On Achieving Fair and Throughput-Optimal Scheduling for TCP Flows in Wireless Networks

Yi Chen, *Student Member, IEEE*, Xuan Wang, and Lin Cai, *Senior Member, IEEE*

***Abstract*— Throughput-optimal scheduling has been heavily investigated given its ability to fully utilize network resources and maintain network stability. Most of the existing throughput-optimal algorithms, including the classic queue-length based MaxWeight algorithm and flow-delay-based MaxWeight algorithm, however, may bring a severe unfairness problem when scheduling transmission control protocol (TCP) controlled flows. As TCP is the dominant transport layer protocol in the Internet and it controls the majority of Internet traffic, we study how to design the scheduling algorithm that can ensure both throughput optimality and be compatible to TCP flows. In this paper, we analyze the reason behind the incompatibility between the existing scheduling algorithms and TCP, and then investigate the properties of the head-of-line access delay-based scheduling algorithm (HOLD) we proposed. We prove that the proposed HOLD can fairly schedule TCP flows in wireless networks with time-varying channel conditions and achieve throughput optimality with flow-level dynamics. Simulations using OMNeT++ 4 have been conducted to validate our analytical results, and compare the performance of different scheduling algorithms comprehensively.**

***Index Terms*— Communication system traffic control, cross layer design, TCPIP, wireless networks.**

I. Introduction

TRANSMISSION Control Protocol (TCP) is the dominant transport layer protocol in the Internet, and it has been extensively studied in the literature [1]. It performs decently even with the scale of the Internet growing by several orders of magnitude in the past three decades. The basic congestion control mechanism of TCP was designed to probe for the available bandwidth while maintaining certain level of fairness among co-existing flows. In practice, all TCP variants, both the widely used loss-based variants and the other delay-based ones, have their own clock timing, which relies on the end-to-end acknowledgement packets (ACKs). Based on the received ACKs, a TCP sender determines whether and how many packets should be injected into the network by updating the size of the congestion window (*cwnd*). This protocol was originally designed for wired networks. As an increasing number of wireless devices are involved in the Internet, it becomes increasingly important to investigate the compatibility between TCP and the lower layer wireless scheduling algorithms [2], [3]. The adjustment of *cwnd* was designed to achieve fairness among all TCP flows. However, the control mechanism of some existing throughput-optimal scheduling algorithms conflicts with TCP congestion control. How TCP can be compatible with throughput-optimal scheduling algorithms in wireless networks when multiple users share a radio link is the research interest of this paper.

In the link layer, scheduling algorithms in wireless networks have been extensively studied in the literature. For example, considering the differentiated services, a number of scheduling algorithms and MAC protocols have been designed according to the QoS requirements of various applications [4]. On the other hand, considering the wireless channel dynamics, opportunistic scheduling algorithms can exploit the multi-user diversity gain to improve the overall performance. Among them, in the past decade, the Queue-length based MaxWeight scheduling algorithm (QMW) has been thoroughly studied because of its desirable features of throughput-optimality and utility-optimal operation, with the assumptions that there are a fixed number of flows (users) in the system and that a certain type of flow control scheme is adopted [5]. Assuming that only one user can be scheduled in every time slot, the scheduling rule of QMW can be found in Algorithm 1, in which the scheduler tries to maximize the selected transmission rate, weighted by the queue length.

Algorithm 1: Let $Q_i(t)$ denote the i-th flow at time t, and the corresponding queue length is $|Q_i(t)|$. QMW seeks user i to transmit which satisfies the following condition at the beginning of time slot t:

$$i^*(|Q_i(t)|, r_i(t)) \in \underset{1 \leq i \leq N(t)}{\arg\max} |Q_i(t)| \cdot r_i(t), \tag{1}$$

with uniform tie-breaking if there are more than one users satisfying the condition.

In (1), $r_i(t)$ is the transmission rate of $Q_i(t)$ at time t, and $N(t)$ is the total number of users in the system at time t. The scheduling decision is made in every time slot independently. Unfortunately, some TCP flows will suffer from a severe unfairness or starvation problem if the QMW scheduling is adopted. In fact, not only QMW, but also most of the existing throughput-optimal scheduling algorithms can cause the unfairness problem when scheduling TCP flows [6], which gives the motivation of our study in this paper.

The main contributions of this paper are three-fold. First, we reveal why the existing throughput-optimal scheduling algorithms are not compatible with TCP flows. Second, we propose the throughput-optimal Head-Of-Line access Delay

Manuscript received November 26, 2015; revised April 12, 2016 and July 13, 2016; accepted September 1, 2016. Date of publication September 16, 2016; date of current version December 8, 2016. The associate editor coordinating the review of this paper and approving it for publication was M. Rossi.

Y. Chen and L. Cai are with the University of Victoria, Victoria, BC V8W2Y2, Canada (e-mail: chenyi@ece.uvic.ca; cai@ece.uvic.ca).

X. Wang was with the University of Victoria, Victoria, BC V8W2Y2, Canada (e-mail: xuan_wang@ece.uvic.ca).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TWC.2016.2610430

based scheduling algorithm (HOLD) and apply it to schedule the TCP flows in wireless networks. We prove that it can schedule TCP flows under homogeneous or heterogeneous channel conditions with certain level of fairness guarantee. Third, simulations using OMNeT++ 4 have been conducted to validate our theoretical findings, which show that the HOLD algorithm can outperform the other throughput-optimal scheduling algorithms in supporting TCP flows in wireless networks.

The rest of the paper is organized as follows. Sec. II explains the related concepts, including the system capacity region, throughput-optimality, and presents the insights of why the joint behaviours of TCP and the existing scheduling algorithms do not result in a desirable performance. Sec. III introduces the system model, including the channel and queueing models. In Sec. IV, the HOLD scheduling algorithm is introduced, its throughput-optimality is studied and fairness performance is analyzed. Performance evaluation is given in Sec. V, followed by the concluding remarks in Sec. VI.

II. Incompatibility Between TCP and Existing Scheduling Algorithms

In this section the existing queue-length based throughput-optimal scheduling algorithms are introduced. Since QMW is the origination of these algorithms, we use QMW as an example to investigate the unfairness problem with TCP flows. We also reveal the unfairness problem of the delay-based scheduling algorithm with TCP flows. With the observation of unfairness in the example, we further discuss the motivation and the approach to find the throughput-optimal scheduling algorithm for fair TCP flow scheduling.

A. System Capacity and Throughput-Optimal Scheduling Algorithms

The wireless network capacity region Λ is defined as the closure of all arrival rate vectors that can be stably transmitted in the network, considering all possible scheduling policies. An arrival rate vector can be stably transmitted when the queueing stability is assured. The queueing stability of a discrete time process $Q(t)$ is defined as that $Q(t)$ is strongly stable if it satisfies $\limsup_{t\to\infty}(1/t)\sum_{\tau=0}^{t-1}\mathbb{E}[|Q(\tau)|] < \infty$ [7]. Λ is fixed and only depends on the channel statistics of the system. A scheduling algorithm is throughput-optimal if it is able to ensure the queueing stability as long as the vector of average arrival rates is within the capacity region [8].

QMW is provable to be throughput-optimal with the condition that the number of users in the system does not change over time [5]. Due to its desirable throughput-optimality feature and low complexity to implement, its performance has been extensively studied [7]–[9]. Other queue-length based scheduling algorithms including the Exponential rule and Log rule were proposed in [10] and [11] to improve the delay performance. The applications of throughput-optimal scheduling algorithms can be found in [12]–[15].

In the networks with a dynamic number of flows over time referred as flow-level dynamics, QMW is no longer applicable due to the instability problem [16]. The capacity region for systems with flow-level dynamics is different from that without flow-level dynamic, which will be given in Sec. III. Several scheduling solutions were proposed for systems with flow-level dynamics. The Max-Rate scheduling algorithm (MR) was designed in [17], but the pure MR scheduling is an off-line algorithm, and requires the full knowledge of the channel distribution in advance, which is difficult and sometimes impossible to obtain in practical systems. A modified MR in the same paper uses the history information to learn the channel variance, but how to design the learning window is an open question. The Flow-Delay based MaxWeight (F-D-MW) scheduling algorithm was studied in [18] to stabilize the systems with flow-level dynamics. The proof shows that F-D-MW is throughput-optimal, but the drawbacks are the complexity of implementation and the undesirable delay performance.

The above scheduling algorithms mainly focus on how to achieve throughput-optimality, and have no special consideration of how to schedule TCP controlled flows. In the next subsection we will use an example to show the incompatibility between TCP and the queue-length based scheduling.

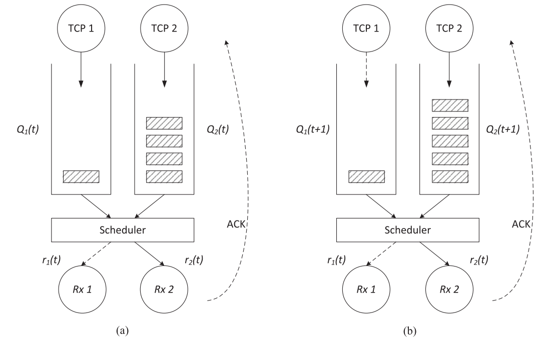


Fig. 1. Incompatibility between TCP and QMW scheduling.

B. An Example

Fig. 1 illustrates the interaction of QMW and TCP. We assume that the packet arrivals are regulated by a loss-based TCP congestion controller (TCP-Reno [19] or TCP-SACK [20]). For the simplicity of the explanation, we assume that only one packet will be transmitted when a flow is scheduled, and that all the packets have the same size of one maximum segment size (MSS). The queueing time and transmission delay in the wireless access links dominate the variation of the Round Trip Time (RTT).

Suppose that before time t, the second flow $Q_2(t)$ has been in the system for a while and its TCP congestion window size at time t has already been increased to be larger than one MSS, i.e., $cwnd_2(t) > 1$ MSS; while the first flow $Q_1(t)$ is a new one entering the system, and its TCP congestion window size is small, e.g., $cwnd_1(t) = 1$ MSS. Fig. 1(a) shows the example that the queue lengths of these two flows are $|Q_1(t)| = 1$ MSS and $|Q_2(t)| = 4$ MSS at time t. Assume $r_1(t) = r_2(t)$. Since $|Q_2(t)| > |Q_1(t)|$, according to the scheduling policy of QMW in Algorithm 1, a packet from $Q_2(t)$ is transmitted. Rx 2, the receiver of $Q_2(t)$, generates an ACK after receiving the packet, and sends the ACK to the TCP sender, i.e., TCP 2 in Fig. 1. After receiving the ACK, TCP 2 slides and increases the congestion window size at time slot $t+1$, i.e., $cwnd_2(t+1) > cwnd_2(t)$, and sends one or more packet(s) into $Q_2(t)$. On the other hand, since no packet is transmitted in flow 1 at time slot t, TCP 1 receives no ACK, and thus its congestion window size remains one. As a result, no new packet is added to $Q_1(t)$ and we still have $|Q_2(t+1)| > |Q_1(t+1)|$ in time slot $t+1$ as shown in Fig. 1(b). Eventually, $|Q_1(t)|$ will hardly increase and the first flow suffers from the starvation, while the second flow dominates the usage of the resources.

QMW also makes the old flows suffer from a long delay before their last few packets are transmitted. This problem is considered as the last-packet problem of QMW, which is also the reason why QMW is not applicable with flow-level dynamics [21]. Consider that flow one has a finite amount of data to transmit. Before finishing the whole transmission, it is possible that only one or a couple of packets are left in its queue. If another flow has many packets waiting in the queue at this time, the last few packets in flow one have to wait without being scheduled until the number of packets in the other flow's queue decreases to a sufficiently small value.

Simulation results in Sec. V will show the severe unfairness problem of the joint behaviour of TCP and QMW both at the beginning and the end of each flow's transmission. Other variants of QMW, such as the Exponential rule and the Log rule [10], [11], all directly or indirectly use the queue length as the weight for the scheduling decision, and thus they can be categorized as queue-length based scheduling. They encounter the same unfairness and starvation problem when working with TCP flows. In the rest of the paper, we only take QMW as the representative one in this category.

C. F-D-MW

The scheduling rule of F-D-MW can be found in Algorithm 2.

Algorithm 2: Let $Q_i(t)$ denote the i-th flow at time t. $D_i(t)$ is the sojourn time of $Q_i(t)$ which is measured from the time instant when $Q_i(t)$ arrives in the network waiting for being scheduled. F-D-MW seeks user i to transmit which satisfies the following condition at the beginning of time slot t:

$$i^*(D_i(t), r_i(t)) \in \underset{1 \le i \le N(t)}{\arg\max}\, D_i(t) \cdot r_i(t),$$

with uniform tie-breaking if there are more than one users satisfying the condition.

F-D-MW is throughput-optimal with a dynamic number of users in the system [8], [22]. However, F-D-MW is not compatible with TCP flows in wireless networks either. Because an F-D-MW scheduler always assigns a higher weight to the existing TCP flows in the network, the new flows have a much lower instantaneous throughput when they enter the system, and thus they suffer the long start-up latency and may even be starved at the beginning. This may not be desirable for the applications with stringent delay requirements. In most operating systems (Windows, MacOS, etc.), when a node begins to establish a TCP connection, it sends out the TCP SYN control packet and will wait up to 75 seconds (20 seconds in Unix) for the SYN-ACK packet from the destination node. When the timer expires, the attempt to establish the TCP connection will be abandoned. With F-D-MW (and QMW), it is possible that the connection may be abandoned due to the long start-up latency.

D. Further Discussion

The objective of an optimal scheduler is to allocate resources to stabilize the system whenever possible. Typically, a resource allocation problem can be modelled as a utility maximization problem, and solved by the approaches as those in [23]–[25]. By using dual-decomposition, the problem can be decomposed into a rate control problem and a Maximum Weighted Matching problem (scheduling problem). In such an approach, the rate control is explicitly performed for the scheduling algorithm, and the congestion signal of the rate control is the queue length, which is a required feedback information to the sender. Since the throughput-optimal scheduling (including QMW) can be explained as a generalization of the scheduling algorithm developed by this approach, a rate control may be needed to cooperate in real operation to prevent undesirable performance degradation.

In the Internet, however, the queue-length based rate control is not likely to be widely used, so long as TCP is the dominant transport layer protocol [6]. The popular TCP variants, such as TCP Reno, TCP New Reno [26] and TCP SACK, are all window-based congestion control using the packet loss as the congestion signal. Since it is not likely to drastically modify TCP to be compatible with the scheduling algorithms due to the backward compatibility concern, how to design an efficient scheduling algorithm in the link layer to be compatible with the existing TCP protocol is a critical issue.

We have two types of methods to design scheduling algorithms to be compatible with the current window-based TCP. The first is to use a utility based non-throughput-optimal scheduling algorithm in the MAC layer, whose stability region is less than the system capacity region, which results in less efficient channel utilization. A typical example of such algorithms is the Proportional Fairness scheduling algorithm (PF), which has been widely adopted in the cellular systems such as the LTE networks. PF is able to fairly allocate the channel resources for all the users in the system according to their previous resource allocation while considering the multi-user diversity gain. But it has been shown that PF is not throughput-optimal [27]. Reference [28] has proven that utility based scheduling, including the PF scheduling, is not throughput-optimal, and thus in general the stability region is less than the capacity region. The second is to develop new throughput-optimal scheduling algorithms compatible with TCP. There are two main approaches to design the throughput-optimal scheduling algorithms, i.e., the queue-length-based and the delay-based approaches. As QMW is not desirable for supporting TCP flows, a newly designed queue-length based scheduling was proposed in [6], which uses network coding and the computation of a threshold when deciding the weight of each user. The algorithm shows throughput-optimality and fairness with a fixed number of long-lived TCP flows, but the design

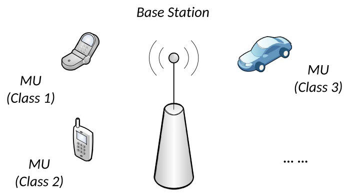


Fig. 2. The downlink of a wireless network with different classes of flows.

of the threshold and network coding brings the complexity in implementation. Furthermore, this algorithm is not designed for the networks with flow-level dynamics. Therefore we focus on the second type, the delay-based approach. The existing delay-based solution, F-D-MW, however, is not compatible with TCP either as explained earlier. This motivates us to study our own delay-based scheduling algorithm.

III. System Model

In this paper, we consider the downlink of a single-hop centralized wireless access network as illustrated in Fig. 2, which works in slotted time. Our work can also be applied to uplink scheduling if a centralized scheduler exists, which is omitted due to space limit. The network consists of one central controller, such as the base station (BS), and $N(t)$ mobile users (MU) at time t. Each MU is associated with a distinct TCP flow. As each user is associated with one flow, we do not distinguish the concept of "user" and "flow" thereafter. Flows are categorized into different classes according to their channel profiles, and the central controller in the network selects one flow to transmit for each time slot.

A. Networks With Flow-Level Dynamics

In the network with flow-level dynamics [16], we have the arrival of new flows and the departure of old ones. We consider that there are K classes of flows in the system, and each class of flows is defined according to the channel profiles. At time slot t, the i-th flow of class k is denoted by $Q_{ki}(t)$, the number of flows of class-k is $N_k(t)$, and the total number of flows in the systme is $N(t) = \sum_{k=1}^{K} N_k(t)$. For every $Q_{ki}(t)$, there is a finite amount of data to transmit. After all the data are delivered through the radio link, the corresponding flow will leave the system. As a result, the number of flows in the current time slot may not be the same as that in the next time slot. Thus the number of flows in the system is time-varying.

B. Channel Model

Let $r_{ki}(t)$ denote the transmission rate of the wireless channel at time t for $Q_{ki}(t)$. The unit of the channel rate is $bit/slot$. The BS can transmit at most $r_{ki}(t)$ bits at time slot t for $Q_{ki}(t)$. $r_{ki}(t)$ may vary over time as a result of channel fading. For each class k, we assume that $r_{ki}(\cdot)$ are i.i.d. copies of positive random variable R_k with finite first and second order moments, and $r_{ki}(t) \in \{R_{k1}, R_{k2}, \cdots, R_{km_k}\}$. Different classes have different channel profiles, which give the channel rate distributions, i.e., what is the probability that the channel rate is equal to a centain value. The maximum possible transmission rate of the class-k flows is defined as $R_k^{\max} := \sup\{r : \mathbb{P}\{R_k = r\} > 0\}$, and the maximum possible transmission rate of the system is defined as $R^{\max} := \max_{1 \leqslant k \leqslant K} \{R_k^{\max}\}$. The flows in the same class have the same channel rate distribution, and thus they have the same upper bound, lower bound, the average rate and the channel rate variance. It is possible that the flows in the same class have different instantaneous channel rates as the channel rate is a random variable.

C. Queueing Model

We assume that new flows can arrive at the scheduler at any time in a time slot. The number of new class-k flows arriving during time slot t is $A_k(t)$, which is the i.i.d. copy of a random variable A_k with a finite mean $\lambda_k = \mathbb{E}[A_k(\cdot)]$, where $\mathbb{E}[\cdot]$ denotes expectation. The packets of the i-th flow in class k are stored in a dedicated buffer. We consider that the amount of data stored at the sender side is $B_{ki}(t)$, and the buffer is large enough to avoid buffer overflow. $B_{ki}(t)$ is the i.i.d. copy of an integer random variable B_k and has a finite mean $\beta_k = \mathbb{E}[B_{ki}(\cdot)]$. We assume that the second moments of A_k and B_k are both finite. TCP is used as the end-to-end transport protocol. For each flow, TCP determines the amount of data delivered from $B_{ki}(t)$ to the transmission queue $Q_{ki}(t)$ of the scheduler. The amount of data delivered by TCP from $B_{ki}(t)$ to $Q_{ki}(t)$ is denoted by $s_{ki}(t)$. We suppose that the scheduling decision is made at the beginning of every time slot, so that any of the data packets that arrives after the beginning of slot t, i.e., any $B_{ki}(t)$ of $\forall k = \{1, 2, \cdots, K\}$ and $\forall i = \{1, 2, \cdots, N_k(t)\}$ can only be transmitted in the following slots. We define $|Q_k(t)| := \sum_{i=1}^{N_k(t)} |Q_{ki}(t)|$ as the class-k backlog and $|Q(t)| := \sum_{k=1}^{K} |Q_k(t)|$ as the system backlog. The queue dynamic is given by

$$|Q_{ki}(t+1)| = \max[|Q_{ki}(t)| - r_{ki}(t) + s_{ki}(t), 0]. \qquad (2)$$

If there is no more traffic arrival for $Q_{ki}(t)$ from the current TCP session, $Q_{ki}(t)$ will leave the system. With the above model, we can define the capacity region of a flow-level dynamic network. Let γ_k represent the expected number of time slots required for the service of a class-k flow if served with $R_k^{\max}$, and then we have $\gamma_k = \mathbb{E}\left\lceil \frac{B_k}{R_k^{\max}} \right\rceil$. Let $\rho_k = \lambda_k \gamma_k$ denote the traffic intensity of class-k flows, and $\rho = \sum_{k=1}^{K} \rho_k$ denote the system traffic intensity. The system capacity region is defined as $\mathcal{S} = \{(\lambda_1, \lambda_2, \ldots, \lambda_K), (\gamma_1, \gamma_2, \ldots, \gamma_K) : \rho < 1\}$. For any arrival process that lies in the capacity region, if the system is strongly stable, i.e., $\limsup_{T \to \infty} \frac{1}{T} \sum_{t=0}^{T-1} \mathbb{E}[|Q(t)|] < \infty$, then the correspondingly adopted scheduling algorithm is throughput-optimal.

With the models above, intuitively, if the system is stable, the total amount of data in the system should be finite. If the system is unstable, the total amount of data will grow into infinity when $t \to \infty$ considering infinite buffer size. Note that the traffic intensity represents on average how many slots are required to transmit the arrived data in one time slot if the maximum transmission rate is adopted. If the traffic intensity is smaller than 1, it means that the average amount of the data arrived in one time slot can be transmitted in less than one time slot by the maximum transmission rate, and thus there exists at least one scheduling algorithm to obtain the system stability. When the traffic intensity is larger than 1, it means that on average more than one time slot is required to transmit the amount of arrival data in one time slot, and thus the packets that cannot be transmitted in time will accumulate, which leads to system instability. From this perspective, the traffic intensity $\rho < 1$ is defined as the system capacity region. Any arrival rate in the capacity region can be stably transmitted by the throughput-optimal scheduling algorithms without admission control [16].

IV. HOL Access Delay Based Scheduling

We first give the definition of the Head-Of-Line (HOL) access delay which we will use in our scheduling.

Definition 1 (The HOL Access Delay $H_{ki}(t)$): Let $I_{ki}^{H}(t)$ denote the head-of-line bit in $Q_{ki}(t)$ which will be the first bit to be transmitted once $Q_{ki}(t)$ is scheduled. The HOL access delay of $Q_{ki}(t)$ is defined as $H_{ki}(t) = t - t_0$, where t is the current time, and t_0 is the time at which $I_{ki}^{H}(t)$ becomes the first bit in $Q_{ki}(t)$.

HOL access delay can be viewed as the waiting time of $Q_{ki}(t)$ being served from $Q_{ki}(t)$'s previous transmission, and is calculated according to the following equation:

$$H_{ki}(t+1) = (H_{ki}(t) + 1)(1 - \mathbf{1}_{ki}(t)), \tag{3}$$

where $\mathbf{1}_{ki}(t)$ is the indicator function such that $\mathbf{1}_{ki}(t) = 1$ only when $Q_{ki}(t)$ is scheduled at time slot t, and $\mathbf{1}_{ki}(t) = 0$ otherwise. With the system model and the definition of HOL access delay, we propose the following HOL access delay based scheduling algorithm.

Algorithm 3: HOL access Delay based MaxWeight scheduling algorithm (HOLD) seeks the flow $\{k, i\}$ for transmission that satisfies the following condition at the beginning of time slot t:

$$\{k, i\}^{*}(H_{ki}(t), r_{ki}(t)) \in \underset{1 \le k \le K, 1 \le i \le N_k(t)}{\arg\max} H_{ki}(t) \cdot r_{ki}(t), \tag{4}$$

with uniform tie-breaking if there are more than one flow satisfying the condition. The scheduling decision is made in every time slot independently.

HOLD is different from F-D-MW. For F-D-MW, the flow delay increases along the time until the flow leaves the system. While for HOLD, HOL access delay of one flow returns back to zero once it is scheduled to transmit.

Remarks: Similar to QMW and F-D-MW, we also assume that we schedule one flow in each time slot in HOLD. In the current LTE systems, resource blocks can be assigned to different flows in the same time slot, and this does not conflict with the main results of our work. Taking OFDMA as an example, although multiple flows can be scheduled in the same time slot, in any sub-channel, we can only schedule one flow at a time. According to the structure of the latest 3GPP framework for LET system (shown in [29, Fig. 6.4-1]), each UE has a dedicated buffer for data storage, and multiple queues exist in this data storage for different types of flows. All the flows are connected to the scheduler which selects the most desirable flow. After each scheduling decision, the queues are updated and ready for the next round of scheduling. Different flows can be categorized into different classes, and thus it is possible to implement the proposed HOLD scheduling algorithm considering the system model in our work.

In the following we first investigate the throughput-optimality of HOLD, and then study the fairness issue using HOLD to schedule TCP flows. The fairness is closely related to HOL access delay. It can be known from Definition 1 that the HOL access delay is actually the access waiting time, which indicates how long one flow has to wait between two consecutive transmissions. If two flows in the network need to equally share the channel time, their average access waiting time should be the same. Longer HOL access delay means that less channel time is allocated to the corresponding flow. Thus, we will study the fairness performance of HOLD by investigating the HOL access delay. To measure the fairness of HOLD, we use the fairness defined for weighted fair queueing in terms of allocated channel time, in which each flow can be allocated a share of channel time proportional to its weight. In HOLD, the weight of the allocated channel time of each flow is its maximum channel rate, i.e., for $Q_{ki}(t)$, the weight w_{ki} is $R_k^{\max}$. With this definition, and considering the variance of channel condition and the uniform tie-breaking rule, all the flows should share the channel time with the average access waiting time correlated by w_{ki} in order to achieve a fair scheduling. According to [30], if

$$H_{ki}(t)/H_{mj}(t) = w_{mj}/w_{ki}, \tag{5}$$

the weighted average fairness of HOLD in terms of the allocated channel time can achieve the maximum, considering that the allocated channel time can be viewed as the reciprocal of HOL access delay.

A. Throughput-Optimality

We first focus on the throughput-optimality of HOLD in flow-level dynamic systems. We only present the most important steps, and the details and simulation results can be found in [31].

The proof of the throughput-optimality of HOLD involves three steps. Let $r(t)$ denote the real transmission rate of the network at time t. First, if a class-k flow $Q_{ki}(t)$ is scheduled, i.e., $r(t) = r_{ki}(t)$, the sufficient condition for the network with flow-level dynamics to be stable for any arrival rate that lies in the capacity region is

$$\lim_{t \to \infty} \mathbb{P}\{r(t) < R_k^{\max} | \text{a class-}k \text{ flow is scheduled}\} = 0. \tag{6}$$

Second, we draw the conclusion that, for a single-class ($K = 1$) flow-level dynamic multi-user wireless system with HOLD, we can obtain $\mathbb{P}\{r(t) < R^{\max}\} = p^{\tilde{N}(t)}$, and thus (6) is true if the system is unstable. Here $\tilde{N}(t)$ is an increasing function of $N(t)$, and $p = \mathbb{P}\{r_i(t) \neq R_i^{\max}\}$. Third, we obtained the result that HOLD is throughput-optimal for a single-class ($K = 1$) flow-level dynamic system by following the result in step two, and then we extended the throughput-optimality to a heterogeneous system with $K > 1$ by explaining that if $\exists k$ such that $N_k(t) \to \infty$, we have $\forall i \in \{1, 2, \cdots, K\}$,

$N_i(t) \to \infty$, i.e., if one class is unstable, then all the classes in the network are unstable.

Remarks: The proof of (6) involves the definition of a Lyapunove function regarding the workload of the system at time t. The intuitive explanation of the above theorem is as follows. If the scheduling algorithm always tries to schedule a flow when it has its possible maximum transmission rate, the system is stable thanks to the maximum utilization of resources. From the definition of the capacity region, we can tell that if a flow is scheduled when it is not in its maximum transmission channel rate, it probably needs more time slots for transmission and hence leads to a waste of resources. However, the above is not a necessary condition for a system to be stable. For example, if there is a large distance between the arrival rate vector and the capacity region boundary, i.e., the traffic intensity of the system is quite low, it is possible that the system is able to deliver all the arrival bits though some transmissions associated with a low transmission rate. But for a network with a very high traffic intensity, i.e., there is a very small gap between the arrival rate vector and the system capacity, the condition in (6) becomes necessary.

B. Fairness Analysis

Proposition 1: Given i.i.d. channel transmission rate distribution for all the system flows, HOLD can achieve fair resource sharing, so that all flows obtain an equal share of the channel time.

Proof: In this proof, we only have one class of flows in the system, and thus the class index k in the subscript is omitted for simplicity. We first investigate the simplified scenario in which there are only 2 flows. The proof can be extended to the N-flow cases. Since we have the assumption that all the system flows have i.i.d. channel transmission rate distribution, we can tell $\mathbb{E}[r_1(t)] = \mathbb{E}[r_2(t)]$. Next we will show that $\mathbb{E}[H_1^{sch}(t)] = \mathbb{E}[H_2^{sch}(t)]$, where $H_i^{sch}(t)$ is the HOL access delay of flow i when it is scheduled at time t. To prove this, without loss of generality, we just need to show that $\mathbb{E}[H_1^{sch}(t)] > \mathbb{E}[H_2^{sch}(t)]$ is impossible.

We assume that $\mathbb{E}[H_1^{sch}(t)] > \mathbb{E}[H_2^{sch}(t)]$ is true, i.e., flow 1 has an on average larger head-of-line access delay than that of flow 2. The channel rate of $Q_i(t)$ at time slot t, denoted by $r_i(t)$, only depends on the SINR of the wireless channel at t and the corresponding modulation and coding scheme. As a result, we have $\Pr\{r_i(t) = x | H_i(t) = y\} = \Pr\{r_i(t) = x\}$, which indicates that $H_i(t)$ and $r_i(t)$ are independent. Given that $\mathbb{E}[r_1(t)] = \mathbb{E}[r_2(t)]$ and $r_i(t)$ is independent of $H_i(t)$, according to HOLD which seeks the maximum product of $r_i(t) \cdot H_i(t)$ in every time slot, flow 1 has more chance to transmit than flow 2. Consequently, the average number of time slots that flow 1 has to wait between two of its transmissions is less than that of flow 2. This implies $\mathbb{E}[H_1^{sch}(t)] < \mathbb{E}[H_2^{sch}(t)]$, which contradicts to our assumption here. Thus the assumption that $\mathbb{E}[H_1^{sch}(t]) > \mathbb{E}[H_2^{sch}(t)]$ cannot hold. Similarly we can prove that $\mathbb{E}[H_1^{sch}(t)] < \mathbb{E}[H_2^{sch}(t)]$ is also not possible. Thus we have $\mathbb{E}[H_1^{sch}(t)] = \mathbb{E}[H_2^{sch}(t)]$. This result can be extended to $\mathbb{E}[H_i^{sch}(t)] = \mathbb{E}[H_j^{sch}(t)]$ if we have more than 2 flows in the system. Since the proof is similar to the 2-flow case, the details are omitted. ■

Next we consider the fairness performance of HOLD in heterogeneous networks. For a single flow $Q_{ki}(t)$, it is possible that H_{ki}^{sch} varies from time to time, even with the deterministic channel profile. Considering the variance of HOL access delay, we define $\bar{H}_{ki}^{sch}$ as the average value of $H_{ki}^{sch}(t)$ over time. To measure the fairness of HOLD, we define $\eta_{k,l}^H$ as the ratio of the average HOL access delay of class-k and class-l flows, i.e., $\eta_{k,l}^H = \bar{H}_{ki}^{sch}/\bar{H}_{lj}^{sch}$. We assume that the choice of i in class k and j in class l does not affect the value of $\eta_{k,l}^H$. Similarly, we define $\eta_{k,l}^R$ as the ratio of the channel rates of class k and class l, so as to describe the relationship between the HOL access delay and channel rate with heterogeneous and deterministic channel rate profile.

Proposition 2: Given non-identical (heterogeneous) constant channel rates for the flows, when the number of flows in the system is sufficiently large, HOLD can achieve fair channel time allocation among flows proportionally to their channel rates.

Proof: We assume that every flow in the network has a non-empty queue. For simplicity, we consider that all the flows can be categorized into 2 classes, and $\bar{N}_1$ and $\bar{N}_2$ are the average number of flows in class 1 and class 2, respectively. The deterministic channel rates of class-1 and class-2 flows are R_{11} and R_{21}, respectively. We assume that the channel rate of class-1 flows is smaller than that of class-2 flows, i.e., $R_{11} \leqslant R_{21}$. Note that $\eta_{2,1}^R = R_{21}/R_{11}$ and $\eta_{1,2}^H = \bar{H}_{1i}^{sch}/\bar{H}_{2j}^{sch}$. Because the channel rates of class 1 and class 2 are constant values, we have $H_{ki}^{sch}(t) = \max_{1 \leq i \leq N_k(t)} H_{ki}(t)$. If a flow $Q_{1i}(t)$ from class 1 is scheduled in time slot t, the earliest time for a flow $Q_{2j}(t)$ to be scheduled is time slot $t+1$, thus we have the following relationship:

$$\bar{H}_{1i}^{sch} \cdot R_{11} \geqslant (\bar{H}_{2j}^{sch} - 1) \cdot R_{21}. \tag{7}$$

We further clarify that $\bar{H}_{1i}^{sch}$ is expected to be $\bar{H}_{1i}^{sch} = \bar{N}_1 + \eta_{12}^H \bar{N}_2$, and the explanation is as follows. $\bar{H}_{1i}^{sch}$ is the average time that $Q_{1i}(t)$ waits between the previous and next transmissions. During this time period, each of the other class-1 flows is expected to have one transmission considering that their HOL access delays are larger than $Q_{1i}(t)$ in the first time slot after $Q_{1i}(t)$'s transmission, and thus $\bar{H}_{1i}^{sch} \geqslant \bar{N}_1$. Meanwhile, since we assume $R_{11} \leqslant R_{21}$, during the time period of $\bar{H}_{1i}^{sch}$, each of the class-2 flows can be scheduled once or more. As $\bar{H}_{1i}^{sch}$ and $\bar{H}_{2j}^{sch}$ are the average access waiting times for class-1 and class-2 flows to be scheduled, respectively, $\bar{H}_{1i}^{sch}/\bar{H}_{2j}^{sch}$ represents on average how many times that a class-2 flow can be scheduled during $\bar{H}_{1i}^{sch}$, and thus $(\bar{H}_{1i}^{sch}/\bar{H}_{2j}^{sch}) \cdot \bar{N}_2(t)$ means on average how many times that class-2 flows can be scheduled during $\bar{H}_{1i}^{sch}$. By calculating on average how many times all the other flows can transmit between $Q_{1i}(t)$'s previous transmission and next transmission, i.e., during the period of $\bar{H}_{1i}^{sch}$, we have $\bar{H}_{1i}^{sch} = \bar{N}_1 + (\bar{H}_{1i}^{sch}/\bar{H}_{2j}^{sch})\bar{N}_2$. With (7), we can further have:

$$\frac{\bar{N}_1 + \eta_{1,2}^H \cdot \bar{N}_2}{\dfrac{\bar{N}_1 + \eta_{1,2}^H \cdot \bar{N}_2}{\eta_{1,2}^H} - 1} \geqslant \eta_{2,1}^R.$$

The solution of $\eta^{H}_{1,2}(t)$ can be found by solving the following inequality:

$$\eta^{H}_{1,2} \geqslant \frac{\eta^{R}_{2,1}(\bar{N}_2 - 1) - \bar{N}_1}{2\bar{N}_2} + \frac{\sqrt{(\bar{N}_1 - \eta^{R}_{2,1}(\bar{N}_2 - 1))^2 + 4\eta^{R}_{2,1}\bar{N}_1\bar{N}_2}}{2\bar{N}_2}. \tag{8}$$

When $\bar{N}_2$ is large enough ($\bar{N}_2 \gg 1$), the right-hand-side of (8) converges to

$$\frac{\eta^{R}_{2,1}\bar{N}_2 - \bar{N}_1 + \sqrt{(\bar{N}_1 - \eta^{R}_{2,1}\bar{N}_2)^2 + 4\eta^{R}_{2,1}\bar{N}_1\bar{N}_2}}{2\bar{N}_2} = \eta^{R}_{2,1}.$$

Similarly, if the flow $Q_{2j}(t)$ from class 2 is scheduled, we have

$$(\bar{H}^{sch}_{1i} - 1) \cdot R_{11} \leqslant \bar{H}^{sch}_{2j} \cdot R_{21},$$

which indicates:

$$\frac{\bar{N}_1 + \eta^{H}_{1,2}\bar{N}_2 - 1}{\frac{\bar{N}_1 + \eta^{H}_{1,2}\bar{N}_2}{\eta^{H}_{1,2}}} \leqslant \eta^{R}_{2,1}.$$

By solving this inequality, we have

$$\eta^{H}_{1,2} \leqslant \frac{\bar{N}_2\eta^{R}_{2,1} - \bar{N}_1}{2(\bar{N}_2 - 1)} + \frac{\sqrt{(\bar{N}_2\eta^{R}_{2,1} - \bar{N}_1)^2 + 4\eta^{R}_{2,1}\bar{N}_1(\bar{N}_2 - 1)}}{2(\bar{N}_2 - 1)}. \tag{9}$$

When $\bar{N}_2$ is sufficiently large, (9) converges to $\eta^{H}_{1,2} \leqslant \eta^{R}_{2,1}$. Hence we come to the conclusion that $\eta^{H}_{1,2} = \eta^{R}_{2,1}$ by combining the results above. ■

Proposition 3: Given independent and non-identical (heterogeneous) channel rate distributions for the flows, when the number of flows is sufficiently large, HOLD can achieve fair resource sharing among flows proportional to their maximum channel rates.

Proof: We still consider a 2-class system, where the channel rate of flow $Q_{ki}(t)$ from class k in time slot t is denoted by $r_{ki}(t)$. We use $r^{sch}_{ki}(t^{(k)})$ to denote the channel rate when $Q_{ki}(t^{(k)})$ is actually scheduled in time slot $t^{(k)}$, in which we specifically use $H^{sch}_{ki}(t^{(k)})$ to denote the HOL access delay of $Q_{ki}(t^{(k)})$. Thus we have $H^{sch}_{1i}(t^{(1)})r^{sch}_{1i}(t^{(1)}) \geqslant \max\{H_{2j}(t^{(1)})r_{2j}(t^{(1)})\}$ when $Q_{1i}(t^{(1)})$ is scheduled.

In [31], it has been proved that when the number of flows in the system is sufficiently large, $\mathbb{P}\{r^{sch}_{ki}(t) = R^{\max}_{k}\} = 1$, which means that the scheduler is able to fully utilize the multi-user diversity gain to improve the throughput performance, and hence we have $H^{sch}_{1i}(t^{(1)})R^{\max}_{1} \geqslant \max\{H_{2j}(t^{(1)})r_{2j}(t^{(1)})\}$. Because the earliest following time for a flow of class 2 to be scheduled is $t^{(1)} + 1$, we have $H^{sch}_{1i}(t^{(1)})R^{\max}_{1} \geqslant (H^{sch}_{2j}(t^{(2)}) - 1)R^{\max}_{2}$.

By taking the time average over the above inequality, we have

$$\lim_{T\to\infty}\frac{1}{T}\sum_{t=0}^{T} H^{sch}_{1i}(t)R^{\max}_{1} \geqslant \lim_{T\to\infty}\frac{1}{T}\sum_{t=0}^{T}(H^{sch}_{2j}(t) - 1)R^{\max}_{2}. \tag{10}$$

Because the number of flows in the system is sufficiently large, and thus $H^{sch}_{2j}(t^{(2)}) \gg 1$. From (10) we have $\mathbb{E}[H_{1i}(t)^{sch} \cdot R^{\max}_{1}] \geqslant \mathbb{E}[H_{2j}(t)^{sch} \cdot R^{\max}_{2}]$. Similarly, we can obtain $\mathbb{E}[H^{sch}_{2j}(t) \cdot R^{\max}_{2}] \geqslant \mathbb{E}[H^{sch}_{1i}(t) \cdot R^{\max}_{1}]$. This indicates that when $t \to \infty$, we have

$$\frac{\mathbb{E}[H^{sch}_{1i}(t)]}{\mathbb{E}[H^{sch}_{2j}(t)]} = \frac{R^{\max}_{2}}{R^{\max}_{1}}. \tag{11}$$

■

Remarks: Since the arrival rate of networks with flow-level dynamics refers to the number of new flows generated in one time slot, the arrival rate influences the total number of flows in the network. Given a wireless network, the larger the arrival rate is, the more flows we have in the network. The weighted fairness between flows remains the same no matter whether the arrival rates for HOLD are small or large. The throughput ratio of two flows will change only when their channel profiles change. Considering that the arrival rate does not affect the fairness and the increase of HOL access delay of each individual flow in one time slot so long as each queue is non-empty, in summary, HOLD is able to achieve fair scheduling among flows and thus can be adopted with TCP control schemes given various channel conditions.

One thing that we need to emphasize is that a scheduling algorithm with good fairness performance does not always necessarily mean the same HOL access delay for every flow if the flows belong to different classes. In practical networks, the resource allocation may be related to how much a customer pays for the service, and thus the scheduling algorithm should also accordingly assign the channel resources. To provide this type of differentiated services, we can simply assign a weight to each class, and use the multiplication of the weight and $R^{\max}_{k}$ of each class to ensure the portion of channel time allocated to this class.

On the other hand, with TCP flows, if a throughput-optimal scheduling algorithm suffers the unfairness problem, such as QMW, the desirable throughput-optimality may no longer hold. This is because the TCP controlled flow tends to avoid quick queue length increase, so with QMW scheduling, some flows may not have a sufficiently large size of the queue length to be scheduled. The HOLD scheduling is compatible with TCP as it aims to allocate channel time to flows less dependent on its packet arrival process.

C. Throughput Analysis

With the analysis of the HOL access delay, we can further analyze the throughput relationship between different flows, which follows a η^2-rule as explained in the analysis below. The throughput analysis will be based on the HOL access delay analysis above, and thus we also follow the three cases discussed above.

Case 1: Given i.i.d. channel transmission rate distribution for all the system flows, we have $\mathbb{E}[W_i(t)]/\mathbb{E}[W_j(t)] = 1$, where $W_i(t)$ denotes the throughput of flow $Q_i(t)$ in time slot t.

This conclusion can be drawn from the result that $\mathbb{E}[H_i^{sch}] = \mathbb{E}[H_j^{sch}]$. Since the probability that flow $Q_i(t)$ is scheduled, denoted by p_i, can be calculated as $p_i = 1/\mathbb{E}[H_i^{sch}]$, we know that $p_i = p_j$. Because the flows have i.i.d. channel distribution, considering that $\mathbb{E}[W_i(t)] = p_i \cdot \mathbb{E}[R_i]$, we can come to the conclusion that $\mathbb{E}[W_i(t)]/\mathbb{E}[W_j(t)] = 1$.

Case 2: Consider a 2-class network. Given non-identical (heterogeneous) constant channel rates for the flows, when the number of flows in the system is sufficiently large, we have $\mathbb{E}[W_{1i}(t)]/\mathbb{E}[W_{2j}(t)] = 1/(\eta_{2,1}^R)^2$, where $\eta_{2,1}^R = R_{21}/R_{11}$.

This conclusion can be drawn from the result in Proposition 2 that $\eta_{1,2}^H = \eta_{2,1}^R$. Given this relationship, we have $p_{1i}/p_{2j} = \mathbb{E}[H_{2j}^{sch}]/\mathbb{E}[H_{1i}^{sch}] = 1/\eta_{2,1}^R$, and thus the ratio of throughput $\mathbb{E}[W_{1i}(t)]/\mathbb{E}[W_{2j}(t)] = (R_1/R_2) \cdot (p_{1i}/p_{2j}) = 1/(\eta_{2,1}^R)^2$.

Case 3: Consider a 2-class network. Given independent and non-identical (heterogeneous) channel rate distributions for the flows, when the number of flows in the system is sufficiently large, we have $\mathbb{E}[W_{1i}(t)]/\mathbb{E}[W_{2j}(t)] = 1/(\tilde{\eta}_{2,1}^R)^2$, where $\tilde{\eta}_{2,1}^R = R_2^{\max}/R_1^{\max}$.

This conclusion can be drawn from the result of (11) in the proof of Proposition 3. Given this relationship, we have $p_{1i}/p_{2j} = \mathbb{E}[H_{2j}^{sch}]/\mathbb{E}[H_{1i}^{sch}] = 1/\tilde{\eta}_{2,1}^R$, and the ratio of throughput $\mathbb{E}[W_{1i}(t)]/\mathbb{E}[W_{2j}(t)] = (p_{1i}/p_{2j}) \cdot (R_{1i}^{\max}/R_{2j}^{\max}) = 1/(\tilde{\eta}_{2,1}^R)^2$.

V. Performance Evaluation

To investigate the performance of the HOLD scheduling algorithm for TCP flows, we conducted simulations with OMNeT++ 4.4.1. We compared the performance of HOLD with the MR, QMW, F-D-MW, and PF scheduling algorithms.

A. Network Setting

We consider centralized wireless networks such as the cellular networks in our simulation. The network topology in our simulation is shown in Fig. 3. In this network, a server is connected to the base station (BS) through a router. The BS can exchange messages with a number of wireless devices through the shared wireless channel. In the simulation, each client tries to establish a TCP connection with the server at a certain time, and then sends requests to the server. If the TCP connection is established successfully, the server will send the requested data back to the clients. For each TCP connection, the number of requests per TCP session follows exponential distribution with the mean value of 10 (requests); the request length follows truncated normal distribution with mean value of 20B; the reply length follows exponential distribution with mean value of 100MB; the re-connection interval is 10 seconds.

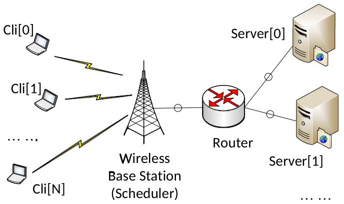


Fig. 3. Network topology.

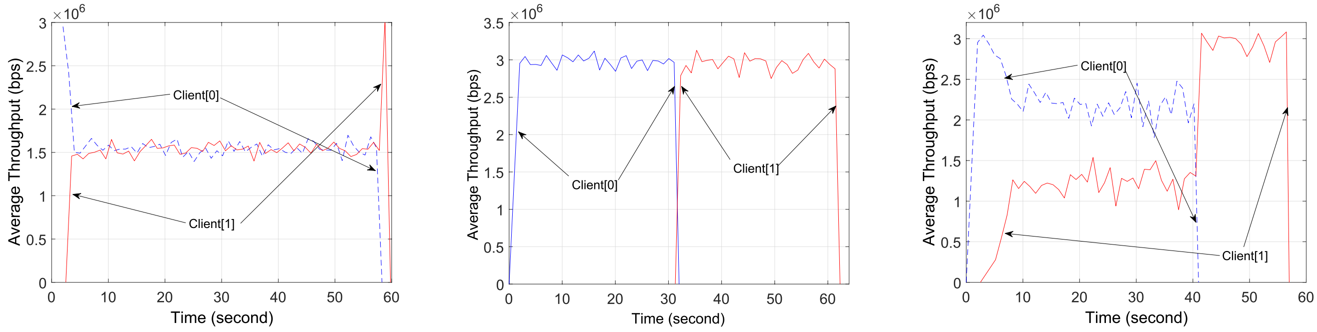
In our simulation, the scheduler is implemented in the BS. Each client has a dedicated queue in the BS. The packets sent to cli[i] is stored in Queue[i] ($i \in \{0, 1, 2, \cdots\}$). Each queue can store 100 packets, which is sufficiently large to avoid frequent packet dropping. How to minimize the buffer size while ensuring no degradation of performance is another important issue and out of the scope of this paper [17]. The scheduler determines which queue is chosen to transmit, and how many packets can be transmitted. When the packets are sent out, they will be delivered to the clients over the wireless channel. Based on the practice that wireless links tend to be the bottleneck link in a network, we assume that the bandwidth between the intermediate routers and the servers are large enough, and the bottleneck link is the wireless channel to/from the client, such that the packet delay jitter in other hops can be ignored compared to the delay in the wireless access links.

B. Homogeneous Networks With HOLD

In this simulation, we focus on the fairness performance of the scheduling algorithms in a homogeneous network, in which the channel rate distribution of each flow is the same. The Jain's fairness index in terms of HOL access delay of each flow is always close to 1 in our simulation with various number of flows in the system, which validates our analysis of the HOL access delay in the homogeneous networks in Proposition 1. More information can be found in the comprehensive simulation results in Fig. 8.

Next we investigate the throughput performance, and we begin with the simplest simulation scenario, in which there are only 2 clients (cli[0,1]). The channel rate of the wireless link in the network can be randomly selected from the set of {2Mbps, 3Mbps, 4Mpbs} with the same probability. The starting time of the TCP connection of cli[0] and cli[1] follows exponential distribution with mean value of 0s and 2.5s, respectively. We compare the performance of HOLD, QMW and F-D-MW in the 2-flow network scenario in Fig. 4. Fig. 4(a) shows the performance of HOLD in terms of throughput, and the y-axis is the throughput of each client averaged over a sliding time window of 1 second. We can tell from the figure that before the beginning of cli[1]'s TCP session, cli[0] used the channel exclusively, but as long as cli[1] started its data request, the two clients in the network began to evenly share the network bandwidth.

Fig. 4(b) shows the fairness performance of classic QMW in terms of throughput, with the same network settings as those in Fig. 4(a). In Fig. 4(b), after the TCP connection between cli[0] and the server has been established, cli[0] dominated the

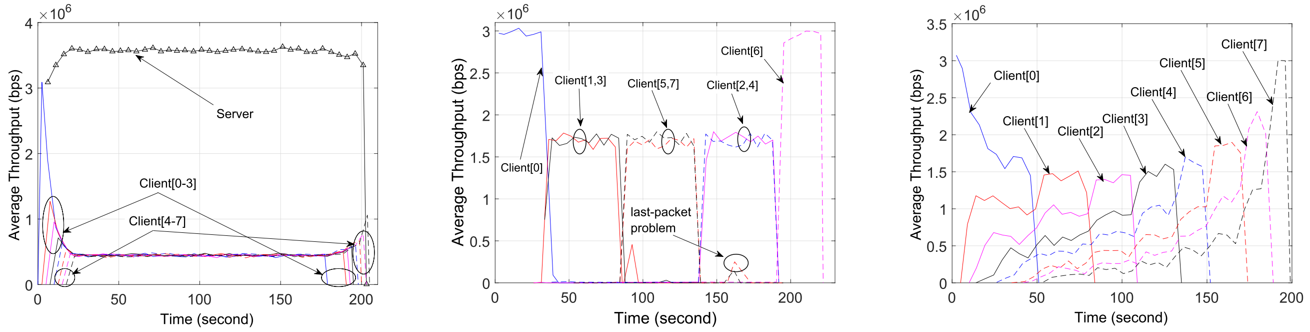


(a) HOLD scheduling algorithm.

(b) QMW scheduling algorithm.

(c) F-D-MW scheduling algorithm.

Fig. 4. Throughput performance in the 2-flow homogeneous network.



(a) HOLD scheduling algorithm.

(b) QMW scheduling algorithm.

(c) F-D-MW scheduling algorithm.

Fig. 5. Throughput performance in the 8-flow homogeneous network.

channel usage. cli[1], however, suffered from the starvation, and was able to begin the transmission only after cli[0] received all the data. Fig. 4(c) shows the performance of F-D-MW in the same setting. Although cli[1] did not have the starvation problem and was able to have data transmission at the same time with cli[0], they could not share the channel resource evenly since the throughput of cli[1] is only about half of that of cli[0], which shows a noticeable unfairness.

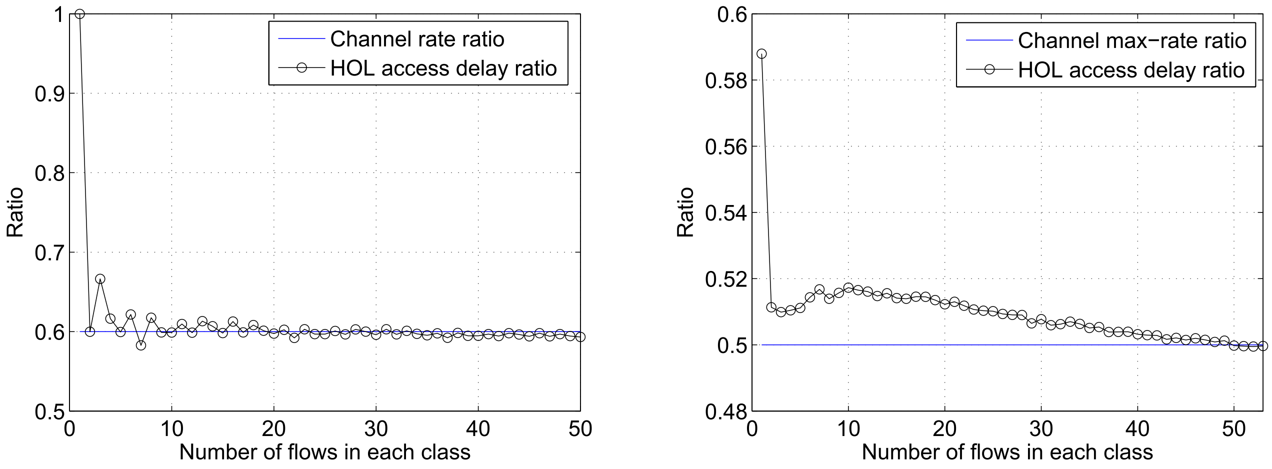
To further verify the fairness performance, we increased the number of flows in the system to 8, and the results are shown in Fig. 5. In this simulation, the channel setting is the same as that in Fig. 4. The starting time of the TCP connection for cli[0-7] follows exponential distribution with mean values of 0s, 2.5s, 5s, 7.5s, 10s, 12.5s, 15s and 17.5s, respectively.

We can observe the performance of HOLD in Fig. 5(a) that before the data request of cli[1], the average throughput of cli[0] kept increasing. But after cli[1-7] began their TCP sessions, instead of dominating the usage of the channel, cli[0] shared the channel resource with the other clients and its throughput decreased quickly to around 0.5Mbps, while the other 7 clients were also able to increase the throughput to about 0.5Mbps. This trend stayed stable for the rest of the simulation time until the end of the transmission. This observation verifies the desirable fairness performance of HOLD. Furthermore, from the server's throughput performance, which indicates the total throughput of the network, we can observe that the network throughput is above the average channel rate. In fact, thanks to the opportunistic scheduling feature of HOLD, we can achieve the multi-user diversity gain. Fig. 5(b) shows the performance of QMW, where cli[0] dominated the channel usage from the beginning to the end of its transmission. The last-packet problem of QMW mentioned in Sec. II-B can also be observed in Fig. 5(b). Fig. 5(c) shows the performance of F-D-MW. The priority of the older flows in F-D-MW can be observed through the whole simulation. As a result, if a long-lived TCP flow exists, other flows may starve.

C. Heterogeneous Networks With HOLD

In this simulation, we focus on the fairness performance of the scheduling algorithms in a heterogeneous network, in which the flows are categorized into two classes according to the various channel rate distributions. Fig. 6(a) shows the ratio of the average HOL access delay of the two classes, which have non-identical (heterogeneous) constant channel rates. Here the channel rates of the two class are 3Mbps and 5Mbps, so the channel rate ratio is 0.6 which is shown by the blue straight line. We can observe that the ratio of the HOL access delay converges quickly to the blue straight line as the number of flows increases. This verifies Proposition 2.

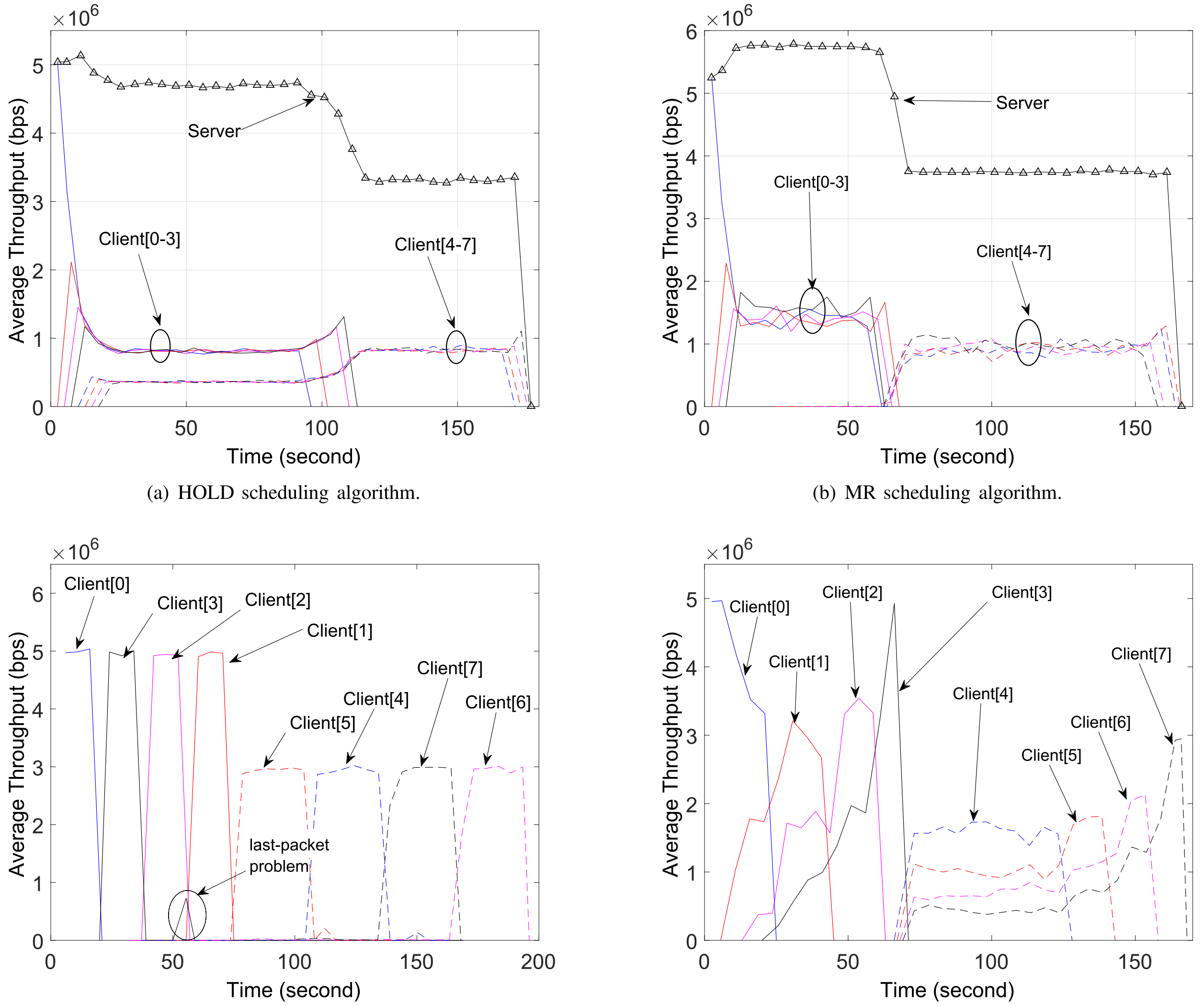
Fig. 6(b) shows the ratio of average HOL access delay of the two classes flows which have independent and non-identical (heterogeneous) channel rate distributions. In this figure, we use the two-state Markov channel model, and the available channel rate set for class 1 and 2 is {2Mbps, 3Mbps}, and {5Mbps, 6Mbps}, respectively. Each rate in the set has the probability of 0.5. The maximum channel rate ratio is 0.5



(a) Without channel variations.

(b) With channel variations.

Fig. 6. HOL access delay ratio in the heterogeneous network.



(a) HOLD scheduling algorithm.

(b) MR scheduling algorithm.

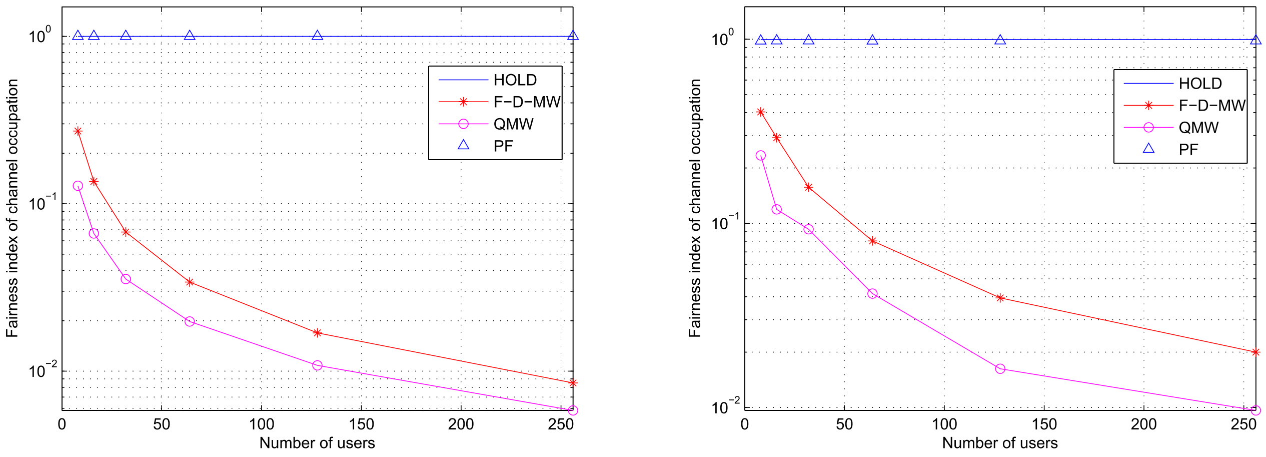
(c) QMW scheduling algortihm.

(d) F-D-MW scheduling algorithm.

Fig. 7. Throughput performance in the 8-flow heterogeneous network with channel variations.

which is shown by the blue straight line. We can also observe that the ratio of the HOL access delay converges to the blue straight line as the number of flows increases. This also verifies the analysis in Proposition 3.

Fig. 7(a) shows the fairness performance of HOLD in an 8-flow heterogeneous system, in which cli[0-3] belong to class 1 and cli[4-7] belong to class 2. The simulation results are averaged over a sliding time window of 5 seconds.



(a) Homogeneous network.

(b) Heterogeneous network.

Fig. 8. Jain's fairness index in terms of channel occupation.

The available channel rate sets for class 1 and 2 are {4Mbps, 5Mbps, 6Mbps} and {2Mbps, 3Mbps, 4Mbps}, respectively. The starting time of the TCP connection for cli[0-7] is 0s, 2.5s, 5s, 7.5s, 10s, 12.5s, 15s and 17.5s, respectively. We can observe that before 10s, cli[0-3] evenly shared the channel resource, which is the same as what we can expect in a homogeneous network. After 10s, cli[4-7] took turns to begin their TCP sessions. In this case, instead of suffering from starvation, cli[4-7] were able to occupy a portion of channel time. After 17.5s, the ratio of the mean window-averaged throughput between cli[0-3] and cli[4-7] is approximately 0.447, which is very close to the square of the maximum rate ratio ($\eta_{1,2}^{R} = 0.444$). This observation also verifies our throughput analysis in Sec. IV-C.

For comparison, Fig. 7(b) shows the performance of MR in the heterogeneous network with channel variations. In Fig. 7(b), MR shows a very desirable total network throughput performance. Since MR tries to select the flow with the maximum possible channel rate to transmit, the total network throughput is very close to 6Mbps. But the flows from class 1 occupied the major part of the throughput, and the flows from class 2 had very little share of the network throughput due to the difference on the channel conditions. This observation shows the unfairness problem in MR in heterogeneous networks.

The performances of QMW and F-D-MW are presented in Figs. 7(c) and 7(d). In Fig. 7(c) we can see that, for QMW, after the TCP connection of cli[0] was established, the data transmission for cli[0]'s TCP session dominated the whole network throughput, while the other clients had to yield the channel resource to cli[0]. Since all (or most) of the requested data has been received, cli[0] finished its channel usage by 20s. Only after this time, one of the other clients had a chance to be scheduled. But similarly, this client also dominated the transmission in the whole network, and the remaining clients continued to suffer from starvation. This observation remains the same through the whole simulation time. Similar to Fig. 5(b), we can see the last-packet problem of QMW. We note that only one flow is scheduled at a time here while two clients can be schedule simultaneously in Fig. 5(b). This is because the RRT is smaller here according to the parameter settings, and the number of packets in one flow's queue can grow fast to earn enough priority such that the TCP connection of the other flows cannot be established. The performance of F-D-MW in Fig. 7(d) is similar to the one in Fig. 7(b), where the newer flows suffer from starvation till the old ones leave.

The fairness index in terms of the channel occupation time with an increasing number of flows in the homogeneous network is shown in Fig. 8(a). In this simulation, we used long-lived TCP flows in the homogeneous networks and counted the channel occupation of each flow over the period of 512 seconds. The Jain's index of all the flows in terms of channel occupation was investigated here. Let c_{ki} denote the channel occupation of $Q_{ki}(t)$. The fairness index of the homogeneous system $\mathcal{J}_{homo}$ is calculated as

$$\mathcal{J}_{homo} = \frac{(\sum_{k=1}^{K}\sum_{i=1}^{N_k(t)} c_{ki})^2}{N(t)\sum(c_{ki})^2}.$$

With the increase of the number of flows in the system, the fairness of HOLD is not affected, and very close to 1 as the PF scheduling algorithm. However, for F-D-MW and QMW, since the channel time is shared by only one or a few number of flows in the system in a long period, with the increasing of flows in the system, the fairness index is monotonically decreasing.

In heterogeneous networks, we use weighted fairness index to measure the fairness of HOLD. The weight of each flow w_{ki} is the maximum rate that can be achieved, and the channel occupation time is proportional to w_{ki}, so we use the normalized channel occupation time to measure the fairness in heterogeneous networks. The fairness index of the system $\mathcal{J}_{hete}$ is calculated as

$$\mathcal{J}_{hete} = \frac{(\sum_{k=1}^{K}\sum_{i=1}^{N_k(t)} c_{ki} w_{ki}^{-1})^2}{N(t)\sum(c_{ki} w_{ki}^{-1})^2}.$$

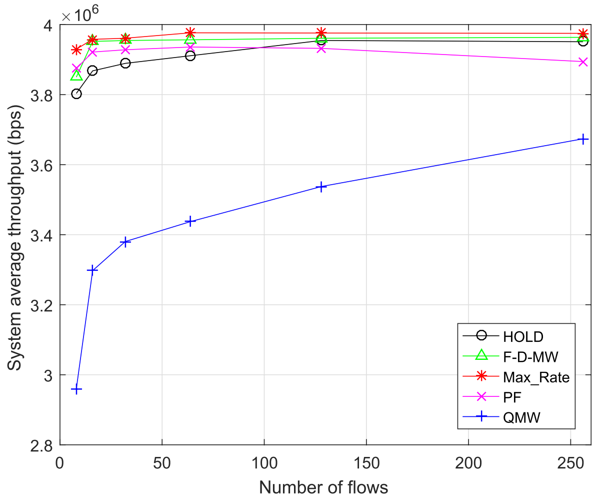


Fig. 9. System throughput of different algorithms with increasing number of flows.

With this definition, the weighted fairness of HOLD is also verified to be as good as PF through simulation as shown in Fig. 8(b). Although QMW and F-D-MW are not designed to achieve weighted fairness, we put them together in Fig. 8(b) as a reference to the interested readers.

The corresponding system throughput of Fig. 8 is shown in Fig. 9, which indicates that the proposed HOLD algorithm not only provides fair resource allocation as PF, but also maintains the throughput performance in very high level as that of MR and F-D-MW, especially when the number of flows in the system is sufficiently large. Different from those with HOLD, MR and F-D-MW, with the increase of the number of flows, the throughput of PF decreases since its first priority is to guarantee fairness. On the contrary, the throughput of QMW is noticeably lower.

D. System Stability

The simulation in Fig. 10 investigates the throughput-optimality of different scheduling algorithms with flow-level dynamics. We used a two-state channel model, and considered heterogeneous network with 5 classes of flows with the traffic intensity 0.99. The number of flows in Fig. 10 is the number of simultaneous backlogged flows in the network in any given slot. Consider that each flow arrives in the system with a finite amount of data to transmit, and leaves the system once all the data are transmitted. When the system is stable, we have a finite number of flows in the system. When the number of flows grows into infinity when $t \to \infty$, the total amount of data in the system also grows into infinity and thus we have system instability [16]. In the simulation, the number of flows in the system can be stabilized by MR and F-D-MW which are proved to be throughput-optimal in the literature, while the proposed HOLD algorithms has almost the identical performance compared with MR and F-D-MW which confirms the throughput-optimality of HOLD. In contrast, the system cannot be stabilized by QMW, which is shown not to be throughput-optimal with flow-level dynamics.

Since in the current cellular networks, the most widely used scheduler is the PF algorithm, we also included PF in the simulation. The history update window was set to be 1000 slots as recommended in [32]. Fig. 10 shows that PF is not throughput-optimal because it is not able to stabilize the system when other provable throughput-optimal schedulers can. This result also validates the conclusion in [27] and [28], which offered a number of examples to show the instability of PF, and the theoretical proof showing that the stability region is less than the capacity region, respectively.

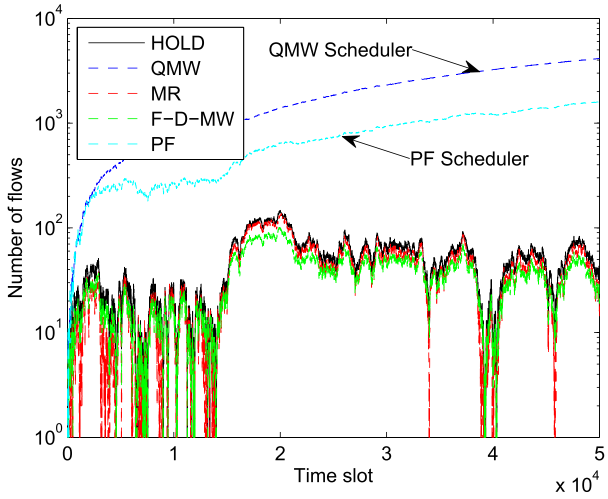


Fig. 10. Throughput-optimality test.

VI. Conclusion

In this paper, we have studied the compatibility between the TCP congestion control scheme and HOLD scheduling algorithm. Since we observed that other classic throughput-optimal scheduling algorithms, e.g., QMW and F-D-MW, encounter the starvation problem when scheduling TCP regulated flows, we designed HOLD scheduling algorithm, which is shown to be compatible with TCP flows through theoretical analysis. To verify the theoretic results, comprehensive simulations have been conducted to compare the performances of HOLD with QMW, F-D-MW and MR in both homogeneous and heterogeneous systems. Simulation results have validated the theoretical analysis, and demonstrated the superior performance of HOLD when serving TCP flows compared to the existing solutions in terms of fairness. The result of system stability have validated that HOLD is throughput-optimal, while the PF scheduler is not throughput-optimal.

For the future work, we consider to design the scheduling algorithm which is able to provide the differentiated services based on the application QoS requirements, such that not only the throughput of the system can be improved, but also the QoS of different applications can be taken into consideration.

References

[1] L. Cai, X. Shen, and J. W. Mark, *Multimedia Services in Wireless Internet: Modeling and Analysis*. Hoboken, NJ, USA: Wiley, 2009.

[2] L. Cai, X. Shen, J. Pan, and J. W. Mark, “Performance analysis of TCP-friendly AIMD algorithms for multimedia applications,” *IEEE Trans. Multimedia*, vol. 7, no. 2, pp. 339–355, Apr. 2005.

[3] X. Wang and Z. Li, “Joint optimization of TCP congestion control and distributed CSMA scheduling,” in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2012, pp. 5729–5733.

[4] S. Xiang and L. Cai, "Transmission control for compressive sensing video over wireless channel," *IEEE Trans. Wireless Commun.*, vol. 12, no. 3, pp. 1429–1437, Mar. 2013.

[5] L. Tassiulas and A. Ephremides, "Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks," *IEEE Trans. Autom. Control*, vol. 37, no. 12, pp. 1936–1948, Dec. 1992.

[6] H. Seferoglu and E. Modiano, "TCP-aware backpressure routing and scheduling," *IEEE Trans. Mobile Comput.*, vol. 15, no. 7, pp. 1783–1796, Jul. 2016.

[7] M. J. Neely, "Stochastic network optimization with application to communication and queueing systems," *Synthesis Lectures Commun. Netw.*, vol. 3, no. 1, pp. 1–211, 2010.

[8] M. Andrews, K. Kumaran, K. Ramanan, A. Stolyar, R. Vijayakumar, and P. Whiting, "Scheduling in a queuing system with asynchronously varying service rates," *Probab. Eng. Inform. Sci.*, vol. 18, no. 2, pp. 191–217, Apr. 2004.

[9] X. Wang and L. Cai, "Limiting properties of overloaded multiuser wireless systems with throughput-optimal scheduling," *IEEE Trans. Commun.*, vol. 62, no. 10, pp. 3517–3527, Oct. 2014.

[10] S. Shakkottai and A. L. Stolyar, "Scheduling for multiple flows sharing a time-varying channel: The exponential rule," *Trans. Amer. Math. Soc.*, vol. 2, pp. 185–202, Dec. 2002.

[11] B. Sadiq, S. J. Baek, and G. de Veciana, "Delay-optimal opportunistic scheduling and approximations: The log rule," *IEEE/ACM Trans. Netw.*, vol. 19, no. 2, pp. 405–418, Apr. 2011.

[12] J. Chen, W. Xu, S. He, Y. Sun, P. Thulasiraman, and X. S. Shen, "Utility-based asynchronous flow control algorithm for wireless sensor networks," *IEEE J. Select. Areas Commun.*, vol. 28, no. 7, pp. 1116–1126, Sep. 2010.

[13] L. Zheng and L. Cai, "A distributed demand response control strategy using Lyapunov optimization," *IEEE Trans. Smart Grid*, vol. 5, no. 4, pp. 2075–2083, Jul. 2014.

[14] X. Wang, Y. Chen, L. Cai, and J. Pan, "Scheduling in a secure wireless network," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Apr. 2014, pp. 2184–2192.

[15] J. Chen, W. Xu, S. He, Y. Sun, P. Thulasiraman, and X. Shen, "Utility-based asynchronous flow control algorithm for wireless sensor networks," *IEEE J. Sel. Areas Commun.*, vol. 28, no. 7, pp. 1116–1126, Sep. 2010.

[16] P.V. Ven, S. Borst, and S. Shneer, "Instability of MaxWeight scheduling algorithms," in *Proc. IEEE INFOCOM*, Apr. 2009, pp. 1701–1709.

[17] S. Liu, L. Ying, and R. Srikant, "Throughput-optimal opportunistic scheduling in the presence of flow-level dynamics," *IEEE/ACM Trans. Netw.*, vol. 19, no. 4, pp. 1057–1070, Aug. 2011.

[18] B. Sadiq and G. D. Veciana, "Throughput optimality of delay-driven MaxWeight scheduler for a wireless system with flow dynamics," in *Proc. 47th Annu. Allerton Conf. Commun. Control, Comput.*, Sep. 2009, pp. 1097–1102.

[19] J. Padhye, V. Firoiu, D. F. Towsley, and J. F. Kurose, "Modeling TCP Reno performance: A simple model and its empirical validation," *IEEE/ACM Trans. Netw.*, vol. 8, no. 2, pp. 133–145, Apr. 2000.

[20] M. Mathis, J. Mahdavi, S. Floyd, S. Floyd, and A. Romanow, *TCP Selective Acknowledgment Options*, document RFC 1-12, Oct. 1996.

[21] B. Ji, C. Joo, and N. B. Shroff, "Exploring the inefficiency and instability of back-pressure algorithms," in *Proc. IEEE INFOCOM*, Apr. 2013, pp. 1528–1536.

[22] B. Li, A. Eryilmaz, and R. Srikant, "On the universality of age-based scheduling in wireless networks," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Apr. 2015, pp. 1302–1310.

[23] X. Lin and N. B. Shroff, "Joint rate control and scheduling in multihop wireless networks," in *Proc. IEEE Conf. Decision Control (CDC)*, vol. 2. Dec. 2004, pp. 1484–1489.

[24] Y. Yu and G. B. Giannakis, "Joint congestion control and OFDMA scheduling for hybrid wireline-wireless networks," in *Proc. IEEE Int. Conf. Comput. Commun. (INFOCOM)*, May 2007, pp. 973–981.

[25] A. Eryilmaz and R. Srikant, "Fair resource allocation in wireless networks using queue-length-based scheduling and congestion control," *IEEE/ACM Trans. Netw.*, vol. 15, no. 6, pp. 1794–1803, Dec. 2005.

[26] L. A. Grieco and S. Mascolo, "Performance evaluation and comparison of Westwood+, New Reno, and Vegas TCP congestion control," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 34, no. 2, pp. 25–38, Apr. 2004.

[27] M. Andrews, "Instability of the proportional fair scheduling algorithm for HDR," *IEEE Trans. Wireless Commun.*, vol. 3, no. 5, pp. 1422–1426, Oct. 2004.

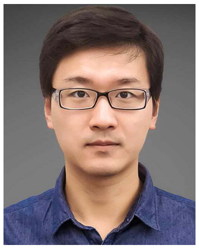
[28] X. Wang and L. Cai, "Stability region of opportunistic scheduling in wireless networks," *IEEE Trans. Veh. Technol.*, vol. 63, no. 8, pp. 4017–4027, Oct. 2014.

[29] *3GPP Technical Specification 36.300 V13.2.0, 3GPP, Release 13*, Dec. 2015.

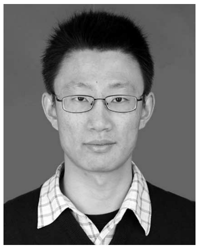
[30] R. Elliott, "A measure of fairness of service for scheduling algorithms in multiuser systems," in *Proc. IEEE Can. Conf. Electr. Comput. Eng. (CCECE)*, vol. 3. May 2002, pp. 1583–1588.

[31] Y. Chen, X. Wang, and L. Cai, "HOL delay based scheduling in wireless networks with flow-level dynamics," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2014, pp. 4898–4903.

[32] A. Jalali, R. Padovani, and R. Pankaj, "Data throughput of CDMA-HDR a high efficiency-high data rate personal communication wireless system," in *Proc. IEEE Spring 51st Semiannu. Veh. Technol. Conf.*, vol. 3. May 2000, pp. 1854–1858.



Yi Chen (S'14) received the B.Eng. and M.A.Sc. degrees in communications and information engineering from the Northwestern Polytechnical University, Xi'an, China, in 2008 and 2011, respectively. He is currently pursuing the Ph.D. degree with the Department of Electrical and Computer Engineering, University of Victoria, Canada. His current research interests include scheduling design and resource allocation in wireless networks.



Xuan Wang received the B.Eng. degree in information security and the M.S. degree in signal and information processing from the Beijing University of Posts and Telecommunications, in 2007 and 2010, respectively, and the Ph.D. degree in electrical engineering from the Department of Electrical and Computer Engineering, University of Victoria, under the supervision of Prof. Lin Cai. His research interests include scheduling, resource allocation, and cross-layer design in wireless networks.

Lin Cai (S'00–M'06–SM'10) received the M.A.Sc. and Ph.D. degrees in electrical and computer engineering from the University of Waterloo, Waterloo, Canada, in 2002 and 2005, respectively. Since 2005, she has been with the Department of Electrical and Computer Engineering, University of Victoria, where she is currently a Professor. Her research interests span several areas in communications and networking, with a focus on network protocol and architecture design supporting emerging multimedia traffic over wireless, mobile, ad hoc, and sensor networks.

Dr. Cai has served as a TPC Symposium Co-Chair of the IEEE Globecom'07, Globecom'10, and Globecom'13. She served as an Associate Editor of the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS, the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, the *EURASIP Journal on Wireless Communications and Networking*, the *International Journal of Sensor Networks*, and the *Journal of Communications and Networks*, and as the Distinguished Lecturer of the IEEE VTS Society. She was a recipient of the NSERC Discovery Accelerator Supplement Grants in 2010 and 2015, respectively, and the Best Paper Awards of the IEEE ICC 2008 and the IEEE WCNC 2011.