1] i.e., $\lim_{M\to\infty} \frac{L_M}{M} = 0$. The zero-regret strategy is guaranteed to converge to an optimal strategy after enough rounds are played [41]. Thus, the proposed algorithm is proved to be asymptotically approaching the optimal strategy [42].

2) Time Complexity Analysis: To analyze the scalability of the proposed resource sharing scheme in terms of the number of flows (F), we evaluate the time complexity for each stage in Fig. 3 in one interval. First, the resource demand estimation module runs Algorithm 1 for each flow separately, which leads to $O(F)$ time complexity. Then, in Algorithm 2, the reward is calculated for all the arms, where flow f has $(U^{(n)}_{f,m}+1)$ arms, according to its resource demand. Thus, the complexity is proportional to the total number of arms, $\sum_{f\in\mathcal{F}_n}(U^{(n)}_{f,m}+1)$. The overall number of resource blocks required by flows should be comparable to the number of available resource blocks ($I^{(n)}$). Otherwise, the network would incur unstable queue accumulation. We obtain the complexity of reward calculation in Algorithm 2 as $O(I^{(n)})$. After that, each resource block is allocated by comparing the marginal reward for candidate flows, as given in Algorithm 3. The maximal time complexity is $O(F)$, in which all the F flows belong to the candidate set. As the comparison is conducted at most for $I^{(n)}$ times, the

[1] A strategy whose average regret per round tends to zero when the horizon tends to infinity is defined as zero-regret strategy [41].

TABLE I
PARAMETERS OF TRAFFIC FLOW

Number of flows (F)	50
Average traffic rate	15 Mbps
Delay violation tolerance (ε)	0.5 %
Per-hop delay requirement - number of flows	0.5 ms - 10
	0.7 ms - 15
	1.5 ms - 25

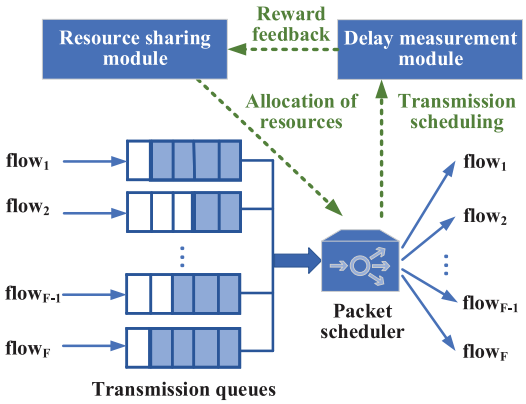


Fig. 5. Illustration of the simulation scenario.

time complexity of Algorithm 3 is $O(FI^{(n)})$. Therefore, combining the complexity of three algorithms, the time complexity of the resource sharing scheme is $O(FI^{(n)})$.

Since $I^{(n)}$ is determined by the amount of available resources and the resource block size, we can set the value of $I^{(n)}$ by adjusting the block size. If we set $I^{(n)}$ as a fixed value, the complexity is reduced to $O(F)$. However, when the number of flows increases, the resource sharing scheme with the fixed $I^{(n)}$ may have a degraded performance. In particular, if there are more flows than resource blocks, i.e., $F > I^{(n)}$, the resources allocated to different flows are distinct even if all the flows have same features. To avoid this inconsistency between the allocated resources and the observed features due to insufficient resource blocks, we set $I^{(n)} \propto F$, and the polynomial time complexity, $O(F^2)$, is achieved. Note that if the resource block size has to be set as a small value with the increase of F in order to satisfy $I^{(n)} \propto F$, the improvement on delay satisfaction by allocating one resource block could be insignificant, leading to a low marginal gain. Hence, a lower limit on the resource block size needs to be determined properly, considering the allocation efficiency, which remains an important and open issue for the proposed algorithm.

IV. NUMERICAL RESULTS

To demonstrate the effectiveness of the proposed resource sharing scheme, numerical results are presented.

A. Simulation Scenario

In our work, the resource allocation algorithm is designed for each switch, aiming at satisfying the per-hop delay requirement. To evaluate the performance of the proposed algorithm, we consider the packet transmission at a network switch in

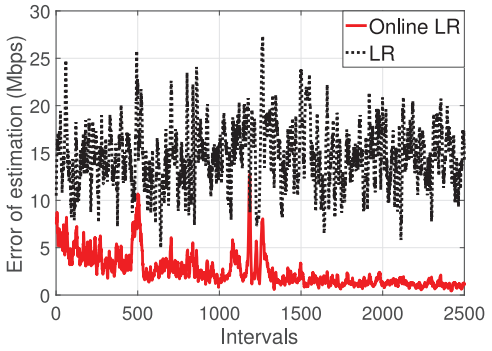


Fig. 6. Error of resource demand estimation.

the simulation. For the F traffic flows traversing the switch, the resource sharing among them is simulated and the delay of each packet passing the switch is measured, as shown in Fig. 5. The network switch, consisting of the transmission queues, packet scheduler, resource sharing module, and delay measurement module, is emulated on a high-performance computer server using a Python IDE called PyCharm [43]. For each traffic flow, its packets enter the corresponding transmission queue and wait for transmission scheduling. Given the amount of resources allocated to each traffic flow, the packet scheduler makes packet forwarding decisions, i.e., adjusting the forwarding rate for each flow. During the packet transmission, the delay measurement module records the packet reception time and departing time to measure the packet delay at this switch. At the end of each resource sharing interval, the delay measurement module calculates the delay satisfaction ratio for each flow, which is utilized by the resource sharing module to make future resource allocation decisions.

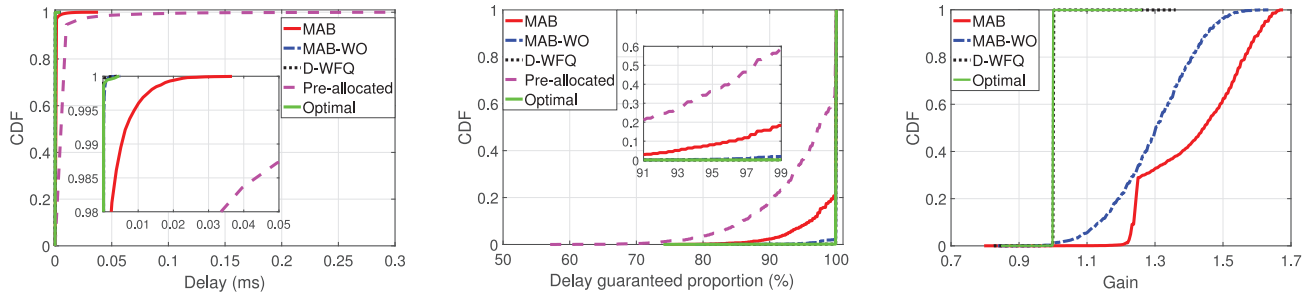
The detailed parameters of the traffic flows are given in Table I. To simulate flows of different services, we divided the 50 flows into 3 subsets with different per-hop delay requirements. The duration of each resource sharing interval is 5 ms. From Table I, the average traffic rate at this switch is 750 Mbps, considering 50 flows traversing the switch with average traffic rate of 15 Mbps. In reality, the traffic volumes are different during peak and off-peak hours, which leads to varying average resource utilization of the switch. To simulate these variations, we set the switch forwarding resources for the off-peak hour case as 1250 Mbps (i.e., 60% resource utilization) and for the peak hour case as 833 Mbps (i.e., 90% resource utilization). In addition to traffic load variations, we consider different pre-allocated resource conditions that impact the demand of resource sharing, including over-provisioning and on-demand matching cases. The pre-allocated conditions are determined by both the pre-allocated forwarding resources and the average traffic rate of each flow (15 Mbps in our simulation). Parameters of these cases are given in Table II, where the minimal resource allocation ratio is the ratio of overall pre-allocated resources over the switch forwarding resources.

B. Performance of Resource Demand Estimation

To evaluate the accuracy of resource demand estimation, we divide the traffic data trace into two sets, one is used

TABLE II
PARAMETERS OF SWITCH

Network traffic condition	Off-peak		Peak
Switch forwarding resources (Mbps)	1250		833
Available resource block size (Mbps)	3.5	5	0.83
Pre-allocated resources (Mbps / flow)	18	15	15
Minimal resource allocation ratio	0.72	0.6	0.9
Pre-allocated condition	Over-provisioning	Matching traffic	



(a) Packet delay (b) Delay guaranteed proportion (c) Resource allocation gain

Fig. 7. The performance in a off-peak hour condition with matching traffic resources.

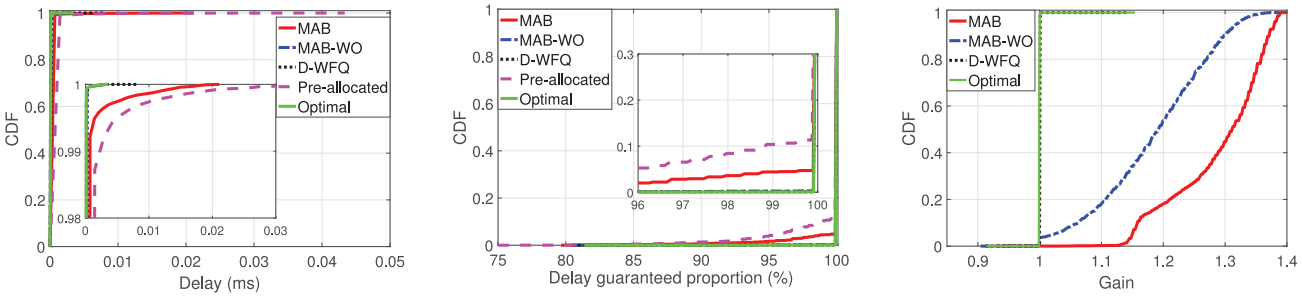
as training data set (including 489,930 observations) and the remaining is used for testing. Each observation represents the flow transmission status during one resource sharing interval, consisting of input vector (including the number of arrived packets, initial queue length, and the measured delay bound to satisfy the per-hop delay violation ratio), and the output (allocated forwarding resources). In the training stage, all the observations are fed into an LR module given by (5), and a weight vector $\mathbf{W} \in \mathbb{R}^5$ is obtained by the LR module. Then, we apply the online resource demand estimation algorithm with the initial vector $\mathbf{W}$ to estimate the required resources. The estimation error in each interval represents the difference between an estimated amount and an actual amount of allocated resources. To show the performance of resource demand estimation, the moving average of errors during 10 consecutive intervals is shown in Fig. 6 where online LR refers to the proposed method and LR method utilizing $\mathbf{W}$ for estimation without weights update. Without the weights update, the error of LR method varies between 15% and 35%. However, due to the weights update in (6), the estimation error of online LR shows a decreasing trend in Fig. 6, and becomes stable around 1% after 1500 intervals. Therefore, the proposed resource estimation module is capable of providing an accurate resource demand estimation for the allocation of available resources module.

C. Delay Reduction via Resource Sharing

To evaluate the delay performance of the resource sharing scheme, we determine the proportion of flows that meet their delay requirements (i.e., satisfying $\rho_{f,m}^{(n)} \geq 1-\varepsilon$), referred to as delay guaranteed proportion. In terms of cost, we measure the resource allocation fraction including both pre-allocated resources and allocated available resources. For comparison between different resource sharing schemes, the cumulative distribution functions (CDF) of packet delay, delay guaranteed proportion, and resource allocation gain which is the ratio of delay guaranteed proportion over resource allocation ratio are calculated for 4,000 intervals.

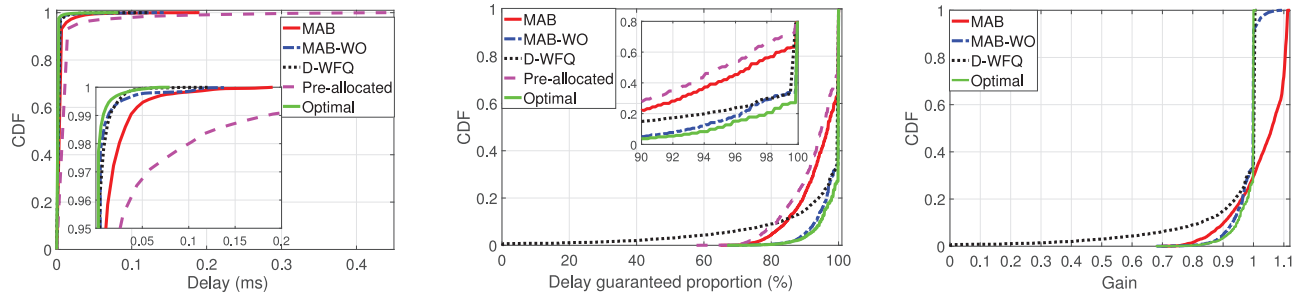
The proposed MAB based resource sharing scheme is compared with an on-switch resource allocation approach, the dynamic WFQ (D-WFQ) method [16]. The D-WFQ is an enhanced version of WFQ under dynamic traffic conditions, considering the differentiated per-hop delay requirements of traffic flows. We also compare with the resource sharing scheme without resource demand estimation module (MAB-WO). In the MAB-WO method, the allocation of available resources module directly uses the per-hop delay requirements, the predicted traffic arrival, and the queue lengths as input, instead of using the estimated resource demands as shown in Fig. 3. To show the performance improvement achieved by utilizing available resources, we simulate the packet transmissions with only pre-allocated resources given in Table II (pre-allocated). If the forwarding resources are fully used, we make a fixed resource sharing decision to achieve the optimal accumulative delay guaranteed proportion during the simulation intervals (optimal).

When the flows are pre-allocated with the resources matched their packet arrival rates, the packet delay experiences a long-tailed distribution as shown in Fig. 7. It leads to the delay guaranteed proportion less than 90% among 20% of intervals. However, during the off-peak hours, all four resource sharing schemes achieve nearly 100% delay guaranteed proportion through allocating available resources. Since MAB and MAB-WO schemes require less forwarding resources than D-WFQ and optimal schemes, they have higher resource allocation gain, as shown in Fig. 7. This is because MAB schemes can learn to allocate the available resources to the flows for the highest marginal gain, through the greedy allocation algorithm



(a) Packet delay (b) Delay guaranteed proportion (c) Resource allocation gain

Fig. 8. The performance in a off-peak hour condition with over-provisioning resources.



(a) Packet delay (b) Delay guaranteed proportion (c) Resource allocation gain

Fig. 9. The performance in a peak-hour condition with matching traffic resources.

in Algorithm 3. Comparing MAB and MAB-WO schemes, the resource demand estimation module makes it possible to identify the flows with stringent delay requirements, and set higher upper bounds of allocated available resource blocks. As more available resources can be allocated to the flows with stringent requirements, MAB outperforms MAB-WO in most resource sharing intervals in terms of resource allocation gain. In over-provisioning situations, we simulate the case that each flow is pre-allocated with more resources than necessary on average. Better delay guaranteed proportion is shown in Fig. 8, compared with the on-demand matching case. However, due to the high resource allocation ratio in the over-provisioning case, its resource allocation gain is lower than the on-demand matching case.

In the peak-hour scenario, all four resource sharing schemes tend to make full use of the resources to support the heavy traffic, and experience delay guaranteed proportion higher than 90% among 80% of the simulation intervals, as shown in Fig. 9. Comparing with transmission upon pre-allocated resources, the MAB method has a higher delay guaranteed proportion by avoiding the long-tailed distribution of packet delay. Due to the small amount of available resources during peak hours, it is unlikely to allocate the required amount of available resources to a flow, based on the estimated resource demand. Thus, comparing with the off-peak hour case, the difference between MAB and MAB-WO methods vanishes during peak hours.

The highest resource allocation gain is obtained when 100% delay guaranteed proportion is achieved with the pre-allocated resources. Thus, resource allocation gain is bounded by the reciprocal of minimal resource allocation ratio (i.e., the ratio of overall pre-allocated resources over the switch forwarding resources). Since the optimal and D-WFQ schemes make full use of available forwarding resources, the resource allocation gain is bounded by 1. However, MAB and MAB-WO schemes can achieve better delay guaranteed proportion with less resources, and their gains outperform the optimal and D-WFQ schemes. Due to the errors at the starting stage of resource demand estimation, the MAB scheme experiences over-allocation of available resources compared with the subsequent stable stages. Thus, there is a step change in the resource allocation gain, which becomes negligible with more fine-grained available resource block sizes. As shown in Fig. 7 to Fig. 9, the step shrinks with the available resource block size decreases from 5 Mbps, to 3.5 Mbps, and to 0.83 Mbps, respectively.

D. Adaptive Resource Sharing

Due to insufficient forwarding resources, instantaneous delay degradation happens during traffic peaks, such as in intervals 25, 87, and 125, as shown in Fig. 10, where the real-time packet arrival rate is normalized to the average traffic rate. Although all three schemes experience degraded performance, both MAB and MAB-WO schemes outperform D-WFQ, due to the proactive resource allocation.

Furthermore, to adapt to traffic dynamics, it requires a fast response of resource allocation when traffic peak comes. Fig. 11 shows how the MAB scheme adaptively allocates resources. During traffic peak, packet delay increases, and more forwarding resources are needed. To obtain the response time to traffic peak, we focus on the time instants where

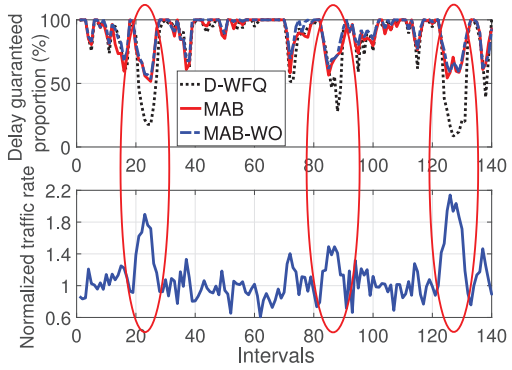
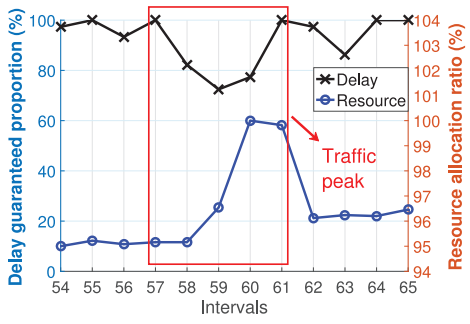
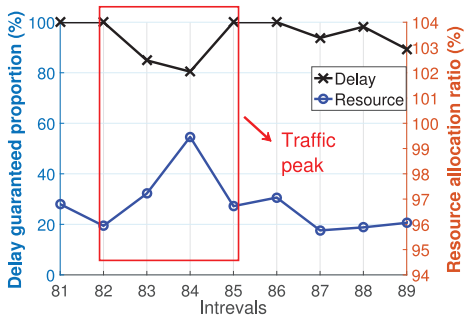


Fig. 10. Resource sharing performance in a peak condition with matching traffic resources.



(a) Traffic peak 1



(b) Traffic peak 2

Fig. 11. The response of the proposed resource sharing scheme to traffic peaks.

resource allocation ratio starts to increase or falls back to a stable level. Comparing with the traffic peak duration, it is observed that the response time is at most one interval when traffic peak appears and disappears, which is 5 ms in our simulation. Therefore, the proposed resource sharing scheme demonstrates adaptation capability to traffic dynamics, by allocating suitable resources to the flows.

V. Conclusion

In this article, we have proposed a novel learning-based proactive resource sharing scheme to maximize resource utilization efficiency with delay satisfaction. Two modules for estimating online resource demand and allocating available resources are developed jointly to achieve efficient resource sharing at each network switch. To learn the implicit relation between the allocated resources and differentiated delay requirements from traffic flows of different services, a multi-armed bandit learning-based resource allocation scheme is proposed, which enables fast and proactive resource adjustment upon traffic variations. During the data transmission, delay satisfaction ratios are measured as the reward feedback to refine the learning parameters for better convergence. The proposed scheme is proved to be asymptotically approaching the optimal strategy with the polynomial time complexity. Extensive simulation results are presented to demonstrate both the advantages of the proposed resource sharing scheme over conventional schemes and the robustness to traffic dynamics. For future work, the proposed joint resource demand estimation and resource allocation framework will be extended for reliable end-to-end packet transmission.

Acknowledgment

The authors would like to thank the undergraduate research assistant, Mingxi Zhang, for her help in conducting simulations.

References

[1] S. Manjunath and G. Raina, "Stability and performance of compound TCP with a proportional integral queue policy," *IEEE Trans. Control Syst. Technol.*, vol. 27, no. 5, pp. 2139–2155, Sep. 2019.

[2] M. Dong, Q. Li, D. Zarchy, P. B. Godfrey, and M. Schapira, "PCC: Re-architecting congestion control for consistent high performance," in *Proc. 12th USENIX Conf. Netw. Syst. Design Implement.*, May 2015, pp. 395–408.

[3] N. Cardwell, Y. Cheng, C. S. Gunn, S. H. Yeganeh, and V. Jacobson, "BBR: Congestion-based congestion control," *Commun. ACM*, vol. 60, no. 2, pp. 58–66, Feb. 2017.

[4] S. Floyd, K. Ramakrishnan, and D. L. Black, "The addition of explicit congestion notification (ECN) to IP," IETF, RFC 3168, Sep. 2001.

[5] G. Carlucci, L. De Cicco, and S. Mascolo, "Controlling queuing delays for real-time communication: The interplay of E2E and AQM algorithms," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 46, no. 3, pp. 1–7, Jul. 2018.

[6] J. Han, K. Xue, Y. Xing, P. Hong, and D. S. Wei, "Measurement and redesign of BBR-based MPTCP," in *Proc. ACM SIGCOMM Conf. Posters Demos*, Aug. 2019, pp. 75–77.

[7] J. Chen *et al.*, "SDATP: An SDN-based traffic-adaptive and service-oriented transmission protocol," *IEEE Trans. Cogn. Commun.*, vol. 6, no. 2, pp. 756–770, Jun. 2020.

[8] C. Gao, V. Rajabian-Schwart, W. Zhang, G. Xue, and J. Tang, "How would you like your packets delivered? An SDN-enabled open platform for QoS routing," in *Proc. IEEE/ACM 26th Int. Symp. Qual. Services (IWQoS)*, Banff, AB, Canada, Jun. 2018, pp. 1–10.

[9] J. W. Guck, A. Van Bemten, and W. Kellerer, "DetServ: Network models for real-time QoS provisioning in SDN-based industrial environments," *IEEE Trans. Netw. Service Manag.*, vol. 14, no. 4, pp. 1003–1017, Dec. 2017.

[10] Q. Ye, W. Zhuang, X. Li, and J. Rao, "End-to-end delay modeling for embedded VNF chains in 5G core networks," *IEEE Internet Things J.*, vol. 6, no. 1, pp. 692–704, Feb. 2019.

[11] C. Dovrolis, D. Stiliadis, and P. Ramanathan, "Proportional differentiated services: Delay differentiation and packet scheduling," *IEEE/ACM Trans. Netw.*, vol. 10, no. 1, pp. 12–26, Feb. 2002.

[12] E. Davies, M. A. Carlson, W. Weiss, D. Black, S. Blake, and Z. Wang, "An architecture for differentiated services," IETF, RFC 2475, Dec. 1998.

[13] A. Demers, S. Keshav, and S. Shenker, "Analysis and simulation of a fair queueing algorithm," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 19, no. 4, pp. 1–12, Sep. 1989.

[14] M. Shreedhar and G. Varghese, "Efficient fair queuing using deficit round-robin," *IEEE/ACM Trans. Netw.*, vol. 4, no. 3, pp. 375–385, Jun. 1996.

[15] M. Menth, M. Mehl, and S. Veith, "Deficit round Robin with limited deficit savings (DRR-LDS) for fairness among TCP users," in *Proc. Int. Conf. Meas. Model. Eval. Comput. Syst. (MMB)*, Jan. 2018, pp. 188–201.

[16] C.-C. Li, S.-L. Tsao, M. C. Chen, Y. Sun, and Y.-M. Huang, "Proportional delay differentiation service based on weighted fair queuing," in *Proc. IEEE 9th Int. Conf. Comput. Commun. Netw. (ICCCN)*, Las Vegas, NV, USA, Oct. 2000, pp. 418–423.
[17] K. Bao, J. D. Matyjas, F. Hu, and S. Kumar, "Intelligent software-defined mesh networks with link-failure adaptive traffic balancing," *IEEE Trans. Cogn. Commun. Netw.*, vol. 4, no. 2, pp. 266–276, Jun. 2018.
[18] K. Chen and L. Huang, "Timely-throughput optimal scheduling with prediction," *IEEE/ACM Trans. Netw.*, vol. 26, no. 6, pp. 2457–2470, Dec. 2018.
[19] M. D. F. De Grazia, D. Zucchetto, A. Testolin, A. Zanella, M. Zorzi, and M. Zorzi, "QoE multi-stage machine learning for dynamic video streaming," *IEEE Trans. Cogn. Commun. Netw.*, vol. 4, no. 1, pp. 146–161, Mar. 2018.
[20] J. Li, W. Shi, N. Zhang, and X. Shen, "Delay-aware VNF scheduling: A reinforcement learning approach with variable action set," *IEEE Trans. Cogn. Commun. Netw.*, early access, Apr. 21, 2020, doi: 10.1109/TCCN.2020.2988908.
[21] L. Wang *et al.*, "Scheduling with machine-learning-based flow detection for packet-switched optical data center networks," *IEEE J. Opt. Commun. Netw.*, vol. 10, no. 4, pp. 365–375, Apr. 2018.
[22] I.-S. Comşa *et al.*, "Towards 5G: A reinforcement learning-based scheduling solution for data traffic management," *IEEE Trans. Netw. Service Manag.*, vol. 15, no. 4, pp. 1661–1675, Dec. 2018.
[23] J. Luo, X. Su, and B. Liu, "A reinforcement learning approach for multipath TCP data scheduling," in *Proc. IEEE 9th Annu. Comput. Commun. Workshop Conf. (CCWC)*, Las Vegas, NV, USA, Jan. 2019, pp. 0276–0280.
[24] J. Zhu, Y. Song, D. Jiang, and H. Song, "A new deep-Q-learning-based transmission scheduling mechanism for the cognitive Internet of Things," *IEEE Internet Things J.*, vol. 5, no. 4, pp. 2375–2385, Aug. 2018.
[25] O. Avner and S. Mannor, "Multi-user communication networks: A coordinated multi-armed bandit approach," *IEEE/ACM Trans. Netw.*, vol. 27, no. 6, pp. 2192–2207, Dec. 2019.
[26] A. Mukherjee, S. Misra, V. S. P. Chandra, and M. S. Obaidat, "Resource-optimized multiarmed bandit-based offload path selection in edge UAV swarms," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4889–4896, Jun. 2019.
[27] W. Wu, N. Cheng, N. Zhang, P. Yang, W. Zhuang, and X. Shen, "Fast mmwave beam alignment via correlated bandit learning," *IEEE Trans. Wireless Commun.*, vol. 18, no. 12, pp. 5894–5908, Dec. 2019.
[28] P. Auer, "Using confidence bounds for exploitation-exploration tradeoffs," *J. Mach. Learn. Res.*, vol. 3, pp. 397–422, Nov. 2002.
[29] W. Chu, L. Li, L. Reyzin, and R. Schapire, "Contextual bandits with linear payoff functions," in *Proc. 14th Int. Conf. Artif. Intell. Stat. (AISTATS)*, Apr. 2011, pp. 208–214.
[30] M. M. Tajiki, S. Salsano, L. Chiaraviglio, M. Shojafar, and B. Akbari, "Joint energy efficient and QoS-aware path allocation and VNF placement for service function chaining," *IEEE Trans. Netw. Service Manag.*, vol. 16, no. 1, pp. 374–388, Mar. 2019.
[31] Q. Ye, J. Li, K. Qu, W. Zhuang, X. Shen, and X. Li, "End-to-end quality of service in 5G networks: Examining the effectiveness of a network slicing framework," *IEEE Veh. Technol. Mag.*, vol. 13, no. 2, pp. 65–74, Jun. 2018.
[32] *OpenFlow Switch Specification* Open Netw. Found., Menlo Park, CA USA, Mar. 2015.
[33] M. Shahbaz *et al.*, "PISCES: A programmable, protocol-independent software switch," in *Proc. ACM SIGCOMM Conf.*, Aug. 2016, pp. 525–538.
[34] O. Alhussein *et al.*, "A virtual network customization framework for multicast services in NFV-enabled core networks," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 6, pp. 1025–1039, Jun. 2020.
[35] O. Alhussein and W. Zhuang, "Robust online composition, routing and NF placement for NFV-enabled services," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 6, pp. 1089–1101, Jun. 2020.
[36] L. Li, W. Chu, J. Langford, and R. E. Schapire, "A contextual-bandit approach to personalized news article recommendation," in *Proc. ACM Int. Conf. World Wide Web (WWW)*, Apr. 2010, pp. 661–670.
[37] X. Yan and X. Su, *Linear Regression Analysis: Theory and Computing*. Hackensack, NJ, USA: World Sci., 2009.
[38] Y. Feng and D. P. Palomar, "A signal processing perspective on financial engineering," *Found. Trends Signal Process.*, vol. 9, nos. 1–2, pp. 1–231, 2016.
[39] E. Hazan, A. Rakhlin, and P. L. Bartlett, "Adaptive online gradient descent," in *Proc. Conf. Neural Inf. Process. Syst. (NIPS)*, Dec. 2007, pp. 65–72.
[40] L. Zhou, "A survey on contextual multi-armed bandits," Aug. 2015. [Online]. Available: arXiv:1508.03326.
[41] J. Vermorel and M. Mohri, "Multi-armed bandit algorithms and empirical evaluation," in *Proc. Eur. Conf. Mach. Learn. (ECML)*, Oct. 2005, pp. 437–448.
[42] P. Yang, N. Zhang, S. Zhang, L. Yu, J. Zhang, and X. Shen, "Content popularity prediction towards location-aware mobile edge caching," *IEEE Trans. Multimedia*, vol. 21, no. 4, pp. 915–929, Apr. 2019.
[43] *Pycharm*. Accessed: Aug. 2019. [Online]. Available: https://www.jetbrains.com/pycharm/

Jiayin Chen received the B.E. and M.S. degrees from the School of Electronics and Information Engineering, Harbin Institute of Technology, Harbin, China, in 2014 and 2016, respectively. She is currently pursuing the Ph.D. degree with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada. Her research interests are in the area of vehicular networks and machine learning, with current focus on intelligent transport system and big data.

Peng Yang (Member, IEEE) received the B.E. degree in communication engineering and the Ph.D. degree in information and communication engineering from the Huazhong University of Science and Technology (HUST), Wuhan, China, in 2013 and 2018, respectively. He was with the Department of Electrical and Computer Engineering, University of Waterloo, Canada, as a Visiting Ph.D. Student from September 2015 to September 2017, and a Postdoctoral Fellow from September 2018 to December 2019. Since January 2020, he has been a Faculty Member with the School of Electronic Information and Communications, HUST. His current research focuses on next-generation networking, mobile edge computing, video streaming, and analytics.

Qiang Ye (Member, IEEE) received the Ph.D. degree in electrical and computer engineering from the University of Waterloo, Waterloo, ON, Canada, in 2016. He was a Postdoctoral Fellow with the Department of Electrical and Computer Engineering, University of Waterloo from December 2016 to November 2018, where he was a Research Associate from December 2018 to September 2019. He has been an Assistant Professor with the Department of Electrical and Computer Engineering and Technology, Minnesota State University, Mankato, MN, USA, since September 2019. His current research interests include 5G networks, software-defined networking and network function virtualization, network slicing, artificial intelligence and machine learning for future networking, protocol design, and end-to-end performance analysis for the Internet of Things.

Weihua Zhuang (Fellow, IEEE) has been with the Department of Electrical and Computer Engineering, University of Waterloo, Canada, since 1993, where she is currently a Professor and a Tier I Canada Research Chair of wireless communication networks. She was a recipient of the 2017 Technical Recognition Award from the IEEE Communications Society Ad Hoc and Sensor Networks Technical Committee, and several best paper awards from IEEE conferences. She was the Technical Program Symposia Chair of the IEEE GLOBECOM 2011 and the Technical Program Chair/Co-Chair of the IEEE VTC in 2016 and 2017. She was an IEEE Communications Society Distinguished Lecturer from 2008 to 2011. She was the Editor-in-Chief of the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY from 2007 to 2013. She is a Fellow of the Royal Society of Canada, the Canadian Academy of Engineering, and the Engineering Institute of Canada. She is an Elected Member of the Board of Governors and VP Publications of the IEEE Vehicular Technology Society.

Xuemin (Sherman) Shen (Fellow, IEEE) received the Ph.D. degree in electrical engineering from Rutgers University, New Brunswick, NJ, USA, in 1990.

He is currently a University Professor with the Department of Electrical and Computer Engineering, University of Waterloo, Canada. His research focuses on network resource management, wireless network security, Internet of Things, 5G and beyond, and vehicular ad hoc and sensor networks. He received the R. A. Fessenden Award in 2019 from IEEE, Canada, the Award of Merit from the Federation of Chinese Canadian Professionals (Ontario) in 2019, the James Evans Avant Garde Award in 2018 from the IEEE Vehicular Technology Society, the Joseph LoCicero Award in 2015, and Education Award in 2017 from the IEEE Communications Society, and the Technical Recognition Award from Wireless Communications Technical Committee in 2019, and AHSN Technical Committee in 2013. He has also received the Excellent Graduate Supervision Award in 2006 from the University of Waterloo and the Premier's Research Excellence Award in 2003 from the Province of Ontario, Canada. He served as the Technical Program Committee Chair/Co-Chair for IEEE Globecom'16, IEEE Infocom'14, IEEE VTC'10 Fall, and IEEE Globecom'07, and the Chair for the IEEE Communications Society Technical Committee on Wireless Communications. He is the elected IEEE Communications Society Vice President for Technical & Educational Activities, the Vice President for Publications, a Member-at-Large on the Board of Governors, the Chair of the Distinguished Lecturer Selection Committee, a Member of IEEE ComSoc Fellow Selection Committee. He was/is the Editor-in-Chief of the IEEE INTERNET OF THINGS JOURNAL, IEEE NETWORK, *IET Communications*, and *Peer-to-Peer Networking and Applications*. He is a Registered Professional Engineer of Ontario, Canada, an Engineering Institute of Canada Fellow, a Canadian Academy of Engineering Fellow, a Royal Society of Canada Fellow, a Chinese Academy of Engineering Foreign Member, and a Distinguished Lecturer of the IEEE Vehicular Technology Society and Communications Society.

Xu Li received the B.Sc. degree in computer science from Jilin University, China, in 1998, the M.Sc. degree in computer science from the University of Ottawa in 2005, and the Ph.D. degree in computer science from Carleton University in 2008. He worked as a Research Scientist (with tenure) with Inria, France. He is a Senior Principal Researcher with Huawei Technologies Canada. He contributed extensively to the development of 3GPP 5G standards through over 90 standard proposals. He has published over 100 refereed scientific papers and is holding over 30 issued U.S. patents. His current research interests are focused in 5G. He is/was on the editorial boards of *IEEE Communications Magazine*, the IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, the *Wiley Transactions on Emerging Telecommunications Technologies*, and a number of other international archive journals.