MAMS: Mobility-Aware Multipath Scheduler for MPQUIC

Wenjun Yang, *Graduate Student Member, IEEE*, Lin Cai, *Fellow, IEEE*, Shengjie Shu, *Student Member, IEEE*, Jianping Pan, *Fellow, IEEE, Senior Member, ACM*, and Amir Sepahi, *Graduate Student Member, IEEE*

***Abstract*— Multi-homing technologies are promising to support seamless handoff and non-interrupted transmissions. Scheduling packets across multiple paths, however, has the known issue of out-of-order (OFO) due to the heterogeneity of the paths, which is detrimental to users' quality of experience (QoE). Wireless link characteristics undergo a fast change over time in mobile environments, thus aggravating the OFO issue. In this paper, we present a novel mobility-aware multipath QUIC (MMQUIC) framework in which interactions between link and transport layers are introduced so that the scheduler at a mobile sender is aware of uplink variations, and a new ACK packet structure is designed to inform the scheduler of downlink variations when the receiver is mobile. Based on MMQUIC, a Mobility-Aware Multipath Scheduler (MAMS) is developed, which forecasts the path conditions in successive time slots based on historical and current end-to-end (E2E) path conditions, along with wireless uplink/downlink conditions, and pre-allocates packets on multiple paths accordingly. We conduct a series of experiments to evaluate the performance of MAMS using network simulator 3 (ns-3). Simulation results demonstrate that MAMS effectively leverages the information related to mobility, achieving substantial performance gains w.r.t. the goodput and packet delay distribution under different mobility patterns.**

***Index Terms*— Out-of-order, multipath QUIC, packet scheduling, goodput, mobility management.**

I. Introduction

THE next-generation cellular networks, 6G, are envisioned to provide ubiquitous communication coverage around the world [1]. End users can leverage various access networks, such as unmanned aerial vehicle (UAV) swarms and dense terrestrial networks (e.g., LTE and WiFi). On the other hand, not only end-users are mobile in 6G, but also even backbones (e.g., satellite networks) and access networks (e.g., UAV) could also be mobile, which leads to frequent handoffs and possible connection breakages [2], [3]. In this case, multihoming technologies, such as multipath TCP (MPTCP) [4] and multipath QUIC (MPQUIC) [5], have a great potential to support seamless connectivity migration without interruption [6], [7].

However, different paths may have drastically different characteristics. When scheduling packets onto multiple paths, the issue of out-of-order (OFO) arrivals at the multihomed sink node is prevalent and problematic [8], [9], [10], [11], [12], [13]. A stable goodput, that is, the amount of in-order packets received at the transport layer per time unit, is preferable for many applications, e.g., web browsing, video streaming, gaming, and other delay-sensitive applications.

To deal with the OFO issue, state-of-the-art multipath schedulers, e.g., BLEST [9], DAPS [10], and OTIAS [11], applied the idea of Earliest Delivery Path First (EDPF) [14]. Specifically, they try to estimate the arrival time on each path in the successive time slots by obtaining path characteristics such as propagation delay, queuing delay, and congestion window (cwnd) from acknowledgments (ACKs). Then, the packet with the smallest sequence number is placed on the path with the earliest arrival time so that packets can arrive in sequence. Their estimation models cannot cover the scenarios with complicated packet loss events. Inspired by that, DPSAF [12] developed a loss-based throughput estimation model in conjunction with a SACK feedback detection mechanism, mitigating the OFO issue in lossy heterogeneous networks. Considering retransmission timeout (RTO), LATE [13] redesigned a more comprehensive transmission model to estimate the expected throughput on each path and schedule packets accordingly. However, in the mobile environment, the above schedulers fail to cope with the challenges raised by the user or network mobility. For instance, packet losses may arise from either handoff procedures when signal strength is too weak on an access link, or from overflow dropping when a link data rate reduction causes the access link to be congested [15]. The presence of different types of loss events and dynamic loss rates in a mobile scenario conflicts with the assumption of a stationary packet loss process in the existing schedulers.

In this paper, by extending our previous work [16] to address more challenging cases, i.e., both uplinks and downlinks are affected by mobility, we propose a Mobility-Aware Multipath Scheduler (MAMS) for MPQUIC. The main contributions of this paper are three-fold:

- We present a novel mobility-aware multipath QUIC (MMQUIC) framework, which allows the transport layer agent to obtain uplink variations when the sender is in motion. In the case that the mobile user is a receiver, MMQUIC redesigns the ACK packet so that the sender on the other end can be informed about downlink variation

Manuscript received 9 August 2023; revised 27 January 2024; accepted 14 March 2024; approved by IEEE/ACM Transactions on Networking Editor I.-H. Hou. This work was supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC), and in part by the Compute Canada. *(Corresponding author: Lin Cai.)*

Wenjun Yang, Lin Cai, and Amir Sepahi are with the Department of Electrical and Computer Engineering, University of Victoria, Victoria, BC V8P 5C2, Canada (e-mail: wenjunyang@uvic.ca; cai@ece.uvic.ca; amirsepahi@uvic.ca).

Shengjie Shu and Jianping Pan are with the Department of Computer Science, University of Victoria, Victoria, BC V8P 5C2, Canada (e-mail: shengjies@uvic.ca; pan@uvic.ca).

Digital Object Identifier 10.1109/TNET.2024.3382269

caused by the mobility of the receiver or the access network.

- On top of MMQUIC, we develop the MAMS scheduler for goodput enhancement. Being aware of link variations, MAMS can better estimate error rate, and retransmission delay, as well as the capacity of bottleneck links. Then, MAMS uses a probabilistic model to estimate the expected throughput of each path under the impact of mobility. After that, it makes an intelligent scheduling decision accordingly to enhance the quality of services.
- We implement the MMQUIC framework and associated MAMS scheduler by extending the QUIC code [17] in network simulator 3 (ns-3). Through extensive experiments and performance comparisons, the proposed MAMS achieves substantial performance gains in terms of goodput and packet delivery delay compared to the state-of-the-art schemes.

The rest of this paper is organized as follows. Section II reviews related works. Section III describes an overview of the baseline MPQUIC model and our design objective. Section IV presents the design of the MMQUIC framework. Section V elaborates on a detailed multipath scheduling design. Section VI shows numerical results along with analyses, followed by concluding remarks and further research issues in Section VII.

II. Related Works

A well-studied multipath transport protocol that provides in-order delivery services is MPTCP. It is adopted in the Access Traffic Steering, Switching, and Splitting (ATSSS) [18] architecture by the 3rd Generation Partnership Project (3GPP) for ensuring seamless handoff. CMT-SCTP is a stream-aware multipath transport protocol that can utilize information about each sub-stream to make better scheduling or congestion control decisions, as opposed to MPTCP which makes decisions for each connection [19]. With the ever-increasing demands for stringent QoS guarantees, a novel streaming-aware multipath transport protocol MPQUIC was proposed. Compared to MPTCP, MPQUIC is more desirable in mobile environments for two reasons: First, MPQUIC spends 1 RTT to initialize a subflow if the subflow has never been established before, or 0 RTT otherwise. In the event of a handover failure, MPQUIC would consume 0 RTT to restore the disconnected subflow. Furthermore, each MPQUIC connection is associated with a 64-bit connection ID (CID) instead of a four-tuple set [15]. Thus even if the address and/or port are changed due to mobility, the connection remains active.

For multipath transmission protocols, the reordering issue has been an active research topic with three categories of approaches: congestion control, path scheduling design, and forward error correction (FEC) based multipath techniques.

Congestion control for multipath protocols aims to adjust the sending rate across multiple paths to minimize the reordering delay at the receiver and improve the goodput. Authors in [20] observed that the goodput is near optimal when the end-to-end delays of two transmission paths are very close. Hence, they proposed a cwnd adaptation algorithm for the MPTCP source (CWA-MPTCP), which dynamically adjusts the cwnd according to the ratio of the maximum path delay over the minimum path delay, to mitigate the variation of end-to-end path delay. Pokhrel [21] took into account the loss and delay characteristics of the routes, and then employed a queueing-theoretic approach to proving the relationship between the queue size and reordering delay, which can be the guidance for sending rate adjustment. Recently, DEFT [22] formulated the reordering issue as an optimization problem and proved that the reordering delay is minimized if the equilibrium round-trip-time (RTT) of all paths is equalized, which aligns with the principle of the former designs. However, equalizing the RTT of all paths is hard to achieve in practice due to random events, e.g., link failures, and packet retransmission, etc.

In the context of packet scheduling, the Earliest Delivery Path First (EDPF) [14] has served as the baseline. Considering path characteristics such as propagation delay, queuing delay, and cwnd, EDPF predicts the arrival time over each path and then selects the path with the earliest arrival time to deliver the packets with the smallest sequence number, ensuring packets arrive in sequence. Delay-Aware Packet Scheduler (DAPS) [10], OTIAS [11], BLEST [9], and STTF [23] fall into the EDPF category. They suffer from performance degradation in highly lossy networks as they fail to consider retransmission delay in their arrival time estimation model. Therefore, EDPF variants, e.g., DPSAF [12] and LATE [13] were proposed. DPSAF stresses the importance of packet loss in estimating packet arrival time over each path especially when the delay of paths has a large difference. Its loss-based throughput estimation model in conjunction with SACK feedback detection can mitigate the OFO issue. DPSAF considers only two cases: no packet loss and fast retransmission, and it does not consider RTO. As observed by Padhye et al. [24], there were more timeout events than fast retransmit events in almost all of their experimental traces, and the majority of window decreases are due to time-outs, rather than fast retransmits. On the other hand, DPSAF chooses the maximum probability of different cases to conduct the estimation, which cannot appropriately show the expected value of throughput within a certain period. To compensate for the limitation of DPSAF, LATE redesigned the transmission model in which the situations of fast retransmission and RTO are comprehensively discussed. Then it uses a probabilistic approach to estimate the expected throughput on each path and schedules packets out of order accordingly, further improving the estimation accuracy and goodput in lossy networks. However, it mainly focuses on the goodput improvement by minimizing the reordering delay without consideration of the throughput maximization.

In addition to applying Automatic Repeat reQuest (ARQ) to improve reliability, FEC-based multipath approaches precode enough information in redundant packets to decode the lost packets with high probability without the need to wait for retransmissions. For instance, a Reed-Solomon (RS) code is employed in ADMIT [25], a rateless Raptor code in FMTCP [26], and a systematic random linear code in SC-MPTCP [27]. These works demonstrated that FEC is well suited to maximizing the aggregated goodput across multiple paths. The closest work to our focus on the OFO issue is Stochastic Earliest Delivery Path First (SEDPF) which uses the streaming code [28] to encode the data to reduce the in-order

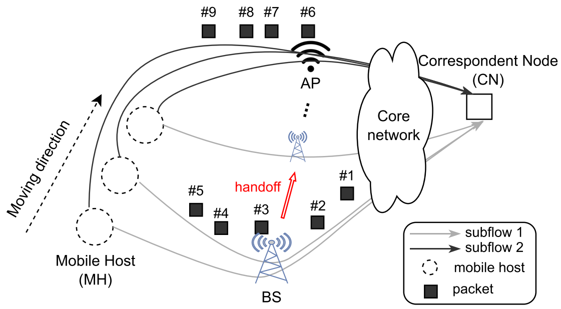


Fig. 1. The behavior of MPQUIC in mobile networks.

delivery delay. However, these approaches come at the cost of bandwidth, which is not desirable for some throughput-intensive applications in the mobile environment where the bandwidth fluctuates over time.

Overall, in the mobile environment, it is hard for the above to cope with changes in the wireless link condition due to mobility, leading to the prediction errors w.r.t. cwnd or packet arrival time: it is because conventional layered architecture does not well support inter-layer interactions. To the best of our knowledge, there is no existing solution to address the multipath OFO issue considering the link variations due to mobility, which motivates this work.

III. System Model

A. System Overview

Consider a scenario in which access points (APs) and base stations (BSs) are densely distributed over a certain area, as shown in Fig. 1, involving various types of communication technologies, such as WiFi and cellular. A mobile host (MH) with $\mathcal{P}$ interfaces is communicating with a correspondent node (CN) via APs/BSs and the Internet core network. Since MH is MPQUIC-enabled, $\mathcal{P}$ MPQUIC connections (a.k.a. subflows or paths) can be established between MH and CN. Within a short period of time, MH moves along a straight line at a constant speed towards the same direction. $R_a(t)$ and $R_b(t)$ represent the Euclidean distance between the MH and its connected AP and BS, respectively, in real time. Service outages would occur once the $R_i(t), i \in \{a, b\}$, exceeds the maximum coverage distance of the AP/BS. Thus, when an access link's received signal-to-noise ratio (SNR) falls below a certain threshold σ_i, the associated subflow has to hand off from the current AP/BS to another. Given these network settings, we present the MPQUIC scheduling model, reordering delay model, and problem statement. The main notations used throughout this paper are listed in Table I.

TABLE I

The Main Notations and Definitions

Notation	Definition
D_k	delivery time for packet k.
$\mathcal{K}, k$	the set of segments, a segment identifier.
$\mathcal{M}, m$	available paths set, a path index.
$\mathcal{P}, p$	used paths set, a path index.
$\mathcal{V}, v$	set of scheduling updates, an index of update.
ΔT_v	the duration of each scheduling cycle v.
$\Theta, \boldsymbol{\theta_p}$	scheduling policy, transmission task on p.
γ_p	retransmission delay till successful arrival.
C_p	the link capacity.
$q_p(\Delta T_v)$	transmitted packets through path p within ΔT_v.
$\mathcal{R}_{p,v}, r$	transmission rounds on p in v, a round index.
$s_{u\|r}$	the u-th sub-state given the r-th round.
$\mathbb{P}\left(s_{u\|r}, s_{u'\|(r+1)}\right)$	state transition probability from $s_{u\|r}$ to $s_{u'\|(r+1)}$.
$w_p(s_{u\|r})$	the cwnd of path i when the sub-state is $s_{u\|r}$.
τ_p	the RTT on path p.
g_p	the slow start threshold on p.
$T^C_{\langle r,p\rangle}$	time to let all $w_p(r)$ packets arrive at the receiver.
$T^B_p(s_{r,u})$	leftover time budget when the r-th round begins.
x_r	number of delivered packets in the r-th round.
δ_k	the reordering delay.
Ω, u	the entire states set, the sub-state index.
$\boldsymbol{\Lambda_p}$	mobility error rate.

B. MPQUIC Scheduling Model

The streaming service request coming from the CN node contains a set of packets, denoted by $\mathcal{K} = \{1, \ldots, K\}$.[1] Let $\mathcal{M} = \{1, \ldots, M\}$ be the set of available MPQUIC paths. Denote by w_m and τ_m the congestion window and RTT on path m, respectively. To respond to the request, the scheduling policy $\mathbf{\Pi}$ running at the MPQUIC source has two responsibilities: First, it selects the path subset $\mathcal{P}$ out of $\mathcal{M}$ for data dissemination, i.e., $\mathcal{P} \subseteq \mathcal{M}$. Second, the scheduler makes the decisions about which packets are scheduled onto path $p \in \mathcal{P}$ when the path has an available window. The decisions can be denoted by $\Theta = \langle \boldsymbol{\theta_1}, \ldots, \boldsymbol{\theta_p}, \ldots, \boldsymbol{\theta_P} \rangle$ where $\boldsymbol{\theta_p}$ represents a set of packet numbers assigned to path p. Therefore, $\mathbf{\Pi}$ can be characterized by a 2-tuple of $\langle \mathcal{P}, \Theta \rangle$.

To adapt to network condition changes, the scheduler has to keep updating $\mathbf{\Pi}$ until the completion of K packets' transmissions. The frequency to update $\mathbf{\Pi}$ varies for different scheduling algorithms. The MPQUIC Round-Robin (MPQUIC-RR) [5] and minRTT [30] schedulers refresh θ_p for each path independently and asynchronously, i.e., they change θ_p right after path p receives new ACKs or on the event that the path has room for a packet. The independent scheduling update is undesirable for solving OFO issues, so BLEST [9], DAPS [10], and LATE [13] update θ_p for each path simultaneously after all paths receive ACKs or timeout signals, which depends on the longest path RTT.

According to the scheduling mechanism adopted by [9], [10], and [13], the scheduling updating cycle can be indexed by $\mathcal{V} = \{1, 2, \ldots\}$. $\mathcal{K}_v$, $\mathcal{P}_v$, and $\Theta(v)$ represent the set of remaining buffered data, the updated set of selected paths, and the updated packet scheduling decision, respectively, at the start of scheduling cycle $v \in \mathcal{V}$. Denote by ΔT_v the time duration of cycle v, which is determined by the path with the longest RTT, i.e., $\max\limits_{p \in \mathcal{P}_v}\{\tau_p\}$. If there are paths that cannot receive ACK before the RTO timer triggers, denoted by $p' \in \mathcal{P}_v$, ΔT_v would be regulated by both the timer threshold $\mu_{p'}$ and the RTT τ_p. Hence, we have $\Delta T_v = \max\limits_{p,p' \in \mathcal{P}_v}\{\tau_p, \mu_{p'}\}$. Since MPQUIC allows ACKs to return from different paths,

[1] MPQUIC has two levels of schedulers: stream scheduler and packet scheduler [29]. The former is to make decisions for each stream, e.g., the priority of each stream, while the latter is to assign packets of the selected streams to available paths. In this paper, we adopt the packet scheduling model, so we map a service request that might consist of multiple resource streams (e.g., CSS, video, and Javascript) to a set of packets.

not necessarily the original one, ΔT_v could be as low as $(\max_{p\in\mathcal{P}_v}\{\tau_p\} + \min_{p\in\mathcal{P}_v}\{\tau_p\})/2$.

Within ΔT_v, each path $p \in \mathcal{P}_v$ takes one or multiple round trips to fulfill the transmission task $\boldsymbol{\theta_p} \in \Theta(v)$ until the task is finished or the next scheduling cycle $v+1$ starts. As a result, the time on path p during cycle v can be further divided into round trips, which are indexed by $\mathcal{R}_{p,v} = \{1, \ldots, r, \ldots, R\}$. Note R depends on the path RTT τ_p and the window size w_p.

C. Reordering Delay Model

Due to the path heterogeneity, the issue of OFO packet arrivals is predominated in the context of multipath transmission, leading to the packet reordering delay. Note that path heterogeneity refers to the differences in RTT, cwnd, and loss rate over paths.

We denote by δ_k the reordering delay that packet k due to heterogeneous routes. It can be expressed with

$$\delta_k = D_k - a_{k,p}, \tag{1}$$

where $a_{k,p}$ stands for the arriving time for a packet $k \in \boldsymbol{\theta_p}$, and D_k means the time that packet k can be delivered to the upper layer application. D_k is determined by the arrival time of all preceding $k-1$ packets, i.e.,

$$D_k = \max_{p_1,p_2,\ldots,p_k\in\mathcal{P}}\{a_{1,p_1}, a_{2,p_2}, \ldots, a_{k,p_k}\}. \tag{2}$$

Let t_v be the time in seconds when the v-th scheduling cycle starts. The time-varying parameters, e.g., C_p, w_p and τ_p should be rewritten as $C_p(t_v)$, $w_p(t_v)$ and $\tau_p(t_v)$, respectively. For readability, we ignore the notation (t_v) in the equations hereafter.

For a packet $k \in \boldsymbol{\theta_p}$, assuming its sending order is i-th and its arriving time can be estimated by

$$a_{k,p} = t_v + \sum_{n=2}^{i}\Delta t_{n,n-1} + OWD_p + \gamma_{k,p}, \tag{3}$$

where $\Delta t_{n,n-1}$, OWD_p, and $\gamma_{k,p}$ are defined below.

1) $\Delta t_{n,n-1}$: the packet sending interval between the $(n-1)$-th and the n-th packets. If they are sent within the same round $r \in \mathcal{R}_{p,v}$, the sending interval is dominated by the delay of sending a packet out over the access link, such as the access link transmission delay, or the queuing delay if other flows share the same link. If the two neighboring packets are sent at different rounds, which means the latter packet needs to wait for the window availability, $\Delta t_{n,n-1}$ would be dominated by τ_p.

2) OWD_p: the one-way delay of path p is estimated to be approximately half of the total delay τ_p, which includes propagation delay, queuing delay, and other delays along the path. In mobile environments, the OWD_p among different paths exhibits large differences and undergoes changes over time.

3) $\gamma_{k,p}$: the delay caused by the retransmission of packet k over path p. Its starting point is the packets' expected arrival time and the length depends on the severity of packet loss which is discussed in Section V-B.

D. Problem Formulation

Given the task of transmitting K packets, our objective is to find a scheduling policy $\mathbf{\Pi}$ that improves the aggregated network throughput while minimizing the average reordering delay, i.e.,

$$\max_{\mathbf{\Pi}:\langle\mathcal{P},\Theta\rangle} \sum_{v=1}^{V}\sum_{p\in\mathcal{P}_v}\frac{q_p(\Delta T_v)}{\Delta T_v} - \frac{1}{K}\sum_{v=1}^{V}\sum_{p\in\mathcal{P}_v}\sum_{k\in\boldsymbol{\theta_p}}\delta_k \tag{4}$$

where $q_p(\Delta T_v)$ means the number of packets that successfully arrive at the receiver through path p within a period of ΔT_v.

Here V refers to the maximum cycles the MPQUIC sender takes to finish the task of K packets' transmissions. $\sum_{v\in\mathcal{V}}\sum_{p\in\mathcal{P}_v} q_p(\Delta T_v)$ stands for the total number of packets arrived at the receiver within V cycles, so we have $\sum_{v\in\mathcal{V}}\sum_{p\in\mathcal{P}_v} q_p(\Delta T_v) = K$ and the first term of (4) is maximized when $\sum_{v\in\mathcal{V}}\Delta T_v$ is minimized.

Remark 1: To minimize the completion time, the intuition is to select as many paths as possible. However, if the amount of buffered data is small, employing more paths can result in poorer performance [31]. First, we sort all paths in ascending order according to their RTT $\tau_m, m \in \mathcal{M}$ and obtain the new path set $\mathcal{M}' = \{m_1, m_2, \ldots, m_M\}$ in which $\tau_{a_1} \leq \tau_{a_2} \leq \tau_{a_M}$. Given the remaining buffered data $\mathcal{K}_v$ at time t_v, the number of elements in the set $\mathcal{K}_v$ is denoted by $|\mathcal{K}_v|$. If $|\mathcal{K}_v| \geq \sum_{m\in\mathcal{M}'} q_m(\Delta T_v)$, we should use all M paths, i.e., $\mathcal{P} = \mathcal{M}'$, to obtain the maximum aggregated throughput in the current cycle, thus speeding up the completion in the rest of scheduling cycles. Otherwise, we choose the number of paths that is sufficient to finish the buffered data while maintaining the minimum largest RTT. In this case, the number of selected paths, P, is expressed with

$$P = \min\{P^* | \sum_{i=1}^{P^*} q_{m_i}(\Delta T_v) \geq |\mathcal{K}_v|\}, m_i \in \mathcal{M}' \tag{5}$$

and thus $\mathcal{P} = \{m_1, \ldots, m_P\} \subseteq \mathcal{M}'$.

Once $\mathcal{P}$ is determined, the minimization of the second term in (4) is mainly dependent on the design of Θ because Θ affects the packet scheduling time and arrival time.

Therefore, we conclude that the key to solving the problem in (4) is to find out $q_p(\Delta T_v)$ for each path in mobile environments and then design Θ accordingly.

IV. MMQUIC Framework

Making an accurate estimation of $q_p(\Delta T_v)$ in mobile networks is challenging for two reasons. The first challenge stems from the different sources of packet losses. For instance, packet losses may be due to the handoff procedure in which the signal strength of an access link becomes very weak, or due to the overflow dropping in access links since MPQUIC has no knowledge of the link capacity changes. The second challenge is the time-varying channel capacity resulting from different mobility patterns.

The above issues boil down to one basic cause: the link variations are transparent to the transport layer. Therefore, in this section, we present the MMQUIC framework as shown in Fig. 2. Similar to MPQUIC and MPTCP, MMQUIC also

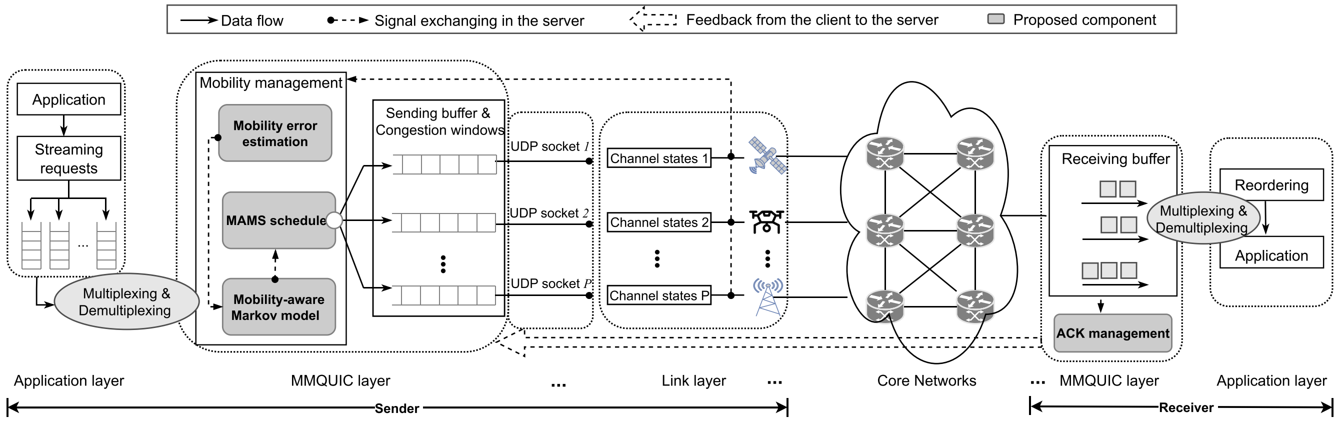


Fig. 2. Overview of the MMQUIC architecture.

follows the conventional TCP/IP layering architecture. The key contributions of MMQUIC are the components in the dark gray blocks in Fig. 2 and the established interfaces across different layers or components. In MMQUIC, schedulers at the sender side can realize channel condition changes of both uplinks and downlinks, to make an optimal multipath scheduling policy. Next, we look at the data delivery process to illustrate how the MMQUIC-enabled sender is informed of the variations of uplinks and downlinks.

A. Handling Uplink Variation for the Mobile Sender

The uplink between a mobile sender and its corresponding access networks may suffer from link quality fluctuations whenever the mobile sender is moving while transmitting. To make the MMQUIC layer aware of such fluctuations of the link layer, as shown in Fig. 2, we use the cross-layer design philosophy to add an information exchange module between the two layers. Through the cross-layer design, the channel states, e.g., transmission power, and fading parameters, can be timely obtained by the MMQUIC layer and be the input of the mobility management module which will be discussed in the next section.

B. Handling Downlink Variation for the Mobile Receiver

In contrast to the case when the sender is in motion, the mobile receiver will cause the link quality to vary at downlinks of the last mile. The mobility management module at the sender cannot be updated with the downlink dynamics directly in this case. Therefore, we add a component, ACK management, which not only incorporates a mobility information field in each ACK packet but also manages the ACK returned path to keep the sender informed quickly.

Fig. 3 shows the new MMQUIC ACK packet, consisting of two blocks: MMQUIC header, and ACK frame. To distinguish it from the standard QUIC ACK design in [32], we highlight new fields with dark gray.

1) MMQUIC Header: To make the sender react quickly to downlink variations to the mobile receiver, MMQUIC tries to return the ACK packet as soon as possible. It allows ACK packets to return from any of the paths, including the fast path and the slow path, as opposed to MPTCP whose acknowledgment is supposed to return on the same subflow. The *PID* field with a 4-bit length in the header section is to help the sender identify which subflow the ACK packet acknowledges. Upon the reception of an ACK, the sender will examine the *PID* field and update the subflow states accordingly.

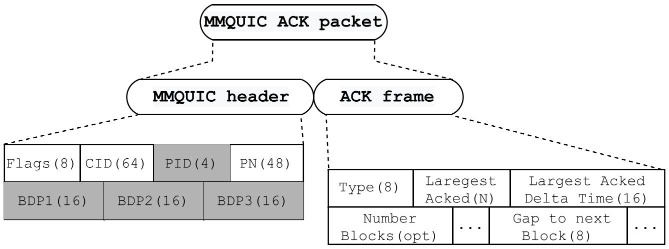


Fig. 3. MMQUIC ACK packet structure.

Moreover, to provide the sender with insights into the downlink variations, some information regarding the path bandwidth-delay-product (BDP) information would be appended in the ACK packet and bounced back from the receiver. For the backward-compatibility purpose, we leverage the existing *option* field in the header to insert the BDP information. As shown in Fig. 3, three fields related to the mobility parameters are appended in the MMQUIC header block, i.e., *BDP1*, *BDP2*, and *BDP3*. These three fields reveal the bandwidth-delay-product (BDP) of an E2E connection at present, in 0.5 seconds and 1 second, respectively. This BDP information keeps updated upon the reception of data and will be piggybacked to keep the sender informed. Note that the interval of 0.5 seconds is empirically selected, it is customizable to choose different values for different scenarios.

2) MMQUIC ACK Frame Structure: The ACK frame structure inherits the design in [32], with E2E path information including the *largest Acked* and *gap to the next block*, which indicates the path loss situation and the end-to-end latency, where the length of *N* bits denotes a variable length.

V. MAMS SCHEDULER

Using the MMQUIC framework which ensures the sender is informed, we further develop a Mobility-Aware Multipath

Scheduler (MAMS) to improve goodput in dynamic wireless systems. Three major components are included: mobility error estimation, mobility-aware Markov model for the sender, and MAMS scheduler.

A. Mobility Error Rate Estimation

In wireless networks, packet losses are often due to wireless channel fluctuations or path failure but not by link congestion [33]. In other words, the errors caused by mobility dominate the packet loss rate (PLR). To estimate $q_p(\Delta T_v)$, we first give an analysis of the mobility-associated loss rate $\boldsymbol{\Lambda_p}$ calculation as follows.

The signal-to-noise ratio (SNR) is a crucial factor in deriving the packet-level loss rate in wireless channels. In practice, wireless NICs report the real-time measurements in the channel state information (CSI) format, which serves as the proxy of the true SNR [34]. On the other hand, many mathematical formulas are proposed to derive the bit error rate (BER) based on SNR and further estimate the PLR. For example, in [35], the relationship between BER $\bar{\mathbb{P}}_b$ and SNR is

$$\bar{\mathbb{P}}_b = \frac{2\left(1-\frac{1}{L}\right)}{\log_2 L} Q\left[\sqrt{\left[\frac{3\log_2 L}{L^2-1}\right]\left[\frac{2SNR}{\log_2 M}\right]}\right], \tag{6}$$

where L is the number of levels in each dimension of the M-ary modulation system. $Q(x)$ is the Gaussian error function and is given by [36]

$$Q(x) = \int_x^{\infty} \frac{1}{\sqrt{2\pi}} e^{\frac{-y^2}{2}} dy \tag{7}$$

Under the assumption that FEC coding can recover a few bits per packet, $\boldsymbol{\Lambda_p}$ can be calculated as follows,

$$\begin{aligned} \boldsymbol{\Lambda_p} &= 1 - \mathbb{P}[\text{at most } \Psi \text{ bits are erroneous}] \\ &= 1 - \sum_{i=0}^{\Psi} \binom{Z}{i} (\bar{\mathbb{P}}_b)^i (1-\bar{\mathbb{P}}_b)^{Z-i}, \end{aligned} \tag{8}$$

where Z is the length of each packet in bits, and Ψ is the maximum number of error bits the FEC coding can handle.

Note that it is challenging to use a universal model to estimate BER based on SNR for different wireless technologies because they have distinct channel models and modulation mechanisms. In practice, it is more efficient to look into the SNR-to-BER mapping table to obtain the BER value. There have been lots of investigations such as [34], [37], and [38] shed light on the mapping from SNR to BER, which can be referred to.

B. Mobility-Aware Markov Model for the Sender

As the congestion control algorithm of QUIC inherits TCP's mechanism and most implementations of QUIC use the New Reno variant [39], we model the window behavior for each MMQUIC subflow based on New Reno congestion control.

The mobility-aware state transition process of MMQUIC subflow is shown in Fig. 4, in which two main states, application-limited (AL) and network-limited (NL), are presented.[2] It is similar to the model in [13], except that the transition from AL to NL is triggered by not only the congestion loss but also transmission failures due to mobility.

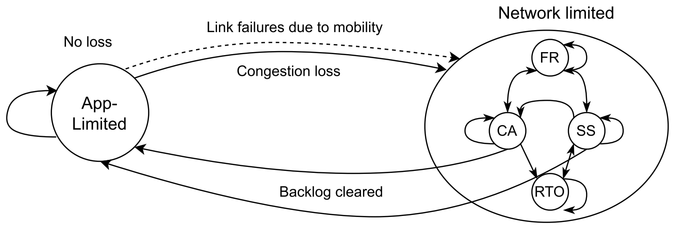


Fig. 4. State transition diagram of MMQUIC subflow.

In a network-limited state, four sub-states are defined in the set $\boldsymbol{\Omega}$ as below,

$$\begin{aligned} \boldsymbol{\Omega} = \{&\text{Slow Start (SS), Congestion Avoidance (CA),} \\ &\text{Fast Recovery (FR), Retransmission TimeOuts (RTO)}\} \end{aligned}$$

In the presence of packet losses, the time required for a packet to reach the receiver successfully varies depending on the severity of the loss event. We denote by $w_p(s_r)$ the congestion window over path p given the network state is in s_r ($s_r \in \boldsymbol{\Omega}$) during the r-th transmission round, $T^C_{\langle s_r,p\rangle}$ the total time consumed by the successful delivery of all $w_p(s_r)$ packets, including the initial shot of packet transmission in round r, and lost packet retransmission delay γ_p in the next round $r+1$. So $T^C_{\langle s_r,p\rangle}$ can be obtained by

$$T^C_{\langle s_r,p\rangle} = \tau_p/2 + \gamma_p, \tag{9}$$

where the relationship between γ_p and the sub-states s_r is summarized as

$$\gamma_p = \begin{cases} 0, & \text{if } s \in \{\text{CA}, \text{SS}\}, \\ \tau_p, & \text{if } s == \text{FR}, \\ RTO_p, & \text{if } s == \text{RTO}. \end{cases} \tag{10}$$

We refer interested readers to [13] for more discussions on (10).

C. Calculation on $q_p(\Delta T_v)$

Given a time budget of ΔT_v, multiple rounds of packets may be delivered through the path p, and at each round, the data transmission could suffer from either outage error or handover loss which is dependent on $\boldsymbol{\Lambda_p}$.

1) Recursive Problem Formulation: Let $T^B_p(s_r)$ be the leftover time budget when the r-th round of transmission starts. At the beginning, when $r = 1$, we have $T^B_p(s_1) = d^{(TD)}_{Q+1,2} + OWD_j$, and at least x_1 ($x_1 \le w_p(s_1)$) packets are delivered in the first round. Then we look into the leftover

[2] Unlike greedy flows, such as FTP, where the source rate is limited by the network, the sending rate of some flows (e.g., VoIP) is a function of media encoding and, thus, may or may not be NL. We refer to the periods where the source rate is not limited by the network as AL periods. The state transition from AL to NL is driven by the hybrid packet loss events with rate $\boldsymbol{\Lambda_p}$. The system transitions back to an application-limited state when the MMQUIC sender matches its input and output rates (e.g. when the packet backlog is cleared).

budget $T_p^B(s_2)$ which has three possibilities as below,

$$T_p^B(s_2) = \begin{cases} T_p^B(s_1) - \tau_p, \text{ if } s_2 \in \{\text{CA}, \text{SS}\}, \\ T_p^B(s_1) - \tau_p, \text{ if } s_2 == \text{FR}, \\ T_p^B(s_1) - RTO_p, \text{ if } s_2 == \text{RTO}. \end{cases}$$

As a result, for the second round, we use $s_{u|2}$ to denote the possible sub-state of the network system, where $u \in \mathcal{U} = \{1,2,3\}$. Specifically, if $u = 1$, then $s_{u|r} \in \{\text{CA}, \text{SS}\}$. If $u = 2$, then $s_{u|r} = \text{FR}$. Otherwise, $s_{u|r} = \text{RTO}$.

Comparing $T_p^B(s_{u|2})$ with $T^C_{\langle 2,p \rangle}$, the sender can determine whether the second round of transmission is allowed, if so, all $w_p(s_1) - x_1$ lost packets during the first round along with new data sitting in the sending buffer would be sent out in the second round. Repeatedly, the leftover time budget $T_p^B(s_{u|r})$ ($r \in \{3, 4 \ldots\}$) and the value of x_r during each round r are recalculated, which would be stopped till the time budget is less than $\tau_p/2$.

In conclusion, the calculation of the expected number of $q_p(\Delta T_v)$ is a recursive process. We have

$$\begin{aligned} q_p(\Delta T_v) &= q_p(T_p^B(s_1)) \\ &= x_1 + \sum_{u=1}^{3} \mathbb{P}\left(s_1, s_{u|2}\right) q_p(T_p^B(s_{u|2})), \end{aligned} \tag{11}$$

where $\mathbb{P}\left(s_1, s_{u|2}\right)$ represents the probability of network state transition from s_1 to $s_{u|2}$, and $q_p(T_p^B(s_{u|2})$ can be further calculated by

$$\begin{aligned} q_p(T_p^B(s_{u|2})) = x_{u|2} + \sum_{u'=1}^{3} \mathbb{P}\left(s_{u|2}, s_{u'|3}\right) \cdot \\ q_p(T_p^B(s_{u'|3})). \end{aligned} \tag{12}$$

Then, $q_p(T_p^B(s_{u|r}))$ is generalized as

$$\begin{aligned} q_p(T_p^B(s_{u|r})) = x_{u|r} + \sum_{u_2=1}^{3} \mathbb{P}\left(s_{u|r}, s_{u'|(r+1)}\right) \cdot \\ q_p(T_p^B(s_{u'|(r+1)})), \forall r \in \{2, 3, \ldots\}, \end{aligned} \tag{13}$$

2) Recursive Problem Decomposition: To get a closed-form expression for the recursive calculation, we decompose it in the following way.

We denote by x_r the expected number of successfully delivered packets within the r-th round. $q_p(\Delta T_v)$ is the summation of x_r over all possible rounds, i.e.,

$$q_p(\Delta T_v) = \sum_{r=1}^{r_{\max}} \mathbb{E}\left[x_r\right], \tag{14}$$

where the $r_{\max}$ is given by

$$r_{\max} = \lfloor \Delta T_v / \min\{2T^C_{\langle 1,p \rangle}\} \rfloor = \lfloor \Delta T_v / \tau_p \rfloor.$$

Note that $\lfloor \cdot \rfloor$ means floor function.

It has been demonstrated that, for mobile users, the wired parts of both the Cellular and WiFi routes are high-speed and free from impairments, and the wireless access links to the AP/BS are the bottlenecks [40], [41], [42]. Therefore, the value of x_r is not only the function of cwnd and loss rate but also the bottleneck capacity C_p.

Here the transmission rate of wireless links is dynamic as the distance R_p between MH and APs changes, which is written as

$$C_p = \eta \cdot W_p \log_2(1 + SINR_p), \tag{15}$$

where W_p is the allocated channel bandwidth to interface p of the MH, $SINR_p$ in (16) is the received signal to interference plus noise ratio of the access link, and $\eta \in (0,1)$ is a coefficient of the communication system, depending on several factors, e.g., hardware and software design, as well as the modulation and coding schemes.

$$\text{SINR}_\text{p} = \frac{Y_p(R_i)^{-\beta}}{\sigma^2 + I_p}, \tag{16}$$

where Y_p stands for the transmission power of interface p, β is the path loss exponent, σ^2 is the background noise on the frequency channel, and I_p is the associated interference.

Let $\Phi_p(s_r)$ be the maximum number of packets that can be sent out at time t when the path p is in state t_r, it is given by

$$\Phi_p(s_r) = \min\{w_p(s_r), C_p(t)\tau_p(t)\}. \tag{17}$$

3) x_r Calculation: At the very beginning (i.e., $r = 1$), the sender just receives new feedback from the receiver, and thus the network state is deterministic. Denoted by $\mathbb{P}(x_1|\Phi_p(s_1))$ the probability that x_1 out of $\Phi_p(s_1)$ packets are lost. Since the loss probability of each packet is $\boldsymbol{\Lambda_p}$ and packet losses are independent of each other [24], [43], [44], the value of $\mathbb{P}(x_1|\Phi_p(s_1))$ obeys the Binomial formula, i.e.,

$$\mathbb{P}(x_1|\Phi_p(s_1)) = \binom{\Phi_p(s_1)}{\Phi_p(s_1) - x_1} \cdot \boldsymbol{\Lambda_p}^{\Phi_p(s_1) - x_1} \cdot (1 - \boldsymbol{\Lambda_p})^{x_1}. \tag{18}$$

Hence, x_1 can be obtained with

$$\mathbb{E}\left[x_1\right] = \sum_{x_1=0}^{\Phi_p(s_1)} x_1 \cdot \mathbb{P}(x_1|\Phi_p(s_1)). \tag{19}$$

To derive the x_r where $r \in \{2, 3, \ldots\}$, we first define a conditional path characteristic set

$$\begin{aligned} \mathcal{I}(s_{u|r}) = \{w_p(s_{u|r}), g_p(s_{u|r}), \tau_p(s_{u|r}), \\ \boldsymbol{\Lambda_p}(s_{u|r}), T_p^B(s_{u|r})\}. \end{aligned}$$

Here $g_p(s_{u|r})$ refers to the slow start threshold over p given the state of $s_{u|r}$. All elements in $\mathcal{I}(s_{u|r})$ will be updated in a way that aligns with (6)-(21) in [13]. Additionally, $\Phi_p(s_{u|r})$ should be updated as follows and incorporated into the set $\mathcal{I}(s_{u|r})$.

$$\Phi_p(s_{u|r}) = \min\{w_p(s_{u|r}), C_p(t')\tau_p(t')\}, \tag{20}$$

where $t' = t_0 + OWD_j - T_p^B(s_{u|r})$ indicates the successive time slot associated with state $s_{u|r}$, and t_0 is the starting point when $r = 1$.

Furthermore, as the network state transitions are highly dependent on the number of lost packets out of $\Phi_p(s_{u|r})$, n, the state transition probability $\mathbb{P}\left(s_{u|(r-1)}, s_{u'|r}\right)$ is given by [13], [24], [43], [44]

$$\mathbb{P}\left(s_{u|(r-1)}, s_{u'|r}\right) = \sum_{n=a}^{b} \binom{\Phi_p(s_{u|(r-1)})}{n} \cdot (\boldsymbol{\Lambda_p})^n.$$

Algorithm 1 The Path Selection Algorithm

```
Input  : $\mathcal{I}(s_1) = \{w_m(s_1), g_m(s_1), \tau_m(s_1)\}$,
         $\forall m \in \mathcal{M} = \{1, \ldots, M\}$
Output: $\mathcal{P}_v$.
$Con\_I \Leftarrow$ the sender is moving // True or False
$Con\_II \Leftarrow$ the receiver is moving
$\mathcal{M}' = \{m_1, m_2, \ldots, m_M\}$
 //Sort all paths $m \in \mathcal{M}$ in ascending order by $\tau_m$
$P, Q_P \leftarrow 0$
for $i = 2; i \leq M; i++$ do
    $T_{\max} = \tau_{m_i}/2$
    for $j = 1; j \leq i; j++$ do
        $r \leftarrow 1,\ T_j^B(s_r) = T_{\max},\ q_j(T_j^B(s_r)) \leftarrow 0$
        while $T_j^B(s_r) \geq T^C_{\langle s_r, j\rangle}$ do
            if $Con\_I$ then
                $BDP^{\langle u\rangle}(t) \leftarrow$
                 read from link layer information //
                 uplink
            if $Con\_II$ then
                $BDP^{\langle d\rangle}(t) \leftarrow$ read from ACK frame //
                 downlink
            $\boldsymbol{\Lambda_j} := f_1(R_j(t))$ // Eq. (8)
            $\Phi_j(s_r) :=$
             $f_1(BDP^{\langle u\rangle}(t), BDP^{\langle d\rangle}(t), w_j(s_r))$ // Eq.
             (17)
            if $r$ == $1$ then
                $x_r := f_2(\Phi_j(s_r), \boldsymbol{\Lambda_j})$ // Eq. (19)
            else
                for each $u \in \mathcal{U} = \{1, 2, 3\}$ do
                    $x_{u|r} := f_2(\Phi_j(s_r), \boldsymbol{\Lambda_j})$ // Eq. (19)
                $x_r := f_3(x_{u|r})$ // Eq. (22)
            $q_j(T_j^B(s_r)) \leftarrow q_j(T_j^B(s_r)) + x_r$
            $r \leftarrow r + 1$
            $T_j^B(s_r) \leftarrow T_j^B(s_r) - (\gamma_{j,s_r})^+_{\tau_j}$ // Eq. (10)
            $t$  is updated with Eq. (3)
        $Q_P+ = q_j(T_j^B(s_r))$
    $P = i$
    if $Q_P \geq |\mathcal{K}_v|$ then
        break
$\mathcal{P}_v := \{m_1, \ldots, m_P\}$
return $\mathcal{P}_v$
```

$$(1 - \boldsymbol{\Lambda_p})^{\Phi_p(s_{u|(r-1)})-n}, u, u' \in \{1, 2, 3\} \tag{21}$$

in which the values of a and b for different states are summarized in Table II.

With the knowledge of $\Phi_p(s_{u|r})$, along with (13) and (21), the expected value of x_r can be derived as follows,

$$\mathbb{E}\left[x_r\right]$$
$$= \sum_{u_1=1}^{3} \sum_{u_2=1}^{3} \ldots \sum_{u_{r-1}=1}^{3} \mathbb{P}\left(s_1, s_{u_1|2}\right) \mathbb{P}\left(s_{u_1|2}, s_{u_2|3}\right) \ldots$$

TABLE II

THE SETTINGS OF a AND b WITH DIFFERENT SUB-STATES

Sub-state Classification	Parameter Settings
u' = 1	a = 0, b = 0
u' = 2	a = 1, $b = \Phi_p(s_{u\|(r-1)}) - 3$
u' = 3	$a = \Phi_p(s_{u\|(r-1)}) - 2$, $b = \Phi_p(s_{u\|(r-1)})$

$$\mathbb{P}\left(s_{u_{r-2}|(r-1)}, s_{u_{r-1}|r}\right) \mathbb{E}\left[x_{u_{r-1}|r} | \Phi_p(s_{u_{r-1}|r})\right]$$
$$= \sum_{u_1=1}^{3} \sum_{u_2=1}^{3} \ldots \sum_{u_{r-1}=1}^{3} \mathbb{P}\left(s_1, s_{u_1|2}\right) \cdot \prod_{v=2}^{r-1} \mathbb{P}\left(s_{u_{v-1}|v}, s_{u_v|(v+1)}\right)$$
$$\cdot \sum_{x_{u_{r-1}|r}=0}^{\Phi_p(s_{u_{r-1}|r})} \mathbb{P}\left[x_{u_{r-1}|r} | \Phi_p(s_{u_{r-1}|r})\right] \cdot x_{u_{r-1}|r}. \tag{22}$$

Using (19) and (22), (14) is rewritten as

$$q_p(\Delta T_v) = \mathbb{E}\left[x_1\right] + \sum_{r=2}^{r_{\max}} \mathbb{E}\left[x_r\right]. \tag{23}$$

To maximize the aggregated throughput, $\mathcal{P}$ could be derived as described in **Algorithm** 1 given the estimated $q_p(\Delta T_v)$.

In the next subsection, we will demonstrate a scheduling policy Θ design to minimize the average reordering delay.

D. Packet Scheduling Policy Θ

For a packet k sent over path p^*, the reordering delay δ_k is subject to the delay difference between p^* and other paths that have longer delays. Let $p_s \in \mathcal{P}$ be a path that is relatively slower than the paths in the path set $\mathcal{F}$, such that $\forall p_f \in \mathcal{F} \subset \mathcal{P}, \tau_{p_f} \leq \tau_{p_s}$.

The core idea of Θ is to select packets for a slower path to reduce the blocking time on its relatively faster paths. The detailed design is given in **Algorithm** 2.

For example, when the *for* loop in line 7 of **Algorithm** 2 starts, the slowest path p_s is path m_P, and the rest paths with shorter delay are in set $\mathcal{F}$. Within OWD_P which is the time duration to get a packet arrived at the receiver through p_s, $\sum_{p_f \in \mathcal{F}} q_{p_f}(OWD_P)$ packets can be processed by the relatively faster paths of p_s. To avoid the reordering delay caused by the late arrival on path p_s, we reserved the first $\sum_{p_f \in \mathcal{F}} q_{p_f}(OWD_P)$ packets for the faster paths, and select packets starting from $\sum_{p_f \in \mathcal{F}} q_{p_f}(OWD_P) + 1$ for the path p_s. To further determine how to distribute the reserved $\sum_{p_f \in \mathcal{F}} q_{p_f}(OWD_P)$ packets over all paths $p_f \in \mathcal{F}$, the for loop repeats the above procedures until all $\sum_{p_f \in \mathcal{F}} q_{p_f}(OWD_P)$ packets are appropriately scheduled. The above steps make the schedule decision for the first $\sum_{p_f \in \mathcal{F}} q_{p_f}(OWD_P) + 1$ packets while only one packet is assigned to the slowest path p_s. To make the most of the available window without exceeding the bottleneck capacity on the slowest path, we use a for loop and repeat the process until the number of packets scheduled on the slowest path approaches $\Phi_P(t_v)$ as defined in Eq. (17). This is achieved using a while loop, as shown in line 6 of **Algorithm** 2.

Algorithm 2 The Packet Scheduling Algorithm

```
   Input  : $\mathcal{I}(s_1) = \{w_p(s_1), g_p(s_1), \tau_p(s_1)\}, \forall p \in \mathcal{P}_v = \{m_1, \ldots, m_P\}$
   Output: $\Theta(v) = \langle \boldsymbol{\theta_{m_1}}, \ldots, \boldsymbol{\theta_{m_P}} \rangle$.
1  Initialization at scheduling time $t_v$
2      $LIndex, RIndex \leftarrow 0$
3      $\omega \leftarrow 0$
4      for each $p \in \mathcal{P}$ do
5          $\boldsymbol{\theta_p} \leftarrow \emptyset$
6  for $i = P; i \geq 1; i - -$ do
7      $p_s \leftarrow m_i, \mathcal{F} \leftarrow \{m_1, \ldots, m_{i-1}\}$
8      $Q_{p_s} \leftarrow 0$,
9      $FP \leftarrow$ the first packet in $\mathcal{K}_v$
10     for each $p_f \in \mathcal{F}$ do
11         Calculate $q_{p_f}(OWD_{p_s})$ according to Algorithm 1
12         $Q_{p_s} + = q_{p_f}(T^B_{p_f}(s_r))$
13     $LIndex = Q_{p_s} + FP$
14     if $i == P$ then
15         $\boldsymbol{\theta_{p_s}} = \boldsymbol{\theta_{p_s}} \cup \{LIndex\}$
16         $\mathcal{K}_v = \mathcal{K}_v \setminus \{LIndex\}$
17         continue;
18     else if $i == 1$ then
19         $RIndex = LIndex$
20         $LIndex \leftarrow FP$
21     else
22         $RIndex = LIndex$
23         $LIndex = Q_{p_s} + 1$
24     $\boldsymbol{\theta_{p_s}} = \boldsymbol{\theta_{p_s}} \cup \{LIndex, \ldots, RIndex\}$
25     $\mathcal{K}_v = \mathcal{K}_v \setminus \{LIndex, \ldots, RIndex\}$
26 return $\langle \boldsymbol{\theta_{m_1}}, \ldots, \boldsymbol{\theta_{m_P}} \rangle$
```

E. Time Complexity Analysis

In this subsection, we will briefly analyze the complexity of our algorithms in terms of time consumption.

Algorithm 1 involves three nested loops. The outer loop runs from $i = 2$ to $i = M$ and the middle loop runs from $j = 1$ to $j = i$, leading to $\sum_{i=2}^{M} i = \frac{M(M+1)}{2} - 1$ iterations. The inner loop (i.e., the *while* loop) is a recursive process, the number of iterations is determined by the $T_{\max}$. If $T_{\max}$ is less than the smallest time cost $T^C_{\langle s_r, j \rangle}$, the recursion stops. This base case is reached in constant time, $O(1)$. Otherwise, in each recursive call, three subproblems are generated, corresponding to the three states (line 19). Each subproblem involves a recursive call with a reduced time budget (line 24). Let's assume that $T_{\max}$ is reduced by a factor $\hat{\gamma}$ at each step. In that case, the number of recursive calls could be $\log_{\hat{\gamma}}(T_{\max})$. If we use a tree structure to depict the above process, the depth of the recursion tree is $\log_{\hat{\gamma}}(T_{\max})$, and at each level, there are three branches. The work done at each node is to compute the expected number of packets, which involves constant time operations. Therefore, The inner loop contributes a time complexity of $O(3^{\log_{\hat{\gamma}}(T_{\max})})$. Combining three loops, **Algorithm** 1 has the time complexity of $O(3^{\log_{\hat{\gamma}}(T_{\max})} M^2)$.

Similarly, **Algorithm** 2 involves two loops starting at lines 6 and 10, respectively. The total number of iterations is $\sum_{i=1}^{P}(i-1) = \frac{P(P-1)}{2}$ iterations. Given each step in the iterations is operating at a constant time, the time complexity of **Algorithm** 2 is $O(P^2)$.

It is noteworthy that the execution time of our algorithms imposes minimal overhead. Firstly, we can employ memorization techniques to avoid redundant calculations, significantly reducing the depth of the recursion tree and consequently lowering the actual time complexity. Secondly, in real-world scenarios, the number of communication interfaces (M) for each device is limited, and the latency differences among paths are comparatively small. This implies that the magnitude of $\log_{\hat{\gamma}}(T_{\max})$ is within acceptable limits. In our experiments, as depicted in Fig. 17, the execution time for completing **Algorithm** 1 is a mere 0.08 milliseconds when $M = 8$. Last but not least, both **Algorithm** 1 and **Algorithm** 2 occur infrequently. They are activated only at the commencement of each new scheduling cycle, with each activation spaced a few seconds apart.

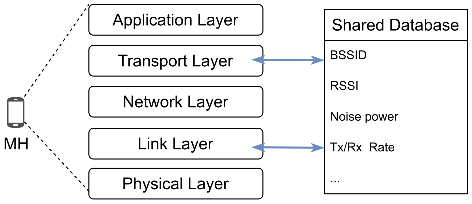


Fig. 5. Cross-layer architectural blueprint.

VI. Performance Evaluation

We use ns-3 [45] to implement the proposed MAMS over the MMQUIC framework and investigate its performance in this section, comparing it with the following two related multipath scheduling policies over MPQUIC.

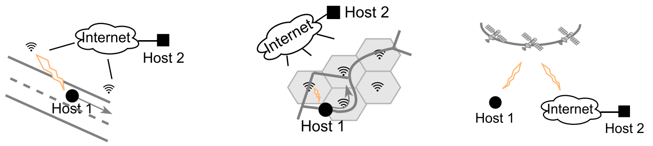
- MPQUIC-RR is the easiest but unintelligent policy to distribute packets over multiple paths in a round-robin fashion.
- LATE [13] was originally designed for MPTCP to mitigate the OFO issue in heterogeneous networks. A comprehensive analysis of loss situations and a loss-aware throughput estimation model is at the heart of it.

A. Prototype Implementation

To begin with, we extended the existing QUIC module [17] to MPQUIC based on network simulator 3 (ns-3) in accordance with the Internet Engineering Task Force (IETF) draft[3] on MPQUIC. We refer readers to our paper [46] and the public release source code at https://github.com/ssjShirley/mpquic for the implementation details of MPQUIC-ns3.

Further, as illustrated in Fig. 5, we introduce a shared database to enable the cross-layer MMQUIC framework.

[3] https://datatracker.ietf.org/doc/draft-deconinck-quic-multipath/



(a) Highway (b) Urban (c) ITSN

Fig. 6. Mobile scenarios.

Through the creation of interfaces that connect the layers to the shared database, the data stored in the database is accessible to protocol functions across different layers. For instance, in the event of changes to the access link conditions, the link layer acquires parameters such as Basic Service Set Identifier (BSSID), Received Signal Strength Indicator (RSSI), etc. These parameters are then written into the database, enabling the MMQUIC layer to retrieve this information. Note that the presence of the shared database does not change the conventional five-layer structure, and the encapsulation and decapsulation processes do not involve the data stored in the database. Therefore, the MMQUIC framework is distinguished by its backward compatibility.

On top of the MMQUIC framework, we implement the MAMS scheduler and benchmark algorithms. The source code can be accessed online.[4]

B. Experimental Settings

Fig. 6 depicts the three mobile scenarios employed in our experiments: Highway, Urban, and the integrated terrestrial and satellite network (ITSN).

In the Highway scenario, we assume a straight street with a predictable user movement direction within a defined time interval. The APs/BSs are sparsely deployed, resulting in a degraded average link bandwidth for end hosts. However, handovers occur less frequently in this scenario. In simulations, we place APs/BSs in a line and the interval is 4 Km. According to the studies in [47], the actual achieved bandwidth of a WiFi or LTE user is far below the ISP's claimed bandwidth in rural areas, so we set the bandwidth to a comparatively lower value, which is respectively 5 Mbps and 10 Mbps for LTE and WiFi paths in this scenario.

In contrast, the city roads in the Urban scenario are more winding, featuring numerous intersections. This leads to a more random user movement direction due to factors such as unexpected traffic jams. The APs/BSs are densely deployed, resulting in a comparatively higher average link bandwidth [48]. In simulations, each AP/BS is in the center of a hexagon and they are 200 meters away. The link bandwidth of LTE and WiFi is set to 50 Mbps and 200 Mbps, aligning with the measurements in [48].

Diverging from the characteristics of the first two scenarios, in ITSN, the user mobility is negligible. However, due to the high-speed movement of satellites which form the access networks and/or backbone networks, the ground users, either the sender or the receiver, are relatively moving, and the uplinks between ground users/stations and an ingress satellite and the downlinks between an egress satellite and another ground users/stations undergo frequent handoff or link failure issues. The specific settings for this scenario are summarized in Table III [15].

TABLE III
SATELLITE SIMULATION PARAMETERS

Parameters	Value
Distance between satellites and end nodes	500 Km
Speed of satellites v	7–7.5 Km/s
Uplink bandwidth W	5 MHz
Inter/intra-satellite link bandwidth	50 Mbps
Satellite transmission power	20 dBW
End-user transmission power	30 dBW
Noise power density at the satellite antenna	-164 dBm
Noise power density at end devices	-144 dBm
Path fading model	Rician
Path loss exponent	2.5

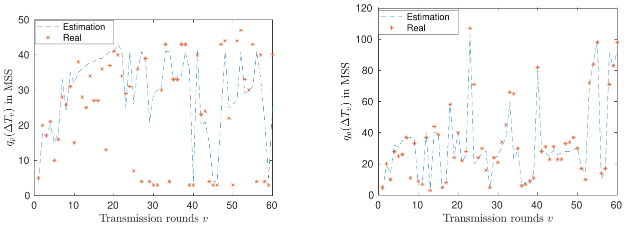
By pinging several servers using mobile phones, we realize that the E2E delay for LTE and WiFi in practice is within the ranges of 23 – 74 ms and 22 – 320 ms, respectively. Therefore, we respectively select 50 ms and 150 ms as the minimum RTT for LTE and WiFi paths in the above scenarios.

The "Internet" as depicted in Fig. 6, comprises nodes arranged in an $N \times N$ grid topology, interconnected by wired links. We can easily customize the value of N in different scenarios. For instance, we set N to 8 when there are 8 paths established between the sender and the receiver. Each wired link shares the same settings as follows: bandwidth is 500 Mbps, the delay is 2 ms, and the packet transmission error rate is 0.001%. Throughout all experiments, the settings of wired links remain unchanged.

Based on the above three scenarios, we set up three use cases to investigate the performance of MAMS: 1) the sender moves only. 2) The receiver moves only, and 3) the access networks along with backbone networks are moving. Concerning the application type, we select file downloading and Dynamic Adaptive Streaming over HTTP (DASH) to study the non-delay-sensitive yet dominated traffic and delay-sensitive real-time traffic, respectively. Therefore, we have two test combinations: file downloading over MAMS with different mobility scenarios, and DASH over MAMS given a certain mobility scenario. We employ the *BulkSendHelper* module in ns-3 to simulate a 5 MB file downloading process with different mobility cases. For the test of DASH over MAMS, we refer to the source code at [49].

For both MAMS and benchmarks, two main performance metrics are measured: goodput and the cumulative distribution function (CDF) of per-packet delivery delay. Specifically, the goodput is calculated by averaging the bytes of received in-order packets over the time duration once the receiver receives new packets with the expected sequence number. The per-packet delivery delay refers to the elapsed time from the moment that a packet is sent to the moment that it is read by an upper application.

[4] https://github.com/WenjunYang2021/MAMS. The key module of this project is located in the *quic* folder, and the most important modifications regarding the multipath scheduling are included in files quic-socket-tx-scheduler.cc, quic-socket-tx-buffer.cc, and quic-socket-base.cc.



(a) LATE

(b) MAMS

Fig. 7. The estimation model accuracy validation in the mobile uplink case.

We measure the goodput performance whenever the receiver receives in-order arrived packets, the data points collected range from 2000 to 3000, depending on the in-order arrival frequency. We track the reordering delay for around 3400 packets given the file size is 5 MB while the maximum segment size (MSS) is 1460 bytes.

C. Model Validation

As the estimation on $q_p(\Delta T_v)$ in Section V-C is a key component of MAMS, the estimation model accuracy is critical to MAMS's performance. Fig. 7 and 8 present the differences between the estimation value and real value of $q_p(\Delta T_v)$ in different scenarios. To distinguish the accuracy model of existing solutions in mobile scenarios, we incorporate LATE and our first version of MMQUIC presented in [16] here for comparison. To avoid confusion, we rename the first version of MMQUIC as MMQUIC-v1. Note that Fig. 7 displays the mobile uplink case, i.e., when the sender moves only, in which MAMS and MMQUIC-v1 behave similarly, so we present MAMS only in this case. Their differences mainly appear in the mobile downlink case where the receiver moves, which are studied in Fig. 8.

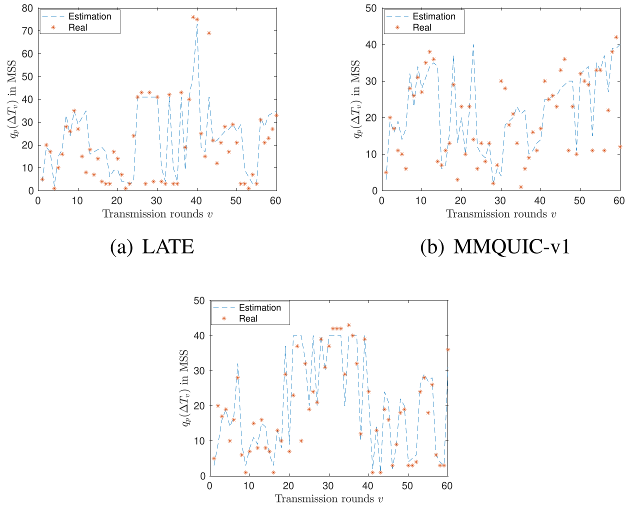
Fig. 7(a) shows that there is a big gap between the estimation and real values at some points when running LATE. In Fig. 7(b), the estimation and real values are very close, so MAMS demonstrates a desirable model accuracy in the mobile uplink case.

Regardless of mobile uplink or downlink cases, LATE is unaware of the link condition changes proactively, so Fig. 8(a) exhibits the same trend as Fig. 7(a), showing the mismatch between the estimation and real values. By analyzing Fig. 8(b) and 8(c), we can see the improvement of MAMS against MMQUIC-v1 with respect to the model accuracy is significant. This mainly benefits from the new design of the MMQUIC ACK packet structure.

In summary, MAMS can accurately calculate the path throughput in various mobile scenarios.

D. Results and Analysis

1) Mobile Uplinks: The mobile uplinks refer to the case where the sender is moving. Assuming the sender (i.e., Host 1 in Fig. 6(a)) moves along a straight line in the `Highway` scenario. Once the handoff decision indicator, Received Signal Strength (RSS), is below the threshold σ, MH would switch to another access point. Based on threshold studies in [50], we set σ to -90 dBm in our experiments.

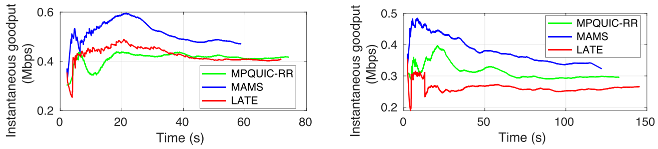


(a) LATE

(b) MMQUIC-v1

(c) MAMS

Fig. 8. The estimation model accuracy validation in the mobile downlink case.



(a) MH moves at 20 m/s

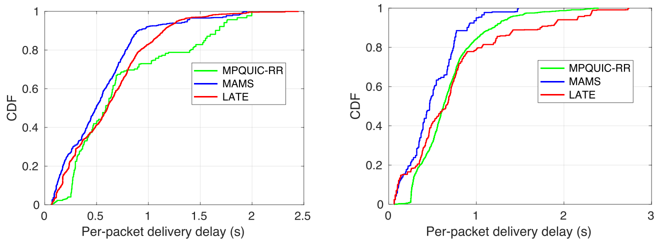
(b) MH moves at 40 m/s

Fig. 9. The instantaneous goodput over time observed at the receiver when the sender is in motion.

Fig. 9 depicts our setting and final results with respect to the goodput of three algorithms: MPQUIC-RR, MAMS, and LATE. As shown in Fig. 9, an increase in moving speed leads to a more rapid channel variations in the channel, leading to more frequent link failures and a consequent degradation in goodput across all algorithms. On the other hand, MAMS always achieves the highest goodput compared with the others, as it benefits from the consideration of the likelihood of link failure loss $\boldsymbol{\Lambda_p}$ and the cross-layer design in MMQUIC so that the MAMS deployed at the sender side can timely obtain uplink variations and make adaptive scheduling decisions. In the calculation of $\boldsymbol{\Lambda_p}$, the Nakagami fading parameter m_s and the path-loss exponent α are set to 1 and 2 [51], respectively. Comparatively, LATE is no longer as effective in mobile scenarios. Its goodput decreases as the moving speed increases and becomes even worse than MPQUIC-RR when the speed is up to 40 m/s as shown in Fig. 9(b).

MAMS makes 18% and 44% improvement against LATE and MPQUIC-RR in the 90-th percentile per-packet delivery delay, respectively as shown in Fig. 10(a), and 50% and 35% improvement compared to LATE and MPQUIC-RR in the 99-th percentile per-packet delivery delay, respectively, when the speed is up to 40 m/s as shown in Fig. 10(b).

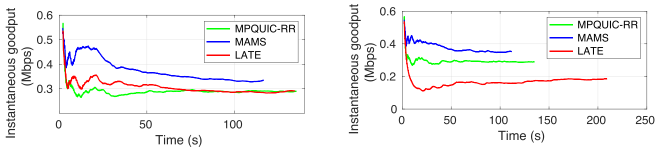
2) Mobile Downlinks: Mobile downlinks depict the use case where the receiver is the mobile host, which is the most common mobile pattern on the Internet today. To verify the effectiveness and efficiency of MAMS in this case, we conduct a series of experiments in both `Highway` and `Urban`



(a) MH moves at 20 m/s

(b) MH moves at 40 m/s

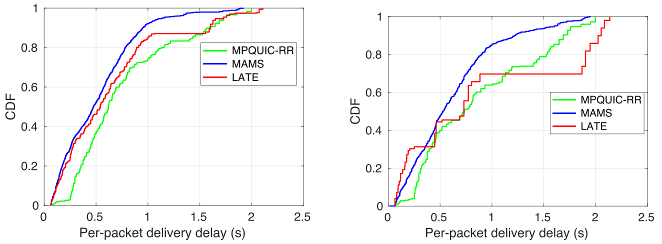
Fig. 10. Associated CDF of per-packet delivery time with different moving speeds.



(a) MH moves at 20 m/s

(b) MH moves at 40 m/s

Fig. 11. The instantaneous goodput over time observed at the receiver when the receiver is in motion in `Highway` scenario.



(a) MH moves at 20 m/s

(b) MH moves at 40 m/s

Fig. 12. Associated CDF of per-packet delivery time with different moving speeds.

scenarios with different settings w.r.t. bandwidth, mobility pattern, speed, etc.

Fig. 11 and Fig. 12 depict the performance of goodput and the delay CDF in the `Highway` scenario.

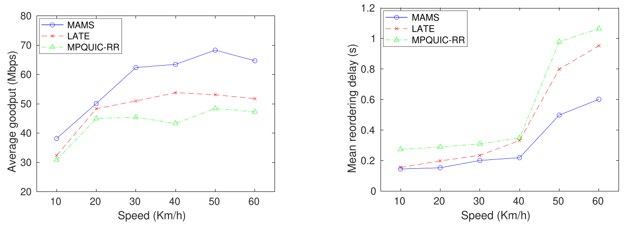
In Fig. 11(a) where the receiver is moving at 20 m/s, the mobility prediction model in LATE offers some goodput improvement against MPQUIC-RR within the first 70 s, while the advantage is diminished afterward, because LATE has no component to get the loss rate and bottleneck capacity updated, and the prediction errors accumulate over time. However, LATE still finishes the 5 MB file transmission 1.5 s faster than MPQUIC-RR. On the other hand, MAMS always tries to obtain the updated link conditions from the newly received ACK packets, leading to 17.3%-55.3% goodput enhancement against LATE and 13.4%-76.8% enhancement against MPQUIC-RR. Fig. 11(b) further presents the situations when MH moves faster. Compared to Fig. 11(a), we observe that the faster the moving speed, the worse the LATE performs. Unlike LATE, MAMS has a clear clue about the trend of path conditions including the link failure rate and the bottleneck capacity. It can largely alleviate the impact of the increased moving speed, thereby maintaining the highest goodput values. However, it is worth mentioning that MAMS would suffer from some performance degradation if it deals with the mobile downlink cases since the ACK packets have to take some time to be piggybacked so that the downlink information carried by the returned ACK frame is outdated to some extent. Then we can explain why the overall goodput results of MAMS of Fig. 11 get lower than that of Fig. 9 where MAMS can obtain the uplink variations timely.

Fig. 12 describes the per-packet delay distribution which reveals the reordering delay of each packet. Comparing Fig. 12(a) with Fig. 12(b), one obvious fact is that the per-packet delay with the LATE algorithm is largely increased as the MH speeds up to a certain level. Numerically, 90% of packets are consumed by upper applications within less than 1.603 s when using the LATE scheduler in Fig. 12(a), and 90% of packets take less than 2.053 s to be consumed in Fig. 12(b). On the contrary, the 90-th percentile delay of MAMS is increased from 0.95 s to 1.21 s as the speed increases, and that of MPQUIC-RR is increased from 1.56 s to 1.68 s. Although the per-packet delivery delay is overall increased, MAMS and MPQUIC-RR have fewer delay fluctuations in terms of the delay distribution. The increased delay in LATE is not only caused by the inaccurate loss rate estimation and accumulated prediction errors, but also the ignorance of bottleneck capacity and consequent congestion delay. Hence, we conclude that the existing prediction-based algorithms fail to address the challenges of mobility and even perform worse than conventional round-robin ways in high-speed mobile networks, and the MAMS does compensate for their weaknesses in this aspect.

Compared to the `Highway` scenario, `Urban` scenario is featured by higher link bandwidth yet more dynamic moving speed and trajectory. Under this scenario, we aim to observe how different algorithms react to more complicated mobility patterns. Given the user's moving speed is more random due to factors such as unexpected traffic jams, we assume that the moving speed is a uniformly distributed random variable. Fig. 13 shows the performance of average goodput and reordering delay with varied mean values of the speed.

In Fig. 13(a), three key conclusions can be drawn. Firstly, the increment of speed contributes to an enhanced goodput across all algorithms when the speed remains below 30 Km/h. This is because fast-moving can prevent the user from connecting to a network with poor conditions for a long time, which is detrimental to the average goodput. Secondly, as the speed surpasses 30 Km/h, the handoff happens more frequently and the path conditions become more dynamic such that all three algorithms undergo a slight performance degradation. Specifically, the goodput of MPQUIC-RR, LATE, and MAMS begins to decrease after reaching speeds of 30, 40, and 50 Km/h, respectively. Thirdly, MAMS consistently outperforms LATE and MPQUIC-RR across all conditions, particularly when the speed exceeds 20 Km/h. This suggests that MAMS excels in balancing the trade-offs associated with high speeds.

Fig. 13(b) demonstrates the fact that the mean reordering delay with all three algorithms keeps increasing as the speed increases. When the speed exceeds 40 Km/h, the reordering delay increases significantly. MPQUIC-RR is the most susceptible to the considerable dynamics, followed by LATE. MAMS maintains up to 50% improvement in terms of delay performance.



(a) The goodput vs speed (b) The reordering delay vs speed

Fig. 13. The impact of varied moving speed in the `Urban` scenario.

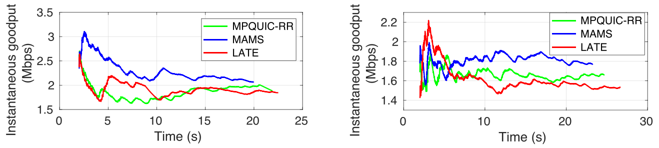
3) Network Mobility: Last but not least, we investigate the case where both uplinks and downlinks undergo dynamic changes. Here the experiments are running over the `ITSN` scenario as shown in Fig. 16(c).

Under the impact of network mobility, the goodput achieved by the three algorithms is given in Fig. 14. It can be seen that the achievable goodput of each algorithm is constrained by the uplink or downlink capacities, even though the inter/intra-satellite links provide such a high data rate as 50 Mbps. On the other hand, the fast-changing error rate caused by high-speed mobility plays a pivotal role in shaping the trend of goodput. Benefiting from the consideration of these two factors, MAMS has a more accurate prediction on the achievable throughput of each path and is more adaptive to the network dynamics, as well as higher goodput against both LATE and MPQUIC-RR. On the contrary, the comparison between LATE and MPQUIC-RR reveals that LATE has no performance gains when the satellite speed is either 7 Km/s or 7.5 Km/s.

More specifically, as shown in Fig. 14(a), MAMS has 5.06% – 48.9% and 10.3% – 64.8% goodput improvement over MPQUIC-RR and LATE, respectively. The average goodput of LATE is improved by 19.83% compared to that of MPQUIC-RR within 5 – 10 s, whereas it drops dramatically and even performs worse than MPQUIC-RR during the rest of the time slots. This aligns with the delay distribution shown in Fig. 15(a) where MAMS has the lowest 90-th percentile per-packet delay which is 0.28 s, and the counterpart of MPQUIC-RR and LATE is 0.31 s and 0.34 s respectively. From Fig. 14(b), we observe that the goodput improvement of MAMS and the performance degradation of LATE are both significant. The reason for the large reordering delay in LATE is due to the cumulative prediction errors under a high-speed mobile environment. As shown in Fig. 15(b), LATE has 17.9% and 41.9% larger 90-th percentile per-packet delay compared to MPQUIC-RR and MAMS, respectively. Additionally, LATE has a heavily long tail, some packets suffer from extremely long reordering delay of up to 1.8 s which is harmful to QoE.

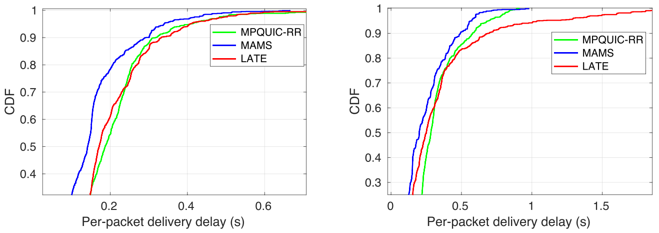
Based on the above analysis, we can conclude that MAMS is effective in reducing reordering delays, even in a challenging environment where both uplink and downlink undergo frequent changes at the same time.

4) DASH Over MAMS: To validate if the MAMS scheduler works for the real-time video streaming application, we employ the DASH application to run over MAMS. Considering the expiration of packets in video streaming is useless and regarded as packet loss [25], the per-packet delivery delay distribution is still used as a performance criterion.



(a) Satellites move at 7 Km/s (b) Satellites move at 7.5 Km/s

Fig. 14. The instantaneous goodput measured at the receiver in `ITSN` scenario.



(a) Satellites move at 7 Km/s (b) Satellites move at 7.5 Km/s

Fig. 15. Associated CDF of per-packet delivery time with different moving speeds.

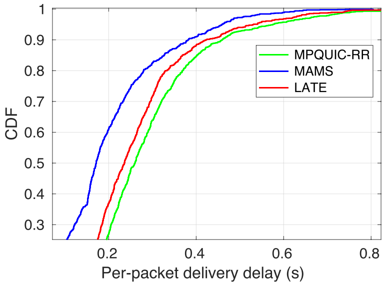
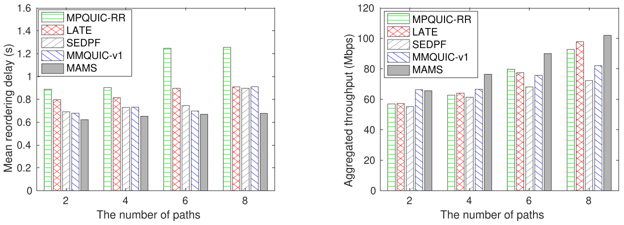


Fig. 16. Per-packet delivery delay distribution when a video client requests the video data as encoded in [52].

With respect to mobility patterns, we employ the case of mobile downlinks. The network settings are identical to the settings in Section VI-D.2. The video client, MH who is moving at 20 m/s, requests one video segment each time. The video data was provided by the authors of [52] and is consistent with real-world DASH video encoding. Assuming MH can tolerate an E2E delay of no more than 500 ms. Next, we will examine how each algorithm performs.

As shown in Fig. 16, MAMS effectively shortens the delivery delay for each packet, ensuring 97.24% of packets arrive within 500 ms. Comparatively, 94.03% and 92.7% of packets meet the deadline when using LATE and MPQUIC-RR, respectively. Therefore, choosing MAMS as the packet scheduler for the DASH application can provide users with a more pleasant experience.

5) The Impact of the Number of Paths: Compared to our initial version of MMQUIC, MMQUIC-v1, MAMS not only has an addition of minimizing the reordering delay in mobile downlink cases but also strives for throughput optimization by selecting the optimal group of paths. Therefore, in this subsection, we will compare MAMS with MMQUIC-v1 to validate the effectiveness of MAMS's throughput optimization design in the case where the number of paths is greater than 2. On the other hand, as we introduced in Section II, SEDPF [53] is the relevant one that employs FEC to address the multipath



(a) The impact on reordering delay (b) The impact on throughput

Fig. 17. The impact of a different number of paths.

OFO issue. As a result, we also incorporate SEDPF into the comparison.

Fig. 17 shows the mean reordering delay and aggregated throughput performance when the MH fetches a 10 MB file in the `Urban` scenario. Provided that the access networks are diversely consisting of WiFi, LTE, and satellite technology, the MH can choose to connect to multiple networks to establish multiple paths between the MH and the server.

First of all, according to the results given in Fig. 17(a) and 17(b), we observe that MMQUIC-v1 neither shortens the reordering delay nor improves the throughput compared to benchmark algorithms. MAMS successfully compensates for the limitations of MMQUIC-v1 in dealing with these issues. Specifically, Fig. 17(a) demonstrates that MAMS reduces the mean reordering delay by up to 0.233 seconds. Meanwhile, as shown in Fig. 17(b), MAMS achieves up to 24.2% throughput improvement.

As shown in Fig. 17(a), the mean reordering delay is increasing as the number of paths increases especially for MPQUIC-RR, which implies that the more the paths, the more severe the OFO issue. The results show that the FEC adopted by SEDPF is promising to mitigate the OFO issue compared to ARQ-based approaches (i.e., MPQUIC-RR and LATE). However, MAMS outperforms SEDPF in terms of reordering delay, especially in the scenario where 8 paths exist. One reason relies on the fact that MAMS helps reduce the reordering delay caused by path differences while SEDPF helps reduce the reordering delay caused by lost packets. The reordering delay caused by path differences is predominated when the number of paths is large enough.

In addition to minimizing the reordering delay, MAMS also makes path selections to maximize the aggregated throughput. Fig. 17(b) presents the throughput comparison across benchmark algorithms. Overall, the larger number of paths offers a higher aggregate throughput. By looking into the throughput of SEDPF, MPQUIC-RR, and LATE, SEDPF shortens the reordering delay at the cost of throughput degradation compared to MPQUIC-RR and LATE. The more paths created, the more packet redundancy the SEDPF has, and the more substantial the throughput degradation is. On the contrary, MAMS offers a promising throughput gain while minimizing the reordering delay, which benefits from the path selection strategy as described in **Algorithm** 1.

Note that the overhead of maintaining our model is intuitively increasing as the number of paths increases. Through experiments, we observe that the most overhead of maintaining our model is the calculation of $q_p(\Delta T_v)$ which is a recursive process as described in Section V-C. The computation time peaks at 0.08 milliseconds for eight paths and can be reduced to less than 1 microsecond for two paths. Consequently, the computational overhead of MAMS is generally manageable.

VII. Conclusion and Future Works

To cope with mobility impact, in this paper, we present MMQUIC, a novel framework that involves an information exchange module and a new ACK packet structure to keep the sender informed of changes in the wireless uplink/downlink. Based on MMQUIC, a new multipath scheme MAMS is developed. Using a probabilistic model, MAMS can forecast the achievable throughput of the faster subflow in successive time slots, so that an equivalent amount of packets would be pre-allocated to the faster subflow to ensure they are not blocked by packets on the slower subflow. Theoretical analysis backed up by experimental validation shows that the MAMS scheduler effectively mitigates the impact of either the user's movement or network mobility on the OFO issues, achieving substantial performance gains in terms of goodput and throughput compared to the state of the arts in various mobility cases.

We currently focus on the negative impacts of OFO issues among different paths. The expected reordering delay within the same path mainly depends on the severity of packet loss and retransmission types (e.g., RTO, and Fast Retransmission), which are beyond the control of the scheduler policy and not within the scope of this work. How to mitigate the negative impacts caused by OFO arrivals within the same path would be a future research issue. In addition, we plan to study the coordination of congestion control and packet scheduling over multiple paths. Since MMQUIC has the capability of predicting the bottleneck bandwidth, we can take advantage of that knowledge to design a coupled congestion control algorithm for multiple subflows to avoid some issues caused by mobility, e.g., bandwidth under-utilization or overshooting. Currently, we choose the widely deployed congestion control algorithm New Reno to model the window behavior of MMQUIC subflows. Upon the readiness of a new congestion control, the Markov model and estimation model in this paper can be further improved. Lastly, migrating our design from the simulator to a real system is crucial for assessing the design's robustness and performance under real-world conditions.

References

[1] Y. Wang, W. Feng, J. Wang, and T. Q. S. Quek, "Hybrid satellite-UAV-terrestrial networks for 6G ubiquitous coverage: A maritime communications perspective," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 11, pp. 3475–3490, Nov. 2021.

[2] M. Giordani and M. Zorzi, "Non-terrestrial networks in the 6G era: Challenges and opportunities," *IEEE Netw.*, vol. 35, no. 2, pp. 244–251, Mar./Apr. 2020.

[3] S. Kota and G. Giambene, "6G integrated non-terrestrial networks: Emerging technologies and challenges," in *Proc. IEEE Int. Conf. Commun. Workshops (ICC Workshops)*, Jun. 2021, pp. 1–6.

[4] R. Khalili, N. Gast, M. Popovic, and J.-Y. Le Boudec, "MPTCP is not Pareto-optimal: Performance issues and a possible solution," *IEEE/ACM Trans. Netw.*, vol. 21, no. 5, pp. 1651–1665, Oct. 2013.

[5] Q. De Coninck and O. Bonaventure, "Multipath QUIC: Design and evaluation," in *Proc. 13th Int. Conf. Emerg. Netw. Exp. Technol.*, Nov. 2017, pp. 160–166.

[6] J. Pan, R. Jain, S. Paul, and C. So-in, "MILSA: A new evolutionary architecture for scalability, mobility, and multihoming in the future internet," *IEEE J. Sel. Areas Commun.*, vol. 28, no. 8, pp. 1344–1362, Oct. 2010.

[7] V. Ishakian, J. Akinwumi, F. Esposito, and I. Matta, "On supporting mobility and multihoming in recursive internet architectures," *Comput. Commun.*, vol. 35, no. 13, pp. 1561–1573, Jul. 2012.

[8] T. Viernickel, A. Froemmgen, A. Rizk, B. Koldehofe, and R. Steinmetz, "Multipath QUIC: A deployable multipath transport protocol," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2018, pp. 1–7.

[9] S. Ferlin, Ö. Alay, O. Mehani, and R. Boreli, "BLEST: Blocking estimation-based MPTCP scheduler for heterogeneous networks," in *Proc. IFIP Netw. Conf. (IFIP Networking) Workshops*, 2016, pp. 431–439.

[10] G. Sarwar, R. Boreli, E. Lochin, A. Mifdaoui, and G. Smith, "Mitigating receiver's buffer blocking by delay aware packet scheduling in multipath data transfer," in *Proc. 27th Int. Conf. Adv. Inf. Netw. Appl. Workshops*, Mar. 2013, pp. 1119–1124.

[11] F. Yang, Q. Wang, and P. D. Amer, "Out-of-order transmission for in-order arrival scheduling for multipath TCP," in *Proc. 28th Int. Conf. Adv. Inf. Netw. Appl. Workshops*, 2014, pp. 749–752.

[12] K. Xue et al., "DPSAF: Forward prediction based dynamic packet scheduling and adjusting with feedback for multipath TCP in lossy heterogeneous networks," *IEEE Trans. Veh. Technol.*, vol. 67, no. 2, pp. 1521–1534, Feb. 2018.

[13] W. Yang, P. Dong, L. Cai, and W. Tang, "Loss-aware throughput estimation scheduler for multi-path TCP in heterogeneous wireless networks," *IEEE Trans. Wireless Commun.*, vol. 20, no. 5, pp. 3336–3349, May 2021.

[14] K. Chebrolu and R. Rao, "Communication using multiple wireless interfaces," in *Proc. IEEE Wireless Commun. Netw. Conf. Record. WCNC*, vol. 1, 2002, pp. 327–331.

[15] W. Yang, S. Shu, L. Cai, and J. Pan, "MM-QUIC: Mobility-aware multipath QUIC for satellite networks," in *Proc. 17th Int. Conf. Mobility, Sens. Netw. (MSN)*, 2021, pp. 608–615.

[16] W. Yang, L. Cai, S. Shu, and J. Pan, "Scheduler design for mobility-aware multipath QUIC," in *Proc. GLOBECOM IEEE Global Commun. Conf.*, 2022, pp. 2849–2854.

[17] *Quic-ns-3*. Accessed: Oct. 2022. [Online]. Available: https://github.com/signetlabdei/quic-ns-3

[18] M. Simon, E. Kofi, L. Libin, and M. Aitken, "ATSC 3.0 broadcast 5G unicast heterogeneous network converged services starting release 16," *IEEE Trans. Broadcast.*, vol. 66, no. 2, pp. 449–458, Jun. 2020.

[19] B. Jonglez, M. Heusse, and B. Gaujal, "SRPT-ECF: Challenging round-robin for stream-aware multipath scheduling," in *Proc. IFIP Netw. Conf. (Networking)*, Jun. 2020, pp. 719–724.

[20] D. Zhou, W. Song, and M. Shi, "Goodput improvement for multipath TCP by congestion window adaptation in multi-radio devices," in *Proc. IEEE 10th Consum. Commun. Netw. Conf. (CCNC)*, Jan. 2013, pp. 508–514.

[21] S. R. Pokhrel and M. Mandjes, "Improving multipath TCP performance over WiFi and cellular networks: An analytical approach," *IEEE Trans. Mobile Comput.*, vol. 18, no. 11, pp. 2562–2576, Nov. 2019.

[22] C. Lee, J. Jung, and J.-M. Chung, "DEFT: Multipath TCP for high speed low latency communications in 5G networks," *IEEE Trans. Mobile Comput.*, vol. 20, no. 12, pp. 3311–3323, Dec. 2021.

[23] P. Hurtig, K.-J. Grinnemo, A. Brunstrom, S. Ferlin, Ö. Alay, and N. Kuhn, "Low-latency scheduling in MPTCP," *IEEE/ACM Trans. Netw.*, vol. 27, no. 1, pp. 302–315, Feb. 2019.

[24] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose, "Modeling TCP throughput: A simple model and its empirical validation," in *Proc. ACM SIGCOMM Conf. Appl., Technol., Architectures, Protocols Comput. Commun.*, 1998, pp. 303–314.

[25] J. Wu, C. Yuen, B. Cheng, M. Wang, and J. Chen, "Streaming high-quality mobile video with multipath TCP in heterogeneous wireless networks," *IEEE Trans. Mobile Comput.*, vol. 15, no. 9, pp. 2345–2361, Sep. 2016.

[26] Y. Cui, L. Wang, X. Wang, H. Wang, and Y. Wang, "FMTCP: A fountain code-based multipath transmission control protocol," *IEEE/ACM Trans. Netw.*, vol. 23, no. 2, pp. 465–478, Apr. 2015.

[27] M. Li, A. Lukyanenko, and Y. Cui, "Network coding based multipath TCP," in *Proc. IEEE INFOCOM Workshops*, Mar. 2012, pp. 25–30.

[28] M. Karzand and D. J. Leith, "Low delay random linear coding over a stream," in *Proc. 52nd Annu. Allerton Conf. Commun., Control, Comput. (Allerton)*, 2014, pp. 521–528.

[29] V. A. Vu and B. Walker, "On the latency of multipath-QUIC in real-time applications," in *Proc. 16th Int. Conf. Wireless Mobile Comput., Netw. Commun. (WiMob)*, Oct. 2020, pp. 1–7.

[30] A. Nikravesh, Y. Guo, F. Qian, Z. M. Mao, and S. Sen, "An in-depth understanding of multipath TCP on mobile devices: Measurement and system design," in *Proc. 22nd Annu. Int. Conf. Mobile Comput. Netw.*, Oct. 2016, pp. 189–201.

[31] W. Yang et al., "A MPTCP scheduler for web transfer," *Comput., Mater. Continua*, vol. 57, no. 2, pp. 205–222, 2018.

[32] R. Hamilton, J. Iyengar, I. Swett, and A. Wilk. (Jul. 2016). *QUIC: A UDP-Based Secure and Reliable Transport for HTTP/2 Internet Engineering Task Force, Internet-Draft Draft-hamilton-early-deployment-quic-00*. [Online]. Available: https://datatracker.ietf.org/doc/html/draft-hamilton-early-deployment-quic-00

[33] J. Wu, C. Yuen, B. Cheng, Y. Shang, and J. Chen, "Goodput-aware load distribution for real-time traffic over multipath networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 26, no. 8, pp. 2286–2299, Aug. 2015.

[34] D. Halperin, W. Hu, A. Sheth, and D. Wetherall, "Predictable 802.11 packet delivery from wireless channel measurements," in *Proc. ACM SIGCOM Comput. Commun. Rev.*, vol. 40, Oct. 2010, pp. 159–170.

[35] R. A. Shafik, M. S. Rahman, and A. R. Islam, "On the extended relationships among EVM, BER and SNR as performance metrics," in *Proc. Int. Conf. Electr. Comput. Eng.*, Dec. 2006, pp. 408–411.

[36] A. Goldsmith, *Wireless Communications*. Cambridge, U.K.: Cambridge Univ. Press, 2005.

[37] K. Jamieson and H. Balakrishnan, "PPR: Partial packet recovery for wireless networks," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 37, no. 4, pp. 409–420, 2007.

[38] G. Aniba and S. Aissa, "Cross-layer designed adaptive modulation algorithm with packet combining and truncated ARQ over MIMO Nakagami fading channels," *IEEE Trans. Wireless Commun.*, vol. 10, no. 4, pp. 1026–1031, Apr. 2011.

[39] R. Marx, J. Herbots, W. Lamotte, and P. Quax, "Same standards, different decisions: A study of QUIC and HTTP/3 implementation diversity," in *Proc. Workshop Evol., Perform., Interoperability QUIC*, 2020, pp. 14–20.

[40] S. Ferlin, T. Dreibholz, and Ö. Alay, "Multi-path transport over heterogeneous wireless networks: Does it really pay off?" in *Proc. IEEE Global Commun. Conf.*, Dec. 2014, pp. 4807–4813.

[41] S. Deng, R. Netravali, A. Sivaraman, and H. Balakrishnan, "WiFi, LTE, or both?: Measuring multi-homed wireless internet performance," in *Proc. Conf. Internet Meas. Conf.*, Nov. 2014, pp. 181–194.

[42] S. Sundaresan, N. Feamster, and R. Teixeira, "Home network or access link? Locating last-mile downstream throughput bottlenecks," in *Proc. Int. Conf. Passive Act. Netw. Meas.* Heraklion, Greece: Springer, 2016, pp. 111–123.

[43] S. Fu and M. Atiquzzaman, "Performance modeling of SCTP multi-homing," in *Proc. GLOBECOM EEE Global Telecommun. Conf.*, vol. 2, 2005, pp. 6–11.

[44] S. Fortin-Parisi and B. Sericola, "A Markov model of TCP throughput, goodput and slow start," *Perform. Eval.*, vol. 58, nos. 2–3, pp. 89–108, Nov. 2004.

[45] *Ns-3 Simulator*. Accessed: Oct. 2022. [Online]. Available: https://www.nsnam.org

[46] S. Shu, W. Yang, J. Pan, and L. Cai, "A multipath extension to the QUIC module for Ns-3," in *Proc. Workshop Ns-3*, 2023, pp. 86–93.

[47] T. Daengsi and P. Wuttidittachotti, "Quality of service as a baseline for 5G: A recent study of 4G network performance in Thailand," in *Proc. IEEE Int. Conf. Commun., Netw. Satell. (Comnetsat)*, Dec. 2020, pp. 395–399.

[48] X. Yang et al., "Mobile access bandwidth in practice: Measurement, analysis, and implications," in *Proc. ACM SIGCOMM Conf.*, 2022, pp. 114–128.

[49] *DASH-NS3*. Accessed: Apr. 2023. [Online]. Available: https://github.com/haraldott/dash

[50] K. Yang, I. Gondal, B. Qiu, and L. S. Dooley, "Combined SINR based vertical handoff algorithm for next generation heterogeneous wireless networks," in *Proc. IEEE GLOBECOM IEEE Global Telecommun. Conf.*, Nov. 2007, pp. 4483–4487.

[51] V. A. Aalo, C. Mukasa, and G. P. Efthymoglou, "Effect of mobility on the outage and BER performances of digital transmissions over Nakagami-m fading channels," *IEEE Trans. Veh. Technol.*, vol. 65, no. 4, pp. 2715–2721, Apr. 2016.

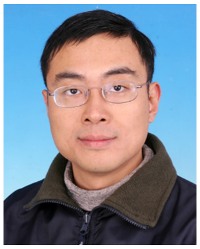
[52] H. Ott, K. Miller, and A. Wolisz, "Simulation framework for HTTP-based adaptive streaming applications," in *Proc. Workshop Ns-3 (WNS3)*, 2017, pp. 95–102.

[53] A. Garcia-Saavedra, M. Karzand, and D. J. Leith, "Low delay random linear coding and scheduling over multiple interfaces," *IEEE Trans. Mobile Comput.*, vol. 16, no. 11, pp. 3100–3114, Nov. 2017.

Wenjun Yang (Graduate Student Member, IEEE) is currently pursuing the Ph.D. degree with the Department of Electrical and Computer Engineering, University of Victoria, BC, Canada. His research interests include multipath QUIC/TCP, video streaming, deep reinforcement learning, and the next generation of network architecture.

Lin Cai (Fellow, IEEE) has been with the Department of Electrical and Computer Engineering, University of Victoria, since 2005. 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 ubiquitous intelligence. She is also a NSERC E. W. R. Steacie Memorial Fellow, a Canadian Academy of Engineering (CAE) Fellow, and an Engineering Institute of Canada (EIC) Fellow. In 2020, she was elected as a member of the Royal Society of Canada's College of New Scholars, Artists and Scientists, and the 2020 "Star in Computer Networking and Communications" by N2Women. She received the NSERC Discovery Accelerator Supplement (DAS) Grants in 2010 and 2015, respectively. She co-founded and chaired the IEEE Victoria Section Vehicular Technology and Communications Joint Societies Chapter. She has been elected to serve the IEEE Vehicular Technology Society (VTS) Board of Governors (2019–2024) and served as the VP Mobile Radio from 2023 to 2024. She served as a Board Member for IEEE Women in Engineering from 2022 to 2024 and a Board Member for IEEE Communications Society (ComSoc) from 2024 to 2026.

Shengjie Shu (Student Member, IEEE) received the M.Sc. degree in computer science from the University of Victoria, Victoria, BC, Canada, in 2023. Her research interests include the next generation of network architecture and related issues, such as multipath QUIC/TCP, AI for networking, and software development.



Jianping Pan (Fellow, IEEE) received the bachelor's and Ph.D. degrees in computer science from Southeast University, Nanjing, Jiangsu, China. He is currently a Professor in computer science with the University of Victoria, BC, Canada. He did his postdoctoral research with the University of Waterloo, ON, Canada. He was also with Fujitsu Laboratories and NTT Laboratories. His area of specialization is computer networks and distributed systems. His current research interests include protocols for advanced networking, performance analysis of networked systems, and applied network security. He received IEICE Best Paper Award in 2009, Telecommunications Advancement Foundation's Telesys Award in 2010, WCSP 2011 Best Paper Award, IEEE Globecom 2011 Best Paper Award, JSPS Invitation Fellowship in 2012, IEEE ICC 2013 Best Paper Award, NSERC DAS Award in 2016, IEEE ICDCS 2021 Best Poster Award, and DND/NSERC DGS Award in 2021. He has been serving on the technical program committees for major computer communications and networking conferences, including IEEE INFOCOM, ICC, Globecom, WCNC, and CCNC. He was the Ad Hoc and Sensor Networking Symposium Co-Chair of IEEE Globecom 2012 and an Associate Editor of IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY. He is a Senior Member of the ACM.

Amir Sepahi (Graduate Student Member, IEEE) received the M.Sc. degree in communication networks from the Isfahan University of Technology in 2019. He is currently pursuing the Ph.D. degree with the Department of Electrical and Computer Engineering, University of Victoria, BC, Canada. His research interests include wireless communications, internet protocols, artificial intelligence, and network security.