QoS-Oriented Multipath Protocol Design in Mobile Networks

by

Wenjun Yang

A Dissertation Submitted in Partial Fulfillment of the
Requirements for the Degree of

DOCTOR OF PHILOSOPHY

in the Department of Electrical and Computer Engineering

QoS-Oriented Multipath Protocol Design in Mobile Networks

by

Wenjun Yang

Supervisory Committee

Dr. Lin Cai, Supervisor
(Department of Electrical and Computer Engineering)

Dr. Amirali Baniasadi, Departmental Member
(Department of Electrical and Computer Engineering)

Dr. Kui Wu, Outside Member
(Department of Computer Science)

ABSTRACT

Emerging applications demand stringent quality of services (QoS). Meanwhile, future networks are featured by ubiquitous mobility. How to meet users' QoS requirements in highly mobile environments remains an open issue, which motivates our research on QoS-oriented multipath transport layer protocol design in mobile networks.

First, multipath transfer is promising in tackling mobility issues for a seamless handoff. Scheduling packets across multiple paths, however, has the issue of out-of-order (OFO) arrival due to the heterogeneity of the paths. In this regard, we put forward a Mobility-Aware Multipath Scheduler (MAMS), ensuring that the reordering delay of each packet is minimized in various mobility scenarios and thus the QoS is significantly improved.

Enabling multipath transfer in the Integrated Terrestrial and LEO Satellite Network (ITSN) is promising. However, the existing multipath congestion control algorithms in ITSN suffer from bandwidth under-utilization or overshooting issues due to the high-speed network movement. Therefore, a novel Mobility-Aware COngestion control (MACO) algorithm is developed.

As applications are the driving force for protocol design, we investigate the performance of video streaming applications using multipath transfer. Assuming the QoS requirements of the application are known by the sender, we adopt a lightweight learning framework, a contextual multi-armed bandit (CMAB), to discover the underlying relationship between dynamic network states and QoS performance, which can intelligently select access networks and adapt FEC coding to trade off delay, reliability, and throughput.

Furthermore, 360-degree videos are not only bandwidth-intensive but also highly sensitive to delays. Ensuring both high video quality and smooth playback experience remains a critical issue. Therefore, we introduce a QoE-oriented Deadline-driven (RIDE) algorithm for multipath scheduling at the frame level. RIDE employs a dependency tree to understand deadlines for different types of frames and considers the negative impacts of Field of View (FoV) changes on scheduling decisions. Utilizing an actor-critic framework to train the neural network enables the scheduler agent to adapt to dynamic environments, including network and FoV dynamics.

Contents

List of Tables

List of Figures

ACKNOWLEDGEMENTS

First, I would like to express my deepest gratitude to my supervisor, Professor Lin Cai, for her continuous support and guidance throughout my PhD program. Prof. Cai is an outstanding and influential scholar in the field of Communication and Networking. She introduced me to various research areas and granted me the freedom to choose topics that aligned with my interests. This led me to focus on intelligent network protocols and multimedia systems. Every time I am stuck with hard problems, she can always give inspiring suggestions to help me out. Whenever I am satisfied with my outcome, she tries to challenge me from a different perspective to motivate me to think deeper and wider. Prof. Cai dedicates much of her life to mentoring students, as evidenced by her weekly individual meetings with each student, despite her numerous other commitments. I could not have imagined having a better PhD supervisor.

I would also like to extend my thanks to my supervisory committee members: Professor Amirali Baniasadi and Professor Kui Wu, and the external examiner Professor Mohamed Hefeeda. Prof. Amirali raised many interesting questions in my candidacy exam, which shaped my final thesis. I audited one of Prof. Kui's courses, that is, computer networks. The theory learned from that course has been and will continue to be integral to my research. Prof. Mohamed's impactful papers on video streaming inspired the work presented in Chapters 4 and 5 of this dissertation. This dissertation would not have been possible without their contributions.

Additionally, I would like to thank Professor Jianping Pan, who has acted as my co-supervisor and beyond. He has supported me in all aspects of my research, life, and career, directing me to the right path. His many wise sentences such as "fail to plan is plan to fail", have served as my motto and subtly influenced my behavior. His constant smile and positive attitude equip me with power and confidence during challenging times.

I am also grateful to my former research advisor, Professor Pingping Dong, for her guidance during my master's studies at Hunan Normal University, China. Her encouragement sparked my enthusiasm for research and motivated me to pursue a PhD. Thanks to her connections, I got a chance to be connected to my current PhD supervisor, Prof. Cai. Without Prof. Dong's favor, none of the wonderful experiences of my PhD journey would have been possible.

My lab fellows are both my close friends and amazing research collaborators. We shared many unforgettable moments, which brought joy to my days and motivated

me to move forward.

I would also like to thank all the staff members in the Department of Electrical and Computer Engineering at the University of Victoria. Their support has been invaluable to my PhD journey.

Last but not least, I would like to thank all my family members including but not limited to my loving and caring wife Li, my little adorable daughter Ellie, my parents, my parents-in-law, and my sister. Because of them, I am full of love, joy, and energy, which drives me to finish my PhD program.

Chapter 1

Introduction

1.1 Background

Emerging applications like holographic communication and digital twins require stringent quality of service (QoS) parameters that surpass the capacity of 5G networks. For example, latency requirements shift from tens of milliseconds to sub-millisecond levels, and throughput escalates to tens of terabits per second. On the other hand, future networks are expected to offer ubiquitous global coverage [15, 136], utilizing a range of access technologies such as satellite networks, unmanned aerial vehicle (UAV) swarms, and terrestrial networks (e.g., LTE, and WiFi). The handoff events occur as frequently as every few seconds, posing challenges for maintaining connection continuity.

In dealing with the more stringent requirements and challenges of mobility in future networks, multipath transport-layer protocols are feasible and desirable. By maintaining multiple connections simultaneously for communication, the aggregated throughput will be improved, and the connections will be more resilient to link failures.

Nevertheless, enabling multipath transmission to provide the user with satisfactory QoS in mobile networks is still challenging, motivating our study in this dissertation.

1.2 Research Issues

In this dissertation, we have studied the following research issues:

1. **Out-of-order (OFO) issues with multipath scheduling:** When scheduling

packets onto multiple paths, the issue of OFO arrivals at the multihomed sink node is prevalent and problematic due to the heterogeneity of the paths [42,119, 131,154,157,164], which is detrimental to user's quality of experience (QoE). In the presence of mobility, wireless link characteristics undergo a fast change over time, packet losses may arise from either the handoff procedures when signal strength is too weak on an access link, or from overflow dropping due to link data rate reduction or congestion [166]. Different types of loss events and dynamic loss rates in a mobile scenario conflict with the assumption of a stationary packet loss process in the existing schedulers. Therefore, the OFO issue is more severe and challenging in mobile networks. It is worthwhile to investigate how to offset the negative impacts of mobility and make more intelligent multipath scheduling decisions so that the application-level throughput (i.e., goodput) can be enhanced.

2. **Multipath congestion control over Integrated Terrestrial and Satellite Networks (ITSN):** The ITSN is characterized by high bandwidth-delay-product (BDP) and high-speed movement. In this case, existing congestion control algorithms would suffer from bandwidth underutilization or overshooting issues due to the unawareness of the fast-changing conditions of ground-to-satellite uplinks and satellite-to-ground downlinks. Under interference such as handovers and atmospheric factors, the congestion signals cannot be accurately identified, resulting in unnecessary congestion window (cwnd) reduction and consequent throughput degradation. Therefore, how to devise the multipath congestion control for the emerging mobile scenario is crucial.

3. **QoS guarantee for video streaming in mobile networks:** Video applications require high reliability in delivering the most significant video data within tight deadlines to avoid playback interruptions. Many uncertain factors cause the randomness of the QoS metrics w.r.t. delay, reliability, and throughput. For instance, the traffic load of each access point is dynamic, the received signal depends on the time-varying distances of the access links, the random congestion at the network side may trigger uncertain queuing delays, etc. To cope with these uncertainties, many existing designs leveraged learning-based approaches to discover the dependencies between the network environments and QoS performance and make intelligent decisions. However, video streaming has stringent delay requirements and a high volume of data, so control decisions

must be made at ms-level. Many learning-based algorithms take a long time to converge in response to the abrupt changes in link conditions, so they are too slow or too costly to meet the stringent QoS requirements. Therefore, how to ensure stringent QoS requirements for multipath video streaming in mobile networks is an open issue.

4. **QoE provisioning for 360 video streaming:** When it comes to video streaming, 360° video streaming has fueled many interests in the research community since it can provide users with an immersive pleasant experience. However, such challenges as vast bandwidth consumption and rebuffering-time sensitivity are encountered. The existing solutions either fail to predict the field-of-view (FoV) accurately or suffer from a long rebuffering time. Therefore, how to give the core video tiles the highest priority to deliver to save bandwidth and reduce rebuffering time inspires us to come up with a new approach to improve the QoE of 360 video users.

1.3 Research Contributions and Impacts

In response to the above research issues, our major contributions and impacts are summarized in the following subsections.

1.3.1 Mobility-Aware Multipath Scheduler for MPQUIC

In Chapter 2, we strive for dealing with the dilemma of **Protocol architecture** and **OFO issue with multipath scheduling**. First, we design a mobility-aware multipath QUIC (MMQUIC) protocol architecture, which utilizes multiple connection IDs (CIDs) to maintain an always-on connection in mobile environments and *enables the transport layer agent to obtain uplink variations* when the sender is in motion. For the case that the mobile user is a receiver, MMQUIC *redesigns the ACK packet structure* so that the sender on the other end can be informed of downlink variation caused by the movement of the receiver or the access network.

On top of MMQUIC, we develop the Mobility-Aware Multipath Scheduler (MAMS) 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 and allocates a certain amount of packets accordingly

to multiple paths, ensuring that the reordering delay of each packet is minimized in various mobility scenarios.

Research impact: To our best knowledge, this work is the *first attempt to comprehensively study the negative impacts of mobility (considering both end-user and network mobility) on multipath scheduling and formulate the reordering delay in mobile environments as a minimization problem.* In this work, we simply tag the packet with its sequence number and ensure the smaller the sequence number the sooner the packet arrives. In reality, *it is feasible to extend our design to more sophisticated cases*, say, tag the packet with its significance, ensuring the more significant the packet the sooner it arrives. Therefore, it would be promising to apply our design to many fields, such as communications in disaster areas where the data has distinguished urgency levels. Along with the theoretical analysis, we have developed an MPQUIC module based on network simulator 3 (ns-3), which has been published on the workshop in ns-3 (wns3). Even though the research on MPQUIC has flourished, to our best knowledge, this is the *first ns-3 based MPQUIC project that is open to the public*, so it will facilitate the development of MPQUIC in 6G network scenarios.

1.3.2 Mobility-aware Multipath QUIC Congestion Control in LEO Network

In Chapter 3, we are motivated to develop a novel Mobility-Aware COngestion control (MACO) algorithm. MACO first leverages the *regularity* of LEO network topology and *high correlation* among multiple paths to forecast the real-time path Bandwidth Delay Product (BDP) on each path and then designs a new *quick start* algorithm, in which the initial congestion window (cwnd) is set to a large yet safe value based on the path BDP, largely shortening the duration of network probing time. Based on estimated BDP, MACO introduces a *new mechanism to detect the congestion event*, avoiding unnecessary cwnd deduction.

Research impact: This work solves three challenges in the Integrated Terrestrial and LEO Satellite Network (ITSN): bandwidth underutilization, inaccurate congestion signal, and unsatisfactory responsiveness. This congestion control design is also applicable to other emerging highly mobile systems such as unmanned aerial vehicle (UAV) networks or other networks where the bottleneck BDP suffers from large fluctuations.

1.3.3 QoS-driven Contextual MAB for Multipath Video Streaming in Mobile Networks

In Chapter 4, we design a QoS-driven contextual MAB (QC-MAB) algorithm for MPQUIC to support video streaming in mobile networks. To improve learning efficiency, QC-MAB identifies the most relevant features that affect QoS requirements and then incorporates them into context space. Each arm of QC-MAB is composed of two actions: access network selection and forward error correction (FEC) configuration. Selecting a reliable access network can effectively mitigate the negative impact of mobility on QoS, and whether a packet-level FEC is enabled or not depends on the reliability and delay requirements, as well as the bottleneck bandwidth, delay, and loss rate of each path. Finally, the reward function of QC-MAB is closely related to the pre-defined QoS requirements. The QC-MAB learning agent performs the Upper Confidence Bound (UCB) algorithm to solve the exploration vs. exploitation dilemma, taking reasonable actions to ensure video service requirements.

Research impact: Fulfilling a deterministic QoS guaranteeing (e.g., never more than 1 μs of packet delay across a network) is generally hard in highly dynamic systems, while we shed light on how to render a statistical QoS guarantee (e.g., guaranteeing less than 0.001% packet overdue) for video streaming in ultra-dense mobile networks.

1.3.4 QoE-oriented 360-Degree Video with Multipath Delivery

In Chapter 5, we further investigate how to improve the QoE performance for real-time 360-degree video streaming. In this work, we propose a QoE-oRIented DEadline-driven (RIDE) 360-degree video streaming with MPQUIC. RIDE solves three key challenges: 1) Interactive streaming has a precise frame deadline, 2) The deadline for different frames is different, and 3) The FoV changes in 360-degree video make the scheduling more complicated. By utilizing an actor-critic framework to train the neural network, RIDE enables the scheduler agent to adapt to dynamic environments, so that achieving high frame quality without causing serious rebuffering issues.

Research impact: Our research on 360-degree video streaming has very broad and promising use cases including immersive gaming, remote health care, online education, etc.

Chapter 2

MAMS: Mobility-Aware Multipath Scheduler for MPQUIC

In this chapter, we investigate the mobility challenges and opportunities. To address mobility challenges, we take advantage of the robustness of multipath delivery and put forward a mobility-aware multipath QUIC (MMQUIC) transport protocol. Packet scheduling and congestion control are two fundamental elements for the MMQUIC, while the existing multipath scheduling and congestion control algorithms are undesirable for futuristic applications with stringent QoS requirements in mobile scenarios. Therefore, in this chapter, we investigate the multipath scheduling design to minimize the negative effects of mobility and maximize user's QoE. The congestion control design for MMQUIC will be studied in Chapter 3.

2.1 Introduction

The next-generation cellular networks are envisioned to provide ubiquitous communication coverage around the world [136]. 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, 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 [54,81]. In this case, multihoming technologies, such as multipath TCP (MPTCP) [77] and multipath QUIC (MPQUIC) [32], have a great potential to support seamless connectivity migration without interruption [66, 107].

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 [42, 119, 131, 154, 157, 164]. Even though the sink node may use a large buffer to store the OFO packets, the OFO arrivals will delay the packet acknowledgment and consequently degrade the sending rate at the sender. 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 [42], DAPS [119], and OTIAS [157], applied the idea of Earliest Delivery Path First (EDPF) [21]. 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 [154] 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 [164] 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 [166]. 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.

By extending our previous work [159] 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 work 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 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 [6] 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.

2.2 Related Work

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) [125] 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 [74]. 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 [166]. 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 [176] 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 [110] 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 [82] 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) [21] 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) [119], OTIAS [157], BLEST [43], and STTF [65] 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 [154] and LATE [164] 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 J. Padhye et al. [105], 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 [143], a rateless Raptor code in FMTCP [28], and a systematic random linear code in SC-MPTCP [84]. 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 [76] to encode the data to reduce the in-order 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.

2.3 System Model

2.3.1 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. 2.1, 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

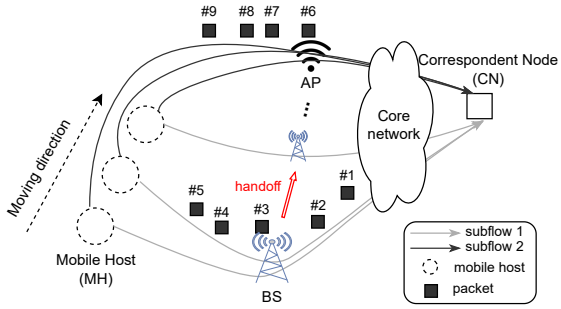


Figure 2.1: The behavior of MPQUIC in mobile networks.

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 chapter are listed in Table 2.1.

2.3.2 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 $\langle \mathcal{P}, \Theta \rangle$.

[1]MPQUIC has two levels of schedulers: stream scheduler and packet scheduler [132]. 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 work, 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.

Table 2.1: 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.

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) [32] and minRTT [100] 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 [42], DAPS [119], and LATE [164] 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 [42, 119, 164], 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., $\max_{p\in\mathcal{P}_v}\{\tau_p\}$. 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 $\Delta T_v = \max_{p,p'\in\mathcal{P}_v}\{\tau_p, \mu_{p'}\}$. Since MPQUIC allows ACKs to return from different paths, 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.

2.3.3 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

$$\delta_k = D_k - a_{k,p}, \tag{2.1}$$

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.,

$$D_k = \max_{p_1,p_2,\ldots,p_k\in\mathcal{P}}\{a_{1,p_1}, a_{2,p_2}, \ldots, a_{k,p_k}\}. \tag{2.2}$$

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

$$a_{k,p} = t_v + \sum_{n=2}^{i} \Delta t_{n,n-1} + OWD_p + \gamma_{k,p}, \tag{2.3}$$

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 packet loss rate which is discussed in Section 2.5.2.

2.3.4 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.,

$$\max_{\mathbf{\Pi}:\langle \mathcal{P},\Theta\rangle} \quad \sum_{v=1}^{V}\sum_{p\in\mathcal{P}_v}\frac{q_p(\Delta T_v)}{\Delta T_v} - \frac{1}{K}\sum_{v=1}^{V}\sum_{p\in\mathcal{P}_v}\sum_{k\in\boldsymbol{\theta_p}}\delta_k \tag{2.4}$$

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 (2.4) 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 [165]. 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 $\tau_{a_1} \le \tau_{a_2} \le \tau_{a_M}$. Given the remaining buffered data $\mathcal{K}_v$ at time t_v, the num-*

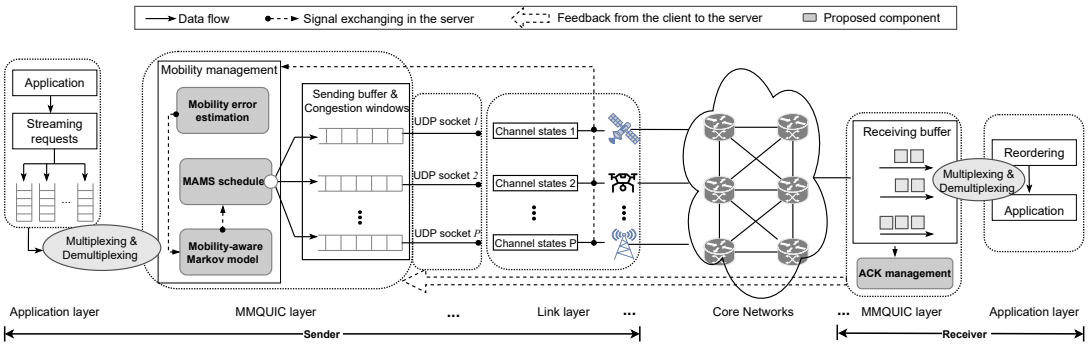


Figure 2.2: Overview of the MMQUIC architecture.

ber 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

$$P = \min\{P^* | \sum_{i=1}^{P^*} q_{m_i}(\Delta T_v) \geq |\mathcal{K}_v|\}, m_i \in \mathcal{M}' \tag{2.5}$$

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

Once $\mathcal{P}$ is determined, the minimization of the second term in (2.4) 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 (2.4) is to find out $q_p(\Delta T_v)$ for each path in mobile environments and then design Θ accordingly.

2.4 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

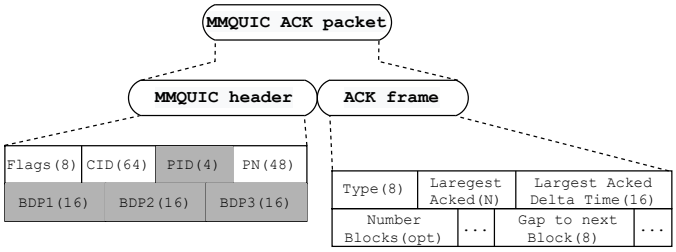


Figure 2.3: MMQUIC ACK packet structure.

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. 5.3. Similar to MPQUIC and MPTCP, MMQUIC also follows the conventional TCP/IP layering architecture. The key contributions of MMQUIC are the components in the dark gray blocks in Fig. 5.3 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.

2.4.1 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. 5.3, 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.

2.4.2 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. 2.3 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 [60], we highlight new fields with dark gray.

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.

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. 2.3, 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.

MMQUIC ACK Frame Structure

The ACK frame structure inherits the design in [60], with E2E path information including the *largest Acked* and *gap to the next block*, where *largest Acked* indicates the largest acknowledged packet sequence number over multiple paths. s

2.5 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.

2.5.1 Mobility Error Rate Estimation

In wireless networks, packet losses are often due to wireless channel fluctuations or path failure but not by link congestion [142]. 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 [59]. 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 [122], the relationship between BER $\bar{\mathbb{P}}_b$ and SNR is

$$\bar{\mathbb{P}}_b = \frac{2\left(1-\frac{1}{L}\right)}{\log_2 L} Q\left[\sqrt{\left[\frac{3\log_2 L}{L^2-1}\right]\left[\frac{2SNR}{\log_2 M}\right]}\right], \tag{2.6}$$

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 [55]

$$Q(x) = \int_x^\infty \frac{1}{\sqrt{2\pi}} e^{\frac{-y^2}{2}} dy \tag{2.7}$$

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

$$\begin{aligned} \boldsymbol{\Lambda_p} &= 1 - \mathbb{P}[\text{at most } \Psi \text{ bits are erroneous}] \\ &= 1 - \sum_{i=0}^{\Psi} \binom{Z}{i} (\bar{\mathbb{P}}_b)^i (1-\bar{\mathbb{P}}_b)^{Z-i}, \end{aligned} \tag{2.8}$$

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

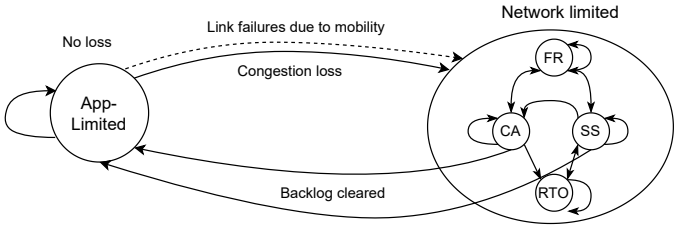


Figure 2.4: State transition diagram of MMQUIC subflow.

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 [11, 59, 71] shed light on the mapping from SNR to BER, which can be referred to.

2.5.2 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 [96], 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. 2.4, in which two main states, application-limited (AL) and network-limited (NL), are presented [2].It is similar to the model in [164], except that the transition from AL to NL is triggered by not only the congestion loss but also transmission failures due to mobility.

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

$$\mathbf{\Omega} = \{\text{Slow Start (SS), Congestion Avoidance (CA),}$$
$$\text{Fast Recovery (FR), Retransmission TimeOuts (RTO)}\}$$

In the presence of packet losses, the time required for a packet to reach the receiver successfully varies depending on the loss event rate. We denote by $w_p(s_r)$ the congestion window over path p given the network state is in s_r ($s_r \in \mathbf{\Omega}$) during the

[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 $\mathbf{\Lambda_p}$. 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).

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

$$T^C_{\langle s_r,p\rangle} = \tau_p/2 + \gamma_p, \tag{2.9}$$

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

$$\gamma_p = \begin{cases} 0, \text{ if } s \in \{\text{CA}, \text{SS}\}, \\ \tau_p, \text{ if } s == \text{FR}, \\ RTO_p, \text{ if } s == \text{RTO}. \end{cases} \tag{2.10}$$

We refer interested readers to [164] for more discussions on (5.6).

2.5.3 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 $\mathbf{\Lambda_p}$.

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 ($x_1 \leq w_p(s_1)$) packets are delivered in the first round. Then we look into the leftover budget $T^B_p(s_2)$ which has three possibilities as below,

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

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 $T^B_p(s_{u|2})$ 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 $T^B_p(s_{u|r})$ ($r \in \{3, 4 \ldots\}$) 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

$$\begin{aligned} q_p(\Delta T_v) &= q_p(T_p^B(s_1)) \\ &= x_1 + \sum_{u=1}^{3} \mathbb{P}\left(s_1, s_{u|2}\right) q_p(T_p^B(s_{u|2})), \end{aligned} \tag{2.11}$$

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

$$\begin{aligned} q_p(T_p^B(s_{u|2})) =& x_{u|2} + \sum_{u'=1}^{3} \mathbb{P}\left(s_{u|2}, s_{u'|3}\right) \cdot \\ & q_p(T_p^B(s_{u'|3})). \end{aligned} \tag{2.12}$$

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

$$\begin{aligned} q_p(T_p^B(s_{u|r})) = x_{u|r} + \sum_{u_2=1}^{3} \mathbb{P}\left(s_{u|r}, s_{u'|(r+1)}\right) \cdot \\ q_p(T_p^B(s_{u'|(r+1)})), \forall r \in \{2, 3, \ldots\}, \end{aligned} \tag{2.13}$$

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.,

$$q_p(\Delta T_v) = \sum_{r=1}^{r_{\max}} \mathbb{E}\left[x_r\right], \tag{2.14}$$

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 [34, 44, 129]. 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

$$C_p = \eta \cdot W_p \log_2(1 + SINR_p), \tag{2.15}$$

where W_p is the allocated channel bandwidth to interface p of the MH, $SINR_p$ in (3.2) 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.

$$\mathrm{SINR_p} = \frac{Y_p(R_i)^{-\beta}}{\sigma^2 + I_p}, \tag{2.16}$$

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

$$\Phi_p(s_r) = \min\{w_p(s_r), C_p(t)\tau_p(t)\}. \tag{2.17}$$

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 [48, 50, 106], the value of $\mathbb{P}(x_1|\Phi_p(s_1))$ obeys the Binomial formula, i.e.,

$$\mathbb{P}(x_1|\Phi_p(s_1)) = \binom{\Phi_p(s_1)}{\Phi_p(s_1) - x_1} \cdot \boldsymbol{\Lambda_p}^{\Phi_p(s_1)-x_1} \cdot (1 - \boldsymbol{\Lambda_p})^{x_1}. \tag{2.18}$$

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: $\mathcal{P}_v$.

$Con\_\mathrm{I} \Leftarrow$ the sender is moving // True or False
$Con\_\mathrm{II} \Leftarrow$ the receiver is moving
$\mathcal{M}' = \{m_1, m_2, \ldots, m_M\}$ //Sort all paths $m \in \mathcal{M}$ in ascending order by τ_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. (2.8)
      $\Phi_j(s_r) := f_1(BDP^{\langle u\rangle}(t), BDP^{\langle d\rangle}(t), w_j(s_r))$ // Eq. (2.17)
      if $r == 1$ **then**
        $x_r := f_2(\Phi_j(s_r), \boldsymbol{\Lambda_j})$ // Eq. (2.19)
      else
        for each $u \in \mathcal{U} = \{1, 2, 3\}$ **do**
          $x_{u|r} := f_2(\Phi_j(s_r), \boldsymbol{\Lambda_j})$ // Eq. (2.19)
        $x_r := f_3(x_{u|r})$ // Eq. (5.3)
      $q_j(T_j^B(s_r)) \leftarrow q_j(T_j^B(s_r)) + x_r$
      $r \leftarrow r + 1$
      $T_j^B(s_r) \leftarrow T_j^B(s_r) - (\gamma_{j,s_r})^+_{\tau_j}$ // Eq. (5.6)
      t is updated with Eq. (2.3)
    $Q_P += q_j(T_j^B(s_r))$
  $P = i$
  if $Q_P \geq |\mathcal{K}_v|$ **then**
    break
$\mathcal{P}_v := \{m_1, \ldots, m_P\}$
return $\mathcal{P}_v$

Hence, x_1 can be obtained with

$$\mathbb{E}[x_1] = \sum_{x_1=0}^{\Phi_p(s_1)} x_1 \cdot \mathbb{P}(x_1 | \Phi_p(s_1)). \tag{2.19}$$

Table 2.2: The settings of a and b with different sub-states

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

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

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

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 [164]. Additionally, $\Phi_p(s_{u|r})$ should be updated as follows and incorporated into the set $\mathcal{I}(s_{u|r})$.

$$\Phi_p(s_{u|r}) = \min\{w_p(s_{u|r}), C_p(t')\tau_p(t')\}, \tag{2.20}$$

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 [48, 50, 106, 164]

$$\begin{aligned}\mathbb{P}\left(s_{u|(r-1)}, s_{u'|r}\right) = &\sum_{n=a}^{b} \binom{\Phi_p(s_{u|(r-1)})}{n} \cdot (\boldsymbol{\Lambda_p})^n \cdot \\ &(1 - \boldsymbol{\Lambda_p})^{\Phi_p(s_{u|(r-1)})-n}, u, u' \in \{1, 2, 3\}\end{aligned} \tag{2.21}$$

in which the values of a and b for different states are summarized in Table 2.2.

With the knowledge of $\Phi_p(s_{u|r})$, along with (2.13) and (2.21), the expected value

of x_r can be derived as follows,

$$\begin{aligned}\mathbb{E}\left[x_r\right] = &\sum_{u_1=1}^{3}\sum_{u_2=1}^{3}\cdots\sum_{u_{r-1}=1}^{3}\mathbb{P}\left(s_1, s_{u_1|2}\right)\mathbb{P}\left(s_{u_1|2}, s_{u_2|3}\right)\cdots \\ &\mathbb{P}\left(s_{u_{r-2}|(r-1)}, s_{u_{r-1}|r}\right)\mathbb{E}\left[x_{u_{r-1}|r}|\Phi_p(s_{u_{r-1}|r})\right] \\ =&\sum_{u_1=1}^{3}\sum_{u_2=1}^{3}\cdots\sum_{u_{r-1}=1}^{3}\mathbb{P}\left(s_1, s_{u_1|2}\right)\cdot\prod_{v=2}^{r-1}\mathbb{P}\left(s_{u_{v-1}|v}, s_{u_v|(v+1)}\right) \\ &\cdot\sum_{x_{u_{r-1}|r}=0}^{\Phi_p(s_{u_{r-1}|r})}\mathbb{P}\left[x_{u_{r-1}|r}|\Phi_p(s_{u_{r-1}|r})\right]\cdot x_{u_{r-1}|r}.\end{aligned} \tag{2.22}$$

Using (2.19) and (5.3), (2.14) is rewritten as

$$q_p(\Delta T_v) = \mathbb{E}\left[x_1\right] + \sum_{r=2}^{r_{\max}}\mathbb{E}\left[x_r\right]. \tag{2.23}$$

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.

2.5.4 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

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

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. (2.17). This is achieved using a while loop, as shown in line 6 of **Algorithm** 2.

2.5.5 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. 2.17, 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.

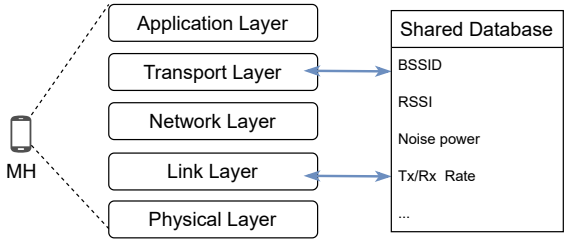


Figure 2.5: Cross-layer architectural blueprint.

2.6 Performance Evaluation

We use ns-3 [4] 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 [164] 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.

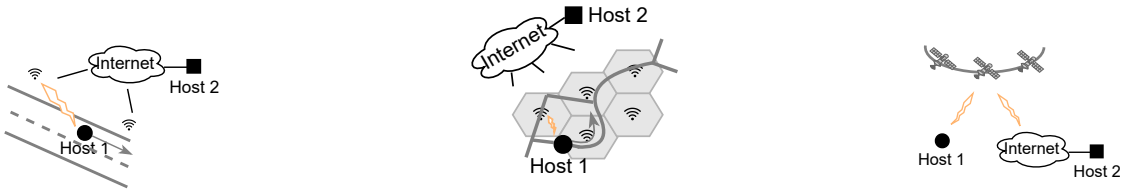
2.6.1 Prototype Implementation

To begin with, we extended the existing QUIC module [6] 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 [124] and the public release source code at https://github.com/ssjShirley/mpquic for the implementation details of MPQUIC-ns3.

Further, as illustrated in Fig. 3.7, we introduce a shared database to enable the cross-layer MMQUIC framework.

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

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



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

Figure 2.6: Mobile scenarios.

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].

2.6.2 Experimental Settings

Fig. 2.6 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 [29], 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 [167]. In simulations, each AP/BS is in the center of a hexagon and they are 200 meters away. The link bandwidth

[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`.

Table 2.3: 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

of LTE and WiFi is set to 50 Mbps and 200 Mbps, aligning with the measurements in [167].

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 2.3 [166].

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. 2.6, 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 [1].

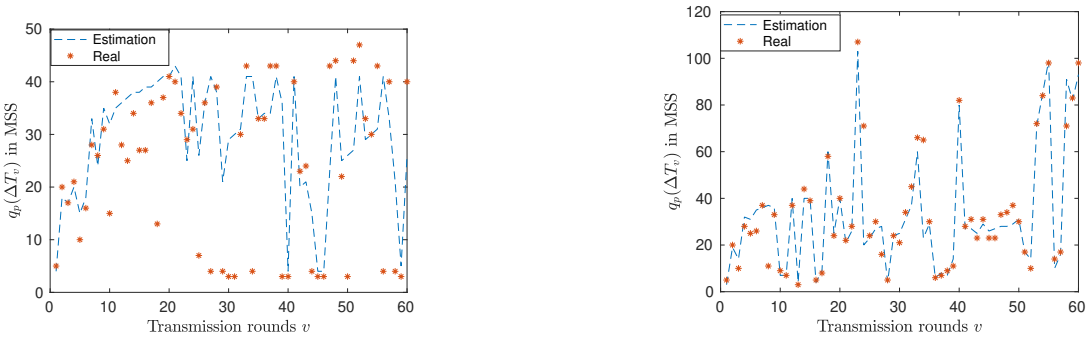
For both MAMS and benchmarks, two main performance metrics are measured: goodput and the cumulative distribution function (CDF) of per-packet delivery delay. Specifically, the goodput is calculated by averaging the bytes of received in-order packets over the time duration once the receiver receives new 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.

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.

2.6.3 Model Validation

As the estimation on $q_p(\Delta T_v)$ in Section 2.5.3 is a key component of MAMS, the estimation model accuracy is critical to MAMS's performance. Fig. 2.7 and 2.8 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 [159] here for comparison. To avoid confusion, we rename the first version of MMQUIC as MMQUIC-v1. Note that Fig. 2.7 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. 2.8.

Fig. 2.7(a) shows that there is a big gap between the estimation and real values at some points when running LATE. In Fig. 2.7(b), the estimation and real values are very close, so MAMS demonstrates a desirable model accuracy in the mobile uplink case.



(a) LATE (b) MAMS

Figure 2.7: The estimation model accuracy validation in the mobile uplink case.

Regardless of mobile uplink or downlink cases, LATE is unaware of the link condition changes proactively, so Fig. 2.8(a) exhibits the same trend as Fig. 2.7(a), showing the mismatch between the estimation and real values. By analyzing Fig. 2.8(b) and 2.8(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.

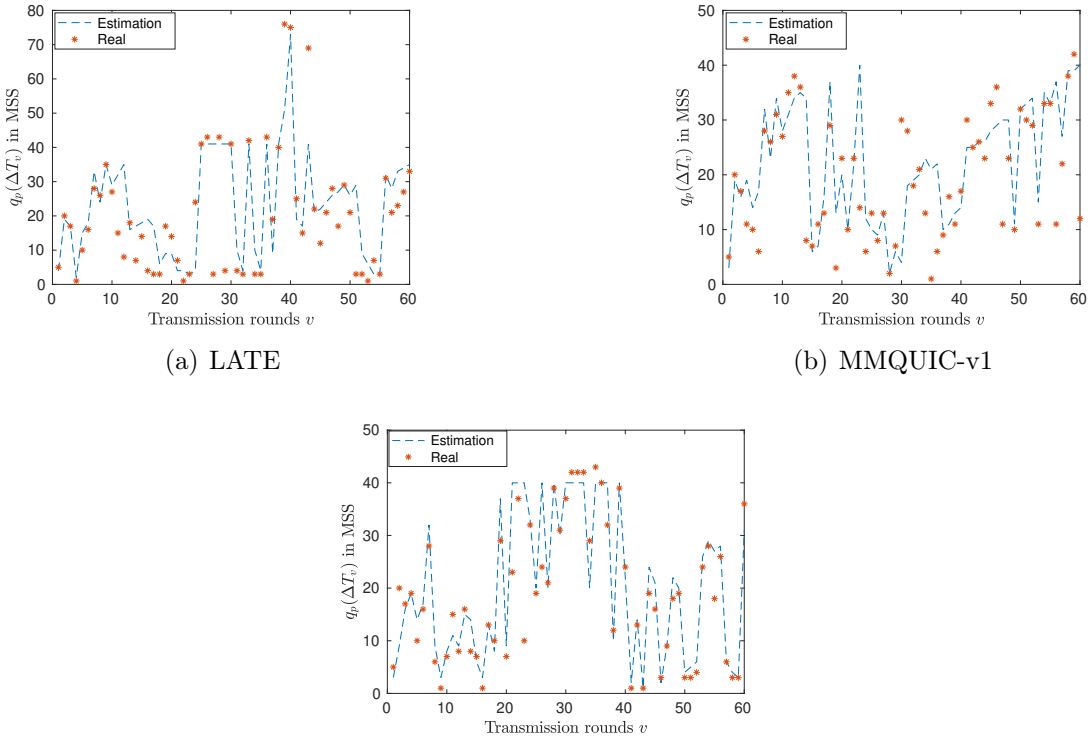
In summary, MAMS can accurately calculate the path throughput in various mobile scenarios.

2.6.4 Results and Analysis

Mobile Uplinks

The mobile uplinks refer to the case where the sender is moving. Assuming the sender (i.e., Host 1 in Fig. 2.6(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 [158], we set σ to -90 dBm in our experiments.

Fig. 2.9 depicts our setting and final results with respect to the goodput of three algorithms: MPQUIC-RR, MAMS, and LATE. As shown in Fig. 2.9, 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 $\boldsymbol{\Lambda_p}$



(a) LATE

(b) MMQUIC-v1

(c) MAMS

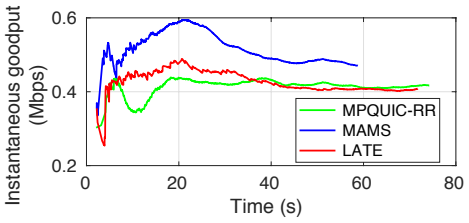
Figure 2.8: The estimation model accuracy validation in the mobile downlink case.

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 $\mathbf{\Lambda_p}$, the Nakagami fading parameter m_s and the path-loss exponent α are set to 1 and 2 [7], 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. 2.9(b).

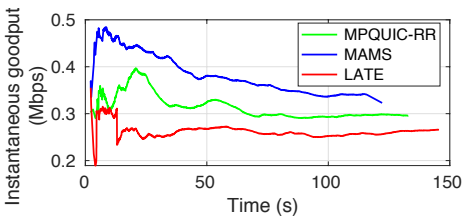
MAMS makes 18% and 44% improvement against LATE and MPQUIC-RR in the 90-th percentile per-packet delivery delay, respectively as shown in Fig. 2.10(a), and 50% and 35% improvement compared to LATE and MPQUIC-RR in the 99-th percentile per-packet delivery delay, respectively, when the speed is up to 40 m/s as shown in Fig. 2.10(b).

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

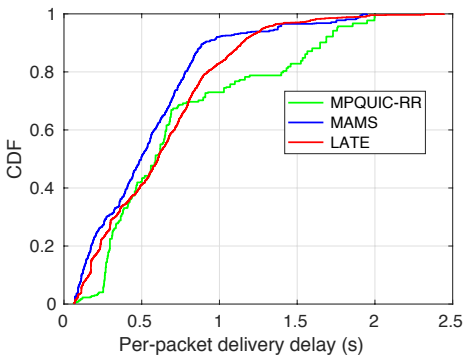


(a) MH moves at 20 m/s

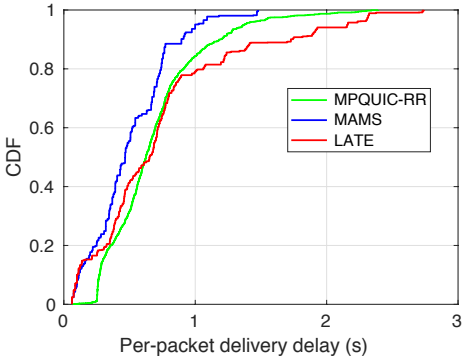


(b) MH moves at 40 m/s

Figure 2.9: The instantaneous goodput over time observed at the receiver when the sender is in motion.



(a) MH moves at 20 m/s



(b) MH moves at 40 m/s

Figure 2.10: Associated CDF of per-packet delivery time with different moving speeds.

and efficiency of MAMS in this case, we conduct a series of experiments in both `Highway` and `Urban` scenarios with different settings w.r.t. bandwidth, mobility pattern, speed, etc.

Fig. 2.11 and Fig. 2.12 depict the performance of goodput and the delay CDF in the `Highway` scenario.

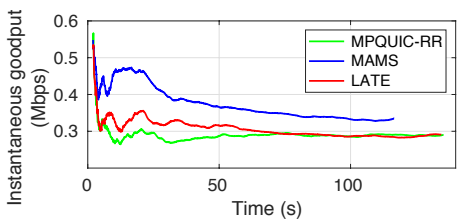
In Fig. 2.11(a) where the receiver is moving at 20 m/s, the mobility prediction model in LATE offers some goodput improvement against MPQUIC-RR within the first 70 s, while the advantage is diminished afterward, because LATE has no component to get the loss rate and bottleneck capacity updated, and the 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. 2.11(b) further presents the situations when MH moves faster.

Compared to Fig. 2.11(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. 2.11 get lower than that of Fig. 2.9 where MAMS can obtain the uplink variations timely.

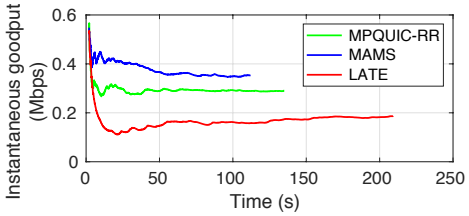
Fig. 2.12 describes the per-packet delay distribution which reveals the reordering delay of each packet. Comparing Fig. 2.12(a) with Fig. 2.12(b), one obvious fact is that the per-packet delay with the LATE algorithm is largely increased as the MH speeds up to a certain level. Numerically, 90% of packets are consumed by upper applications within less than 1.603 s when using the LATE scheduler in Fig. 2.12(a), and 90% of packets take less than 2.053 s to be consumed in Fig. 2.12(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. 2.13 shows the performance of average goodput and reordering delay with varied mean values of the speed.

In Fig. 2.13(a), three key conclusions can be drawn. Firstly, the increment of speed contributes to an enhanced goodput across all algorithms when the speed re-



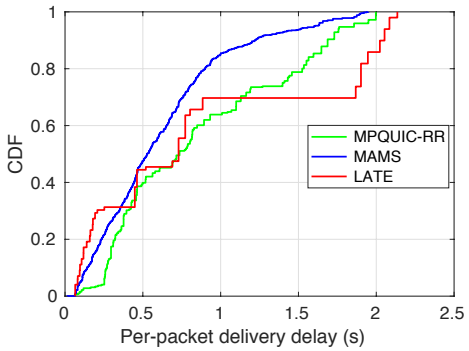
(a) MH moves at 20 m/s



(b) MH moves at 40 m/s

Figure 2.11: 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

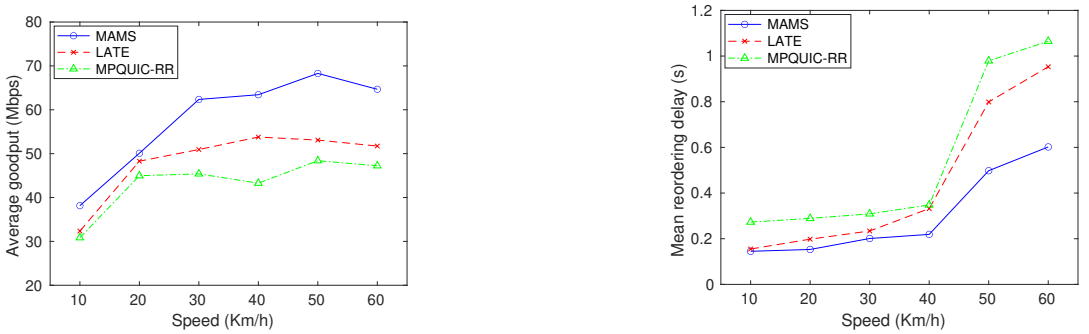


(b) MH moves at 40 m/s

Figure 2.12: Associated CDF of per-packet delivery time with different moving speeds.

mains 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. 2.13(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% improve-



(a) The goodput vs speed

(b) The reordering delay vs speed

Figure 2.13: The impact of varied moving speed in the `Urban` scenario.

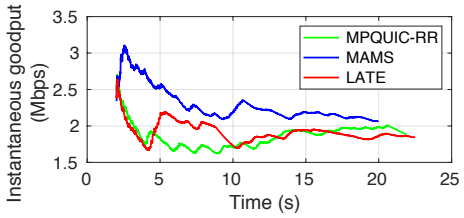
ment in terms of delay performance.

Network Mobility

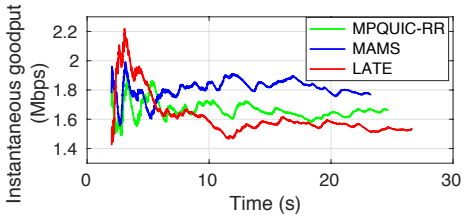
Last but not least, we investigate the case where both uplinks and downlinks undergo dynamic changes. Here the experiments are running over the `ITSN` scenario as shown in Fig. 2.6(c).

Under the impact of network mobility, the goodput achieved by the three algorithms is given in Fig. 2.14. 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. 2.14(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. 2.15(a) where MAMS has the lowest 90-th percentile per-packet delay which is 0.28

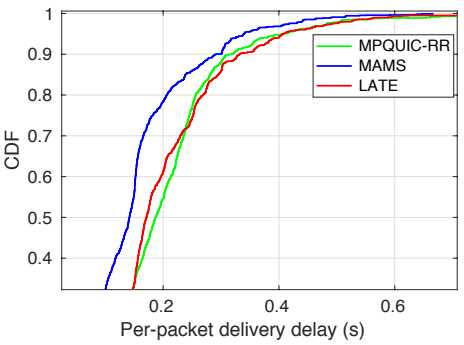


(a) Satellites move at 7 Km/s

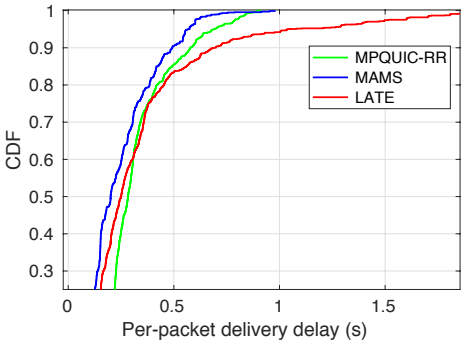


(b) Satellites move at 7.5 Km/s

Figure 2.14: 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

Figure 2.15: Associated CDF of per-packet delivery time with different moving speeds.

s, and the counterpart of MPQUIC-RR and LATE is 0.31 s and 0.34 s respectively. From Fig. 2.14(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. 2.15(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.

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

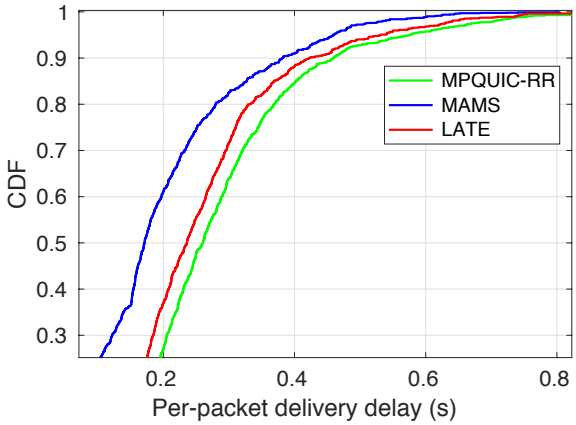


Figure 2.16: Per-packet delivery delay distribution when a video client requests the video data as encoded in [103].

of packets in video streaming is useless and regarded as packet loss [143], the per-packet delivery delay distribution is still used as a performance criterion.

With respect to mobility patterns, we employ the case of mobile downlinks. The network settings are identical to the settings in Section 2.6.4. 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 [103] 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. 2.16, 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.

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 2.2, SEDPF [52] is the relevant one that employs FEC to address the multipath OFO

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

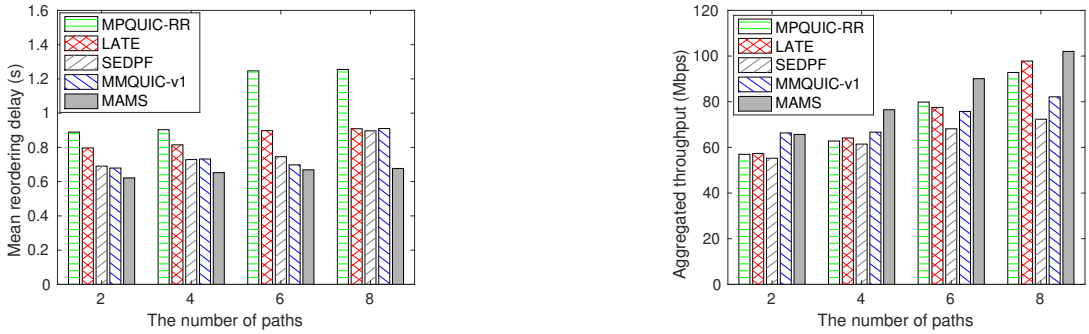
Fig. 2.17 shows the mean reordering delay and aggregated throughput performance when the MH fetches a 10 MB file in the `Urban` scenario. Provided that the access networks are diversely consisting of WiFi, LTE, and satellite technology, the MH can choose to connect to multiple networks to establish multiple paths between the MH and the server.

First of all, according to the results given in Fig. 2.17(a) and 2.17(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. 2.17(a) demonstrates that MAMS reduces the mean reordering delay by up to 0.233 seconds. Meanwhile, as shown in Fig. 2.17(b), MAMS achieves up to 24.2% throughput improvement.

As shown in Fig. 2.17(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. 2.17(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



(a) The impact on reordering delay

(b) The impact on throughput

Figure 2.17: The impact of a different number of paths.

of maintaining our model is the calculation of $q_p(\Delta T_v)$ which is a recursive process as described in Section 2.5.3. 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.

2.7 Summary

To cope with mobility impact, in this work, 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.

Chapter 3

MACO: Mobility-Aware Congestion Control for MMQUIC in Satellite Networks

In Chapter 2, we demonstrate the MMQUIC framework along with the multipath scheduling policy to mitigate the OFO issue. In this chapter, we aim to improve the congestion control performance of MMQUIC in satellite networks.

3.1 Introduction

Next-generation wireless networks are increasingly embracing the Integrated Terrestrial and Low Earth Orbit (LEO) Satellite Networks (ITSN) for ubiquitous coverage [23, 73, 89, 175], which is of paramount importance in many use cases such as disaster rescue and communications in remote areas. Walker-type constellation in Fig. 3.1 has been adopted in LEO networks, e.g., the Starlink developed by SpaceX, which comprises satellites that are uniformly distributed in orbits with the same orbit inclination and altitude.

One typical challenge in ITSN is raised by the satellite movement [30]. LEO satellites move at a high speed (i.e., 5 to 10 km/s), so the handover between users and access satellites will be as frequent as every few seconds, leading to severe connection interruptions.

According to [66, 107], multipath transport protocols, such as multipath TCP (MPTCP) [47] and multipath QUIC (MPQUIC) [32] have a great potential to support

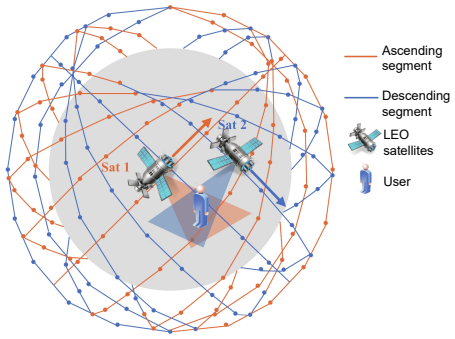


Figure 3.1: Walker-type satellite constellation [87].

seamless connectivity migration during handover. In the context of ITSN, multiple satellites will be visible to a satellite dish, making it feasible to establish multiple connections and apply multipath transfer in ITSN.

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 consumes 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 [166]. Thus even if the address and/or port are changed due to mobility, the connection remains active.

However, the existing congestion control algorithms for MPQUIC in ITSN are undesirable. They can be classified into four categories: loss-based [78,91,137], delay-based [12,18], bandwidth-delay product (BDP) based [20], and learning-based [37,72].

In ITSN, packet loss can occur due to various factors such as handover loss or transmission error caused by atmospheric conditions, while high delay variations can result from routing path detours when certain satellites become unreachable during satellite movement. Therefore, relying solely on packet loss or delay signals is insufficient to accurately identify congestion levels in LEO networks. BDP-based solutions state that the BDP serves as a critical indicator of congestion levels, so obtaining BDP information is a fundamental step. However, they necessitate at least one RTT to probe network conditions to figure out the BDP information. As ITSN has a relatively long propagation delay, using one RTT to do network probing is expensive. Learning-based solutions also suffer from low efficiency at the initial stage because they cannot obtain sufficient data to predict network conditions at the beginning.

Motivated by the above critical issues, our goal is to leverage mobility infor-

mation in congestion control and to better trade off performance metrics including throughput, responsiveness, and TCP friendliness. To achieve this, we present a novel Mobility-Aware COngestion control (MACO) algorithm for MPQUIC in ITSN. In addition to the preliminary results presented in [166], this study aims to tackle more complex scenarios when the bottleneck arises in various locations, including ground-to-satellite access links, inter-satellite links and core networks. The main contributions and novelties of this chapter are three-fold.

First, we propose an efficient BDP estimation approach in the ITSN mobile scenario instead of solely relying on lengthy network probing.

1. To derive the bottleneck bandwidth, we take advantage of the regularity of satellite motion to predict the access link condition changes (Section 3.4.1), and also employ the Adaptive Kalman Filter (AKF) to swiftly offset the estimation errors in cases where the bottleneck arises from non-satellite networks (Section 3.4.1).

2. In the process of delay estimation, we leverage the knowledge of the changing distance between ground users and satellites and statistical analysis of the delay distribution based on measurements spanning one year over Starlink satellites (Section 3.4.1).

Second, we present the MACO algorithm which has two components: BDP-inspired quick start and mobility-aware congestion avoidance.

1. In the quick start phase, MACO sets the initial congestion window (cwnd) to a large yet safe value based on the estimated path BDP as well as learning from history settings across different subflows (Section 3.4.2).

2. In the congestion avoidance design, MACO utilizes a multipath fluid model and the square root function to trade off performance metrics including throughput, convergence, and TCP friendliness. This approach ensures aggressive cwnd growth when the current cwnd is far below the estimated BDP and conservative growth when approaching it, thereby effectively exploring network capacity (Section 3.5.2).

Finally, based on an MPQUIC prototype we developed [124], we implement the MACO and benchmark algorithms. Extensive evaluation results demonstrate that:

1. MACO is more adaptive to dynamics such that its cwnd can reach up to the peak BDP, that is, 300 MSS (maximum segment size), which is 20% and 200% higher than that of BBR [19] and OLIA [78], respectively.

2. Compared to benchmarks, MACO improves the throughput by three times without causing a longer queuing delay.

3. While ensuring TCP friendliness, MACO takes 70.67% less time to recover to the network capacity when packet loss occurs compared to OLIA.

3.2 Related Work

In this section, recent studies on ITSN networks are reviewed, including existing mobility management, MPQUIC-based solutions, and congestion control mechanisms.

3.2.1 Mobility Management

To overcome the mobility issue that stems from the conventional TCP/IP stack in terrestrial networks, new architecture design paradigms have been suggested by the research community. They can be categorized into two schools: 1) clean-slate architectures, e.g., MobilityFirst [116], Named Data Networking (NDN) [170], RINA [31], Content-Centric Networking (CCN) [70], and Data-Oriented Network Architecture (DONA) [80]; 2) evolutionary architectures, e.g., Locator/ID separation protocol (LISP) [39], New IP [85], and Trotsky [97].

The above solutions share a common design rationale that a cleaner separation of identity and location is instrumental to enhancing endpoint mobility. However, the network sides (i.e., satellites) rather than endpoints dominate the mobility and handover issues in the ITSN scenario. As a result, satellite-mobility-driven approaches, such as NDM [30], MSN [173], and QIH [152] are newly developed. They focus on the handover prediction by considering the movement of both user terminals and satellites, followed by a handover-aware protocol design. However, they establish a single TCP connection over terrestrial and space networks, leading to two problems. First, the single connection runs in a break-before-make fashion to deal with the handover, referred to as hard handover, it has the momentary connection breakage issue [134]. Second, the TCP connection has to be re-established with new parameters (another three-way handshake) during a session if the client changes its IP address, which is

prohibitive in ITSN cases where the RTT is generally far higher than the counterpart in terrestrial networks.

3.2.2 Multipath-enabled Transport Protocols

Given the frequent handover in mobile networks, multipath-enabled transport protocol design is more promising. There have been several multipath transport protocols such as DCCP [79], CMT-SCTP [68], MPTCP [47], and MPQUIC [32].

DCCP was mainly invented for mobility support because it is useful for connection migration thanks to the multihoming feature. However, it does not support concurrent multipath transmission, thereby motivating the development of CMT-SCTP. CMT-SCTP has largely improved the throughput by using multiple paths to transmit data simultaneously. However, CMT-SCTP is an extension of SCTP, which has limited deployment compared to more widely adopted protocols such as TCP and UDP. Thus, researchers shift their interests to MPTCP and MPQUIC.

MPQUIC, the latest proposal, has inherited the advantages of both CMT-SCTP and MPTCP, demonstrating great potential in various scenarios including ITSN [123, 146, 166]. First of all, MPQUIC is robust to the connection migration caused by mobility because each QUIC connection is identified with a Connection ID (CID) which can be associated with multiple IP addresses. In addition, different from MPTCP which needs the three-way handshake to establish or restore a connection, MPQUIC takes only 1 RTT to initialize a subflow, or 0 RTT to restore it. This feature is preferable in mobile scenarios. MPQUIC also maintains a global view about the status of each path through a 'PATHS' frame, e.g., whether the path is active, underperforming, or broken. These statistics can be used to speed up the handover process in mobility scenarios [32].

Recent works, such as OLAPS [151], GADaM [26], HBES [149], and MPDTP [123], have explored the usage of multipath transport protocols in mobile scenarios. OLAPS and GADaM utilize learning models to gain a better understanding of network dynamics in mobile scenarios, and then design an optimal path scheduling policy. HBES mainly aims to mitigate the HoL Blocking issues in the presence of mobility, thus improving the quality of experience (QoE) perceived by the receiver. MPDTP focuses on the deadline guarantee by taking into account challenges in the ITSN scenario, such as high loss rate, limited bandwidth, and long round trip time (RTT).

3.2.3 Congestion Control Algorithms

Satellite networks are featured by high bandwidth and long delays. For the traditional TCP/IP stack, at the beginning of a new connection, the sender executes a slow start to probe the usable bandwidth along the path [69], taking a few RTTs, which is not desirable for small file transfer. Authors in [38, 86] argued that the initial window should be set to a larger value against the original value which is typically 1 MSS.

TCP uses a congestion window to explore available bandwidth, which is adjusted in an additive increasing and multiplicative decreasing (AIMD) fashion [17]. The AIMD can ensure a reasonably fair share of bandwidth but can be slow in mobile environments where the channel quality changes fast due to fading, shadowing, and handover [16].

The latest efforts at congestion avoidance can be generally divided into two directions: reactive and proactive congestion control. DCQCN [178], TIMELY [98], and HPCC [86] are the reactive ones, which investigated novel congestion signals to detect congestion. The proactive approaches, e.g., ExpressPass [25], NDP [113], and Homa [99] integrated priority scheduling policies with congestion control to schedule packets for the purpose of congestion-free, but they are unable to get the required information and take action in the first RTT.

In 2016, BBR [19] was proposed as model-based congestion control, which skips the concept of cwnd. It measures bottleneck bandwidth and RTT to control the sending rate directly. It appears to be a great improvement especially when it works with QUIC [67]. However, BBR with multipath support is not yet available as the coupled design of the pacing gain rate for multiple paths remains an open issue. Therefore, MPQUIC has opted for OLIA [78] as the standard congestion control scheme which is loss-based. However, OLIA cannot translate the packet loss accurately in ITSN and incurs performance degradation.

Therefore, in this chapter, we propose the MPQUIC-supported ITSN architecture where MPQUIC deals with smooth handover and fast connection establishment. In addition, we emphasize the importance of collaboration between the end-system and network side, enabling the MPQUIC to be aware of mobility events, and thus it can make a wise and effective decision on congestion control to adapt to the dynamic link capacity.

Table 3.1: The main notations and definitions

Notation	Definition
H_1, H_2	A pair of the sender and the receiver.
G_1–G_3	LEO satellites.
E_1	Ground station.
d_1	Distance traveled by H_1 across the HO area.
d_2	Distance between two adjacent HO areas.
R_1	Distance between two adjacent satellites.
R_2	Radius of the satellite footprint.
w_r	The cwnd of path r.
τ_r	The RTT of path r.
x_r	Transmission rate over path r.
$\Omega_r(t)$	Initial value of cwnd.
$\mathcal{B}_r(t)$	Instantaneous path BDP.
$\overline{\mathcal{B}_r}(t)$	Averaged path BDP over an interval τ_r.
$\mathcal{C}_{r,t}$	Bottleneck bandwidth over path r at time t.
$\mathcal{C}^A_{r,t}$	Bandwidth of ground-satellite links.
$\mathcal{C}^N_{r,t}$	Bandwidth of core networks.
$\mathcal{D}_{r,t}$	Round-trip propagation delay.
$\mathcal{W}_r$	Set of all cwnd when congestion occurs.

3.3 System Model and Problem Statement

In this section, we first model the behavior of MPQUIC subflows under the impact of LEO satellite movements and then identify the problems when applying existing congestion control algorithms to MPQUIC in the ITSN scenario.

3.3.1 System Model

We consider the Walker-type satellite constellation. As of now, most satellite networks operate on a bent pipe principle, which means satellites are used to relay information. Data is transmitted to the satellite, which sends it right back down to the nearest ground station [108, 138], so we focus on this scenario as shown in Fig. 3.2. The main notations used throughout this chapter are listed in Table 3.1.

Fig. 3.2 depicts the ITSN system, consisting of the MPQUIC-capable end hosts H_1 and H_2, the LEO satellites G_1–G_3 in the same orbit plane, and ground station E_1. Assuming all satellites move from south to north at velocity v over time, and H_1 is directly connected to G_1, G_2, G_3, etc. The distance between two adjacent satellites

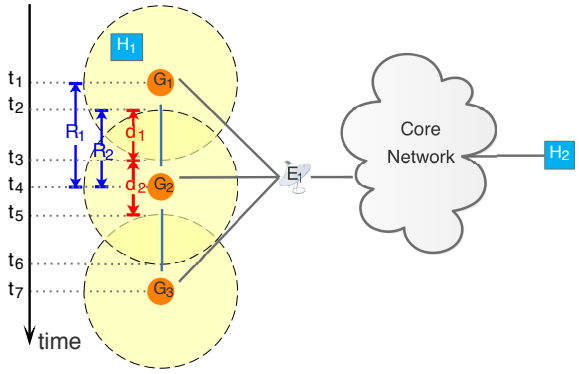


Figure 3.2: MPQUIC-supported ITSN network architecture.

in the same orbit is denoted by R_1, while R_2 represents the radius of the satellite footprint.

At time t_1, H_1 is in the footprint of satellite G_1 by which the first MPQUIC subflow denoted by F_0 is created. In the presence of satellite movement, H_1 enters into the handover (HO) area after a while, in which the HO area refers to the area where the footprint of two adjacent satellites overlaps. Within the HO area, H_1 can create a new subflow F_1 by attaching its second interface to the reachable satellite. Denote by d_1 the distance traveled from the point at which $H1$ enters the HO area to the point at which it exits. When H_1 moves out of the HO area, we assume that the northernmost satellite becomes unreachable and the associated connection has to be migrated to a new coming satellite. For example, at the time t_3 when H_1 approaches to edge of the footprint of G_1, we assume that the received signal strength (RSS) of the interface associated with F_0 is below a threshold, which triggers service outage and MPQUIC immediately migrates the connection of F_0 seamlessly by attaching to the available satellite G_2 while maintaining the connection continuity of F_1. As time goes by, H_1 will re-enter and leave the new HO area periodically. d_2 is the distance between the current HO area and the subsequent one.

3.3.2 Problem Statement

To investigate the problem, we first give the discrete-time topology states as shown in Fig. 3.3. The network topology undergoes periodic changes. Except for state s_1 where only one subflow is established at the beginning, two subflows between H_1 and H_2 are created in subsequent states. In state s_2, as H_1 enters the HO area, the two subflows connect to two different satellites. In state s_3, when H_1 moves to an area

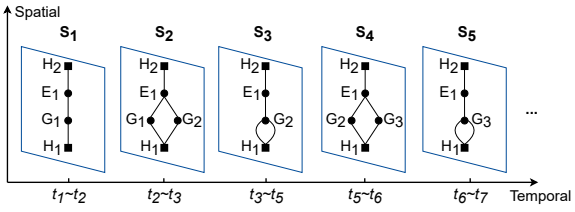


Figure 3.3: A topological breakdown over time.

where only one satellite is visible, both subflows connect to the same satellite. This cyclic pattern of connectivity switching is a characteristic of the network topology in this scenario.

OLIA [78] has been widely adopted as a multipath-based congestion control algorithm for MPTCP and MPQUIC. Let $\mathcal{R}$ be the set of available paths, and w_r and τ_r are the current cwnd and RTT of path $r \in \mathcal{R}$, respectively. The behavior of OLIA during the congestion avoidance (CA) phase is summarized below:

- For each ACK on path r, increase w_r by:

$$\frac{w_r/\tau_r^2}{\left(\sum_{p\in\mathcal{R}} w_p/\tau_p\right)^2} + \frac{\alpha_r}{w_r} \tag{3.1}$$

  where α_r is to guarantee the responsiveness of OLIA. By measuring the current RTT and the number of transmitted bits since the last loss, it determines whether a path is the best path or not. Then α_r is positive for all best paths with a small window and negative for paths that are not the best path while having a maximum window, and it is zero for the path that already has the maximum window.

- For each loss on path r, decrease w_r by $\frac{w_r}{2}$.

For ITSN with large BDP and periodical topological changes, OLIA encounters the following issues.

Bandwidth underutilization: OLIA suffers from bandwidth underutilization given its slow-start probing strategy when initializing new subflows in states s_1 and s_2. The time required by the slow start (SS) to reach a bit rate $\mathcal{C}$ is

$$t_{\text{SS}} = \tau \cdot \left(1 + \log_2 \frac{\mathcal{C}\cdot\tau}{l\cdot\Omega}\right),$$

where l is the average packet length expressed in bits, and Ω represents the initial cwnd. For instance, given that $\mathcal{C} = 200$ Mbps, $\tau = 100$ ms, $l = 1$ KB, and $\Omega = 1$ MSS, we have $t_{\text{SS}} = 1.13$ seconds which is too slow. In addition, upon the reception of a packet loss signal, OLIA will decrease cwnd by half and switch to the CA phase, and increase cwnd additively as in (3.1) to probe for the capacity of high-BDP links, which takes long time.

Inaccurate congestion signal: Under the frequent handover ITSN scenario, packet losses may be due to handover or channel impairments. MPQUIC may misinterpret the unexpected loss signals as congestion and decrease cwnd afterward, degrading the network throughput.

Responsiveness: OLIA mainly relies on α in (3.1) to adapt to network conditions, and α is only updated when receiving an indication of packet loss. However, for unexpected events such as topology change, OLIA cannot perceive it, leading to a slow response in terms of cwnd adjustment. Assuming in the state s_2, the two disjoint paths both have an available bandwidth of 200 Mbps, the cwnd of F_0 and F_1 can be up to about 1000 MSS. After the state changes from s_2 to s_3 if F_0 hands over from G_1 to G_2, F_0 and F_1 have to share the bottleneck links where the bandwidth is 200 Mbps. In this case, if OLIA is unaware of the state change and keeps w_0 and w_1 at 1000 MSS unchanged, the overflow would occur at the bottleneck links.

The root cause of these problems boils down to one key challenge: MPQUIC at the transport layer is unable to differentiate between fluctuations in network conditions caused by congestion and those caused by mobility and handover. Thus, a mobility-aware congestion control algorithm for MPQUIC (MACO) is proposed, consisting of two components: BDP-Inspired Quick Start and Mobility-Aware Congestion Avoidance.

3.4 BDP-Inspired Quick Start

In this section, a quick start (QS) alternative is proposed to tackle the bandwidth under-utilization issue during the SS phase.

Since path BDP reflects the network capacity, it is wise to set the SS threshold (SST) at BDP first, as cwnd can quickly approach SST. Here, two key points should be considered: how to capture the variable BDP accurately at a low cost (**Algorithm 3**), and how to quickly approach the network capacity without severe congestion consequences (**Algorithm 4**).

Algorithm 3: BDP Estimation in ITSN

Input : $\{P_t, d_r, N_r, \mathcal{C}^A_{r,t}, \mathcal{C}^N_{r,t}\}$, $\forall r \in \mathcal{R}$
Output: $\overline{\mathcal{B}_r}(t)$.

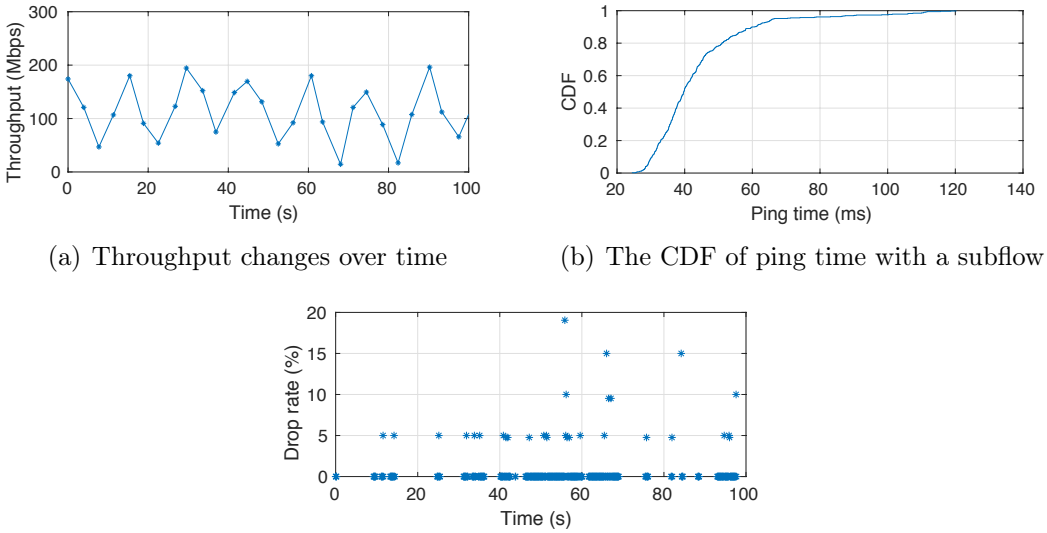
1 **Initialization at time** t_0
2 $\quad t \leftarrow t_0$;
3 $\quad \mathcal{C}^A_{r,t}, \mathcal{C}^N_{r,t} \leftarrow 0$;
4 $\mathcal{C}^A_{r,t} \leftarrow \eta \cdot W \log_2(1 + \frac{P_t g_1 g_2}{(4\pi d_r/\lambda)^2 N_r W})$;
5 Estimate $\mathcal{D}_{r,t}$ using Eq. (3.11);
6 **if** $t == t_0$ **then**
7 $\quad \overline{\mathcal{B}_r}(t) \leftarrow \mathcal{C}^A_{r,t} * \mathcal{D}_{r,t}$;
8 $\quad$ **return** $\overline{\mathcal{B}_r}(t)$;
9 **else**
10 $\quad$ **while** *new ACK received at* t **do**
11 $\quad\quad$ Update S_t, S_{t-1}, τ_r;
12 $\quad\quad \mathcal{C}^N_{r,t} \leftarrow \frac{S_t - S_{t-1}}{\Delta t}$;
13 $\quad\quad$ Update $\mathcal{C}^N_{r,t}$ using AKF;
14 $\quad\quad \mathrm{C}_{r,t} = \min\{\mathcal{C}^A_{r,t}, \mathcal{C}^N_{r,t}\}$;
15 $\quad\quad \overline{\mathcal{B}_r}(t) = \frac{1}{\tau_r} \int_{t-\tau_r}^{t} \mathcal{C}_{r,t} * \mathcal{D}_{r,z} dz$;
16 **return** $\overline{\mathcal{B}_r}(t)$

3.4.1 BDP Estimation in ITSN

By definition, the BDP of path r at time t, $\mathcal{B}_r(t)$, is the product of the available bottleneck bandwidth of the tagged flow, $\mathcal{C}_{r,t}$, and the round-trip propagation delay $\mathcal{D}_{r,t}$. **Algorithm 3** summarizes the procedures of BDP calculation. Lines 1–8 aim to obtain the path BDP in the first RTT, and lines 9–15 present how to utilize the feedback received to further improve the BDP estimation accuracy. The details of these two parts are given in Sections 3.4.1 and 3.4.1, respectively.

Bottleneck at Access Links

As stated in [153], the bandwidth of ground-to-satellite wireless links is relatively low compared to other optical wired or wireless links, so the access link is likely the bottleneck. Under this assumption, we can derive the bottleneck bandwidth without lengthy network probing. In particular, the sender could quickly realize the access link status in the first RTT, as depicted in lines 6–8 of **Algorithm 3**, and initiate the data transmission over each path. This feature proves advantageous in scenarios



(a) Throughput changes over time

(b) The CDF of ping time with a subflow

(c) The drop rate distributions over time

Figure 3.4: Measurements over Starlink LEO networks.

where the RTT is excessively long.

To gain insights into the realistic situation of LEO access networks, we acquired several Starlink dishes and subscribed to LEO services provided by SpaceX so that we can measure the realistic E2E path condition changes over time. By conducting pings to a peered device in Seattle, US from Victoria, Canada, we have captured the fluctuations in network bandwidth, RTT, and packet drop rate over a subflow. The data traces from a 100-second interval on April 21, 2023, are depicted in Fig. 3.4. With the observation from extensive measurement studies, we have noted that the handover between the end host and satellites occurs approximately every 15 seconds. This frequent handover results in significant fluctuations in the throughput performance as shown in Fig. 3.4(a). However, due to the periodic satellite movements, the change of throughput appears good regularity so that predicting the bandwidth of access links without probing is feasible.

Denote by $\mathcal{C}^A_{r,t}$ the time-varying bandwidth associated with ground-satellite links. Recall the description of $\{v, d_1, d_2, R_1, R_2\}$ in Section 3.3.1, $\frac{d_1}{v} = \frac{2R_2 - R_1}{2v}$ defines the interval of state changes from the disjoint to shared state, e.g., from s_2 to s_3, and $\frac{d_2}{v} = \frac{2R_2 - 2d_1}{v}$ characterizes the interval of opposite changes, that is, from the shared to disjoint state. As a result, we can predict which satellite will be associated with subflow r at time t, as well as the distance between access satellites and H_1. Let $d_r(t)$ be the distance of F_r to their accessed satellites at time t, using the path-loss model,

we have [63, 111]

$$\mathrm{SNR_r} = \frac{P_t g_1 g_2}{(4\pi d_r/\lambda)^2 N_r W}. \tag{3.2}$$

In (3.2), P_t stands for the transmit power, g_1 is the transmitting antenna gain, g_2 is the receiving antenna gain, λ is the signal wavelength, N_r is the noise power spectrum density (watts per hertz), and W denotes the bandwidth of the spot beam. Here we omit the symbol t of each parameter for simplicity.

Finally, the data rate of the ground-satellite link can be estimated by

$$\mathcal{C}^A_{r,t} = \eta \cdot W \log_2(1 + \mathrm{SNR_r}), \tag{3.3}$$

where $\eta \in (0, 1)$ is the efficiency coefficient of the system, depending on many factors, e.g., hardware and software design, as well as the modulation and coding schemes.

Bottleneck at the Network Side

In the above subsection, we operated under the assumption of the existence of direct links between end hosts and satellites, where the link capacity typically mirrors the bottleneck bandwidth. In this subsection, we focus on how to determine the bottleneck link capacity in scenarios where direct links between end hosts and satellites are lacking, or when subflows encounter bottlenecks at non-access links, such as inter-satellite links or terrestrial networks, referred to as bottlenecks at the network side. Denote by $\mathcal{C}^N_{r,t}$ the time-varying bottleneck bandwidth at the network side.

Since bottlenecks could occur on any intermediate links that make up the connection, the sender can estimate $\mathcal{C}^N_{r,t}$ by measuring acknowledgments (ACKs). Lines 11–13 of **Algorithm 3** describe two steps to derive $\mathcal{C}^N_{r,t}$: real-time measurement based on ACK packets, and correction with the Adaptive Kalman Filter (AKF).

Given the self-clocking feature, a common approach to measure $\mathcal{C}^N_{r,t}$ is to send a few packets back-to-back over subflow r, and then monitor the timestamps of the received ACKs to estimate bottleneck bandwidth. Assuming S_t and S_{t-1} are the total amounts of data received by the mobile host at successive measurement time instants t and $t-1$, $\mathcal{C}^N_{r,t}$ is measured with

$$\mathcal{C}^N_{r,t} = \frac{S_t - S_{t-1}}{\Delta t}. \tag{3.4}$$

However, the measured $\mathcal{C}^N_{r,t}$ in this manner is likely biased due to many reasons,

e.g., the sending window is too small during slow-start to match the actual network capacity so that the instantaneous $\mathcal{C}_{r,t}^N$ is much lower than it should be, or some neighboring satellites are unreachable due to angle issues so that the routing path is a detour, leading to a longer Δt and consequently an underestimated $\mathcal{C}_{r,t}^N$.

Therefore, we utilize an Adaptive Kalman Filter (AKF) [75] to remove the noise from the estimation of $\mathcal{C}_{r,t}^N$. Given limited measured samples, AKF can still quickly narrow down to the truth by taking a few of those inputs by understanding the variation or the uncertainty of those inputs. The standard Kalman filter goes through two stages to acquire the final estimate as follows.

The first stage is *prediction*, in which the estimated capacity ($\mathcal{C}_{r,t_e}^N$) of path r and the estimation errors $Z_{r,t}$ are updated as

$$\mathcal{C}_{r,t_e}^{N-} = \mathcal{C}_{r,t-1}^{N+} + \delta_{r,t}, \tag{3.5}$$

$$Z_{r,t}^{-} = Z_{r,t-1}^{+} + Q_r, \tag{3.6}$$

where the superscript "+" indicates that the estimate is a posterior, and "-" indicates a prior estimate. $\delta_{r,t}$ stands for white Gaussian noise with zero mean and variance Q_r, which depends on network uncertainties such as satellite mobility.

The second stage is *correction*. The measured network capacity is assumed disturbed by a white Gaussian noise $v_{r,t}$ with zero mean and variance R_r, such that

$$y_{r,t_m} = \mathcal{C}_{r,t_m}^N + v_{r,t}. \tag{3.7}$$

Here $\mathcal{C}_{r,t_m}^N$ is the instantaneously measured value at time t, using (4.10). Then the critical Kalman gain is derived by

$$K_{r,t} = Z_{r,t}^{-} \left(Z_{r,t}^{-} + R_r \right)^{-1}, \tag{3.8}$$

with which we can determine whether the predicted value or measured value is closer to the true value. Let $\hat{C}_{r,t}^N$ and $\hat{Z}_{r,t}$ be the final correction for the predictions of $\mathcal{C}_{r,t_e}^{N-}$ and $Z_{p,t}^{-}$, we have

$$\hat{C}_{r,t}^N = \mathcal{C}_{r,t_e}^{N-} + K_{r,t} \left(y_{r,t_m} - \mathcal{C}_{r,t_e}^{N-} \right), \tag{3.9}$$

$$\hat{Z}_{r,t} = Z_{r,t}^{-} - K_{r,t} Z_{r,t}^{-}. \tag{3.10}$$

The corrected values $\hat{C}_{r,t}^N$ and $\hat{Z}_{r,t}$ will serve as the posterior states influencing

future rounds of corrections. This approach results in a reduction of estimation errors over time, consequently reducing noise within $\mathcal{C}_{r,t}^{N}$[1].

$\mathcal{D}_{r,t}$ Estimation

$\mathcal{D}_{r,t}$ can be decomposed into the propagation delay $D_{\text{ul},t}$ and $D_{\text{dl},t}$ over the ground-to-satellite and satellite-to-ground links, and the delay $D_{\text{terr},t}$ within terrestrial networks. The former has significant variation due to the changing distance between ground stations and satellites. Since we can derive the distance d_r as described in the above subsection, $D_{\text{ul},t}$ and $D_{\text{dl},t}$ can be calculated by dividing the distance by the propagation speed, approximately 3×10^8 m/s. Comparatively, $D_{\text{terr},t}$ is more stable with fewer variations [35]. According to Fig. 3.4(b), the 99th percentile of the RTT for each subflow is 100 ms. Therefore, we can empirically set $D_{\text{terr},t}$ to a constant value at the first RTT. Upon receiving new ACKs in the subsequent rounds, the estimated delay could be updated, e.g., using an exponentially weighted moving average (EWMA).

Therefore, $\mathcal{D}_r$ is given by

$$\mathcal{D}_{r,t} = 2 * (D_{\text{ul},t} + D_{\text{dl},t} + D_{\text{terr},t}). \tag{3.11}$$

BDP Estimation

As summarized in **Algorithm 3**, after obtaining $\mathcal{C}_{r,t}^{A}$ and $\mathcal{C}_{r,t}^{N}$ through lines 4–13, $\mathcal{C}_{r,t}$ is given by

$$\mathcal{C}_{r,t} = \min\{\mathcal{C}_{r,t}^{A}, \mathcal{C}_{r,t}^{N}\}, \tag{3.12}$$

and the path BDP is derived by

$$\mathcal{B}_r(t) = \mathcal{C}_{r,t} * \mathcal{D}_{r,t}. \tag{3.13}$$

$\mathcal{B}_r(t)$ in (4.8) represents instantaneous BDP information immediately following the reception of a new acknowledgment. However, it does not consider the bandwidth fluctuations caused by the rapid movement of satellites during the subsequent transmission round. Therefore, directly utilizing $\mathcal{B}_r(t)$ to regulate cwnd settings is

[1]To implement the Kalman filter effectively, knowledge of the covariances for prediction and measurement noise, denoted as Q_r and R_r, respectively, is essential. However, in our case, these parameters are initially unknown and may vary with changes in the network environment over time. To address this challenge, we can employ the Least Squares method to estimate them based on the autocorrelation of the innovation signal, as outlined in [115].

not ideal. To cope with this challenge, we average out the BDP within the successive RTT in the following manner,

$$\overline{\mathcal{B}_r}(t) = \frac{1}{\tau_r} \int_{t-\tau_r}^{t} \min\{\mathcal{C}_{r,z}^{A}, \mathcal{C}_{r,z}^{N}\} * \mathcal{D}_{r,z} dz, \tag{3.14}$$

in which $\overline{\mathcal{B}_r}(t)$ is used to guide the QS as presented in the next subsection.

3.4.2 QS

The key idea behind the QS is to set SST to the averaged BDP $\overline{\mathcal{B}_r}(t)$ in the first RTT. Next, to approach SST faster, we set the initial window of each subflow r, denoted by $\Omega_r(t)$ to a larger value based on historical information, rather than a small constant all the time. **Algorithm 4** states how the QS performs in a general case where the number of established subflows might be greater than 2.

For ease of understanding, we take a two-subflows example as shown in Fig. 3.5 to elaborate the QS design as follows.

When MACO users join the network and the first subflow F_0 is created, there is no historical information for the F_0. When the w_0 is below 4 maximum segment size (MSS), it is impossible to trigger fast retransmission to recover lost packets, and thus w_0 is initially set to 4 MSS, i.e., $\Omega_0(t_0) = 4$ MSS as shown in Fig. 3.5. Starting with $\Omega_0(t_0)$, w_0 is increased by 1 MSS per ACK until reaching the SST.

Because MPQUIC subflows share some similarities w.r.t. access link capacity, E2E hop count, packet loss rate, etc., the window change trajectory of F_0 is valuable in guiding the selection of $\Omega_1(t)$ for F_1. Suppose F_0 has sent a certain amount of data and experienced network congestion subsequently. We record the associated cwnd information ($w_0^*(t)$) whenever congestion occurs and store them into a set $\mathcal{W}_0$, i.e., $\mathcal{W}_0 = \{w_0^*(t)\}$, as described in the line 4 of **Algorithm 5**. Therefore, when starting up F_1 at t_3 as shown in Fig. 3.5, the BDP of F_1 denoted by $\overline{\mathcal{B}_1}(t_3)$ is first calculated using Eq. (3.14), and then F_1 checks if historical records exist. Since $\mathcal{W}_1$ is empty at t_3 while $\mathcal{W}_0$ has an element of $w_0^*(t_2)$, as depicted in lines 7–12 of **Algorithm 4**, $\Omega_1(t_3)$ would be determined by both $\overline{\mathcal{B}_1}(t_3)$ and $\mathcal{W}_0$ in the following way

$$\Omega_1(t_3) = \gamma \cdot \min\{\overline{\mathcal{B}_1}(t_3), w_0^*(t_2)\}. \tag{3.15}$$

Here we introduce a coefficient γ to ensure the initial window is not only large enough

Algorithm 4: QS

Input : $\mathcal{W}_r$, $\forall r \in \mathcal{R}$
Output: cwnd value during the SS phase.

1 **if** *a new subflow F_r is created or a broken subflow is reconnected* **then**
2 // Set an initial cwnd for F_r
3 **if** $\mathcal{W}_r \neq \emptyset$ **then**
4 $w_{\max} \leftarrow \max\{w_r(t)\}$;
5 $\Omega_r(t) = \gamma \cdot \min\{\overline{\mathcal{B}_r}(t), w_{\max}\}$;
6 **else**
7 **if** $\exists \quad \mathcal{W}_q \quad \neq \emptyset$ **then** $q \in \mathcal{R}(q \neq r)$
8 $w_{\max} \leftarrow 0$;
9 **foreach** $q \in \mathcal{R}(q \neq r)$ **do**
10 **if** $w_{\max} < \max\{w_q(t)\}$ **then**
11 $w_{\max} \leftarrow \max\{w_q(t)\}$;
12 $\Omega_r(t) \leftarrow \gamma \cdot \min\{\overline{\mathcal{B}_r}(t), w_{\max}\}$;
13 **else**
14 $\Omega_r(t) \leftarrow 4$ MSS;
15 $w_r \leftarrow \Omega_r$;
16 **else**
17 // Increase the cwnd exponentially with RTT
18 $w_r \leftarrow w_r +$ # of ACKs ;
19 **return** w_r

but also safe to start, and empirically, γ is set to $1/2$.

After states s_1 and s_2, both F_0 and F_1 have the transmission records and maintain their own set $\mathcal{W}_r$ ($r = \{0, 1\}$) for the subsequent states reference. Thereby, when F_r enters into the SS process again upon the occurrence of timeout or connection breakage, the corresponding $\Omega_r(t)$ can be generalized as

$$\Omega_r(t) = \gamma \cdot \min\{\overline{\mathcal{B}_r}(t), \max\{w_r^*(t)\}\}, \tag{3.16}$$

which corresponds to the lines 3–5 in **Algorithm 4**.

In conclusion, in the QS procedure, MACO strives to start with an appropriate Ω by learning from the recent past and taking advantage of good similarity among different subflows in the ITSN scenario; after the cwnd reaches the SST, MACO steps into CA phase as elaborated in the following subsections.

Algorithm 5: Mobility-Aware Congestion Avoidance

Input : $\{w_r, \tau_r, \mathcal{C}_r(t-1), \mathcal{D}_r(t-1), TP_r(t)\}$, $\forall r \in \mathcal{R}$
Output: cwnd changes during the CA phase.

1 $Con\_\mathrm{I} \Leftarrow \mathbf{1}\{TP_r(t) < \mathcal{C}_r(t-1)\}$;
2 $Con\_\mathrm{II} \Leftarrow \mathbf{1}\{\tau_r(t) > \mathcal{D}_r(t-1) + \Delta_R\}$;
3 **if** *packet loss* $\wedge$ Con_I $\wedge$ Con_II **then**
4 $\mathcal{W}_r \leftarrow \mathcal{W}_r \cup w_r$;
5 $w_r \leftarrow w_r/2$;
6 **else**
7 $T_r = \max_{t \in \{t_0, t_1, \cdots, t_n\}} \tau_r(t)$;
8 **foreach** $k \in \mathcal{R}$ **do**
9 $x_k \leftarrow \frac{w_k(t_n)}{\tau_k(t_n)}$;
10 $w_r \leftarrow w_r + \frac{3 \max_{r \in \mathcal{R}} x_r^2 \sqrt{T_r}}{2\tau_r \left(\sum_{k \in \mathcal{R}} x_k\right)^{\frac{5}{2}}}$;
11 **return** w_r

3.5 Mobility-Aware MPQUIC Congestion Avoidance

Due to high-speed satellite movement, frequent handovers severely impair the quality of experience (QoE) for users [156]. To provide good QoE, the CA design is driven by three goals: 1) Distinguish the congestion loss from handover loss to avoid unnecessary cwnd reduction. 2) Maintain fast convergence to the network capacity. 3) The total throughput of multiple subflows in the handover process should be no less than the throughput that a single-path TCP can achieve. **Algorithm 5** concludes the CA design part.

3.5.1 Mobility-Aware Factor Design

For each subflow r, a mobility-aware factor $h_r \in \{0, 1\}$ is introduced to help identify the true congestion signal. According to lines 1–3 of **Algorithm 5**, MACO takes into account three conditions to detect if the network is congested, i.e., packet loss signal along with two conditions given in (3.17). Only if the three conditions are both satisfied, the network is identified as congested and $h_r = 1$. Otherwise, $h_r = 0$.

$$\begin{cases} TP_r(t) < \mathcal{C}_r(t-1), \\ \tau_r(t) > \mathcal{D}_r(t-1) + \Delta_R, \end{cases} \tag{3.17}$$

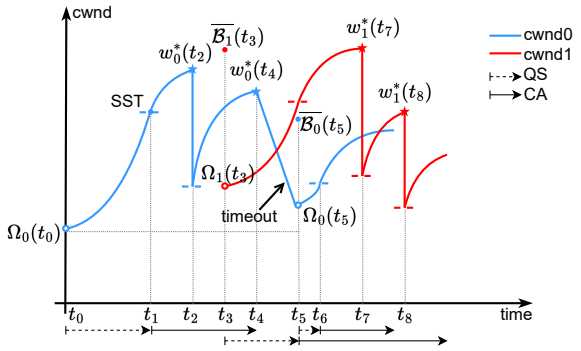


Figure 3.5: Window growth model of MACO.

where $TP_r(t)$ is the available bandwidth measured by (4.10) at the t-th sampling time when receiving an ACK, while $\mathcal{C}_r(t-1)$ is the predicted access link bandwidth by (3.3) at the $(t-1)$-th sampling time right before sending packets. Similarly, $\tau_r(t)$ is the measured RTT at time t, and $\mathcal{D}_r(t-1)$ is the estimated propagation delay using (3.11) at time $t-1$. Considering the unavoidable estimation error of delay, Δ_R is a factor that represents the standard deviation of the differences between the recent RTT measurements and the corresponding estimated ones.

3.5.2 CA Design

In general, the CA algorithm devises a specific control policy governing the increase term (U_r) for each acknowledgment and the decrease term (V_r) triggered by packet loss signals. In our model, V_r is intrinsically tied to system stability and is typically set to half of the current congestion window, i.e., $V_r = \frac{w_r}{2}$. This value remains constant, and our primary focus centers on the design of U_r for each subflow r.

To meet the fast convergence goal, the AIMD scheme is undesirable in high-BDP networks. Instead, a function that grows aggressively when the current cwnd is far away from BDP and grows conservatively when the cwnd is approaching the BDP is more desirable. The square-root function presented in the recent proposal named Elastic-TCP [9] can fulfill our requirements with low complexity, so we employ a similar function to design U_r for each subflow r in the CA phase. Fig. 3.5 depicts how the cwnd is tuned with the square root increasing model during the CA phase.

Authors in [51, 109] formulated the CA process of MPTCP as a fluid model to facilitate the analysis of certain flow level properties, e.g., friendliness, convergence, and stability. [109] concluded that $\frac{\partial \phi_r(\mathbf{X}_\mathcal{R})}{\partial \mathbf{X}_\mathcal{R}}$ determines the convergence speed to the

equilibrium locally if the system is perturbed by unexpected factors (e.g., handover), where ϕ_r is the ratio of U_r to V_r and $\mathbf{X}_{\mathcal{R}}(t) = (x_r(t), r \in \mathcal{R})$, and $x_r = w_r/\tau_r$, indicating the transmission rate on subflow r.

Let $x_r(t) = w_r(t)/\tau_r$ denote the transmission rate of subflow r at time t

To achieve the same convergence property as Elastic-TCP, MACO should have

$$\left(\frac{\partial \phi_r}{\partial \mathbf{X}_{\mathcal{R}}}\right)^{MACO} = \left(\frac{\partial \phi_r}{\partial \mathbf{X}_{\mathcal{R}}}\right)^{Elastic-TCP}. \tag{3.18}$$

In addition, according to the Theorems in [51], an equilibrium of the system does exist if $\phi_r(\mathbf{X}_{\mathcal{R}})$ of MACO can be generalized as

$$\phi_r\left(\mathbf{X}_{\mathcal{R}}\right) = \frac{\theta_r}{x_r}, \tag{3.19}$$

where $\theta_r > 0$ is a variable depending on different convergence rate requirements.

Since

$$\left(\frac{\partial \phi_r}{\partial \mathbf{X}_{\mathcal{R}}}\right)^{Elastic-TCP} = -\frac{3\sqrt{T_r x_r^{TCP}}}{\tau_r^2 (x_r^{TCP})^3}, \tag{3.20}$$

where T_r is the maximum RTT the subflow r has ever experienced, which will be updated upon reception of a new ACK, we have

$$-\frac{\theta_r}{x_r^2} = -\frac{3\sqrt{T_r}}{\tau_r^2 (x_r^{TCP})^{\frac{5}{2}}}. \tag{3.21}$$

In addition, to ensure goal 3) that the multiple subflows maintain at least the same aggregate throughput as co-existing Elastic-TCP, we have $\sum_{k\in\mathcal{R}} x_k = \max_{r\in\mathcal{R}} \left(x_r^{TCP}\right)$. Furthermore,

$$\max_{r\in\mathcal{R}} \left(x_r^{TCP}\right)^{\frac{5}{2}} = \left(\sum_{k\in\mathcal{R}} x_k\right)^{\frac{5}{2}} = \max_{r\in\mathcal{R}} \frac{3x_r^2\sqrt{T_r}}{\theta_r \tau_r^2}. \tag{3.22}$$

Then,

$$\theta_r = \frac{3 \max_{r\in\mathcal{R}} x_r^2 \sqrt{T_r}}{\tau_r^2 \left(\sum_{k\in\mathcal{R}} x_k\right)^{\frac{5}{2}}}. \tag{3.23}$$

Hence, we obtain

$$\phi_r\left(\mathbf{X}_{\mathcal{R}}\right) = \frac{3 \max_{r\in\mathcal{R}} x_r^2 \sqrt{T_r}}{\tau_r^2 x_r \left(\sum_{k\in\mathcal{R}} x_k\right)^{\frac{5}{2}}}. \tag{3.24}$$

Given V_r, we can obtain the solution of the increment term $U_r\left(\mathbf{X}_{\mathcal{R}}\right)$ as below,

$$U_r\left(\mathbf{X}_{\mathcal{R}}\right)=\frac{3\max_{r\in\mathcal{R}} x_r^2\sqrt{T_r}}{2\tau_r\left(\sum_{k\in\mathcal{R}} x_k\right)^{\frac{5}{2}}}. \tag{3.25}$$

Finally, The CA algorithm is summarized in **Algorithm 5**:

- In the CA phase, for each ACK on F_r,

$$w_r \leftarrow w_r+\frac{3\max_{r\in\mathcal{R}} x_r^2\sqrt{T_r}}{2\tau_r\left(\sum_{k\in\mathcal{R}} x_k\right)^{\frac{5}{2}}}, \tag{3.26}$$

3.5.3 Time complexity analysis

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

In **Algorithm** 3, the operations of each step can be done in constant time, so the time complexity is $O(1)$.

For **Algorithm** 4, the time complexity is mainly affected by line 4 and lines 9–12. To find out the maximum $w_r(t)$ in the set $\mathcal{W}_r$ in line 4, we may need to compare each element with the current maximum in the worst case. Therefore, line 4 has a time complexity of $O(m)$ where m stands for the number of occurrences of congestion over path r. The task of lines 9–12 is to search the maximum value across $R-1$ sets, where R indicates the number of subflows that MPQUIC has established and used. Under the assumption that each subflow maintains the same magnitude of the number of congestion occurrences, the running time of lines 9–12 is $mR-m$. As a result, the time complexity of **Algorithm** 4 is $O(mR)$.

Similarly, in **Algorithm** 5, the line 7 and lines 8–9 contribute n and R running times, respectively. Here n represents the number of RTT records for a certain path. Consequently, the overall time complexity is $O(n+R)$.

3.6 Performance Evaluation

In this section, we conduct extensive experiments to verify the performance of MACO in the ITSN scenario, in comparison with the following benchmarks:

1. BBR: Similarly to ours, BBR is a BDP-inspired congestion control algorithm, which has been extensively studied in satellite networks thanks to its BDP-aware

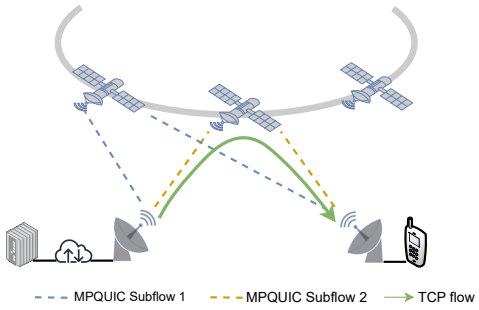


Figure 3.6: Simulation topology.

features. BBR with multipath support is not yet available since the coupled design of the pacing gain rate for multiple paths remains an open issue. Therefore, we currently use BBR as the single-path benchmark in ITSN scenarios.

2. OLIA: OLIA has been widely adopted as the congestion control algorithm for MPQUIC. It aims to achieve the Pareto-optimal cwnd regulation and good fairness towards single-path users. Extensive experiments demonstrate that OLIA achieves great bandwidth utilization under various network settings.

In comparison with single-path-based (i.e., BBR) and multipath-based (i.e., OLIA and MACO) solutions in mobile environments, we can gain insights into the efficiency of multipath transmission in dealing with seamless handover. With respect to the multipath packet scheduling policy, we employ the round-robin algorithm for all multipath-based solutions. On the other hand, the comparison between loss-based (i.e., OLIA) and BDP-based (i.e., BBR and MACO) solutions reveals how they react to high BDP networks and how the E2E QoS performance will be affected.

3.6.1 Prototype Implementation

To begin with, we extended the existing QUIC module [6] to MPQUIC based on ns-3 in accordance with the Internet Engineering Task Force (IETF) draft[2] on MPQUIC. We refer readers to [124] and our open-source code at https://github.com/ssjShirley/mpquic for the implementation details of MPQUIC-ns3.

Based on the MPQUIC prototype, we further create two mobility modules: one for the LEO satellite constellation, and the other for facilitating end devices interacting

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

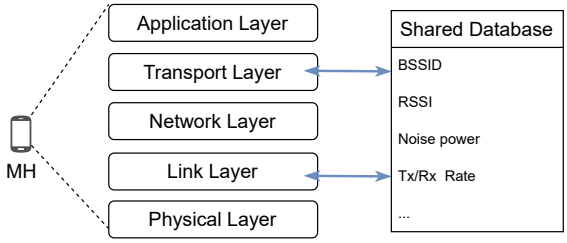


Figure 3.7: Cross-layer architectural blueprint.

with the LEO satellites. They are responsible for configuring and maintaining the position of satellites and end hosts. As satellites move, end hosts can continuously calculate their distance to orbiting satellites, while also assessing the SNR of each interface.

To let the transport layer be aware of the SNR changes at the link layer, as illustrated in Fig. 3.7, we introduce a database in the control plane to enable the cross-layer interactions on which the MACO algorithm is implemented.

Through the creation of interfaces that connect the layers to the database, the data stored in the database is accessible to protocol functions across different layers. For instance, whenever the access link conditions change, the link layer acquires parameters such as Basic Service Set Identifier (BSSID), Received Signal Strength Indicator (RSSI), etc. These parameters are then recorded in the database, enabling the transport layer to retrieve this information. Note that the presence of the database does not change the conventional layered structure, and the encapsulation and decapsulation processes do not involve the data stored in the database. Therefore, this architecture is backward compatible.

3.6.2 Experimental Scenario and Settings

Fig. 3.6 depicts the topology we used, in which a client and a server engage in communication through two satellites. Two MPQUIC subflows are established between the sender and the receiver. Concurrently, background TCP flows are generated to traverse through the same satellites, competing with the MPQUIC flows.

According to the parameters outlined in [118] and the specifications reported by Starlink[3], we simulated an LEO satellite constellation with an altitude of 550 km, and the distance between two adjacent satellites in the same orbit R_1 is set to 150 Km.

[3]https://www.astronomy.com/space-exploration/starlink-satellite-streaks-how-big-of-a-problem-are-they/

Table 3.2: The setting for parameters in (3.2) and (3.3)

Parameters	Value
g_1	37.7 dBi
g_2	37.1 dBi
λ	0.086 m
N_r	1.6×10^{-20} W/Hz
η	0.98
W	12.5 MHz

The satellite's moving speed is 7 km/s, and each satellite has a footprint diameter of about 460 km.

By referring to recent studies [33, 102, 111], some parameter settings in (3.2) and (3.3) are given in Table 3.2. Combining with our measurement results presented in Fig. 3.4(a), which shows the achieved throughput, we configure the transmit power P_t. For instance, considering that the throughput peaks at 200 Mbps when the distance is minimized to 550 km, we assign values of 200 Mbps and 550 km to $\mathcal{C}_{r,t}^{A}$ in (3.3) and d_r in (3.2), respectively. By substituting these values into (3.3) and (3.2), P_t is 35.12 dBm.

3.6.3 Experimental Results

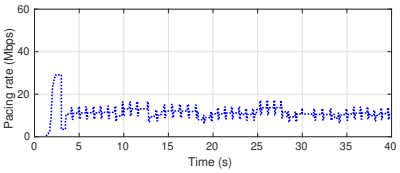
With the settings above, we compare MACO with benchmark algorithms in terms of four aspects: cwnd adaptation, throughput, TCP friendliness, convergence, and the impact of satellite density and BDP.

Congestion Window Adaptation

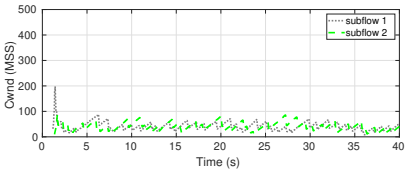
The cwnd is a critical aspect of congestion control design, as it illustrates how an algorithm adjusts its cwnd in response to changing network dynamics. For BBR which does not employ a traditional cwnd concept, we rely on the pacing rate as an indicator of its adaptive behavior. Fig. 3.8 presents the cwnd regulation over time.

With an average RTT of 100 ms, it's noteworthy that 500 MSS is roughly equal to an effective sending rate of 60 Mbps. An insightful observation can be drawn from the data depicted in Fig. 3.8, that is, MACO consistently maintains the largest cwnd on average, followed by BBR, with OLIA displaying the smallest cwnd.

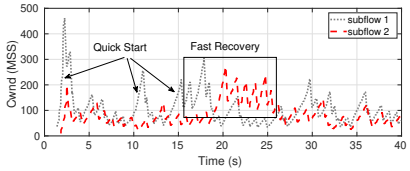
Fig. 3.8(a) illustrates the frequent oscillation in the pacing rate of the BBR congestion control algorithm. This behavior arises from BBR's aim to keep router



(a) The pacing rate adaptation of BBR



(b) cwnd of OLIA



(c) cwnd of MACO

Figure 3.8: The cwnd adaptation of different algorithms over time.

queues at the bottleneck link consistently empty by precisely matching the bottleneck link's rate limit. When a BBR flow coexists with another NewReno flow over a shared bottleneck, the latter tends to aggressively fill up the queue until packet loss occurs. In contrast, BBR triggers its draining function to empty the queue, which, in turn, limits its pacing rate. Consequently, the pacing rate of BBR is suppressed when it shares the bottleneck link with other traffic.

Comparing the cwnd trace of OLIA with MACO as shown in Fig. 3.8(b) and Fig. 3.8(c), respectively, it is evident that MACO maintains a higher average cwnd. Despite MACO showing a higher variation in the cwnd value, it ensures that the lower bound of cwnd is equal to or greater than OLIA's counterpart. This outcome can be attributed to MACO's QS and fast recovery mechanisms. In scenarios involving both mobility and competing traffic, the likelihood of packet losses increases. Unlike MACO, OLIA cannot distinguish congestion-induced losses from losses caused by mobility. As a result, cwnd degradation occurs frequently in OLIA, limiting the cwnd to under 100 MSS after 5 seconds. This value is significantly smaller than MACO's counterpart, which is 300 MSS.

Throughput

Fig. 3.9 demonstrates the throughput of each algorithm under the impact of TCP background traffic. By comparing the throughput of BBR, OLIA, and MACO, we observe that they all suffer from frequent fluctuations in the presence of satellite movement. On the other hand, the throughput of BBR ranges from 5 Mbps to

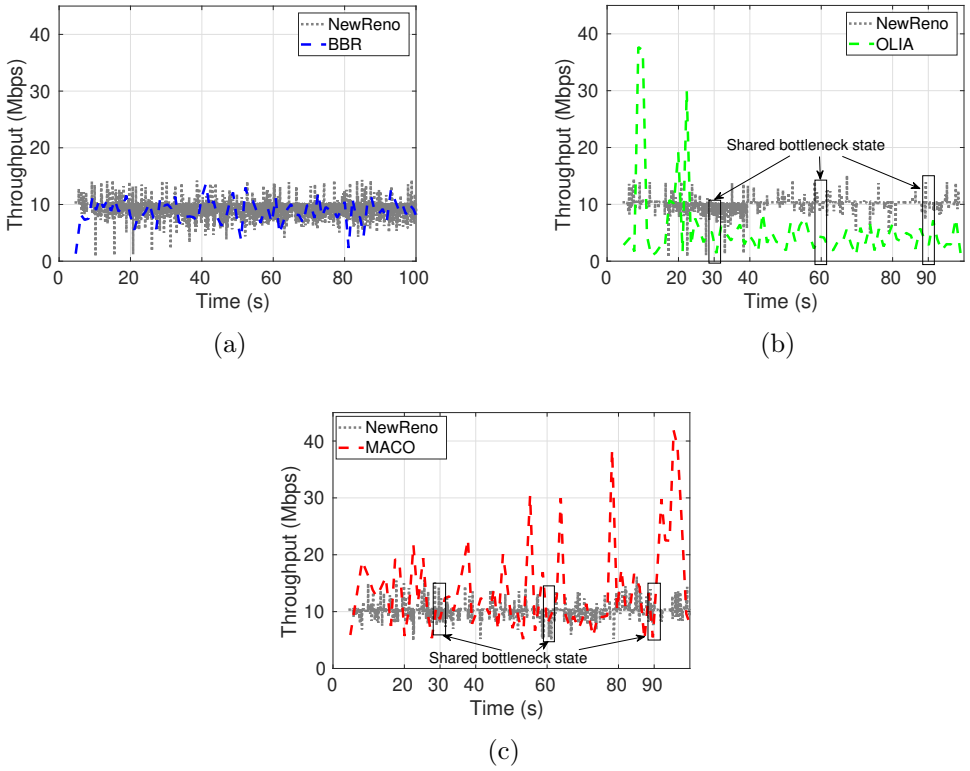


Figure 3.9: Throughput performance of each algorithm when processing QUIC traffic in the presence of TCP background traffic controlled by NewReno: (a) BBR, (b) OLIA, (c) MACO.

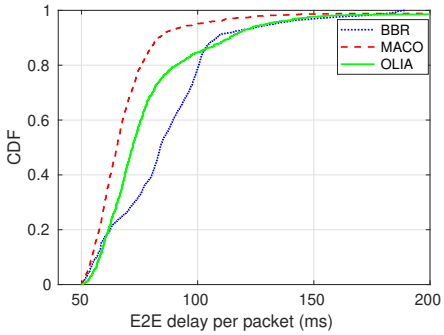


Figure 3.10: The CDF of E2E packet delay.

15 Mbps, which is comparatively low because BBR is a single-path-based solution. Nevertheless, BBR outperforms OLIA after 22 seconds even though OLIA reaches 38 Mbps in the first 22 seconds. This result reveals that the loss-based congestion control is not suitable for the high-BDP as well as highly lossy networks. Compared

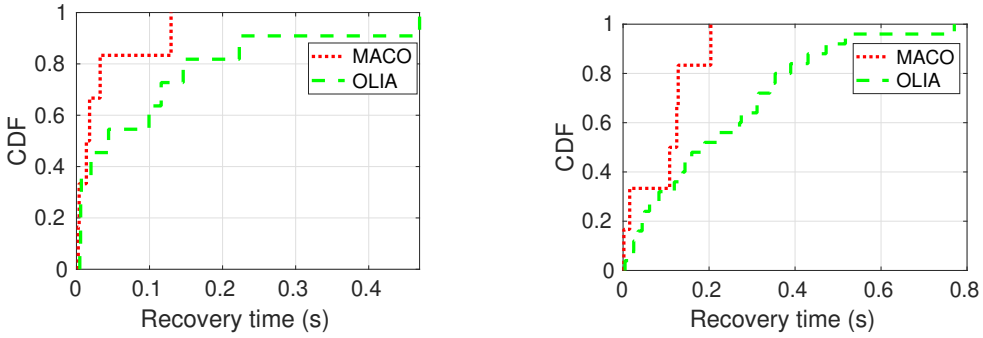
to BBR and OLIA, the throughput curve of MACO fits the Starlink measurements shown in Fig. 3.4(a) better, and that is the reason why MACO can achieve up to three times higher throughput record.

Fig. 3.10 illustrates the CDF of per-packet E2E delay with different algorithms. The 95th percentile of E2E delay is 101 ms for MACO, and the counterparts of OLIA and BBR are 130 ms and 132 ms, respectively. It implies that the throughput improvement achieved by MACO does not lead to longer queuing delays.

As analyzed earlier, filtering out non-congestion-related loss signals from congestion-related loss signals is crucial for throughput improvement as the large proportion of non-congestion-related loss leads to unnecessary cwnd decrement. We measured the number of congestion losses detected by different algorithms. In this scenario, MACO and BBR both receive a similar number of congestion signals, specifically 71 and 69 respectively. However, OLIA experiences a much higher number of congestion losses, totaling 128, leading to significant degradation of its cwnd.

TCP Friendliness

TCP-friendliness is a crucial characteristic of multipath congestion control algorithms. It ensures that the combined throughput of multiple subflows sharing a bottleneck with TCP flows does not exceed the throughput achieved by TCP. When the bandwidth is limited while both TCP and MPQUIC request a large amount of bandwidth, we need to consider it to ensure fairness. However, if the bandwidth resources are sufficient or the TCP user does not request much bandwidth, MPQUIC can still grab more bandwidth as demand. In our experiments, handover occurs every 15 seconds, resulting in the two MPQUIC subflows accessing the same satellite and sharing the bottleneck with TCP background traffic regulated by NewReno every 30 seconds, as shown in Fig. 3.3. The bottleneck-shared events are marked in Fig. 3.9(b) and 3.9(c). The aggregated throughput of OLIA or MACO is nearly equal to or lower than that of TCP during these events, indicating their TCP friendliness. Furthermore, MACO demonstrates TCP friendliness while also striving for higher throughput. It achieves this by aggressively increasing the congestion window when the network capacity is far from being reached and adopting a more conservative approach as it approaches the network capacity. Although OLIA meets the requirement of TCP friendliness, it fails to outperform TCP whenever there is no shared bottleneck.



(a) SS duration (b) CA duration

Figure 3.11: Convergence analysis when background traffic exists.

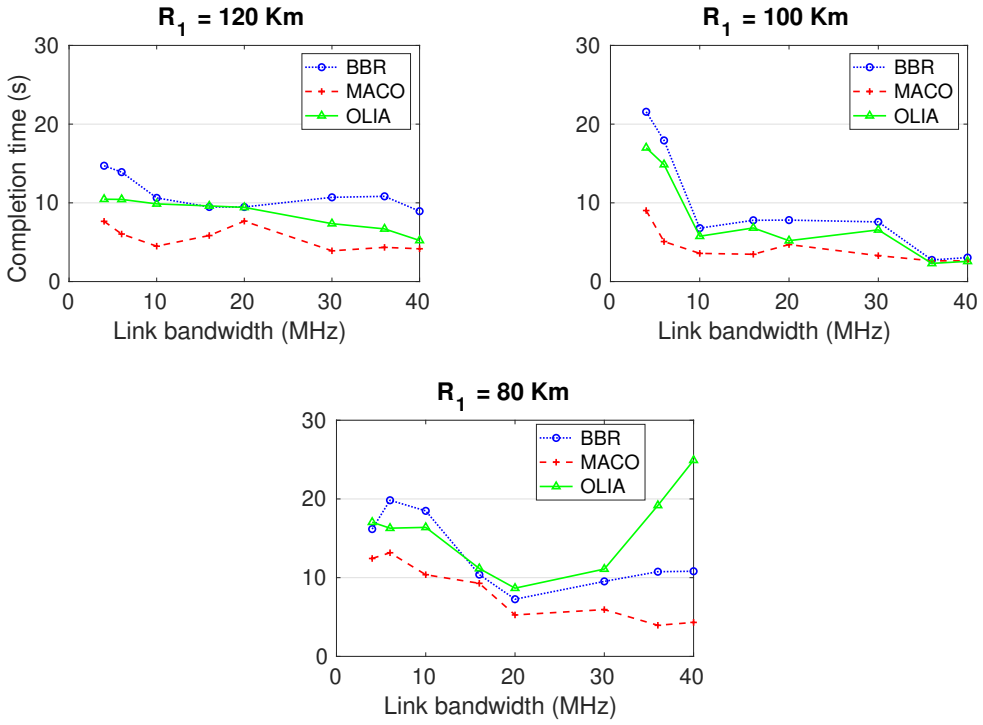


Figure 3.12: The impact of link bandwidth on the completion time at varying satellite density.

Convergence

Since the system could be affected by unexpected factors in ITSN, e.g., satellite availability and weather conditions, maintaining a fast convergence to the network capacity is crucial. To evaluate the convergence performance, we measure the recovery time from the moment that the subflow enters a specific phase (i.e., SS or CA) until

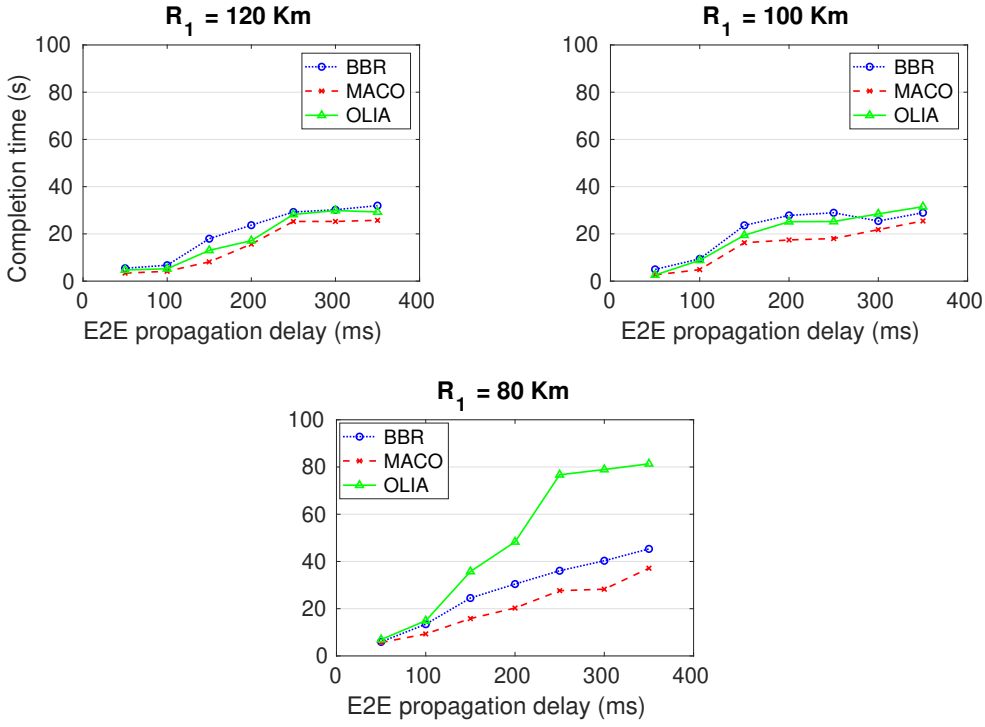


Figure 3.13: The impact of propagation delay on the completion time at varying satellite density.

the moment that the cwnd reaches the path BDP.

In the existence of TCP flows, we observe that MACO is more responsive during either stage compared to OLIA, which is the major reason behind the throughput and delay improvement as depicted in Fig. 3.9 and Fig. 3.10. Fig. 3.11(a) shows that the 99th percentile of the recovery time in the SS phase for MACO and OLIA is 129.4 ms and 470.3 ms, respectively, and the counterpart in the CA phase for MACO and OLIA is 204 ms and 770.3 ms, respectively. On the other hand, compared to the convergence performance in the case without competing traffic, the recovery time of MACO and OLIA in the CA stage is increased by 22.7% and 30.8%, respectively. The comparison results demonstrate that MACO substantially optimizes the system convergence against OLIA in various scenarios.

The Impact of Satellite Density and BDP

Over time, an increasing number of satellites are being launched, resulting in denser satellite distributions across space. In this subsection, we conduct a series of experiments investigating how the satellite density affects the performance of each algo-

rithm. Here we systematically tune the value of inter-satellite distance R_1 to adjust the satellite density. The smaller the R_1, the denser satellites are deployed. For various applications, including file transmission, the time it takes to complete the transfer of a specific file size serves as a critical performance metric that gauges the effectiveness of a congestion control algorithm. Therefore, the completion time is adopted as a metric for performance evaluation.

In addition, we aim to discover the influence of the changing BDP on the completion time of file transmission. Given the BDP is dependent on both bandwidth and delay, we manipulate the two parameters individually to discover how each of them affects the results. Unless specified otherwise, all other configurations, such as transmit power and satellite moving speed, remain consistent with the settings mentioned above.

Fig. 3.12 depicts our findings on how link bandwidth influences the completion time when transmitting a 10 MB file at various R_1 values. From these results, two key observations can be made. Firstly, increasing bandwidth is beneficial for MACO to speed up the file transfer algorithm. However, for BBR and OLIA, incremental bandwidth fails to substantially decrease completion time when R_1 is either too long or too short. This is attributed to the challenges that BBR and OLIA face in fully utilizing bandwidth within highly lossy and mobile network environments. Secondly, higher satellite density proves advantageous in certain scenarios, as denser satellite distributions offer more stable access links and greater available link bandwidth. However, this advantage is not universally applicable, especially in cases where satellites are ultra-dense distributed and the handover happens frequently.

Specifically, when R_1 is less than or equal to 100 Km, we observe a reduction in completion time for all algorithms as bandwidth increases. OLIA decreases BBR's completion time by 7.3% to 38.22%, while MACO reduces OLIA by 8.5% to 64.2%. However, for scenarios where R_1 is 80 Km, completion times for BBR and OLIA initially decrease as bandwidth expands up to 20 MHz, but start to increase as bandwidth exceeds 20 MHz. BBR outperforms OLIA such that it has a 1.3% – 56.5% completion time reduction in this case, while MACO reduces the completion time of BBR and OLIA by up to 6.5 and 20.6 s, respectively. Therefore, MACO consistently achieves the shortest completion times across all scenarios. This relies on the advantages of MACO's ability to adapt to changes in bandwidth and employ multipath transmission in mobile settings.

Fig. 3.13 investigates how algorithms respond to changing delays in diverse mobile

scenarios. In all scenarios, an increase in E2E propagation delay leads to prolonged completion times. MACO and OLIA outperform BBR when the delay is less than 250 ms as shown in Figures where $R_1 = 120$ and 100 Km. However, in scenarios where the delay surpasses this threshold, OLIA experiences performance degradation, resulting in significantly longer completion times compared to MACO and BBR. Conversely, MACO consistently achieves the shortest completion times across all scenarios. This can be attributed to its ability to consider measured path bandwidth and RTT, allowing it to distinguish congestion-related losses from handover losses. This approach mitigates unnecessary cwnd decreases and detrimental retransmissions, particularly in scenarios characterized by high delays.

In summary, the rising satellite density presents a dual-edged impact. While it can enhance link quality, it can also introduce more frequent handovers and a higher drop rate. When compared to BBR and OLIA, MACO consistently benefits from the increasing density. This is primarily due to MACO's ability to proactively reduce its cwnd to mitigate the risk of significant packet loss before handovers take place, and its rapid adaptation to network BDP changes in the presence of network perturbations. Additionally, MACO demonstrates better stability in response to fluctuations in bandwidth and delays.

3.7 Summary

This chapter presents an analytical framework for MPQUIC-enabled ITSN and proposes a multipath congestion control algorithm called MACO, addressing three key challenges. First, MACO tackles the low efficiency of benchmark algorithms in ITSN by leveraging the regularity of LEO topology and predictability of satellite movements to realize network capacity within the first RTT without prohibitive network probing. Second, given the estimated network capacity and the similarity among subflows, MACO develops a quick start (QS) mechanism to reduce the duration of the slow start (SS) phase. Lastly, MACO addresses the issue of unsatisfactory convergence by introducing a multipath-based fluid model to regulate cwnd on multiple subflows. Simulation results show that MACO reduces the completion time by 64.2% and achieves three times higher throughput against benchmark algorithms.

Chapter 4

QoS-driven Contextual MAB for Multipath Video Streaming in Mobile Networks

After the two crucial designs for MPQUIC, scheduling and congestion control, have been finished in the above 2 chapters, in this chapter, we shift our interests onto promising applications in 6G networks, i.e., video streaming applications.

4.1 Introduction

We have witnessed a proliferation of video streaming applications on mobile devices. The use of mobile phones for video conferencing is convenient for people outside of their offices. By accessing real-time video from roadside units, self-driving vehicles can be aware of what is happening behind corners and avoid dangerous blind spots.

To manage network dynamics while maintaining a good user experience, various techniques such as adaptive bit rate (ABR) and scalable video coding (SVC) are employed. Although SVC introduces inherent computational complexity, it holds great promise for handling more advanced video formats, such as 360-degree videos, with finer granularity and greater efficiency.

SVC encodes video once and decodes the bitstream multiple times with different resolutions, frame rates, and video qualities. It enables video applications to demand high reliability in delivering the most significant video data within tight deadlines to avoid playback interruptions. Once the non-interruption condition is met, higher

throughput is preferable for better video quality. However, using single-path transmission is quite hard to provide high-quality video services for high-mobility users due to the dynamic channel quality of the access link [143, 159, 166].

Consequently, multipath video streaming using multiple access links has been extensively investigated in recent years. MPDASH [61] introduces a deadline-aware multipath scheduler into Dynamic adaptive streaming over HTTP (DASH) [128] and takes user preferences into account. However, network states are time-varying especially when mobility is introduced, so MPDASH suffers from performance degradation in dynamic systems. Deep Reinforcement Learning (DRL) is a powerful tool to foresee the variations of network states, allowing for a proactive action to meet QoS demands [57, 94, 148, 169]. However, DRL-based techniques increase the algorithm's reaction time dramatically when the ongoing traffic deviates from the training traffic, so they require a large amount of training data and suffer slow convergence [155]. Hence, Peekaboo [139] was proposed, in which a lightweight Reinforcement Learning (RL) variant, contextual Multi-Armed Bandit (MAB) is employed to learn dynamic characteristics of the heterogeneous paths when scheduling packets using multipath QUIC (MPQUIC).

In addition to learning-based multipath scheduling, when Forward Error Correction (FEC) is used in conjunction with multipath transmission, the on-time packet delivery reliability can be substantially enhanced [24, 28, 84, 143]. ADMIT [143] integrates FEC coding and rate allocation over multiple paths to maximize mobile video quality using multipath TCP (MPTCP). In order to achieve QoS guarantee rather than merely improving QoS, LEAP [24] splits data into multiple subflows according to the data flow's QoS requirements, as well as utilizing cross-path FEC coding to trade off between throughput, delay deadline, and reliability.

However, in mobile environments, the existing learning-based or FEC-combined multipath solutions encounter new challenges. Peekaboo faces challenges in effectively dealing with the highly dynamic end-to-end (E2E) path state (context) in mobile environments, which means the dimension of context keeps increasing as time passes. Therefore, the dependencies between context and QoS performance may take considerable time to discover. Both Peekaboo and LEAP fail to fully explore the diversity of access networks and do not proactively switch to a better one during user movements, leading to significant performance degradation.

Based on the principle that the end system can take the input from the network to optimize the configuration of the transport protocol [15], in this chapter, we develop

a QoS-driven Contextual MAB (QC-MAB) framework for MPQUIC to support video streaming in mobile networks. The main contributions are three-fold.

First, we formulate the access link selection and FEC configuration with the goal of statistical QoS guarantee for video streaming as a contextual bandit problem, considering the impact of mobility. In our approach, we design the context space to include both access link features and E2E path conditions, which enables us to gain insights into the effects of access network dynamics (e.g., caused by user mobility) and backbone network dynamics (e.g., caused by network mobility) on QoS. Within the QC-MAB framework, each arm consists of two actions: access network selection and FEC configuration. Proactive network selection plays a crucial role in mitigating the negative impact of mobility on QoS, and the packet-level FEC configuration can effectively balance the delay, reliability, and goodput. Furthermore, the design of rewards is shaped by the specific features of video streaming, that is, ensuring the on-time delivery of base layer packets takes top priority, followed by the objective of achieving higher goodput.

Second, to improve learning efficiency, we propose a context refinement strategy consisting of two steps: context dimension reduction and context noise removal. By finding correlations among different features observed by the agent, we integrate multiple correlated features into one to reduce the context dimension, thereby reducing the learning time for determining the reward given the context. In addition, we utilize the Adaptive Kalman Filter (AKF) and exponentially weighted moving average (EWMA) approaches to remove the noise of the observed features so that the best arms can be found quickly.

Lastly, we implemented QC-MAB and the benchmark algorithms for MPQUIC in network simulator 3 (ns-3). On the basis of the implementation, we run the video data extracted from a real movie to validate the performance of our proposal in terms of video interruption ratio (IR), mean video quality, E2E delay for all BL packets, and the aggregated goodput. The experimental results demonstrate that QC-MAB outperforms the benchmark algorithms to achieve up to ten times lower IR and three times higher goodput in highly dynamic mobile environments.

4.2 Related Work

The development of video bitrate adaptation and multipath transmission has greatly pushed forward video streaming.

Video rate adaptation. DASH [128] is agnostic of the video codec and can use any codec including H.264/SVC [144] and H.265 [2]. It provides the opportunity to adjust the video bitrate during playback for high QoE. With SVC, we can encode video once and decode the bitstream multiple times with different resolutions, frame rates, and video quality [14,177], so the encoding time and server storage space can be saved, which is particularly important for live or high-quality stored video streaming. There are also approaches for optimizing fixed QoE objectives based on playback statistics such as buffer occupancy [64], or for altering QoE goals based on control theory [168] or reinforcement learning [94, 172]. Recently, [92] designed a decision-tree-based throughput predictor, named Lumos, to select the bitrate of video chunks based on the network capacity. However, these approaches focus on scenarios with single-path transmission and are sensitive to network uncertainties.

Multipath transmission. Multipath transport protocols, such as multipath TCP (MPTCP) [164] and MPQUIC [32] have many potentials, such as bandwidth aggregation, high-reliability provisioning, seamless handover, etc. Various multipath schedulers [36, 112, 132, 133, 164] based on MPTCP or MPQUIC are proposed for different scenarios and objectives. However, they do not consider the video QoS requirements in their design.

The recent efforts for QoS provisioning with multipath delivery can be divided into: 1) estimation model-based, e.g., [58,61,141,179], 2) learning-based policies, e.g., [10, 57, 139, 148, 169], and 3) FEC-combined multipath schemes, e.g., [24, 28, 84, 143].

CMT-DA [141] utilizes the path status estimation to adjust sending rate across multiple paths to minimize the E2E video distortion. MPDASH [61] takes user preferences into account, scheduling packets over multiple paths following the throughput estimation to meet the user's deadline or priority request. Both DEMS [58] and DAMS [179] estimate RTT to schedule the sending order of video blocks with the consideration of the block's deadline.

However, the above solutions are sensitive to sudden network condition changes which may occur frequently in mobile environments, so the learning-based approaches have been investigated. [148] formulated the video streaming process over multiple links as a reinforcement learning (RL) task to meet QoS goals in heterogeneous networks. ReLeS [169] formulated the multipath scheduling problem as an RL task with the consideration of QoS characteristics under various network scenarios. PERM [57] employed an actor-critic network to optimize the multi-path packet scheduling and bitrate adaptation simultaneously. Nevertheless, RL/DRL-based techniques increase

the algorithm's reaction time dramatically [155]. Therefore, the lightweight contextual multi-armed bandit (CMAB) [139] framework is more desirable for applications with stringent QoS concerns. Alzadjali et al. [10] proposed an online MPTCP path manager based on the CMAB to help choose the optimal primary path connection that maximizes throughput and minimizes delay and packet loss. Peekaboo [139] is a novel CMAB-based MPQUIC scheduler to cope with dynamically changed channel conditions of the heterogeneous paths.

On the other hand, in the context of multipath, FEC coding can offer a great chance to balance the tradeoff among QoS metrics. For instance, a Reed-Solomon (RS) code is employed in ADMIT [143], a rateless Raptor code in FMTCP [28], and a systematic random linear code in SC-MPTCP [84]. These works demonstrated that FEC is well suited to maximizing goodput without violating delay requirements. The closest work to our focus on QoS guarantee is LEAP [24] which uses RS code in conjunction with adaptive congestion control over multiple paths to meet the QoS requirements in terms of throughput, delay, and reliability.

In a nutshell, all solutions above have direct or indirect positive impacts on the QoS performance of video streaming with multipath delivery. Yet none of them takes into account the challenges and opportunities in mobile environments.

Mobility in 6G is ubiquitous and diverse. Not only end users but also backbones (e.g., satellite networks) and access networks (e.g., UAV) could be mobile, which leads to the fast-changing traffic density of each access network and the highly dynamic congestion level of core networks. Both the estimation model-based and learning-based solutions lose accuracy if they fail to consider the dynamics caused by mobility.

The access networks visible to an end host are changing all the time along with movements, which implicitly offers a good opportunity to meet stringent QoS requirements if the end host proactively switches to the access networks with better characteristics.

Even though MMSPEED [40] and [101] focused on QoS guarantee with multipath delivery in mobile environments, they differ fundamentally from our work because they targeted sensing data exchange in wireless sensor networks, using the link-layer approach rather than the transport-layer one.

Motivated by mobility challenges, in this chapter, we investigate the impact of mobility on QoS performance for video streaming over MPQUIC.

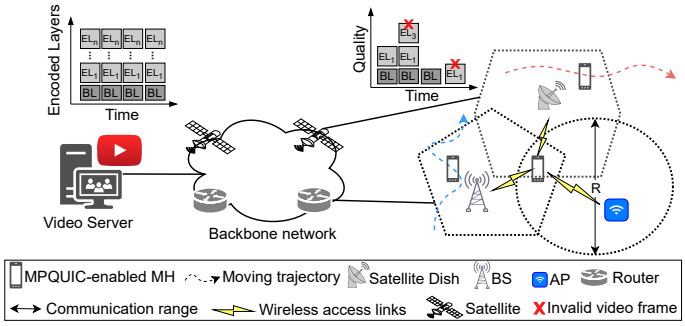


Figure 4.1: Overview of multipath video transmission in mobile networks.

4.3 System Model

4.3.1 Wireless Access Network Model

We consider a scenario where $\mathcal{M}$ access networks are densely distributed over a certain area, as shown in Fig. 4.1. Different types of access points, e.g., LTE/5G base station (BS), WiFi access point (AP), and UAV/satellite AP, have different communication channels associated with different data rates, communication ranges, etc. For simplicity, we use AP to represent all kinds of access points hereafter.

An MPQUIC-enabled mobile host (MH) is requesting video service from the video service provider through multiple access networks simultaneously. MH moves in the Random Walk Mobility Model (RWMM) manner, i.e., moves along a direction $\theta \in [0, 2\pi)$ with a constant velocity within a short time slot. At the end of each time slot, the direction is reshuffled, independent from slot to slot and node to node. If MH moves out of the communication range of the currently serving AP, known as a service outage, a handoff occurs so that the MH can be served by another AP.

4.3.2 Multipath Transmission Model

A library $\mathcal{F} = \{1, \ldots, f, \ldots, F\}$ of video frames is stored in the video server. With certain coding techniques, such as SVC, each frame f is encoded in several layers to provide different qualities. The example in Fig. 4.1 shows the layered coding: base layer (BL) and enhancement layers (ELs). The BL contains the low-resolution information of each frame and renders the video with the minimum quality, and each EL is to provide higher video quality on the top of the lower layers.

The encoded F frames contain a set of packets with distinct significance. Let

$\mathcal{K} = \{1, \ldots, k, \ldots, K\}$ denote the packet sequence number for all F frames. Denote by L_k the layer type associated with packet $k \in \mathcal{K}$. Specifically, $L_k = 0$ if k carries the BL data while $L_k = 1$ if k carries the EL data. Given that the MH is equipped with P interfaces, by which P MPQUIC connections (a.k.a. subflows or E2E paths) can be established between the video server and the MH. Let $\mathcal{P} = \{1, \ldots, p, \ldots, P\}$ be a set of available paths. Each path p independently maintains several time-varying conditions: $w_p(t)$, $RTT_p(t)$ and $\epsilon_p(t)$, which refer to the congestion window (cwnd), round-trip-time (RTT) and packet loss rate at time t, respectively. For simplicity, we ignore the notation t hereafter. With a certain scheduling policy, a portion of packets will be allocated into each path until all path windows are fully filled. A new round of data allocation will begin over the paths receiving a new acknowledgment. This scheduling process repeats until all K packets are sent.

4.3.3 Multipath FEC Model

Using FEC, a block containing n packets is encoded into a longer block with $\hat{n}$ packet, where $n/\hat{n}$ is the coding rate. It is guaranteed that any n of the $\hat{n}$ encoded packets can recover the original data in the n packets. Therefore, if the number of packet losses is no larger than $\hat{n} - n$, the receiver can successfully decode the data without retransmission. Given an MPQUIC path p with a packet loss probability ϵ_p, and let l_p be the decoding failure probability for the block b from path p, we have [28]

$$l_p = \sum_{j=\hat{n}-n+1}^{\hat{n}} \left(\begin{array}{c} \hat{n} \\ j \end{array} \right) \epsilon_p^j \left(1 - \epsilon_p\right)^{\hat{n}-j}. \tag{4.1}$$

4.3.4 Video Quality Model

The minimum acceptable video quality is free of playback interruption [148]. Beyond that, the higher the video resolution, the better the user experience.

To meet minimum video quality requirements, it is crucial for video clients to receive BL packets on time, ensuring that each packet arrives within a specified time constraint referred to as ΔT. However, in highly dynamic systems, offering a deterministic delay guarantee can be a challenging task. Hence, we delve into the concept of a statistical delay guarantee, which is quantified by the reliability Γ. This reliability signifies the probability of successfully achieving the on-time guarantee for BL packets. Furthermore, it is desirable to achieve higher goodput Φ in order to facilitate

the delivery of a larger number of EL packets.

Therefore, the video quality depends on E2E delay, reliability, and goodput.

E2E Delay

Assuming that packet k is scheduled into a path $p^* \in \mathcal{P}$, the application-level E2E latency of packet k denoted as d_k^E can be divided into two main components: network delay (d_k^N) and transport-layer delay (d_k^T). In other words, we have $d_k^E = d_k^N + d_k^T$.

d_k^N includes all of the transmission delay, propagation delay, processing delay and queuing delay along the path when the packet traverses from the source to the destination, referred to as one-way delay (OWD) τ_{p^*}, which is approximately half of the RTT_{p^*}. Note that the transmission delay of the wireless uplinks or downlinks here is uncertain due to user and/or network mobility. As video data from the server to the mobile user go through downlinks, we focus on downlink performance. Let C_{p^*} be the transmission rate of wireless downlinks, we have

$$C_{p^*} = \xi W_m \log_2(1 + SINR_m), \tag{4.2}$$

where $SINR_m$ stands for the received signal to interference and noise ratio of the m-th AP, W_m is the corresponding channel bandwidth, and $\xi \in (0, 1)$ is an efficiency coefficient of the communication system, depending on several factors, e.g., hardware and software design. Therefore, the transmission delay of packet k over the wireless downlinks between the m-th AP and the MH is x_k/C_{p^*} where x_k is the size of packet k.

d_k^T consists of retransmission delay ζ_k and reordering delay δ_k. ζ_k is caused by packet loss events, depending on the congestion control phase, as discussed in [164]. δ_k is due to the out-of-order (OFO) situation, so it has an upper bound based on the maximum difference between the one-way packet delay on routes, that is, $0 \leq \delta_k \leq \max_{p \in \mathcal{P}} \tau_p - \tau_{p^*}$.

Reliability

Considering the feature of SVC video traffic, reliability is defined as the probability of on-time delivering of BL packets, denoted by γ_k. γ_k can be expressed as $\gamma_k = \mathbb{P}\{d_k^E \leq \Delta T\}$.

Without FEC applied, the reliability of BL packets largely depends on the path

error rate including congestion loss and wireless link errors. As long as the time constraint ΔT has not expired, each lost packet can be retransmitted in the subsequent round trips to improve its successful delivery probability. Assuming that packet losses are independent, the process can be represented by a Bernoulli trial. Let $V \in \mathbb{Z}_0^+ = \{0, 1, \cdots\}$ be the maximum retransmission times that can take place before ΔT expires, we have

$$\gamma_k = \sum_{v=0}^{V} \left(\epsilon_{p^*}\right)^v \left(1 - \epsilon_{p^*}\right).$$

Note $v = 0$ refers to the successful packet delivery in the initial transmission.

Similarly, when FEC is applied, γ_k depends on the decoding failure probability l_{p^*} as described in (4.1), and it can be estimated as:

$$\gamma_k = \sum_{v=0}^{V} \left(l_{p^*}\right)^v \left(1 - l_{p^*}\right).$$

Goodput

By definition, the aggregated goodput denoted by Φ stands for the number of useful information bits delivered by multiple paths to the destination per unit of time. Let $\mathcal{J}$ be a subset of $\mathcal{K}$, storing all received decoded packets from multiple paths within the time interval Δt, Φ is expressed as

$$\Phi = \frac{|\mathcal{J}| \cdot MSS \cdot 8}{\Delta t}. \tag{4.3}$$

Here $|\cdot|$ stands for the number of elements inside a set and MSS means the maximum segment size in bytes.

In the context of multipath transmission, the goodput perceived by the user is the summation of the goodput of each path. Denote by ϕ_p the goodput on path p, ϕ_p without FEC can be calculated as the product of the transmission rate and the fraction of packets successfully received, while ϕ_p with FEC can be calculated by subtracting the overhead due to the transmission of redundant packets from the total transmission rate.

Therefore, when FEC is disabled, Φ is estimated as

$$\Phi = \sum_{p \in \mathcal{P}} \phi_p = \sum_{p \in \mathcal{P}} \frac{w_p \cdot (1 - \epsilon_p)}{RTT_p}. \tag{4.4}$$

On the other hand, if FEC is enabled, we have

$$\Phi = \sum_{p \in \mathcal{P}} \phi_p = \sum_{p \in \mathcal{P}} \frac{w_p \cdot (1 - l_p) \cdot (n/\hat{n})}{RTT_p}. \tag{4.5}$$

4.3.5 Problem Formulation

Given the QoS requirements $<\Delta T, \Gamma>$ as described above, our main objective is to develop a policy π that efficiently selects an access network for E2E path. This policy should also dynamically enable or disable the FEC technique in order to meet the criteria for delay and reliability. Additionally, we aim to maximize the goodput while considering all these factors.

Given the QoS requirements $<\Delta T, \Gamma>$ as described above, our main objective is to determine a policy π that effectively selects an access network for each E2E path and dynamically enables or disables the FEC technique to fulfill the delay and reliability criteria, while simultaneously maximizing the goodput. This can be formulated mathematically as follows:

$$\begin{aligned} \max_{\pi} \quad & \Phi \\ \text{s.t.} \quad & \hat{\phi}_p \leq C_p, \quad \forall p \in \mathcal{P} \\ & \gamma_{k_b} \geq \Gamma, \quad k_b \in \{j | L_j = 0, j \in \mathcal{J} \subseteq \mathcal{K}\}, \end{aligned} \tag{4.6}$$

where $\hat{\phi}_p$ is the achieved throughput over path p. If no FEC is applied, it is equivalent to the goodput ϕ_p. Otherwise, $\hat{\phi}_p = \frac{\hat{n}}{n} \cdot \phi_p$.

In reality, there are many uncontrollable factors raised by mobility so it is hard to directly derive the optimal solution of the problem (4.6). For instance, the traffic load of each access point is highly dynamic and the distance from the user to AP is fast-changing, leading to the time-varying data rate C_p and packet error rate ϵ_p over access links. Also, the random congestion on the network side could trigger uncertain queuing delays. To cope with these uncertainties, one intuitive idea is to leverage learning-based approaches to learn the relationship between the network environments and final results from the historical data samples, and then apply the

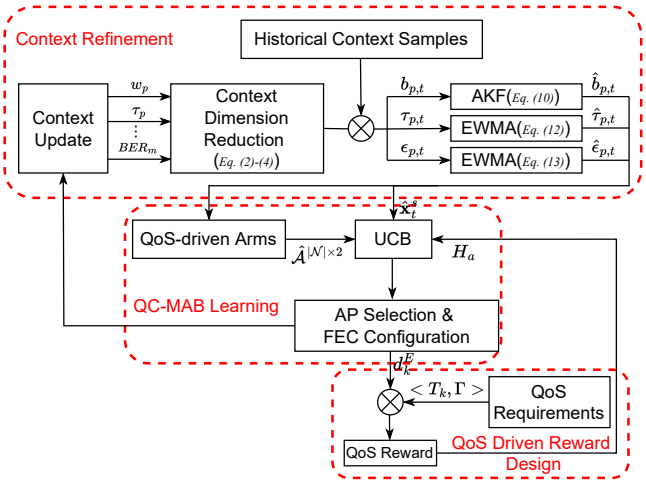


Figure 4.2: QC-MAB framework.

trained model to assist the decision-making.

However, video streaming has stringent delay requirements and a high volume of data, so control decisions must be made at ms-level. Many existing learning-based algorithms take a long time to converge in response to the abrupt changes in link conditions, so they are too slow or too costly to meet stringent QoS requirements. In the next section, we propose a QC-MAB framework to solve the QoS guarantee problem, with which the underlying relationship between context and QoS requirements can be discovered without a lengthy training process, thereby facilitating decision-making in highly dynamic systems.

4.4 QC-MAB for Multipath Video Streaming

As shown in Fig. 4.2, we present a QC-MAB framework for MPQUIC-based video streaming under uncertain network characteristics caused by the user and/or network mobility. First, three components of QC-MAB are designed: Context refinement, QoS-driven arms design, and QoS-driven rewards design. Then, a contextual bandit problem is formulated, and the corresponding solutions are discussed.

4.4.1 Context Refinement (CR)

The networking features that affect the QoS requirements are collectively referred to as context, known as state s. We use context and state interchangeably throughout

Table 4.1: Context space design

E2E path related context
w_p: the cwnd on path p
Ψ_p: bytes-in-flight on path p
τ_p: OWD of path p
ϵ_p: packet loss rate on path p

Wireless links related context
$(\hat{x}, \hat{y})$: MH's location
(x_m, y_m): the m-th AP's location
P_m: received power
σ: background noise
I_m: the interference
W_m: the available bandwidth
BER_m: the link bit error rate

this chapter. Each state s is defined by a vector of features $\mathbf{x}^s = \left(x^s_{E_1}, \cdots, x^s_{E_c}, x^s_{L_1}, \cdots, x^s_{L_d}\right)$ where $(x^s_{E_1}, \cdots, x^s_{E_c})$ represents the E2E path relevant features, and $(x^s_{L_1}, \cdots, x^s_{L_d})$ means features related to the last-mile wireless links. According to the analysis in the last section, 11 critical features related to QoS on path p are summarized in Table 4.1. In this case, $\mathbf{x}^s$ is an 11×1 vector.

However, with a large dimension of contexts, discovering the relationship between context and rewards takes a long time. To mitigate this issue, we try to reduce the dimension of $\mathbf{x}^s$ by finding correlations among features and then integrating several contextual features into one. Furthermore, most contexts are estimated or measured values based on instantaneous feedback from the network, which can be noisy due to sudden changes in network conditions. In order to reveal the intrinsic relationship between contexts and rewards, we need to filter out the noise within contexts. Context dimension reduction and noise removal are referred to as context refinement (CR).

Context Dimension Reduction

w_p and Ψ_p jointly reveal the congestion situation and bottleneck capacity C_p^N at the network side, denoted by $C_p^N := f(w_p, \Psi_p)$. On the other hand, $(\hat{x}, \hat{y})$, (x_m, y_m), P_m, σ_m, I_m and W_m define the achievable access link capacity C_p^A of path p as follows according to Eq. (4.2),

$$C_p^A = \xi W_m \log_2(1 + \frac{P_m(\sqrt{[\hat{x} - x_m]^2 + [\hat{y} - y_m]^2})^{-e}}{\sigma_m^2 + I_m}), \tag{4.7}$$

where e is the path loss exponent.

Therefore, we define a new context b_p, representing the bottleneck bandwidth of

path p. This context is derived by integrating C_p^N and C_p^A in the following manner:

$$b_p = \min\{C_p^N, C_p^A\}. \tag{4.8}$$

In addition, the relationship between the packet loss rate ϵ_p and the link bit error rate BER_m can be expressed with [143]

$$\epsilon_p = \rho + (1-\rho)\left(\sum_{i=a}^{Y}\binom{Y}{i}(BER_m)^i(1-BER_m)^{L-i}\right), \tag{4.9}$$

where ρ stands for the loss rate caused by network congestion, Y is the length of each packet in bits, and a is the minimum number of bits in error that will lead to packet error due to limited error coding capacity in the physical and link layers.

Under the assumption that every ΔT the MH displays a new frame and runs QC-MAB[1], the time at the receiver side can be slotted by $\mathcal{T} = \{1, 2, \ldots, t, \ldots\}$ where t means the t-th display event. Since the correlation among each context can be realized using (4.7)–(4.9), the context space observed by the MH is reduced to a set of three parameters at slot t, i.e., $\mathbf{x}_t^s = (b_{p,t}, \tau_{p,t}, \epsilon_{p,t})$. $b_{p,t}$ and $\tau_{p,t}$ can be directly measured by the learning agent at the MH. In mobile environments, the mobile access link BER far outweighs congestion losses within backbone networks [142]. Hence, $\epsilon_{p,t}$ is roughly equal to BER which can be also directly estimated at the MH side. Meanwhile, the sender will periodically notify the receiver of the estimated loss event rate by measuring the packet loss event rate. Therefore, in the case where backbone networks might be the bottleneck and the congestion loss dominates $\epsilon_{p,t}$, the receiver can use the sender's loss event calculation to estimate $\epsilon_{p,t}$.

Context Noise Removal

In reality, a common approach to obtain $b_{p,t}$ feedback is to monitor the timestamps of the received consecutive packets to estimate bottleneck bandwidth. Assuming S_t and S_{t-1} are the total amounts of data received by the MH at successive measurement

[1]Compared to the video server, the MH has easier access to collect the necessary features locally. This is why we choose to run the QC-MAB at the MH side. In our design, the QC-MAB handles access network selection for the MH and FEC configuration for the video server. By running QC-MAB at the MH, it becomes feasible for the MH to send an acknowledgment (ACK) along with the decision regarding FEC configuration during the subsequent round of transmission back to the video server.

time epochs t and $t-1$, $b_{p,t}$ is measured with

$$b_{p,t} = \frac{S_t - S_{t-1}}{\Delta t}. \tag{4.10}$$

However, the measured $b_{p,t}$ in this manner is likely biased due to many reasons, e.g., the sending window is too small during slow startup to match the actual network capacity, so the instantaneous $b_{p,t}$ is much lower than it should be. Therefore, we utilize an Adaptive Kalman Filter (AKF) [75] to remove the noise within $b_{p,t}$. Given limited measured samples, AKF can still quickly narrow down to the truth by taking a few of those inputs and beyond by understanding the variation or the uncertainty of those inputs. The standard Kalman filter goes through two stages to acquire the final estimate as follows.

The first stage is *prediction*, in which the estimated capacity (b_{p,t_e}) of path p and the estimation errors $Z_{p,t}$ are updated as

$$b^-_{p,t_e} = b^+_{p,t-1} + \omega_{p,t}, \tag{4.11}$$

$$Z^-_{p,t} = Z^+_{p,t-1} + Q_p, \tag{4.12}$$

where the superscript "+" indicates that the estimate is a posterior, and "-" indicates a prior estimate. $\omega_{p,t}$ stands for white Gaussian noise with zero mean and variance Q_p, which depends on network uncertainties such as user mobility.

The second stage is *correction*. The measured network capacity is assumed disturbed by a white Gaussian noise $v_{p,t}$ with zero mean and variance R_p, such that

$$y_{p,t_m} = b_{p,t_m} + v_{p,t}. \tag{4.13}$$

Here b_{p,t_m} is the instantaneously measured value at time t, using (4.10). Then the critical Kalman gain is derived by

$$K_{p,t} = Z^-_{p,t}\left(Z^-_{p,t} + R_p\right)^{-1}, \tag{4.14}$$

with which we can determine whether the predicted value or measured value is closer to the true value. Let $\hat{b}_{p,t}$ and $\hat{Z}_{p,t}$ be the final correction for the predictions of b^-_{p,t_e}

and $Z^-_{p,t}$, we have

$$\hat{b}_{p,t} = b^-_{p,t_e} + K_{p,t}\left(y_{p,t_m} - b^-_{p,t_e}\right), \tag{4.15}$$

$$\hat{Z}_{p,t} = Z^-_{p,t} - K_{p,t}Z^-_{p,t}. \tag{4.16}$$

The corrected $\hat{b}_{p,t}$ and $\hat{Z}_{p,t}$ will be the posterior states that are involved in the future rounds of corrections. In this way, the estimation error diminishes over time, and so does the noise in ${b_{p,t}}^2$.

Finally, we employ the exponentially weighted moving average (EWMA) approach to correct measured $\tau_{p,t}$ and $\epsilon_{p,t}$ as follows:

$$\hat{\tau}_{p,t} = E_1(t) = \alpha \cdot \tau_{p,t} + (1-\alpha) \cdot E_1(t-1), \tag{4.17}$$

$$\hat{\epsilon}_{p,t} = E_2(t) = \beta \cdot \epsilon_{p,t} + (1-\beta) \cdot E_2(t-1). \tag{4.18}$$

The settings of α and β are discussed in Section 4.5.3.

Then, the final context $\hat{\mathbf{x}}^s_t = \{\hat{b}_{p,t}, \hat{\tau}_{p,t}, \hat{\epsilon}_{p,t}\}$ are obtained.

4.4.2 QoS-Driven Arms Design

To find the optimal policy π as presented in problem (4.6), we design arms, a.k.a., actions, from two aspects: access network selection and FEC configuration. The former is for the MH and the latter one is for the video server. As a result, each arm a is associated with a vector of behaviors $\mathbf{x}^a = (x^a_1, x^a_2)$ where x^a_1 is related to the network selection and x^a_2 is associated with FEC option. Given there are $\mathcal{M}$ access networks to be selected, and two options of whether enabling FEC, the set of arm vectors, $\mathcal{A}$, includes $|\mathcal{M}| \times 2$ combinations. Similarly, the larger number of arms, the longer it will take to explore the arms that are never or rarely chosen. Therefore, the arm set is refined by filtering out those arms that do not meet specific criteria. For example, for those paths whose $\hat{\tau}_{p,t}$ is greater than ΔT, we can skip the choice of its associated access network. Therefore, the new arms space $\hat{\mathcal{A}}_t^{|\mathcal{N}|\times 2}$ is yielding ($|\mathcal{N}| \leq |\mathcal{M}|$).

[2]To be able to use the Kalman filter, it is necessary to have the knowledge of the co-variances of the prediction and measurement noise, i.e., Q_p and R_p. However, they are unknown a priori in our case, and they may vary over time as the network environment changes. To address this issue, we can use Least Squares method to estimate them from the innovation signal's autocorrelation as in [115].

$$r|(s,a) = \begin{cases} \left(\sum_{i=1}^{n_b} \mathbb{1}\{d_{b_i}^E \leq \Delta T\} + \sum_{j=1}^{n_e} \mathbb{1}\{d_{e_j}^E \leq \Delta T\}\right)/N, & \text{if} \frac{\sum_{i=1}^{n_b} \mathbb{1}\{d_{b_i}^E \leq \Delta T\}}{n_b} \geq \Gamma \\ \left(\sum_{i=1}^{n_b} \mathbb{1}\{d_{b_i}^E \leq \Delta T\} + \eta \sum_{j=1}^{n_e} \mathbb{1}\{d_{e_j}^E \leq \Delta T\}\right)/N, & \text{otherwise,} \end{cases} \tag{4.19}$$

4.4.3 QoS-Driven Rewards Design

In the QC-MAB framework, the reward function is highly related to the QoS. Also, the posterior reward r is parameterized by state s and action a, denoted by $r|(s,a)$.

In the event that an arm is chosen with a satisfactory QoS value returned, we respond with a positive reward to indicate that the arm is a good candidate to be exploited in future trials. Otherwise, we return a negative/zero reward (penalty). The detailed reward function is designed in (5.13). It is worth noting that the reward is designed for a whole frame instead of a single packet due to the strong correlation between packets within the same video frame, in other words, a reward will be recalculated upon each frame completion.

Suppose each video frame f contains N of packets, which are categorized into n_b BL packets and n_e EL packets. Therefore, the on-time delivery ratio for BL packets within frame f can be calculated by $\gamma_f = \frac{\sum_{i=1}^{n_b} \mathbb{1}\{d_{b_i}^E \leq \Delta T\}}{n_b}, b_i \in \mathcal{K}$. Note $\mathbb{1}\{\cdot\}$ refers to indicator function. Provided that the video client's reliability preference is Γ for BL. If γ_f is greater than Γ, we give a weighted bonus to packets that meet their individual deadline. In this way, the learning agent will strive to guarantee the on-time delivery of BL packets while sending as many EL packets as possible.

Otherwise, if the on-time reliability requirement is not satisfied, the reward for EL packets will be discounted by a factor η, motivating the learning agent to choose some actions (e.g., enable FEC) to reduce the E2E delay for BL packets. Here $\eta = \frac{1}{n_b - \sum_{i=1}^{n_b} \mathbb{1}\{d_{b_i}^E \leq \Delta T\}}$, implying that the more expired BL packets, the less useful of EL packets.

4.4.4 Contextual Bandit Problem

With the QC-MAB framework, the QoS guarantee issue can be formulated into a cumulative rewards maximization problem, known as the contextual bandit problem formulated in **P1**. **P1** indicates that our goal is to find a good policy π that determines

the action a_t given the state s_t to maximize the expected cumulative rewards.

$$\textbf{P1:} \max_{\pi: s_t \to a_t} \mathbb{E}\left[\sum_{t=1}^{T} [r_t|(s_t, a_t)]\right]. \tag{4.20}$$

There are many existing algorithms to solve the MAB problems. The Upper Confidence Bound (UCB) [13] is a lightweight yet efficient bandit algorithm that is proven to produce asymptotically optimal regret performance. Therefore, we incorporate the UCB algorithm into QC-MAB to solve **P1**. Every ΔT the MH displays a new frame, it activates QC-MAB to calculate rewards and proactively select APs for connection. Afterward, the server is notified of the FEC enabling decision along with an acknowledgment, and then it configures FEC accordingly for the next round of data transmission.

4.4.5 UCB Solution

Denote by $\mathcal{S}$ the set of context vectors $\hat{\mathbf{x}}_t^s$ over time. Considering that context features in mobile environments are changing over time, the dimension of $\mathcal{S}$ may increase unbounded as time passes, which poses challenges to learning the relationship between contexts and rewards.

To maintain the context space in an acceptable finite dimension, we create a partition $\mathcal{P}_T$ on the context space $\mathcal{S}$, which splits $\mathcal{S}$ into $(h_T)^D$ sets based on the time horizon T. Here, D is the dimension of a context vector which is 3 in our case, and h_T is a parameter to be designed to determine the number of sets in $\mathcal{P}_T$. Fig. 4.3 gives an example of a 2-dimensional context partition, in which $h_T = 3$ since each feature is partitioned by two thresholds: 10 Mbps and 50 Mbps for the $\hat{b}_{p,t}$, and 10 ms and 50 ms for the $\hat{\tau}_{p,t}$. Here h_T manages the partition granularity, and it can be flexibly customized.

For each 3-dimensional vector $\hat{\mathbf{x}}_t^s \in \mathcal{S}$, it can be mapped to a tuple $q_t \in \mathcal{P}_T$. At the very beginning ($t = 1$), the context is mapped to tuple q_1 and the reward distribution on each arm is unknown yet, so we assume the mean reward $\bar{r}_{a_1|q_1}$ for all arm $a_1 \in \hat{\mathcal{A}}_1$ is identical. The learning agent starts with an arm a_1^* randomly.

Upon a new frame display event at time $t (t \geq 2)$, the learning agent first calculates the conditional reward $r_{a_{t-1}^*|q_{t-1}}(t)$ for the chosen arm in the last time slot using Eq. (5.13). According the Hoeffding's inequality, [62] gives an upper bound of the

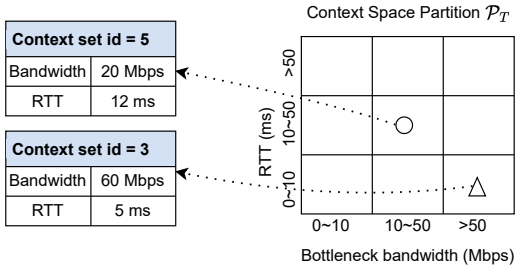


Figure 4.3: Illustration of context space partition and the context set ID identification.

estimated reward $r_{a^*_{t-1}|q_{t-1}}(t)$ as follows,

$$H_{a^*_{t-1}|q_{t-1}} = \frac{R_{a^*_{t-1}|q_{t-1}}}{N_{a^*_{t-1}|q_{t-1}}} + \sqrt{\frac{2\ln t}{N_{a^*_{t-1}|q_{t-1}}}}, \tag{4.21}$$

where $N_{a^*_{t-1}|q_{t-1}}$ stands for the number of times the arm a^* has been selected by time slot $t-1$, and $R_{a^*_{t-1}|q_{t-1}}$ means the total reward of choosing a^* over the past $t-1$ time slots. $R_{a^*_{t-1}|q_{t-1}}$ is calculated by

$$R_{a^*_{t-1}|q_{t-1}} = \sum_{x\in\{y|a_y=a^*_{t-1}\cap q_y=q_{t-1}, y\leq t-1\}} r_{a_x|q_x} \tag{4.22}$$

To determine the action a^*_t given the new context $\hat{\mathbf{x}}^s_t \rightarrow q_t$, the UCB algorithm will compare the reward of the chosen arms with that of non-chosen arms to solve the exploitation vs. exploration dilemma. During the exploitation phase, the UCB algorithm favors actions that have shown higher upper bound $H_{a|q}$ of the estimated reward. By exploiting the currently known best actions, the algorithm aims to maximize the expected reward and make optimal decisions based on the existing knowledge. On the other hand, during the exploration phase, if an action has not been tried very often, or not at all, then $N_{a|q}$ will be small. Consequently the second term of (4.21) will be large, making this action more likely to be selected. This allows the algorithm to collect more data and gain a better understanding of the potential rewards associated with different actions.

With UCB, a^*_t is selected according to

$$a^*_t = \arg\max_{a\in\cap\hat{A}_t}\{H_{a|q_t}\}. \tag{4.23}$$

4.5 Evaluations

In this section, we examine the performance of QC-MAB by comparing it to the two state-of-the-art algorithms below.

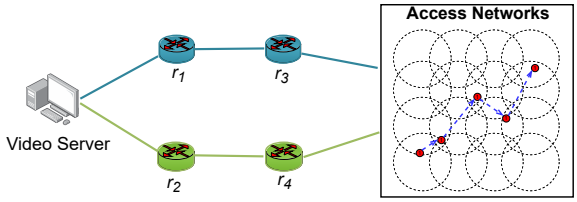
- Peekaboo [139]: it is a novel CMAB-based MPQUIC scheduler. It strives for learning the dynamic characteristics of the heterogeneous path including path RTT and loss rate, and then makes decisions about whether to schedule packets into the path with undesirable conditions immediately or wait for the availability of the path with good conditions.
- LEAP [24]: similar to us, LEAP considers three QoS metrics, i.e., delay, reliability and goodput. It employs FEC to trade off among the three metrics, and uses a bandwidth-aware multipath congestion control algorithm to avoid overshooting issues.

4.5.1 Evaluation Methodology

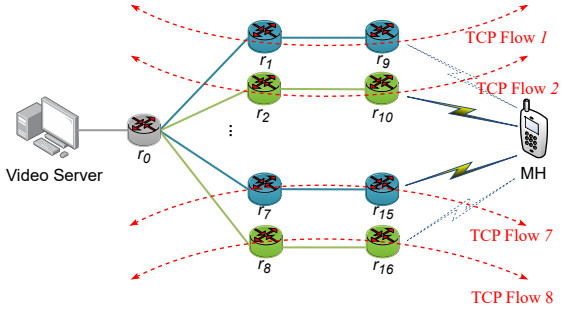
Testbed Construction

To start, we extended the existing QUIC module in network simulator 3 (ns-3) [6] to support MPQUIC, following the guidelines specified in the Internet Engineering Task Force (IETF) drafts [88] on MPQUIC. The publicly available version of our MPQUIC source code can be accessed at [3]. Using this platform, we implemented our proposed solution and incorporated two existing algorithms for comparative analysis. Additionally, we referred to the DASH source code at [1] and made modifications to the MPQUIC source code to enable DASH to operate over MPQUIC.

In order to replicate network bottleneck scenarios that can occur at any point along the E2E path, including access networks and backbone networks, we created two network topologies, as depicted in Fig. 4.4. In Fig. 4.4(a), we evaluated a situation where the end user moves across various access networks with different conditions, some of which may become bottlenecks. Fig. 4.4(b) represents two cases: 1) when the end user remains stationary while multiple background flows compete for backbone resources, causing significant variations in data rate, link delay, and packet drop rate, indicating the backbone network as a potential bottleneck, and 2) when the end user remains stationary and there are no background flows along the E2E paths.



(a) Experimental topology 1



(b) Experimental topology 2

Figure 4.4: Experimental topologies.

Both Fig. 4.4(a) and Fig. 4.4(b) depict that an MH is equipped with a WiFi interface and a 5G interface, allowing for the establishment of two subflows between the MH and the video server. In Fig. 4.4(a), the backbone network comprises a 2×2 node configuration, with circular markers representing the communication range of the AP nodes. We employed the *WiFiHelper* or *NrHelper* tools to configure the access links between the MH and the APs, while the *PointToPointHelper* tool was used to set up the wired links. In Fig. 4.4(b), the backbone network consists of an 8×2 node configuration, with the rightmost column of nodes acting as APs to connect the MH. The solid flash between an AP and the MH represents the currently utilized access link, while the dashed one represents a candidate link.

Video Application Settings

In each simulation, we employed the Dynamic Adaptive Streaming over HTTP (DASH) protocol at the mobile host (MH) side to retrieve video segments from the video server. The video segments used in our simulations were provided by the authors of [103] and were extracted from the movie "Sintel." This dataset comprises 84 video segments, with video bitrates ranging from 0.088 Mbps to 20.5 Mbps, encompassing eight quality increments. The selection of bitrates for different representations was designed to achieve a roughly linear increase in video quality, as measured by the Peak

Signal-to-Noise Ratio (PSNR). Therefore, we can utilize the representation index as an indicator of video quality.

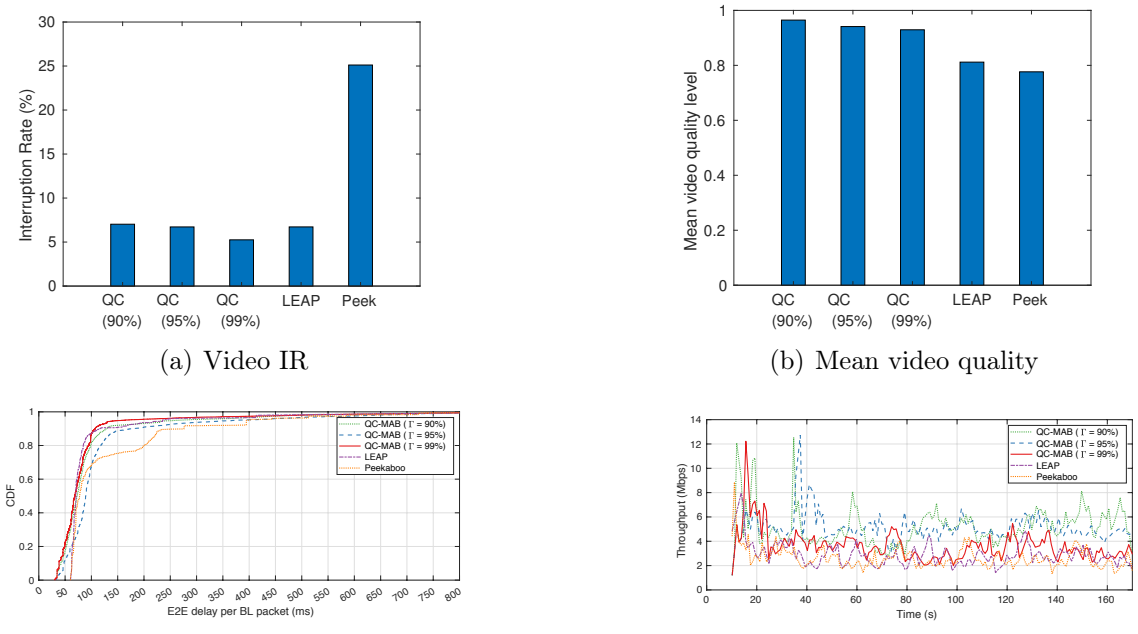
In practical video data protection scenarios, FEC codes are often employed at the bit level, group of pictures (GoP) level, or frame level [171]. In our design, we adopt a frame-level FEC approach at the MPQUIC layer to allocate FEC redundancy for each coding layer within the frame. In accordance with the settings in [171], our SVC configuration includes three layers (one base layer and two enhancement layers), with a FEC redundancy of 30% for each layer. To ensure acceptable video quality, we impose a constraint on the per-packet elapsed delay, limiting it to less than 150 ms. This constraint aligns with previous studies [127, 140] that suggest a one-way delay not exceeding 150 ms to achieve excellent video quality. Consequently, the video player displays one frame every ΔT seconds, where ΔT is set to 0.15. If the base layer of a frame fails to decode successfully for display, a playback interruption occurs. Let n_0 represent the number of playback interruptions, and n_t denote the total number of frames. The interruption ratio (IR) is then defined as IR $= n_0/n_t$, which serves as a critical measure of video user experience.

For all scenarios examined, we measure four key performance metrics: interruption ratio (IR), mean video quality, E2E delay for all BL packets, and aggregated goodput. These metrics enable us to evaluate the impact of our proposed solution and compare it with existing algorithms in terms of video playback experience, video quality, delay, and overall network throughput.

4.5.2 User Movement in Ultra-Dense Networks

Experimental Settings

In this scenario, we utilize the experimental topology depicted in Fig. 4.4(a). Inspired by previous studies [8, 167], we configure the mobile access networks as follows: the APs are evenly distributed in a lattice grid topology, with neighboring APs placed at a distance of 20 meters. The MH initiates its movement from the lower-left corner and randomly moves in a specified direction at a velocity of 10 m/s. The transmission power level (P) of the APs is randomly selected within the range of 100 mW to 200 mW. The path loss exponent (e) is set to 4. We assume no interference and set the noise power (σ) to -180 dBm. The allocated bandwidth (W) of the APs to the MH ranges from 1 MHz to 2 MHz. Based on insights from prior studies [158], we set the MH's handoff threshold to -90 dBm. For the wired links represented by solid lines



(a) Video IR

(b) Mean video quality

(c) E2E delay CDF

(d) Aggregated goodput

Figure 4.5: QoS performance when the access networks or user is in motion.

in Fig. 4.4(a), we initialize the link data rate to 100 Mbps, link packet error rate to 0.01%, and the link delay to a random value ranging from 5 ms to 20 ms, aligning with the settings in [143].

Regarding the QC-MAB algorithm, we define several thresholds to partition the context space, as discussed in Section 4.4.5. Through a series of tests and based on the aforementioned settings, we set the bandwidth threshold to 5 Mbps and 10 Mbps, the OWD threshold to 90 ms and 150 ms, and the error rate threshold to 0.01% and 0.1%. It is important to note that the selection of partition thresholds is highly dependent on the specific scenario, and intelligently tuning these thresholds to adapt to various scenarios remains an open issue. We set the coefficient α in Equation (4.17) to $\frac{1}{8}$, as suggested in [69], and the coefficient β in Equation (4.18) to 0.6, as indicated in [135]. Both QC-MAB and LEAP utilize the QoS requirements as input, and for consistency with LEAP, we set the deadline ΔT of BL packets to 150 ms. The reliability requirement (Γ) for QC-MAB is set at three levels: 90%, 95%, and 99%, while LEAP maintains a reliability requirement of 99%.

IR and E2E Delay

Fig. 4.5 presents the QoS performance under the aforementioned settings. In terms of IR performance (Fig. 4.5(a)), we observe that Peekaboo exhibits an interruption ratio of more than 25%, while QC-MAB and LEAP maintain a ratio of 5%. The higher IR observed in Peekaboo is attributed to its focus on QoS improvement rather than QoS guarantee. Unlike QC-MAB and LEAP, Peekaboo does not penalize the system when BL packets fail to arrive on time, resulting in a higher IR. This observation is further supported by the results of E2E delay in Fig. 4.5(c).

Analyzing Fig. 4.5(c), we observe that QC-MAB with a larger Γ value achieves lower E2E delay, indicating that the actions taken by QC-MAB are indeed driven by QoS requirements. Similarly, LEAP also considers the stringent reliability requirements and employs congestion control and FEC strategies to meet those requirements. However, the probability of achieving an E2E delay below 150 ms is 91.1% for LEAP, whereas QC-MAB achieves 95.9% with a reliability requirement of $\Gamma = 99\%$.

When it comes to reliability, Peekaboo performs the worst at 75.8%. This can be attributed to the increasing BER as the MH moves away from connected APs. While LEAP faces a similar challenge, it utilizes FEC techniques to recover BL packets without retransmission, resulting in better E2E delay performance compared to Peekaboo. In QC-MAB, in addition to employing FEC techniques, the intelligent network selection strategy allows it to proactively explore other suitable APs based on wireless link contexts, effectively mitigating the negative impact of mobility.

Overall, the experimental results demonstrate that QC-MAB outperforms Peekaboo and achieves comparable performance to LEAP in terms of QoS metrics such as IR and E2E delay. The intelligent network selection and proactive AP switching capabilities of QC-MAB enable it to adapt to changing network conditions and ensure reliable and high-quality video streaming in ultra-dense network environments.

Mean Video Quality and Goodput

To achieve high video quality beyond playback smoothness, QC-MAB aims to explore better access networks that can transmit more EL packets without violating the delay requirement of BL packets. Fig. 4.5(b) illustrates the mean video quality results, showing that QC-MAB achieves higher mean quality compared to Peekaboo and LEAP. Additionally, there is a tradeoff between reliability and mean video quality in mobile scenarios, where higher reliability requirements lead to lower mean quality.

The video quality heavily depends on the aggregated goodput over multiple paths. Fig. 4.5(d) presents the changes in goodput over time for different algorithms in the context of mobility. QC-MAB achieves up to three times higher goodput than LEAP and Peekaboo when Γ is set to 90% and 95%. The goodput decreases when a higher reliability requirement, such as 99%, is imposed, aligning with the behavior of lower mean quality with higher reliability. However, even with a reliability requirement of 99%, QC-MAB still improves the goodput compared to LEAP and Peekaboo due to the exploration of diverse APs during movement.

Convergence Analysis

The results above demonstrate that the QC-MAB model can effectively handle network dynamics, thanks to its faster convergence rate. Fig. 4.6 shows the average reward values obtained every 500 packets to analyze the convergence performance. QC-MAB achieves the largest mean reward, while Peekaboo has the least. QC-MAB is about to converge at packet 10,000 (approximately 13.2 seconds), while Peekaboo converges at packet 11,000 (approximately 19.67 seconds). In contrast, LEAP, which is not MAB-based, does not exhibit a clear convergence trend.

The key component that improves the convergence rate in QC-MAB is the CR, which includes context dimension reduction and context noise removal. Fig. 4.7 compares the mean reward performance of QC-MAB with and without CR. Without CR, the mean reward is lower, and the convergence rate is slower. CR reduces the dimensionality of the state space while also removing noise, resulting in faster and more accurate relationship discovery between the state and QoS performance. Without CR, inappropriate choices may be made, leading to larger variations in instantaneous reward.

Impact of Access Network Density

The density of the access network determines the number of visible networks to the end user, which affects the actions set of QC-MAB. In this subsection, we explore the impact of access network density by adjusting the distances between neighboring APs.

The experiments reveal that LEAP and Peekaboo do not proactively select networks, resulting in minimal changes in their IR performance as the distances change. Therefore, we analyze the IR performance with QC-MAB under varied AP distances,

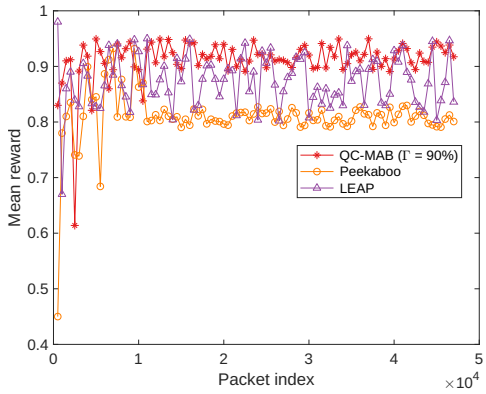


Figure 4.6: The mean reward value over time.

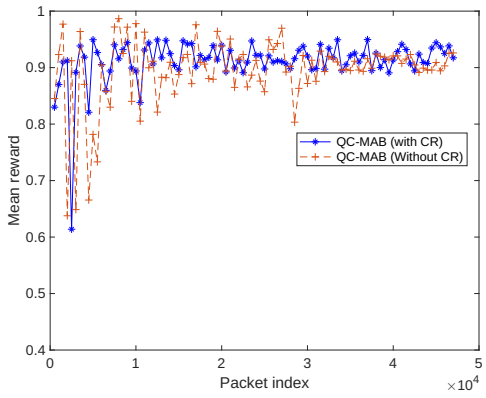


Figure 4.7: The effectiveness of the CR component.

as shown in Fig. 4.8. As the AP distance increases, the IR with QC-MAB initially decreases until a certain threshold and then starts to increase. The analysis of AP density demonstrates the exploration and exploitation dilemma: when more access networks are visible to the end user, QC-MAB has a higher chance of finding better networks to exploit after several rounds of exploration. However, when the density exceeds a certain threshold, continuously exploring new choices may negatively impact the IR performance, as QC-MAB could explore networks with poor conditions.

Overall, the experimental results highlight the advantages of QC-MAB in terms of mean video quality, goodput, convergence rate, and its ability to adapt to varying network densities in ultra-dense environments.

4.5.3 When Backbone Networks Become Bottlenecks

Traditional backbone networks are typically high-capacity and stable. However, in the next-generation network, this may not always be the case. For example, satellite

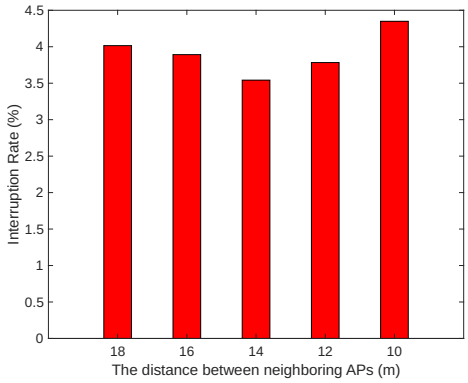
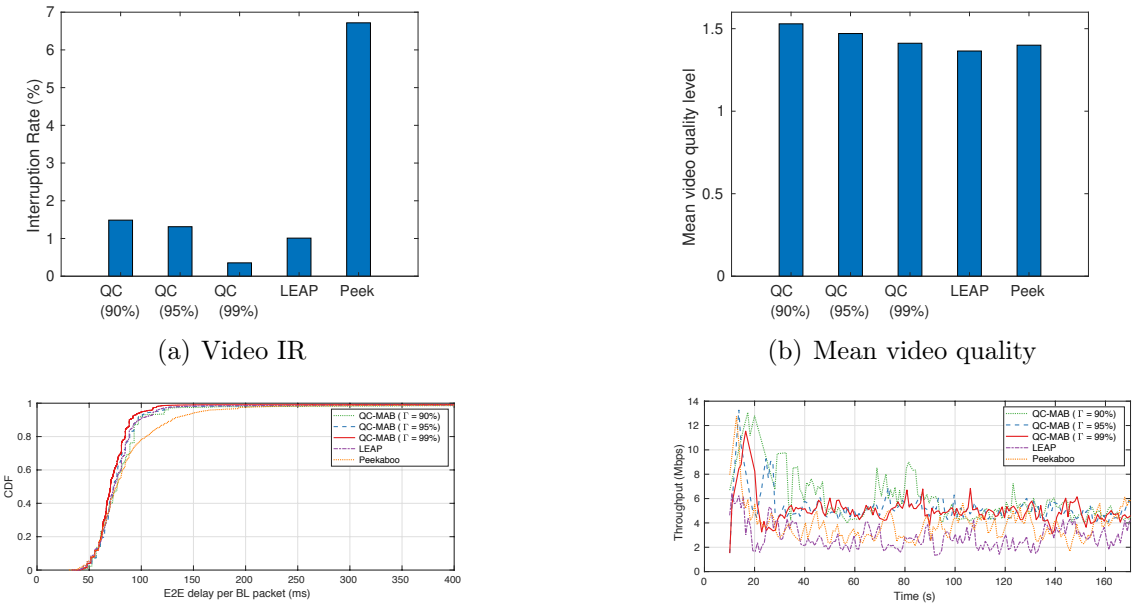


Figure 4.8: IR performance under the varied density of network coverage.

networks can serve as backbone networks, but their inter/intra satellite links are not as stable as terrestrial networks, making them potential bottleneck networks due to satellite movement. To simulate this situation, we constructed the topology shown in Fig. 4.4(b). In this topology, all link settings regarding the delay and error rate remain unchanged, except that wired links have a lower data rate compared to the rightmost access links. Specifically, the bandwidth of access links is set to 30 Mbps, while that of the wired links is 20 Mbps [41]. In this subsection, we start with the case in which background TCP flows as depicted in Fig. 4.4(b) are inactive. In Section 4.5.4, the background TCP flows are active and share the wired links with MPQUIC flows.

Compared to the IR performances in Section 4.5.2, the overall IR values in this scenario are significantly reduced. Specifically, the IR values with Peekaboo, LEAP, and QC-MAB are reduced from an average of 25% to 7%, from 6% to 1%, and from 5% to 0.5%, respectively. The differences in IR performance across different scenarios highlight that QC-MAB and LEAP outperform Peekaboo in terms of QoS guarantee. As shown in Fig. 4.9(c), both QC-MAB and LEAP achieve similar E2E delay guarantee performances, with over 99% of BL packets arriving within 150 ms. Peekaboo achieves a 96% counterpart.

Fig. 4.9(b) presents the mean video quality comparison, showing that while LEAP maintains similar IR values to QC-MAB, it has the lowest mean video quality level. This phenomenon can be explained by the goodput results shown in Fig. 4.9(d). Compared to QC-MAB and Peekaboo, LEAP exhibits the lowest goodput throughout the streaming process. As mentioned earlier, LEAP often compromises bandwidth to guarantee the shortest delay for the most important video frames.



(a) Video IR

(b) Mean video quality

(c) E2E delay CDF

(d) Aggregated goodput

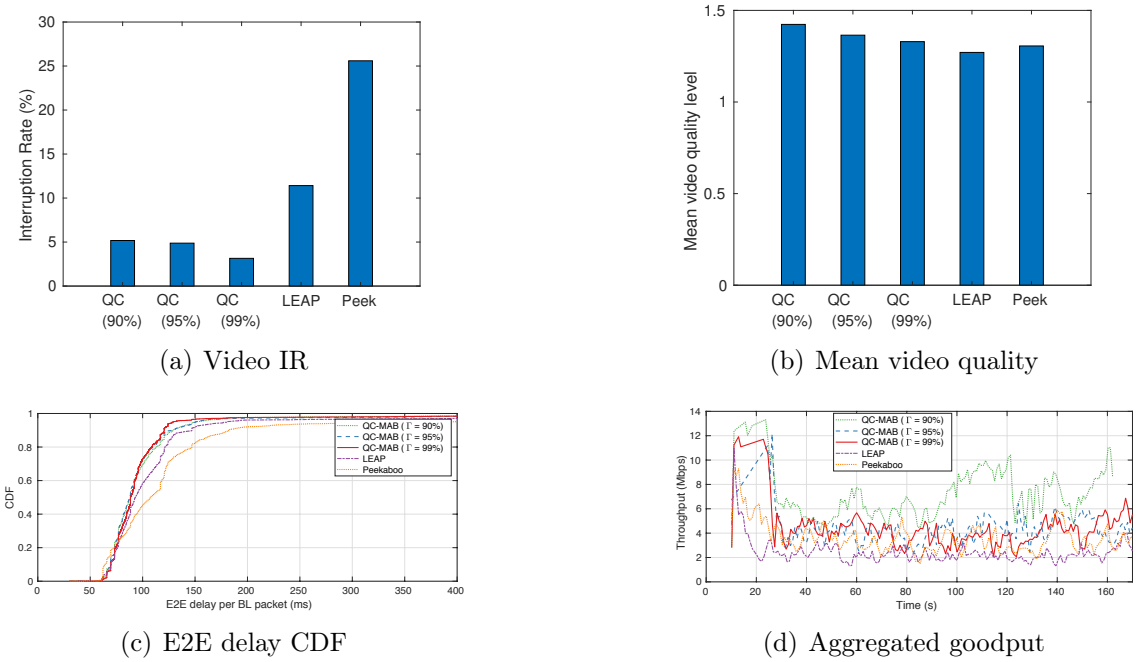
Figure 4.9: QoS performance when the backbone network and access networks are relatively stable.

4.5.4 Impact of Background Traffic in Backbone Networks

Lastly, we examine how QC-MAB performs in a scenario with background traffic competing for bottleneck resources, based on the topology shown in Fig. 4.4(b). In this subsection, except for the presence of background TCP flows, all other experimental settings remain the same as in Section 4.5.3.

IR and E2E Delay

As shown in Fig. 4.10(c), only 82.19% and 91.98% of packets meet the 150 ms deadline when using Peekaboo and LEAP, respectively, while QC-MAB achieves 96.7% reliability with $\Gamma = 99\%$. In Peekaboo, the dynamic bottleneck bandwidth is not taken into account, leading to inappropriate scheduling decisions. LEAP heavily relies on bottleneck bandwidth estimation to determine the cwnd value. However, this estimation is inaccurate when the bandwidth frequently changes over time. In QC-MAB, we emphasize the importance of the bottleneck bandwidth by integrating it into the context space and using an AKF to remove bandwidth noise. This approach ensures that the connection between bandwidth and QoS reward is accurately revealed, re-



(a) Video IR

(b) Mean video quality

(c) E2E delay CDF

(d) Aggregated goodput

Figure 4.10: QoS performance when the backbone network experiences large variations and becomes the bottleneck.

sulting in high reliability. According to Fig. 4.10(a), QC-MAB reduces video IR to around 3.1%, while LEAP and Peekaboo result in 11.4% and 25.11% IR, respectively.

Mean Video Quality and Goodput

From Fig. 4.10(d), we observe that both Peekaboo and LEAP do not have a sharp increase in cwnd at the beginning as shown in Fig. 4.9(d). This is caused by the lack of awareness of bandwidth variations in Peekaboo and LEAP, resulting in overshooting issues. In comparison, QC-MAB with different Γ values achieves higher goodput because it utilizes the UCB algorithm to balance exploration and exploitation. When high variations in routes are detected, QC-MAB can proactively switch to other access points for higher rewards. In comparison to Peekaboo, LEAP achieves lower goodput due to the packet redundancy generated by FEC.

Fig. 4.10(b) illustrates the mean video quality performances, which align with the goodput behaviors. QC-MAB overall improves the mean video quality compared to LEAP and Peekaboo, while LEAP achieves the lowest video quality level.

4.6 Summary

In this chapter, we develop a novel learning-based framework, named QC-MAB, to solve the QoS guarantee issue for multipath video streaming in mobile networks. As the QC-MAB incorporates both wireless link characteristics changes and E2E path conditions change in its context space design, in conjunction with context and arm dimension reduction, it can discover the underlying relationship between network context and QoS performance without a lengthy training process. Given a particular context and pre-defined QoS requirements, QC-MAB can therefore make a reasonable decision with regard to access network selection and FEC coding to trade off delay, reliability, and goodput. Extensive experiments based on ns-3 demonstrate that QC-MAB not only provides an ultra-low video interruption ratio but also high goodput in heterogeneous stationary networks or dynamic mobile systems.

Chapter 5

Tile Scheduling Across Multiple Paths for Smooth Interactive 360-degree Video Streaming

Different from the planar video streaming as described in Chapter 4, 360-degree video streaming aims at providing users with an immersive experience, i.e., a user can navigate in a virtual world by looking around and interacting with the virtual world. As a result, new challenges coming with 360° video streaming are emerging, which motivates us to address them in this chapter.

5.1 Introduction

Next-generation networks (6G) are increasingly embracing virtual reality (VR) and augmented reality (AR) applications. 360-degree video streaming is the key enabler for AR/VR to provide users with an immersive experience. However, a smooth playing experience with 360-degree videos cannot be fully guaranteed all the time. On the one hand, 360-degree videos will consume vast bandwidth to transmit not only wide-angle video frames but also videos with far higher resolutions (e.g., 8K, 16K) compared to conventional video streaming [56]. On the other hand, in the scenario where a mobile device fetches 360-degree video in a moving car, good quality of experience (QoE) is hard to guarantee due to the high dynamics raised by mobility. For example, the video downloading capacity varies over time and is unknown beforehand [49].

To overcome the bandwidth challenges mentioned above, the existing benchmarks

can be categorized into two classes: 1) Field-of-View (FoV) prediction-based methods [22,83,95]. FoV is associated with users' head movement and interest area. Given the limited bandwidth, the video provider would selectively transmit the FoV-associated content instead of the whole video to save bandwidth. 2) Cache based solutions [93, 145]. According to users' browsing history, the video server can prioritize the video segment based on their browsing popularity, and then the most popular one would be pre-cached at edge servers.

However, the FoV-based algorithms could suffer from a wrong prediction, which will cause retransmission and playback pauses. Cache-based algorithms do not work in case some content cannot be obtained beforehand such as the real-time 360-degree videos. In addition, most solutions merely improve QoE in the application layer, e.g., bit rate adaptation, and tiling scheme. They ignore the mutual interaction between the application layer and transport layer, e.g., the bit rate could be affected by the sending rate and scheduling decision at the transport layer.

Therefore, video streaming solutions at the transport layer have attracted much attention in both academic and industrial communities, in particular the multipath transport protocol based designs. Multipath transport protocols, such as multipath TCP (MPTCP) [164] and MPQUIC [32] are promising in aggregating bandwidth and improving the throughput as they can manage multiple end-to-end (E2E) paths simultaneously to transmit data.

In this chapter, we focus on the scheduling design for interactive (i.e., real-time) 360-degree video streaming using MPQUIC. The viewing frame quality and playback smoothness without frequent interruption are two important indicators of the quality of experience (QoE) for video consumers. To provide a satisfactory QoE, we aim to tackle the following pressing challenges:

Interactive streaming has the precise frame deadline: The interactive streaming applications require in-time delivery, which means the video frames need to arrive before an exact time point [143, 179]. Otherwise, the late arrival does not contribute to the video playback or decoding processes, which is detrimental rather than beneficial to the QoE. Existing studies usually control the completion time of video frames within a certain threshold instead of applying a precise deadline for a certain frame.

The deadline for different frames is different: With the introduction of video coding technology, there are different types of frames such as I, B, and P frames, and the importance of each frame in decoding varies. For instance, I frames are paramount

in the decoding process, followed by P frames, with B frames being the least critical. Therefore, compared to P and B frames, I frames should have a larger time window within which the arrival of the I frame remains effective for either the playback itself or for the decoding of subsequent frames. Given this fact, it is necessary to apply differentiated deadlines to each frame type.

The FoV changes in 360-degree video make the scheduling more complicated: Users' FoV is changing over time, involving both spatial and temporal dynamics. For instance, the new FoV may begin with a B frame while the preceding I or P frames, upon which the B frame depends, were not viewed and consequently not fetched in previous FoVs. To successfully display the B frame, the video server must transmit the necessary dependency frames alongside it. This requirement adds a layer of complexity to the scheduling process.

To cope with the above unique challenges, in this chapter, we present a QoE-oRIented DEadline-driven (RIDE) algorithm for the multipath scheduling based on the deep reinforcement learning (DRL) framework. Our contributions are three-fold:

- Unlike existing chunk-based approaches, RIDE's scheduling decisions are made at the frame level. This finer granularity allows for a more detailed analysis of frame dependency relationships. By designing a dependency tree, it is easier to understand the precise deadline for each frame, thus enhancing the efficiency of our scheduling.

- We have tailored RIDE specifically for interactive 360-degree video streaming. This includes identifying unique features that influence users' Quality of Experience (QoE), designing a comprehensive reward function that incorporates multiple critical QoE metrics, and developing an adaptive action space that effectively addresses various streaming dilemmas.

- We have conducted a series of experiments to evaluate RIDE using real-world video and network data traces. Experimental results demonstrate that RIDE can achieve up to four times higher frame quality and lower the interruption rate by 50% against benchmark algorithms.

5.2 Related Work

In this section, recent studies related to video streaming and multipath scheduling are reviewed.

5.2.1 General Multipath Scheduler

There have been several multipath transport protocols such as DCCP [79], CMT-SCTP [68], MPTCP [164], and MPQUIC [32]. MPQUIC, the latest proposal, has inherited the advantages of both CMT-SCTP and MPTCP, thereby receiving much research interest.

Regarding the multipath scheduler, the round-robin (RR) scheduler circularly transmits packets over each path as long as its congestion window (cwnd) is not exhausted. RR does not perceive heterogeneous characteristics of paths, resulting in poor performance under heterogeneous networks [104]. MinRTT, the default scheduler of multipath transport protocols, prioritizes the available path with the smallest Round-trip Time (RTT). However, [114] points out that the minRTT scheduler causes a large number of out-of-order (OFO) packets and requires more memory usage for multipath transport protocols compared to the conventional single-path counterparts. [45] finds that the performance of multipath transport protocols is degraded due to bufferbloat and proposes BM algorithm, which caps the cwnd when the RTT exceeds the minimal RTT (RTTmin) of the path significantly.

5.2.2 Multipath Scheduler for Video Streaming

Improving video streaming service quality using multipath techniques has always been a research highlight. [126] proposes Smart-Flycast, an AI-driven multipath transmission UAV-based live streaming system. They design three novel learning-based multipath transmission mechanisms to improve QoS and QoE in the highly mobile UAV flight environment. MPDASH [61] is a multipath framework for DASH video streaming with the awareness of network interface preferences from users. But it does not focus on per-packet scheduling, exploiting a coarse-grained decision logic that decides whether the cellular sub-flow should be activated or not. XLINK [174] directly captures video QoE intent to control multipath scheduling and path management based on MPQUIC protocol for short videos. OLS [150] schedules packets according to their arrival time and sends a number of redundant packets to avoid the impact of inaccurate estimations, aiming at reducing OFO packets and transmission latency. These efforts all require the support of the application layer. They are tightly coupled to the application. At the same time, they do not fully address variations in network conditions. Our proposed scheduling mechanism relies on more intelligent decision-making to provide universality and adaptability.

5.3 System Model

5.3.1 360-Degree Video Model

A raw panoramic video is broken into a set of consecutive video chunks, and each chunk contains a fixed duration of seconds. Further, each chunk c can be sliced into multiple frames. We index frames by the elements of set $\mathcal{F}_c = \{f_c, f_{c+1}, f_{c+2}, \cdots\}$. In tile-based HTTP adaptive streaming [147], as described in Fig. 5.1, each frame $f_i \in \mathcal{F}_c$ is spatially partitioned into Z tiles which are indexed in raster-scan order, denoted by $\mathcal{Z} = \{1, \cdots z \cdots, Z\}$. The subframe after tiling is denoted by $f_{i,z}$.

To exploit temporal redundancies, a number of consecutive tiles that share the same spatial indicator z are often encoded as a group for the purpose of encoding efficiency. This grouping is referred to as a Group of Picture (GoP), e.g., GoP 1 and GoP 2 in Fig. 5.1, which consists of a sequence of frames, including I-frames, P-frames, and B-frames. Let $\mathcal{G}_z$ be the GoP associated with the tile z, we have $\mathcal{G}_z = \{f_{c,z}, f_{c+1,z}, \cdots, f_{i,z}, \cdots\}$. Note that both the element ordering and the number of elements within set $\mathcal{G}_z$ may vary among tiles, depending on the grouping strategy. $h_{i,z}$ denotes the frame type of $f_{i,z}$, so $h_{i,z} \in \{I, B, P\}$.

On the other hand, each tile is encoded independently at several qualities on demand to adapt to network dynamics. When the network condition is poor, the video server will select the tiles with lower quality to transmit. Denote by $q_{i,z}$ the quality level of frame $f_{i,z}$ belonging to tile z. Noting that the video quality is the hardest element to measure, researchers usually take video bitrate to track. For instance, for a 360-degree video comprising 100 video chunks, with video bitrates ranging from 1 Mbps to 20 Mbps, encompassing eight quality increments. The selection of bitrates for different representations was designed to achieve a roughly linear increase in video quality, as measured by the Peak Signal-to-Noise Ratio (PSNR). Therefore, we can utilize the representation index as an indicator of video quality.

When viewing a 360-degree video, a user can focus on an area at certain direction and with limited horizontal and vertical spans, as known as Field-of-View (FoV). As depicted in Fig. 5.1, a FoV covers multiple tiles and it may change from time to time depending on the head movement. Let a matrix $\mathbf{v} \in \{0,1\}^{I \times Z}$ indicate whether a tile is inside the FoV or not. The element $v_{iz} \in \mathbf{v}$ equals 1 if $f_{i,z}$ is viewed and 0 otherwise. Denote by $\mathcal{V}_s$ the viewed tiles for chunk c, and we have $\mathcal{V}_c = \{f_{i,z} | \forall i, z : f_i \in \mathcal{F}_c, v_{iz} = 1\}$.

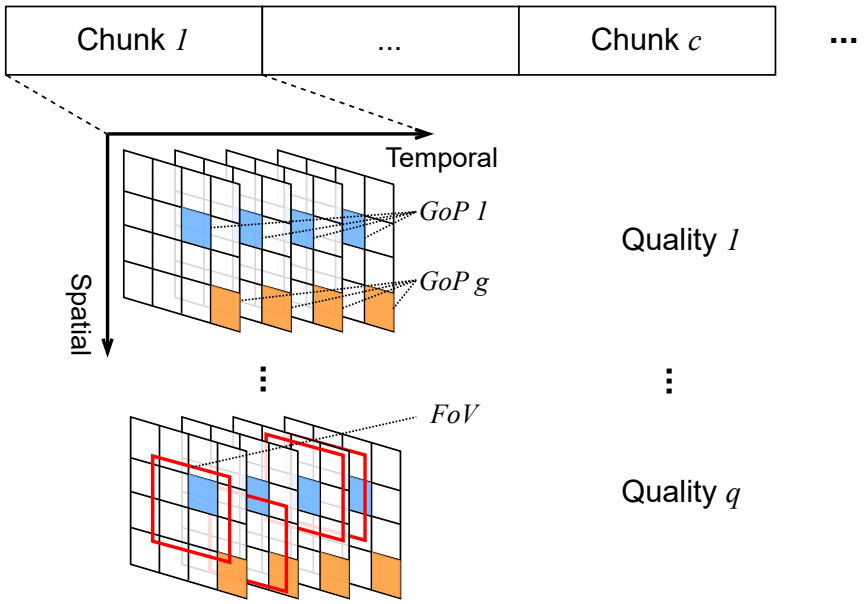


Figure 5.1: Tile-based adaptive streaming for 360-degree video.

5.3.2 Frame Transmission Model

In the context of 360-degree video, data are divided into frames. MPQUIC is an extension of QUIC (Quick UDP Internet Connections), a modern transport protocol developed by Google that operates over UDP (User Datagram Protocol). It supports frame-level scheduling so that the MPQUIC scheduler can determine which frames are transmitted over which path, ensuring that important content is rendered quickly at the receiver side.

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. With a scheduling policy $\mathbf{\Pi}$, the selected frames to transmit on path m are stored in set $\mathcal{S}_m = \{f_{i,z} | f_i \in \mathcal{F}_c, z \in \mathcal{Z}\}$. Based on the sending order of each frame, we rewrite $\mathcal{S}_m$ as $\mathcal{S}'_m = \{f^{(1)}, f^{(2)}, \cdots, f^{(x)}, \cdots\}$, where x indicates the order of $f_{i,z}$ to be scheduled.

Packetization in MPQUIC involves encapsulating frames into UDP packets for transmission over the network. Denote by g_x the number of packets that frame $f^{(x)}$ has. g_x is determined by several factors, such as frame duration and the selected quality level. Assume when scheduling starts at time instance t^*, there are $x-1$ preceding frames before $f^{(x)}$. The latency experienced by $f^{(x)}$ before it arrives at the receiver includes both the queuing time at the sender side and the in-flight time on

the network side. Denote by a_x the arrival time of $f^{(x)}$, we have

$$a_x = t^* + \frac{\sum_{j=1}^{x} g_j \cdot MSS}{\Phi} + r_{\min} \cdot \tau + OWD, \tag{5.1}$$

where MSS means the maximum segment size in bytes, Φ stands for the transmission rate over the wireless or wired links, τ refers to the round-trip time (RTT) over the end-to-end (E2E) path, and OWD is the one-way path delay which is estimated to be approximately half of τ. The term, $r_{\min} * \tau$, considers a variety of extra delays such as the retransmission delay and the delay waiting for the availability of the congestion window, and $r_{\min}$ means the minimum extra rounds of transmission needed to ensure all $\sum_{j=1}^{x} g_j$ packets arriving at the receiver successfully. If the total number of packets belonging the first x frames does not exceed the efficient window (w^*) within one RTT, i.e., $\sum_{j=1}^{x} g_j \leq w^*$, $r_{\min} = 0$. Otherwise, $r_{\min}$ is expressed as follows [160]

$$r_{\min} = \arg\min_{R \geq 2} \{w^* + \sum_{r=2}^{R} \mathbb{E}[n_r] \geq \sum_{j=1}^{x} g_j\}. \tag{5.2}$$

Specifically, w^* depicts the expected number of packets arriving at the receiver successfully within one RTT given the congestion window is w and the packet loss rate is ϵ, it is calculated as $w^* = w\epsilon$ because packet losses are independent of each other and the probability that n out of w packets are lost obeys the Binomial distribution [48, 50, 106, 160]. $\mathbb{E}[n_r]$ in (5.2) can be derived as follows,

$$\begin{aligned} \mathbb{E}[n_r] = & \sum_{u_1=1}^{3} \sum_{u_2=1}^{3} \cdots \sum_{u_{r-1}=1}^{3} \mathbb{P}\left(s_1, s_{u_1|2}\right) \\ & \cdot \prod_{v=2}^{r-1} \mathbb{P}\left(s_{u_{v-1}|v}, s_{u_v|(v+1)}\right) \cdot w(s_{u_{r-1}|r})\epsilon, r \geq 2, \end{aligned} \tag{5.3}$$

where $s_{u_r|r}$ denotes the possible phase in the r-th transmission round, here $u_r \in \{\mathrm{CA}, \mathrm{SS}, \mathrm{RTO}\}$. $\mathbb{P}\left(s_{u_{v-1}|v}, s_{u_v|(v+1)}\right)$ represents the probability of network state transition.

It is worth noting that $f_{i,z}$ and $a_{i,z}$ are equivalent to $f^{(x)}$ and a_x, respectively. The equivalent pair of notations will be used interchangeably throughout this chapter.

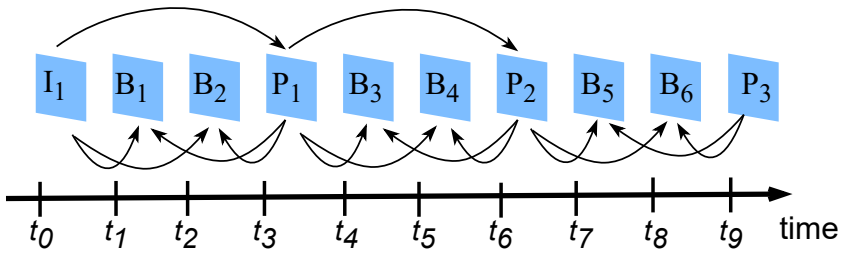


Figure 5.2: A typical MPEG frame sequence.

5.3.3 Video Decoding Model

On the client side, a 360-degree video decoder processes frames from a GoP to reconstruct the immersive video content. A frame decoding failure may cause unsuccessful decoding for the subsequent frames, thus leading to a long freezing time.

Fig. 5.2 illustrates a typical MPEG frame structure [46], featuring three types of frames: I-frames, B-frames, and P-frames. The arrows in the figure represent the decoding dependency relationships among these I, B, and P frames, highlighting how each frame type influences the decoding process of the others.

An I-frame is a fully encoded frame so that its decoding does not depend on any other frame. It serves as an anchor or referencing point for the subsequent P and B frames within the same GoP directly or indirectly. A P-frame contains the changes or differences from the reference frames, and it depends on previous I-frames or P-frames. A B-frame is a bidirectional frame that depends on adjacent I and/or P frame pairs.

In this case, the arriving time of different frames in a GoP can significantly affect decoding and video smoothness, especially in scenarios with limited receiving buffers or variable network conditions.

Assuming the frame sequence within a GoP $\mathcal{G}_z$ is always starting with an I frame and ending before the next I frame. Let $d_{i,z}$ be the time that frame $f_{i,z} \in \mathcal{G}_z$ is successfully decoded and displayed. For I frames, $d_{i,z} = a_{i,z}$. For P frames, the decoding time is determined by both their own arrival time and the decoding time of the preceding I or P frames, so we have $d_{i,z} = \max\{a_{i,z}, d_{i^-,z}\}$ where $i^- = \max\{x | x < i, h_{x,z} \neq B, f_{x,z} \in \mathcal{G}_z\}$. Similarly, for B frames, $d_{i,z} = \max\{a_{i,z}, d_{i^-,z}, d_{i^+,z}\}$ where $i^+ = \min\{x | x > i, h_{x,z} \neq B, f_{x,z} \in \mathcal{G}_z\}$.

5.3.4 QoE Model

To quantify users' perceived quality during the playback time of each chunk c, we take into account three QoE metrics as follows.

1) *Average Quality* $Q_0(\mathcal{V}_c)$. Since only video content in the viewing area contributes to user-perceived quality, the average quality is calculated over all frames in the viewing area, i.e.,

$$Q_0(\mathcal{V}_c) = \frac{1}{IZ}\sum_{i}^{I}\sum_{z}^{Z} v_{iz} \cdot q_{i,z}.$$

where $v_{iz} \in \mathbf{v}$ is the indicator of whether the tile $f_{i,z}$ is inside the viewport, and $q_{i,z}$ represents the quality level. Since the QoE metric is measured for each chunk, I refers to the total number of frames included in a chunk.

2) *Rebuffering time* T_{rebuff}. In the tile-based 360-degree video, the rebuffering event could happen over each tile independently. Let B_z and C be the buffer occupancy in bytes corresponding to tile z and the bitrate, respectively. The left play-time ΔT is then calculated by $\lfloor 8B_z/C \rfloor$. The rebuffering time for each tile is the difference between the left play time and the latency of transmission and decoding successfully, i.e., $T_{\text{rebuff}}(f_{i,z}) = \max\{0, d_{i,z} - \Delta T\}$. By averaging over all tiles, we have

$$T_{\text{rebuff}}(\mathcal{V}_c) = \frac{1}{IZ}\sum_{i}^{I}\sum_{z}^{Z} T_{\text{rebuff}}(f_{i,z}). \tag{5.4}$$

5.3.5 Problem Formulation

With the QoE model defined above, we aim to find an optimal scheduling policy $\mathbf{\Pi}$ to provide a high QoE score for users. The problem is formulated as follows.

$$\max_{\mathbf{\Pi}:\langle f_{i,z},m\rangle} \{Q_0 - \omega_r T_{\text{rebuff}}\} \tag{5.5}$$

Here ω_r is the weight to penalize rebuffering events.

The scheduling policy $\mathbf{\Pi}$ needs to decide path m for each frame $f_{i,z}$, and to achieve the optimized QoE score which is affected by many factors including the application-level and network-level factors. Therefore, it is challenging to make an intelligent decision. The existing heuristic multipath scheduling algorithms fail to handle this problem because the fine-tuned model they proposed is for a specific environment with strong assumptions and it lacks generalization across network conditions and

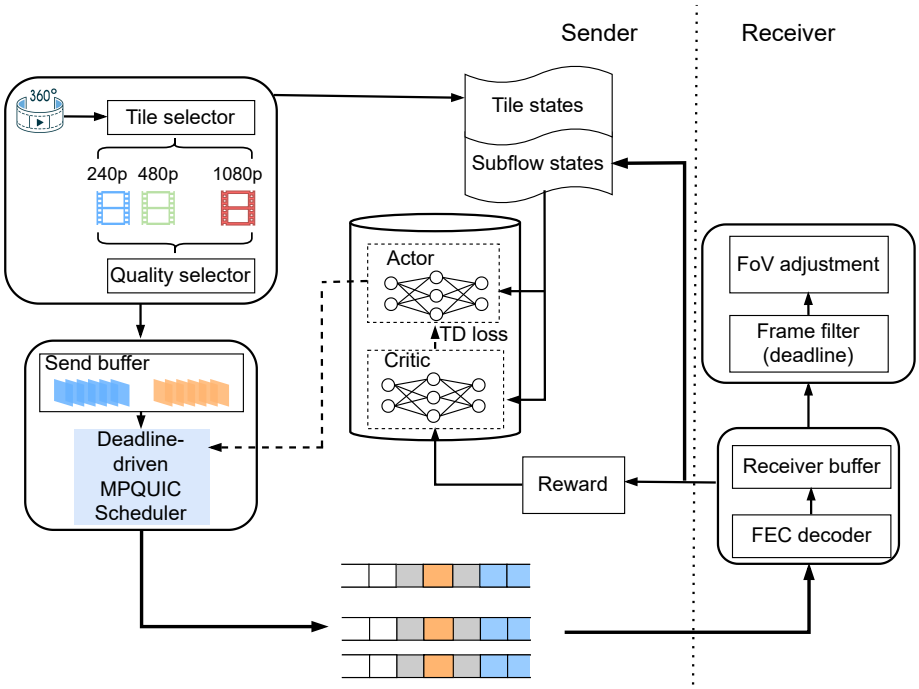


Figure 5.3: The RIDE framework.

QoE objectives.

Inspired by this fact, we present an actor-critic-based RL algorithm RIDE to discover the relationship between the environment and QoE metrics for 360-degree video clients. RIDE learns to make scheduling decisions through observations of the resulting rewards of past actions.

5.4 Our Proposal

In this section, we elaborate on how to build the learning system to solve the formulated QoE maximization problem. Fig. 5.3 gives the overview of the proposed QoE-oRIented DEadline-driven (RIDE) algorithm. The detailed description is given below.

5.4.1 Markov Decision Process

In order to model the complex dynamic process, we model the 360-degree video streaming as a Markov decision process (MDP). In general, the MDP process is

characterized by a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma \rangle$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $\mathcal{P}(s'|s,a)$ is the probability of transitioning to $s' \in \mathcal{S}$ from state $s \in \mathcal{S}$ and action $a \in \mathcal{A}$, $\mathcal{R}$ denotes the reward design and $r(s,a,s')$ is associated with the transition (s,a,s'), and $\gamma \in [0,1)$ is a discount factor. A policy $\pi : \mathcal{S} \rightarrow \Delta(\mathcal{A})$ is defined as a mapping from the state space $\mathcal{S}$ to the probability distribution over the action space $\mathcal{A}$.

As depicted in Fig. 5.3, the sender, i.e., the video server, is knowledgeable about some information, so it plays the role of an intelligent agent. At each coherence time slot t, the agent observes a state $s_t \in \mathcal{S}$ which showcases both the features of the video frames to be fetched and the real-time network conditions, it chooses an action (i.e., the path selection) according to the state-action mapping relationship defined in π: $a_t \triangleq \pi(s_t)$. After choosing a_t, the agent receives a reward $r(s_t, a_t, s_{t+1})$ at time slot $t+1$ based on the reward function $\mathcal{R}$ and the environment turns to the next state s_{t+1} based on the transition probability function $\mathcal{P}(s_{t+1}|s_t, a_t)$.

In the following, we will present how the critical elements of the MDP are designed.

5.4.2 State Space

From the QoE model, we observe that numerous factors are pertinent to QoE metrics for 360-degree video streaming. However, it would cause a high dimension of state space and a lengthy training process if we exhaustively list all potential factors. Therefore, we aim to explore an appropriate state space that accommodates the most important features to ensure learning accuracy and also limits the state dimension to provide learning efficiency.

Similar to most existing solutions, in our RIDE framework, the state space $\mathcal{S}$ also takes into account the following network conditions:

- $\hat{w}_m$: available cwnd over path m. This indicator is critical because it reflects the magnitude of cwnd and in-flight bytes, which affects the queuing delay of a certain frame at the sender side.

- b_m and τ_m: the bottleneck bandwidth and rtt on path m. These two metrics shape the bandwidth-delay product over an end-to-end (E2E) path, in turn affecting the maximum throughput the path can achieve.

- ϵ_m: the loss rate over path m. The reason behind the packet loss in wireless networks is complicated. It could stem from wireless channel fluctuations, path

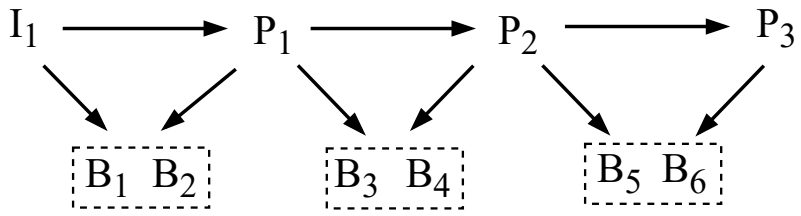


Figure 5.4: Decoding dependency tree.

failure, or link congestion [163]. ϵ_m is crucial for QoE performance because it impacts the reliability of the key frames' transmission.

As we described above, interactive 360-degree videos face unique challenges, such as stringent frame deadlines and FoV changes. Therefore, in addition to the above features of network conditions, RIDE introduces two critical features related to the frame information.

Frame Deadline

To provide a good user experience, existing studies are motivated to control the completion time of video frames within a certain threshold. The in-time frame arrivals contribute to the QoE gain while the overdue frame will be discarded and followed by degraded QoE performance. For instance, designs in [127, 140] demonstrate an improved video streaming performance when they use the threshold of 150 ms.

Applying the same delay constraint to all frames is not desirable for interactive streaming services for two reasons. First, as discussed in Section 5.3.3, in MPEG frame structures, the importance of each frame type varies. I frames are paramount in the decoding process, followed by P frames, with B frames being the least critical. Given this fact, it is necessary to apply differentiated time constraints to each frame type. Second, interactive applications demand that frames arrive by precise deadlines. Imposing a uniform completion time on all frames does not ensure timely arrivals, potentially compromising the effectiveness of the streaming service.

Therefore, we introduce a frame deadline for each tile, denoted by Γ to the state space. Γ indicates the latest time point at which the arrival of a frame remains effective for either the playback itself or for the decoding of subsequent frames.

Fig. 5.4 is a dependency tree created for the frame sequences in Fig. 5.2. We

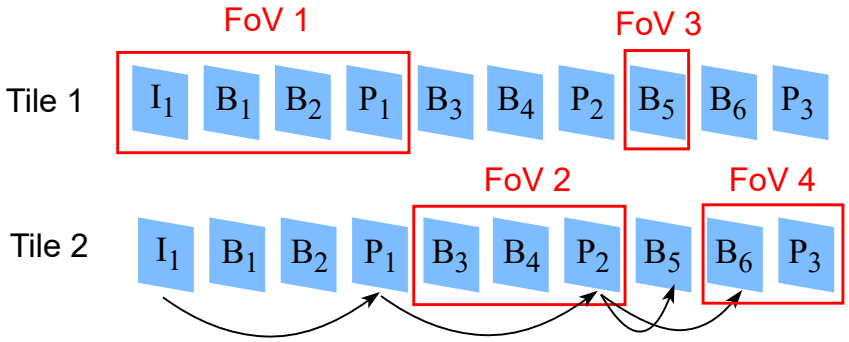


Figure 5.5: An FoV change example: the user's viewpoint switches back and forth between tiles 1 and 2 over time.

observe that I frames serve as root nodes, the child nodes of a P frame include the neighboring B frames and the subsequent P frame, and B frames are located at leaf nodes. Assuming each GoP $\mathcal{G}_z$ starts with an I frame and ends before the next I frame, the deadline of frame $f_{i,z} \in \mathcal{G}_z$, $\Gamma_{i,z}$ can be expressed as

$$\Gamma_{i,z} = t_{\text{playback}}(f_{j,z}), \tag{5.6}$$

where $t_{\text{playback}}(f_{j,z})$ means the playback time of frame $f_{j,z}$ and j is defined as follows.

$$j = \begin{cases} \max\{x | f_{x,z} \in \mathcal{G}_z\}, & \text{if } h_{i,z} = \text{I} \\ \min\{x | x \geq i, h_{x,z} = \text{P}, f_{x,z} \in \mathcal{G}_z\} \quad \text{or} & \\ \max\{x | f_{x,z} \in \mathcal{G}_z\}, & \text{if } h_{i,z} = \text{P} \\ i, & \text{if } h_{i,z} = \text{B}. \end{cases} \tag{5.7}$$

Note that for a P frame, j is calculated by $\max\{x | f_{x,z} \in \mathcal{G}_z\}$ whenever there are no additional P frames following the current one; instead, there may be a sequence of subsequent B frames.

Following the example in Fig. 5.4, $\Gamma_{I_1} = t_{\text{playback}}(f_{B_6})$, $\Gamma_{P_1} = t_{\text{playback}}(P_2)$, and $\Gamma_{B_1} = t_{\text{playback}}(B_1)$.

However, in the context of 360-degree video applications, FoV changes frequently, raising the new challenge of calculating the frame deadline. As shown in Fig. 5.5, the user does not keep staring at tile 1, instead, the viewpoint switches back and forth between tiles 1 and 2. For frames within $\mathcal{G}_1$, the value of Γ is updated as follows. $\Gamma_{I_1} = t_{\text{playback}}(f_{B_5})$ because frame B_6 is not viewed, and $\Gamma_{P_1} = t_{\text{playback}}(P_1)$ because

frames B_3, B_4, and P_2 of tile 1 are not covered by FoV.

Recall that we denote $v_{iz} \in \mathbf{v}$ as an indicator of whether a tile is inside the FoV or not. By taking into account the impact of FoV change, therefore, (5.7) is updated as

$$j = \begin{cases} \max\{x | v_{xz} = 1, f_{x,z} \in \mathcal{G}_z\}, & \text{if } h_{i,z} = \text{I} \\ \text{nextP}, \quad \text{or} & \\ \max\{x | i < x < \text{nextP}, h_{x,z} = \text{B}, f_{x,z} \in \mathcal{G}_z\}, \text{or} & \\ \max\{x | v_{xz} = 1, f_{x,z} \in \mathcal{G}_z\}, & \text{if } h_{i,z} = \text{P} \\ i, & \text{if } h_{i,z} = \text{B}. \end{cases} \tag{5.8}$$

nextP in (5.8) is given by $\min\{x | x \geq i, h_{x,z} = \text{P}, f_{x,z} \in \mathcal{G}_z\}$ and $j = \text{nextP}$ only if nextP is viewed.

The Size of the Frame and Dependencies

In addition to the frame deadline, frame size significantly impacts QoE performance. Therefore, we incorporate the frame size into the state space for better training results. Let $L_{i,z}$ be the frame size of $f_{i,z}$.

However, fluctuations in the FoV and the dependency relationships among frames present a unique challenge. As depicted in Fig. 5.5, FoV 2 encompasses three frames of tile 2. According to decoding theory, frames I_1 and P_1 of tile 2 need to be fetched alongside the viewed frames B_3, B_4, and P_2 to successfully decode these latter three frames. Consequently, the size calculation for frame B_3 must consider not only its own size but also the aggregated size of the frames on which B_3 depends.

In summary, the state space $\mathcal{S}$ is given by $\mathcal{S} = (\hat{\mathbf{w}}, \mathbf{b}, \mathbf{d}, \mathbf{e}, \Gamma, L)$, where $\hat{\mathbf{w}} = \{\hat{w}_m\}$, $\mathbf{b} = \{b_m\}$, $\mathbf{d} = \{\tau_m\}$, $\mathbf{e} = \{\epsilon_m\}$, and $m \in [1, M]$. It is noteworthy that extensive research has been conducted on predicting short-term FoV changes for 360-degree video streaming, demonstrating that prediction accuracy can reach 94.2% within a look-ahead window of 0.2 seconds, equivalent to approximately 7 – 10 frames [53]. Based on this, we assume the short-term FoV trajectory is known, allowing us to primarily focus on the design of multipath scheduling. In our future work, we will consider both the improvement of the FoV prediction and the intelligent scheduling design.

5.4.3 Action Space

In this subsection, we will address three key questions. First, we will define the design of the action space. Second, we will identify which entity is chosen as the agent and explain the rationale for this choice. Third, we will discuss the frequency of action updates.

In the presence of multiple paths available, the action space is a sequence of discrete path numbers, i.e., $\mathcal{A} = (1, 2, \cdots, M)$.

When bandwidth is limited and a frame's deadline is approaching, continuing to transmit a large frame can exhaust the available bandwidth, thereby delaying the arrival of subsequent, more critical frames. Consequently, it may be beneficial to skip the transmission of this frame altogether, without choosing any path for transmission. We represent this skipping option with the digital number -1. After incorporating this action into the action space, we have $\mathcal{A} = (-1, 1, 2, \cdots, M)$.

As illustrated in Fig. 5.3, the output of the actor-network is directed to the scheduler at the sender, meaning the video server acts as the intelligent agent responsible for making scheduling decisions. There are two primary reasons for designating the server as the agent. First, the action space pertains to scheduling decisions, which are typically made by the sender. While it is feasible for the receiver to choose the desired data path, this would necessitate additional signaling between the sender and the receiver in practice. Second, the video server has more comprehensive knowledge about the video content, such as the frame structure and frame types, making it more straightforward for the sender to gather relevant states associated with the frames in the sending buffer.

The RIDE learning model takes frame information as input to determine which path ensures timely delivery or high quality for the entire frame. Consequently, our scheduling decisions are made at the tiled-frame level rather than the packet level. This approach means that RIDE executes action prediction each time the sender handles a new tiled frame.

5.4.4 Reward Design

In RIDE, the reward design is closely aligned with the QoE objectives. Given that QoE is assessed based on two key metrics, i.e., frame quality and rebuffering time, we propose a workflow in Fig. 5.6 for designing the reward component.

Fig. 5.6 illustrates four scenarios that demonstrate the relationship between the

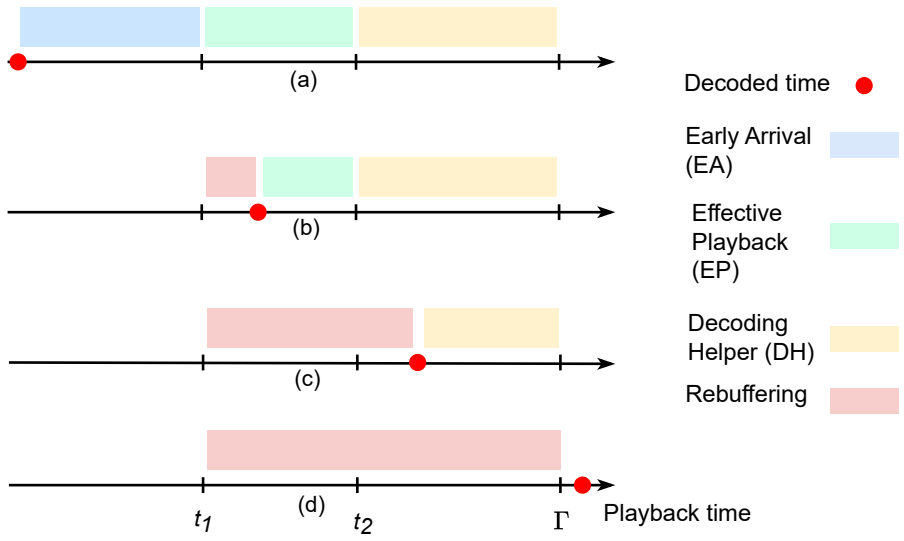


Figure 5.6: Frame decoded time vs QoE performance: (a) being decoded earlier than its playback time; (b) in the middle of the playback duration; (c) missing the playback time itself but helping the subsequent frames decode; (d) being decoded too late to enhance the QoE.

frame's time of being successfully decoded and the QoE performance. These scenarios categorize the frame $f_{i,z}$ into four distinct stages based on its decoded time, $d_{i,z}$.

First, *Early Arrival (EA)* stage. If a frame arrives and is decoded before its scheduled playback time, denoted as $t_1(f_{i,z})$ (i.e., $d_{i,z} < t_1$), it is stored in the receiver's buffer and awaits playback. The earlier a frame is decoded, the more data the receiver can buffer, enabling higher-quality requests for subsequent frames. Consequently, the duration of the EA stage can serve as an indicator of frame quality. Let T_{EA} be the length of EA, we have

$$T_{EA} = \begin{cases} 0, & \text{if } d_{i,z} \geq t_1(f_{i,z}) \\ t_1(f_{i,z}) - d_{i,z}, & \text{otherwise.} \end{cases} \tag{5.9}$$

Second, *Effective Playback (EP)* stage. The time interval $[t_1, t_2]$ represents the effective playback time duration of the frame. For interactive streaming applications, the segment is played in real-time, with the same pace as the video-generating process. Fig. 5.6(a) shows that if the frame has been in the EA stage, the entire frame playback is effective. However, as shown in Fig. 5.6(b), if the frame decoded time is later than the scheduled playback time, i.e., $t_1 \leq d_{i,z} \leq t_2$, the video content for the interval

$[t_1, d_{i,z}]$ would experience freezing, leading to rebuffering events, while the playback from $[d_{i,z}, t_2]$ remains effective. Thus, the length of the EP duration also serves as a critical indicator of frame quality during viewing. We utilize T_{EP} to record the EP length, which is given by

$$T_{EP} = \begin{cases} 0, & \text{if } d_{i,z} \geq t_2(f_{i,z}) \\ t_2(f_{i,z}) - \max\{d_{i,z}, t_1(f_{i,z})\}, & \text{otherwise.} \end{cases} \tag{5.10}$$

Third, *Decoding Helper (DH)* stage. As discussed above, the successful decoding of I and P frames is not only necessary for their own display but also helpful for subsequent frames in decoding. Therefore, we introduce the variable T_{DH} to quantify the impact of a frame's decoded time on the decoding of other frames, and

$$T_{DH} = \begin{cases} 0, & \text{if } d_{i,z} \geq \Gamma_{i,z} \\ \Gamma_{i,z} - \max\{d_{i,z}, t_2(f_{i,z})\}, & \text{otherwise.} \end{cases} \tag{5.11}$$

where $\Gamma_{i,z}$ means the frame deadline of $f_{i,z}$.

Last but not least, *Rebuffering* stage. Here the rebuffering time may consist of two components: the rebuffering time experienced by the current frame and the subsequent frames. As depicted in Fig. 5.6(c) and (d), when $d_{i,z} > t_2$, the display period between $[t_1, t_2]$ experiences freezing. The late decoding of the current frame causes decoding delays for the following frames, leading to further rebuffering. Denote by T_{rebuff} the rebuffering time. As presented in Fig. 5.6, the rebuffering event occurs in three cases and T_{rebuff} is given by

$$T_{\text{rebuff}} = \begin{cases} 0, & \text{if } d_{i,z} \leq t_1(f_{i,z}) \\ \min\{\Gamma_{i,z}, d_{i,z}\} - t_1(f_{i,z}), & \text{otherwise.} \end{cases} \tag{5.12}$$

The reward is finally designed as follows,

$$r = \begin{cases} -\left(\Gamma_{i,z} - t_1(f_{i,z})\right), & \text{if } a = -1 \\ \alpha \cdot T_{EA} + (1-\alpha) \cdot (T_{EP} + T_{DH}) & \\ -T_{\text{rebuff}} - T_{\text{cost}}, & \text{otherwise} \end{cases} \tag{5.13}$$

The condition $a = -1$ indicates that the chosen action is to skip transmitting the frame. As a result, this frame and its potential dependents will experience rebuffer-

ing without any effective playback time. In the second condition of (5.13) where $a \in \mathcal{M}$, both T_{EA}, T_{EP} , and T_{DH} contribute positively to frame quality, whereas T_{rebuff} (rebuffering time) negatively impacts it. Therefore, we use the parameter α to calculate the weighted positive reward. However, unlike the option of skipping the frame, achieving this positive reward incurs resource consumption. To encourage the agent to learn the tradeoff between reward and cost, we include a negative term $-T_{\text{cost}}$ for the reward design under the second condition. Here T_{cost} is the transmission delay for frame $f_{i,z}$, calculated as $T_{\text{cost}} = \frac{L_{i,z} \cdot MSS}{\Phi_a}$.

5.4.5 The Actor-Critic Architecture

As shown in Fig. 5.3, we adopt the actor-critic (AC) architecture to find out the optimal policy π for the above problem. The traditional Q-learning is a two-dimensional tabular method that uses a Q-table to evaluate the system performance of actions taken in each state. However, since the state of the tiled frames and subflows is complex and flexible, the state and action set of the two-dimensional Q-table is limited. It is impractical to use a Q-table to store all actions. It is also time-consuming to frequently search for the corresponding state-action pairs in the Q-table. Therefore, we elected to use AC to train our models.

Specifically, we use Proximal Policy Optimization (PPO) [120] algorithm to solve our formulated stochastic game problem, which follows the actor-critic architecture that is composed of an interactive pair of policy and value networks. The major reason to choose PPO is because it has outstanding performance and lower computational complexity.

5.5 Evaluation

In this section, we examine the performance of RIDE by comparing it to the two state-of-the-art algorithms below.

- Round Robin (RR): RR is the easiest but unintelligent policy to distribute packets over multiple paths in a round-robin fashion, which may cause high latency when two flows have a large difference in bandwidth and RTT.
- minRTT: provided that the congestion window still has space, the path with the lowest measured RTT is preferred.

- Peekaboo [139]: it is a novel CMAB-based MPQUIC scheduler. It strives to learn the dynamic characteristics of the heterogeneous path including path RTT and loss rate, and then makes decisions about whether to schedule packets into the path with undesirable conditions immediately or wait for the availability of the path with good conditions.

5.5.1 Settings

Video Dataset and FoV Trajectory: In our experiments, we adopt a video dataset from [90] consisting of ten 360-degree videos, each with a duration of 60 seconds. These videos are categorized into two types based on their content: computer-generated and natural scenes. They also vary in tempo, ranging from fast-paced to slow-paced motions. Each 60-second video comprises 1800 frames, with each frame lasting approximately 33 milliseconds. Further, each frame is spatially divided into 10×20 tiles to facilitate detailed analysis and processing.

The provided dataset includes both data content information and sensor data traces. The data content information offers detailed descriptions of each frame, including saliency maps and motion maps. Additionally, the sensor data component comprises FoV trajectories collected from dozens of users, providing a rich dataset for analyzing user interactions and viewing behaviors.

Given the provided dataset, we need to use the tools provided by FFmpeg [130] and OpenCV [5] to process the data for later usage. FFmpeg is used to extract comprehensive frame information such as frame size, frame type (I/B/P), frame duration, and frame deadline. Although FFmpeg provides the overall frame size, it does not break down the size by individual tile, which is necessary in our experiments. To address this, we then use OpenCV to manipulate the saliency maps of each frame. We convert these maps into 10×20 matrices of gray values, corresponding to the tile dimensions used in the dataset. These gray values within each frame will be normalized to ensure that their sum equals 1, reflecting the proportional significance of each tile's visual content. With the tile size being proportional to its saliency value, we can determine the size of each tile by multiplying the total frame size by its corresponding normalized saliency value.

Statistically, each GoP comprises 50 frames, which typically include 1 I-frame, approximately 13 P-frames, and 36 B-frames. The size of each viewed tile varies from 10 KB to 200 KB, depending on its saliency value and frame type. In general,

I-frames have larger file sizes compared to P and B-frames due to their comprehensive encoding.

We have selected four videos to serve as the training dataset and one video as the testing dataset.

Network traces: In our experiments, we simulate the multipath transmission consisting of LTE and WiFi networks. The RTT data traces of each path are generated by executing the ping command for 24 hours. During the training stage, we respectively select a data trace over 10 hours for LTE and WiFi paths, and the rest of the data traces are used during the testing stage. To match the frame duration (i.e., 33 ms) of each video, we scale down the average RTT values of WiFi and LTE paths to 20 and 30 ms. The bandwidth traces over WiFi and LTE paths are generated from FCC [27] and HSDPA [117], respectively. The median bandwidth of WiFi and LTE is around 40 Mbps and 20 Mbps. The packet loss rate traces for each path is controlled by a uniform random variable.

Hyper-parameters: It is clear that the proposed RIDE algorithm consists of an actor network and a critic network, each of which has one input layer, two hidden fully-connected layers, and one output layer. Moreover, each hidden layer has 64 neurons. We use the rectified linear unit (ReLU) function to describe the activation function in each hidden layer. The learning rate is set to 0.0003 and the discounted factor is 0.99. In addition, the training process of our proposed algorithm has 30000 episodes, each of which contains around 40 time steps. The number of time steps depends on the number of viewed tiles for each video frame.

5.5.2 Numerical Results

1) Reward: this result is calculated based on (5.13). This metric gives an overview of the QoE performance.

During the testing phase, we conducted a total of 1800 episodes, where each episode corresponds to the processing of a single frame, involving approximately 40 tiles, thereby constituting 40 time steps per episode.

Fig. 5.7 illustrates the cumulative reward over 40 time steps across episodes. Our proposed RIDE algorithm consistently outperforms benchmark algorithms, achieving the highest rewards. It is followed by the RR algorithm and then Peekaboo. The minRTT algorithm scores similarly to Peekaboo but exhibits sharp fluctuations, which are less desirable for managing network dynamics. In contrast, RIDE maintains a

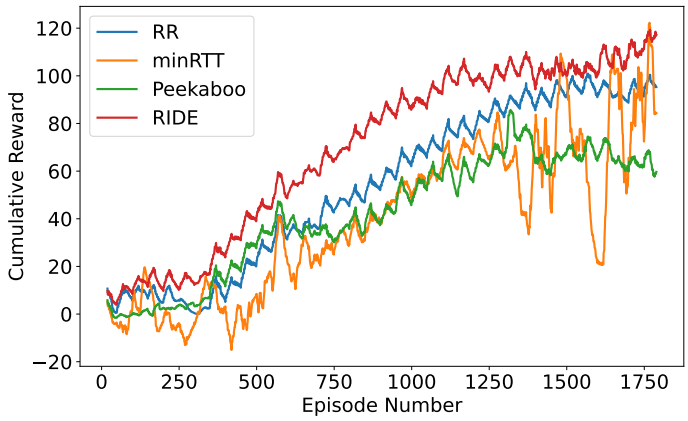


Figure 5.7: The cumulative reward on each episode.

relatively stable reward score, approximately 100, starting from the 1000th episode onwards. This stability underscores its robustness in handling network variations effectively.

Peekaboo has the similar design rationale as RIDE, the former adopts the idea of MAB which is a simpler version of RL algorithm, and the latter employs the action-critic framework. However, the experimental results demonstrate that RIDE substantially outperforms Peekaboo in dealing with 360-degree video streaming. Moreover, even the simplest RR algorithm achieves a higher reward than Peekaboo. The limitation of Peekaboo lies on the fact that the state space and reward design of Peekaboo do not match the demand for interactive streaming applications.

What's more, the significant reward fluctuation with minRTT states that solely considering the delay cannot guarantee a good user experience.

Peekaboo and RIDE share a similar design rationale: Peekaboo utilizes the Multi-Armed Bandit (MAB) approach, which is a simpler variant of RL, while RIDE employs a more complex actor-critic framework. Experimental results reveal that RIDE significantly outperforms Peekaboo in handling 360-degree video streaming. Interestingly, even the simple RR algorithm achieves higher rewards than Peekaboo. The primary limitation of Peekaboo stems from its state space and reward design, which do not truly match the demands of interactive streaming applications.

Furthermore, the significant reward fluctuation with minRTT illustrates that focusing solely on minimizing delay is insufficient to guarantee a good user experience.

2) Frame quality distribution: To quantify the viewing frame quality, we measure the viewing bitrate over the decoded data within a certain interval for each

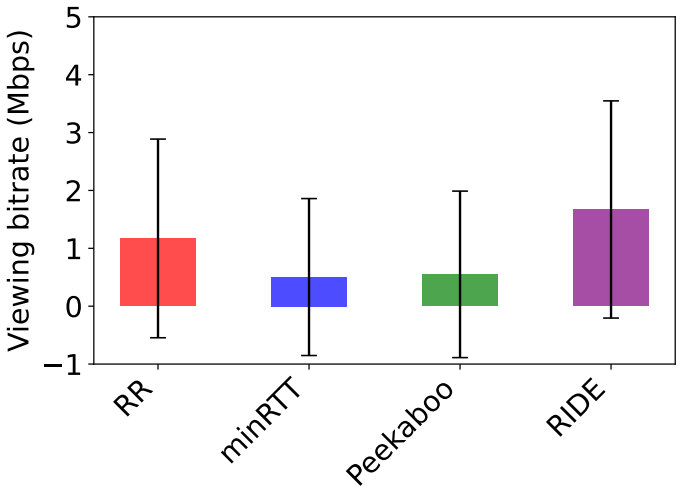


Figure 5.8: The viewing bitrate. The data counts the successfully decoded frames per second.

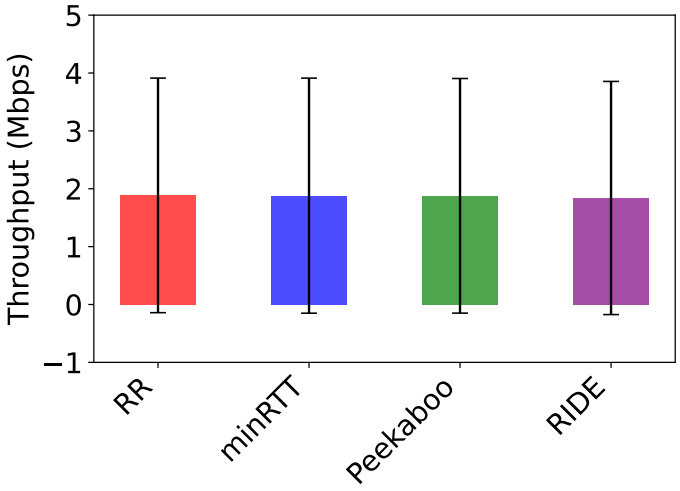


Figure 5.9: Throughput performance. The data counts the arriving data per second regardless of whether it has been decoded.

algorithm. By counting the received and decoded frame in bytes at the receiver every 100 ms, we have the median bitrate and its standard variation as shown in Fig. 5.8.

Fig. 5.8 illustrates that the proposed RIDE algorithm achieves the highest frame quality, averaging close to 2 Mbps. RR delivers the second-highest quality, while Peekaboo and minRTT exhibit lower performance. Comparatively, RIDE's average frame quality is approximately twice that of RR and four times greater than that of Peekaboo and minRTT.

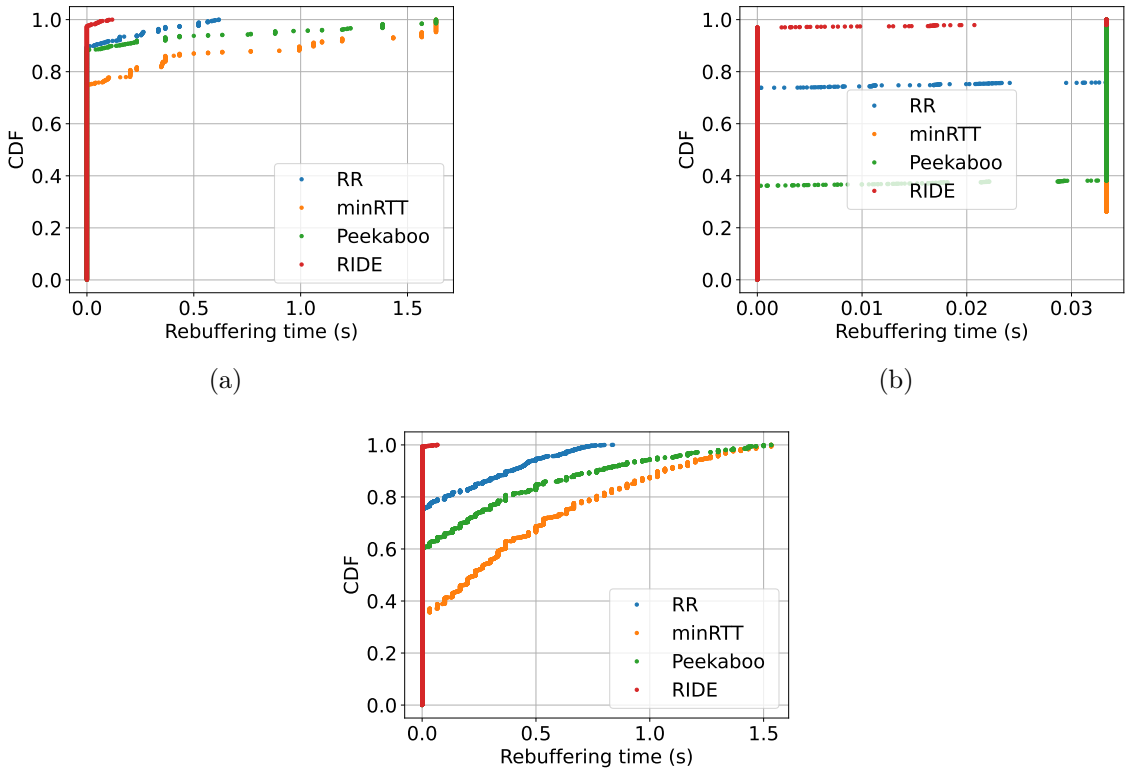


Figure 5.10: CDF of rebuffering time for different types of frames.

These results align with the trends of reward comparisons in Fig. 5.7, where RIDE also outperforms RR, Peekaboo, and minRTT. This consistency demonstrates that taking into account the EA and EP time duration in our reward design is effective and efficient in enhancing the averaging frame quality.

To gain insight into the difference between the decoded data rate and the arriving data rate, we further investigate the throughput data which is only related to the data arriving time and the amount of arrived data but not the data decoded time. Fig. 5.9 shows that all scheduling algorithms maintain the same level of throughput, as opposed to the results shown in Fig. 5.8. On the other hand, compared to benchmark algorithms, RIDE maintains the smallest gap between the viewing bitrate and throughput, which means superior bandwidth utility. Therefore, we conclude that it is important for the agent to understand the decoding dependency among frames and meet the frame deadline to guarantee good viewing quality.

3) CDF of rebuffering time: The rebuffering time is an important factor in evaluating the video playback smoothness. In the following, we will separately

evaluate the cumulative distribution function (CDF) of the rebuffering time that different video frames suffer from. The results are given in Fig. 5.10.

Fig. 5.10(a) presents the CDF results for I frames. Initially, we can determine the interruption rate for I frames by examining the proportion of data points where the rebuffering time equals zero. For RIDE, 96.8% of I frames do not experience rebuffering, resulting in an interruption rate of 3.2%. By comparison, the interruption rates for RR, Peekaboo, and minRTT are 10.3%, 14%, and 24.9%, respectively. Additionally, the maximum freezing time varies among the different scheduling algorithms: RIDE exhibits a maximum freezing time of up to 0.125 seconds, significantly less than Peekaboo and minRTT, which can reach freezing times as long as 1.6 seconds. This comparison highlights RIDE's superior performance in minimizing disruptions during video playback.

Since I frames are essential for the decoding of B and P frames, prolonged freezing times of I frames can result in further delays in decoding B and P frames, leading to a higher freezing ratio for these frames. Fig. 5.10(b) and 5.10(c) present the freezing results for B and P frames, respectively.

In Fig. 5.10(b), the delay primarily occurs in two extreme cases: either 0 or 0.033 seconds. The effective playback window for B frames is quite short, at most 0.033 seconds, making it likely that B frames either meet their deadline or miss their entire playback window. This binary outcome underscores the critical nature of precise scheduling and timely data transmission for maintaining smooth playback of B frames.

Fig. 5.10(b) demonstrates that with the RIDE algorithm, 97.4% of B frames experience no rebuffering and display smoothly. In contrast, the corresponding figures for RR, Peekaboo, and minRTT are 73.9%, 36.7%, and 26.3%, respectively. Similarly, Fig. 5.10(c) reveals that for P frames, the proportion experiencing zero rebuffering is 98.6% with RIDE, 75.1% with RR, 59.9% with Peekaboo, and only 35% with minRTT.

These statistics highlight RIDE's superior performance in minimizing rebuffering, ensuring a smoother and more consistent playback experience for both B and P frames.

4) Frame skipping ratio: In addition to path selection, our action space provides the option to skip sending a frame. We have analyzed the skipping ratio for I, B, and P frames using the RIDE algorithm, with the results presented in Table 5.1. It reveals that the skipping ratios for I, B, and P frames are similarly around

Table 5.1: The skipping action for I/B/P frames

	I	B	P
Total frames	1332	45252	17856
Skipped frames	30	862	356
Skip ratio	2.2%	2%	2%

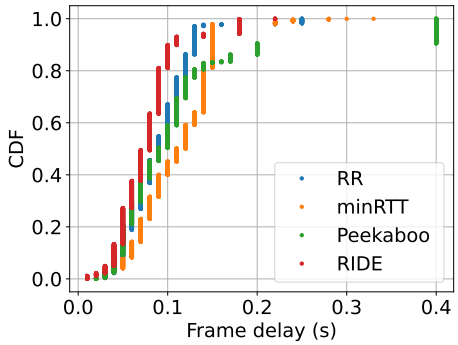


Figure 5.11: CDF of frame delay.

2%. Skipped video frames result in freezing, indicating that the zero rebuffering percentage for the RIDE algorithm in Fig. 5.10 should be adjusted downward by 2%. Consequently, the adjusted zero-rebuffering ratios for RIDE for I, B, and P frames are 96.8%, 95.4%, and 96.6%, respectively. Despite this adjustment, RIDE maintains its superior performance. This demonstrates that the option to skip frames can contribute to quality improvement without leading to significant rebuffering issues.

5) Delay distribution: Interactive streaming applications are highly sensitive to delays. To better understand this, we have analyzed the distribution of delays each frame endures under different scheduling algorithms.

Fig. 5.11 presents the CDF results of the latency experienced by each frame from the time it is scheduled to when it arrives at the receiver. We observe that the 95th percentile of frame latency for RIDE is 0.11 seconds, while for RR, Peekaboo, and minRTT, the corresponding latencies are 0.13, 0.22, and 0.15 seconds, respectively. This reduction in frame delay underscores the effectiveness and efficiency of RIDE's design.

5.6 Summary

Interactive 360-degree video is not only bandwidth-intensive but also highly sensitive to delays. Ensuring both high video quality and smooth playback experience remains a critical issue. In this chapter, we introduce a QoE-oRIented DEadline-driven (RIDE) algorithm for multipath scheduling at the tile level, designed to provide users with a satisfactory Quality of Experience (QoE). RIDE employs a dependency tree to understand deadlines for different types of frames and considers the negative impacts of FoV changes on scheduling decisions. Utilizing an actor-critic framework to train the neural network enables the scheduler agent to adapt to dynamic environments, including network and FoV dynamics. Evaluation results based on the real-world 360-degree video dataset show that RIDE can achieve up to four times higher frame quality and lower the interruption rate by 50% against benchmark algorithms.

Chapter 6

Conclusion

To make the visions of the 6G era a reality, in this dissertation, we conduct a series of research threads focusing on QoE-Oriented Multipath QUIC Protocol Design in 6G Mobile Networks.

The first thread introduces a mobility-aware multipath QUIC (MMQUIC) protocol, which comprehensively studies the negative impacts of mobility (considering both end-user and network mobility) on multipath scheduling and formulates the reordering delay in mobile environments as a minimization problem. Extensive experimental comparison shows that MMQUIC achieves substantial QoE gains. This line of research has led to multiple publications [124,160,163,164] in *IEEE/ACM Transactions on Networking (ToN)*, *IEEE Transactions on Wireless Communications (TWC)*, and *IEEE Globecom*. We are currently discussing the reordering issues within different paths. Mitigating the reordering issue within the same path will be a future research direction.

The second thread gives an analytical framework of MPQUIC-enabled LEO satellite networks (LSNs), and designs a multipath congestion control algorithm, which solves three major challenges in such context: bandwidth underutilization, inaccurate congestion signal, and unsatisfactory responsiveness. This line of research has led to multiple publications [161, 166] in *IEEE Transactions on Mobile Computing (TMC)*, and *IEEE MSN*. In future research, we will investigate new challenges to designing the congestion control algorithm for MPQUIC when the access network is heterogeneous, e.g., MPQUIC accesses to WiFi, 5G, unmanned aerial vehicle (UAV), and satellites simultaneously.

The third one designs a QoS-driven contextual MAB (QC-MAB) framework to support video streaming in mobile networks. Through access network selection and

forward error correction (FEC) configuration, QC-MAB can effectively mitigate the negative impact of mobility on QoS. This line of research has led to multiple (potential) publications [121,162] in *IEEE TMC*, and *IEEE VTC*. In future research, we will consider how to adapt the coding rate to the network conditions with the assistance of advanced learning tools.

Last but not least thread introduces a QoE-oriented Deadline-driven (RIDE) algorithm for multipath scheduling at the tile level, designed to provide users with a satisfactory Quality of Experience (QoE). In future research, we will jointly design the bit rate adaptation at the application layer and multipath scheduling at the transport layer to cope with more sophisticated scenarios.

Bibliography

[1] DASH-NS3. https://github.com/haraldott/dash.

[2] H.265. https://www.itu.int/rec/T-REC-H.265-202108-I/en.

[3] MPQUIC-ns3. https://github.com/ssjShirley/mpquic-ns3. Accessed: 2023-04-20.

[4] NS3 simulator. https://www.nsnam.org.

[5] The opencv library.

[6] quic-ns-3. https://github.com/signetlabdei/quic-ns-3.

[7] Valentine A Aalo, Constantine Mukasa, and George P Efthymoglou. Effect of mobility on the outage and BER performances of digital transmissions over Nakagami-m fading channels. *IEEE Transactions on Vehicular Technology*, 65(4):2715–2721, 2015.

[8] Shuchin Aeron and Venkatesh Saligrama. Wireless ad hoc networks: Strategies and scaling laws for the fixed snr regime. *IEEE Transactions on Information Theory*, 53(6):2044–2059, 2007.

[9] Mohamed A. Alrshah, Mohamed A. Al-Maqri, and Mohamed Othman. Elastic-TCP: Flexible Congestion Control Algorithm to Adapt for High-BDP Networks. *IEEE Systems Journal*, 13(2):1336–1346, 2019.

[10] Aziza Alzadjali, Flavio Esposito, and Jitender Deogun. A contextual bi-armed bandit approach for MPTCP path management in heterogeneous LTE and WiFi edge networks. In *2020 IEEE/ACM Symposium on Edge Computing (SEC)*, pages 307–316. IEEE, 2020.

[11] Ghassane Aniba and Sonia Aissa. Cross-layer designed adaptive modulation algorithm with packet combining and truncated ARQ over MIMO Nakagami fading channels. *IEEE Transactions on Wireless Communications*, 10(4):1026–1031, 2011.

[12] Venkat Arun and Hari Balakrishnan. Copa: Practical delay-based congestion control for the Internet. In *15th USENIX Symposium on Networked Systems Design and Implementation (NSDI 18)*, pages 329–342, 2018.

[13] Peter Auer, Nicolo Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47(2):235–256, 2002.

[14] Lin Cai, Yuanqian Luo, Siyuan Xiang, and Jianping Pan. Scalable modulation for scalable wireless videocast. In *2010 Proceedings IEEE INFOCOM*, pages 1–5. IEEE, 2010.

[15] Lin Cai, Jianping Pan, Wenjun Yang, Xiangyu Ren, and Xuemin Shen. Self-evolving and transformative (set) protocol architecture for 6g. *IEEE Wireless Communications*, 2022.

[16] Lin Cai, Xuemin Shen, and Jon W Mark. *Multimedia services in wireless internet: modeling and analysis*, volume 15. John Wiley & Sons, 2009.

[17] Lin Cai, Xuemin Shen, Jianping Pan, and Jon W Mark. Performance analysis of TCP-friendly AIMD algorithms for multimedia applications. *IEEE Transactions on Multimedia*, 7(2):339–355, 2005.

[18] Yu Cao, Mingwei Xu, and Xiaoming Fu. Delay-based congestion control for multipath TCP. In *20th IEEE International Conference on Network Protocols (ICNP)*, pages 1–10. IEEE, 2012.

[19] Neal Cardwell, Yuchung Cheng, C Stephen Gunn, Soheil Hassas Yeganeh, and Van Jacobson. BBR: Congestion-Based Congestion Control: Measuring bottleneck bandwidth and round-trip propagation time. *Queue*, 14(5):20–53, 2016.

[20] Neal Cardwell, Yuchung Cheng, C Stephen Gunn, Soheil Hassas Yeganeh, and Van Jacobson. BBR: Congestion-based congestion control. *Communications of the ACM*, 60(2):58–66, 2017.

[21] Kameswari Chebrolu and Ramesh Rao. Communication using multiple wireless interfaces. In *2002 IEEE Wireless Communications and Networking Conference Record. WCNC 2002 (Cat. No. 02TH8609)*, volume 1, pages 327–331. IEEE, 2002.

[22] Jinyu Chen, Xianzhuo Luo, Miao Hu, Di Wu, and Yipeng Zhou. Sparkle: User-aware viewport prediction in 360-degree video streaming. *IEEE Transactions on Multimedia*, 23:3853–3866, 2020.

[23] Nan Cheng, HE Jingchao, YIN Zhisheng, ZHOU Conghao, et al. 6G service-oriented space-air-ground integrated network: A survey. *Chinese Journal of Aeronautics*, 35(9):1–18, 2022.

[24] Federico Chiariotti, Stepan Kucera, Andrea Zanella, and Holger Claussen. Analysis and design of a latency control protocol for multi-path data delivery with pre-defined QoS guarantees. *IEEE/ACM Transactions on Networking*, 27(3):1165–1178, 2019.

[25] Inho Cho, Keon Jang, and Dongsu Han. Credit-Scheduled Delay-Bounded Congestion Control for Datacenters. In *Proceedings of the ACM SIGCOMM'17*, page 239–252, New York, NY, USA, 2017.

[26] Tran-Tuan Chu, Mohamed Aymen Labiod, Brice Augustin, and Abdelhamid Mellouk. GADaM on the road - Smart approach to multi-access networks: Analytical and practical evaluation in various urban mobile environments. In *2023 IEEE Wireless Communications and Networking Conference (WCNC)*, pages 1–6. IEEE, 2023.

[27] Federal Communications Commission. Raw data - measuring broadband america, 2016.

[28] Yong Cui, Lian Wang, Xin Wang, Hongyi Wang, and Yining Wang. FMTCP: A fountain code-based multipath transmission control protocol. *IEEE/ACM Transactions on Networking*, 23(2):465–478, 2014.

[29] Therdpong Daengsi and Pongpisit Wuttidittachotti. Quality of service as a baseline for 5G: A recent study of 4G network performance in Thailand. In *2020 IEEE International Conference on Communication, Networks and Satellite (Comnetsat)*, pages 395–399. IEEE, 2020.

[30] Wenying Dai, Hewu Li, Qian Wu, and Xiaomo Wang. NDM: Network Driving IP Mobility Support in Large Scale LEO Satellite Network. In *2020 IEEE Symposium on Computers and Communications (ISCC)*, pages 1–7, 2020.

[31] John Day, Ibrahim Matta, and Karim Mattar. Networking is IPC: a guiding principle to a better internet. In *Proceedings of the 2008 ACM CoNEXT Conference*, pages 1–6, Madrid, Spain, 2008.

[32] Quentin De Coninck and Olivier Bonaventure. Multipath QUIC: Design and evaluation. In *Proceedings of the 13th international conference on emerging networking experiments and technologies*, pages 160–166, 2017.

[33] Inigo Del Portillo, Bruce G Cameron, and Edward F Crawley. A technical comparison of three low earth orbit satellite constellation systems to provide global broadband. *Acta Astronautica*, 159:123–135, 2019.

[34] Shuo Deng, Ravi Netravali, Anirudh Sivaraman, and Hari Balakrishnan. WiFi, LTE, or both? Measuring multi-homed wireless internet performance. In *Proceedings of the 2014 Conference on Internet Measurement Conference*, pages 181–194, 2014.

[35] Jörg Deutschmann, Kai-Steffen Hielscher, and Reinhard German. Satellite Internet performance measurements. In *2019 International Conference on Networked Systems (NetSys)*, pages 1–4. IEEE, 2019.

[36] Enhuan Dong, Mingwei Xu, Xiaoming Fu, and Yu Cao. A loss aware MPTCP scheduler for highly lossy networks. *Computer Networks*, 157:146–158, 2019.

[37] Mo Dong, Tong Meng, Doron Zarchy, Engin Arslan, Yossi Gilad, Brighten Godfrey, and Michael Schapira. {PCC} vivace:{Online-Learning} congestion control. In *15th USENIX Symposium on Networked Systems Design and Implementation (NSDI 18)*, pages 343–356, 2018.

[38] Nandita Dukkipati, Tiziana Refice, Yuchung Cheng, Jerry Chu, Tom Herbert, Amit Agarwal, Arvind Jain, and Natalia Sutin. An argument for increasing TCP's initial congestion window. *ACM SIGCOMM Computer Communication Review*, 40(3):26–33, 2010.

[39] Dino Farinacci, Vince Fuller, David Meyer, and Darrel Lewis. Locator/id separation protocol (lisp). *IETF Network Working Group RFC 6830*, 2013.

[40] Emad Felemban, Chang-Gun Lee, and Eylem Ekici. MMSPEED: Multipath multi-SPEED protocol for QoS guarantee of reliability and timeliness in wireless sensor networks. *IEEE transactions on mobile computing*, 5(6):738–754, 2006.

[41] Simone Ferlin, Özgü Alay, Thomas Dreibholz, David A Hayes, and Michael Welzl. Revisiting congestion control for multipath TCP with shared bottleneck detection. In *Proceedings IEEE INFOCOM*, pages 1–9, 2016.

[42] Simone Ferlin, Özgü Alay, Olivier Mehani, and Roksana Boreli. BLEST: Blocking estimation-based MPTCP scheduler for heterogeneous networks. In *2016 IFIP Networking Conference (IFIP Networking) and Workshops*, pages 431–439. IEEE, 2016.

[43] Simone Ferlin, Özgü Alay, Olivier Mehani, and Roksana Boreli. BLEST: Blocking estimation-based MPTCP scheduler for heterogeneous networks. In *2016 IFIP Networking Conference (IFIP Networking) and Workshops*, pages 431–439. IEEE, 2016.

[44] Simone Ferlin, Thomas Dreibholz, and Özgü Alay. Multi-path transport over heterogeneous wireless networks: Does it really pay off? In *2014 IEEE Global Communications Conference*, pages 4807–4813. IEEE, 2014.

[45] Simone Ferlin-Oliveira, Thomas Dreibholz, and Özgü Alay. Tackling the challenge of bufferbloat in multi-path transport over heterogeneous wireless networks. In *2014 IEEE 22nd International Symposium of Quality of Service (IWQoS)*, pages 123–128. IEEE, 2014.

[46] Chad Fogg, Didier J LeGall, Joan L Mitchell, and William B Pennebaker. *MPEG Video Compression Standard*. Springer Science & Business Media, 2007.

[47] Alan Ford, Costin Raiciu, Mark Handley, and Olivier Bonaventure. RFC 6824: TCP extensions for multipath operation with multiple addresses. *Internet Engineering Task Force*, 2013.

[48] Sophie Fortin-Parisi and Bruno Sericola. A Markov model of TCP throughput, goodput and slow start. *Performance Evaluation*, 58(2–3):89–108, 2004.

[49] Fang Fu, Yunpeng Kang, Zhicai Zhang, F Richard Yu, and Tuan Wu. Soft actor–critic DRL for live transcoding and streaming in vehicular fog-computing-enabled IoV. *IEEE Internet of Things Journal*, 8(3):1308–1321, 2020.

[50] Shaojian Fu and Mohammed Atiquzzaman. Performance modeling of SCTP multihoming. In *GLOBECOM 2005-2005 IEEE Global Communications Conference*, volume 2, pages 6–11. IEEE, 2005.

[51] Kai Gao, Changqiao Xu, Ping Zhang, Jiuren Qin, Lujie Zhong, and Gabriel-Miro Muntean. GCH-MV: Game-Enhanced Compensation Handover Scheme for Multipath TCP in 6G Software Defined Vehicular Networks. *IEEE Transactions on Vehicular Technology*, 69(12):16142–16154, 2020.

[52] Andres Garcia-Saavedra, Mohammad Karzand, and Douglas J Leith. Low delay random linear coding and scheduling over multiple interfaces. *IEEE Transactions on Mobile Computing*, 16(11):3100–3114, 2017.

[53] Ehab Ghabashneh, Chandan Bothra, Ramesh Govindan, Antonio Ortega, and Sanjay Rao. Dragonfly: Higher perceptual quality for continuous 360 video playback. In *Proceedings of the ACM SIGCOMM 2023 Conference*, pages 516–532, 2023.

[54] Marco Giordani and Michele Zorzi. Non-terrestrial networks in the 6G era: Challenges and opportunities. *IEEE Network*, 35(2):244–251, 2020.

[55] Andrea Goldsmith. *Wireless communications*. Cambridge university press, 2005.

[56] Yu Guan, Chengyuan Zheng, Xinggong Zhang, Zongming Guo, and Junchen Jiang. Pano: Optimizing 360 video streaming with a better understanding of quality perception. In *Proceedings of the ACM Special Interest Group on Data Communication*, pages 394–407. 2019.

[57] Yushuo Guan, Yuanxing Zhang, Bingxuan Wang, Kaigui Bian, Xiaoliang Xiong, and Lingyang Song. PERM: Neural adaptive video streaming with multi-path transmission. In *IEEE INFOCOM 2020-IEEE Conference on Computer Communications*, pages 1103–1112. IEEE, 2020.

[58] Yihua Ethan Guo, Ashkan Nikravesh, Z Morley Mao, Feng Qian, and Subhabrata Sen. Accelerating multipath transport through balanced subflow completion. In *Proceedings of the 23rd Annual International Conference on Mobile Computing and Networking*, pages 141–153, 2017.

[59] Daniel Halperin, Wenjun Hu, Anmol Sheth, and David Wetherall. Predictable 802.11 packet delivery from wireless channel measurements. *ACM SIGCOMM Computer Communication Review*, 40(4):159–170, 2010.

[60] Ryan Hamilton, Jana Iyengar, Ian Swett, and Alyssa Wilk. QUIC: A UDP-Based Secure and Reliable Transport for HTTP/2. Internet-Draft draft-hamilton-early-deployment-quic-00, Internet Engineering Task Force, July 2016. Work in Progress.

[61] Bo Han, Feng Qian, Lusheng Ji, and Vijay Gopalakrishnan. MP-DASH: Adaptive video streaming over preference-aware multipath. In *Proceedings of the 12th International on Conference on emerging Networking EXperiments and Technologies*, pages 129–143, 2016.

[62] Wassily Hoeffding. Probability inequalities for sums of bounded random variables. In *The collected works of Wassily Hoeffding*, pages 409–426. Springer, 1994.

[63] Jinhui Huang, Yingxue Su, Wenxiang Liu, and Feixue Wang. Adaptive modulation and coding techniques for global navigation satellite system inter-satellite communication based on the channel condition. *Iet Communications*, 10(16):2091–2095, 2016.

[64] Te-Yuan Huang, Ramesh Johari, Nick McKeown, Matthew Trunnell, and Mark Watson. A buffer-based approach to rate adaptation: Evidence from a large video streaming service. In *Proceedings of the 2014 ACM conference on SIGCOMM*, pages 187–198, 2014.

[65] Per Hurtig, Karl-Johan Grinnemo, Anna Brunstrom, Simone Ferlin, Özgü Alay, and Nicolas Kuhn. Low-latency scheduling in MPTCP. *IEEE/ACM Transactions on Networking*, 27(1):302–315, 2019.

[66] Vatche Ishakian, Joseph Akinwumi, Flavio Esposito, and Ibrahim Matta. On supporting mobility and multihoming in recursive internet architectures. *Computer Communications*, 35(13):1561–1573, 2012.

[67] Jana Iyengar and Martin Thomson. QUIC: A UDP-Based Multiplexed and Secure Transport. RFC 9000, May 2021.

[68] Janardhan R Iyengar, Paul D Amer, and Randall Stewart. Concurrent multipath transfer using SCTP multihoming over independent end-to-end paths. *IEEE/ACM Transactions on networking*, 14(5):951–964, 2006.

[69] Van Jacobson. Congestion avoidance and control. *ACM SIGCOMM computer communication review*, 18(4):314–329, 1988.

[70] Van Jacobson, Diana K Smetters, James D Thornton, Michael F Plass, Nicholas H Briggs, and Rebecca L Braynard. Networking named content. In *Proceedings of CoNEXT'09*, pages 1–12. ACM, 2009.

[71] Kyle Jamieson and Hari Balakrishnan. PPR: Partial packet recovery for wireless networks. *ACM SIGCOMM Computer Communication Review*, 37(4):409–420, 2007.

[72] Nathan Jay, Noga Rotman, Brighten Godfrey, Michael Schapira, and Aviv Tamar. A deep reinforcement learning perspective on Internet congestion control. In *International Conference on Machine Learning*, pages 3050–3059. PMLR, 2019.

[73] Ziye Jia, Min Sheng, Jiandong Li, and Zhu Han. Toward data collection and transmission in 6G space–air–ground integrated networks: Cooperative HAP and LEO satellite schemes. *IEEE Internet of Things Journal*, 9(13):10516–10528, 2021.

[74] Baptiste Jonglez, Martin Heusse, and Bruno Gaujal. SRPT-ECF: challenging Round-Robin for stream-aware multipath scheduling. In *2020 IFIP Networking Conference (Networking)*, pages 719–724. IEEE, 2020.

[75] Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. 1960.

[76] Mohammad Karzand and Douglas J Leith. Low delay random linear coding over a stream. In *2014 52nd Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 521–528. IEEE, 2014.

[77] Ramin Khalili, Nicolas Gast, Miroslav Popovic, and Jean-Yves Le Boudec. MPTCP is not Pareto-optimal: Performance issues and a possible solution. *IEEE/ACM Transactions on Networking*, 21(5):1651–1665, 2013.

[78] Ramin Khalili, Nicolas Gast, Miroslav Popovic, and Jean-Yves Le Boudec. MPTCP is not Pareto-optimal: Performance issues and a possible solution. *IEEE/ACM Transactions On Networking*, 21(5):1651–1665, 2013.

[79] Eddie Kohler, Mark Handley, and Sally Floyd. Designing DCCP: Congestion control without reliability. *ACM SIGCOMM Computer Communication Review*, 36(4):27–38, 2006.

[80] Teemu Koponen, Mohit Chawla, Byung-Gon Chun, Andrey Ermolinskiy, Kye Hyun Kim, Scott Shenker, and Ion Stoica. A data-oriented (and beyond) network architecture. In *Proceedings of the ACM SIGCOMM 2007 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 181–192, 2007.

[81] Sastri Kota and Giovanni Giambene. 6G integrated non-terrestrial networks: Emerging technologies and challenges. In *2021 IEEE International Conference on Communications Workshops (ICC Workshops)*, pages 1–6. IEEE, 2021.

[82] Changsung Lee, Jaewook Jung, and Jong-Moon Chung. DEFT: Multipath TCP for high speed low latency communications in 5G networks. *IEEE Transactions on Mobile Computing*, 20(12):3311–3323, 2020.

[83] Jie Li, Ling Han, Chong Zhang, Qiyue Li, and Zhi Liu. Spherical convolution empowered viewport prediction in 360 video multicast with limited FoV feedback. *ACM Transactions on Multimedia Computing, Communications and Applications*, 19(1):1–23, 2023.

[84] Ming Li, Andrey Lukyanenko, and Yong Cui. Network coding based multipath TCP. In *2012 proceedings IEEE INFOCOM workshops*, pages 25–30. IEEE, 2012.

[85] Richard Li. New IP: Going beyond the Limits of the Internet. In *a Keynote Speech at IEEE Globecom 2019, Hawaii*, 2019.

[86] Yuliang Li, Rui Miao, Hongqiang Harry Liu, Yan Zhuang, Fei Feng, Lingbo Tang, Zheng Cao, Ming Zhang, Frank Kelly, Mohammad Alizadeh, et al. HPCC: High precision congestion control. In *Proceedings of the ACM Special Interest Group on Data Communication*, pages 44–58. 2019.

[87] Xiaoyu Liu, Ting Ma, Zhixuan Tang, Xiaohan Qin, Haibo Zhou, and Xuemin Sherman Shen. UltraStar: A lightweight simulator of ultra-dense LEO satellite constellation networking for 6G. *IEEE/CAA Journal of Automatica Sinica*, 10(3):632–645, 2023.

[88] Yanmei Liu, Yunfei Ma, Quentin De Coninck, Olivier Bonaventure, Christian Huitema, and Mirja Kühlewind. Multipath Extension for QUIC. Internet-Draft draft-ietf-quic-multipath-03, Internet Engineering Task Force. Work in Progress.

[89] Yi Liu, Li Jiang, Qi Qi, Kan Xie, and Shengli Xie. Online computation offloading for collaborative space/aerial-aided edge computing toward 6G system. *IEEE Transactions on Vehicular Technology*, 73(2):2495–2505, 2024.

[90] Wen-Chih Lo, Ching-Ling Fan, Jean Lee, Chun-Ying Huang, Kuan-Ta Chen, and Cheng-Hsin Hsu. 360 video viewing dataset in head-mounted virtual reality. In *Proceedings of the 8th ACM on Multimedia Systems Conference*, pages 211–216, 2017.

[91] Ralf Lübben and Johannes Morgenroth. An odd couple: Loss-based congestion control and minimum RTT scheduling in MPTCP. In *IEEE 44th Conference on Local Computer Networks (LCN)*, pages 300–307. IEEE, 2019.

[92] Gerui Lv, Qinghua Wu, Weiran Wang, Zhenyu Li, and Gaogang Xie. Lumos: Towards better video streaming QoE through accurate throughput prediction. In *IEEE INFOCOM 2022 - IEEE Conference on Computer Communications*, pages 650–659, 2022.

[93] Anahita Mahzari, Afshin Taghavi Nasrabadi, Aliehsan Samiei, and Ravi Prakash. FoV-aware edge caching for adaptive 360 video streaming. In *Proceedings of the 26th ACM international conference on Multimedia*, pages 173–181, 2018.

[94] Hongzi Mao, Ravi Netravali, and Mohammad Alizadeh. Neural adaptive video streaming with Pensieve. In *Proceedings of the conference of the ACM special interest group on data communication*, pages 197–210, 2017.

[95] Yixiang Mao, Liyang Sun, Yong Liu, and Yao Wang. Low-latency FoV-adaptive coding and streaming for interactive 360 video streaming. In *Proceedings of the 28th ACM International Conference on Multimedia*, pages 3696–3704, 2020.

[96] Robin Marx, Joris Herbots, Wim Lamotte, and Peter Quax. Same standards, different decisions: A study of QUIC and HTTP/3 implementation diversity. In *Proceedings of the Workshop on the Evolution, Performance, and Interoperability of QUIC*, pages 14–20, 2020.

[97] James McCauley, Yotam Harchol, Aurojit Panda, Barath Raghavan, and Scott Shenker. Enabling a permanent revolution in internet architecture. In *Proceedings of the ACM Special Interest Group on Data Communication*, pages 1–14. 2019.

[98] Radhika Mittal, Vinh The Lam, Nandita Dukkipati, Emily Blem, Hassan Wassel, Monia Ghobadi, Amin Vahdat, Yaogong Wang, David Wetherall, and David Zats. TIMELY: RTT-based congestion control for the datacenter. *ACM SIGCOMM Computer Communication Review*, 45(4):537–550, 2015.

[99] Behnam Montazeri, Yilong Li, Mohammad Alizadeh, and John Ousterhout. Homa: A receiver-driven low-latency transport protocol using network priorities. In *Proceedings of the ACM SIGCOMM'18*, pages 221–235, 2018.

[100] Ashkan Nikravesh, Yihua Guo, Feng Qian, Z Morley Mao, and Subhabrata Sen. An in-depth understanding of multipath TCP on mobile devices: Measurement and system design. In *Proceedings of the 22nd Annual International Conference on Mobile Computing and Networking*, pages 189–201, 2016.

[101] Hyun Woo Oh, In Tark Han, Kwang Roh Park, and Sang Ha Kim. An enhanced multi-path scheme for QoS guarantee in wireless sensor network. In *2007 IEEE International Symposium on Consumer Electronics*, pages 1–5, 2007.

[102] Ogutu B Osoro and Edward J Oughton. A techno-economic framework for satellite networks applied to low earth orbit constellations: Assessing Starlink, OneWeb and Kuiper. *IEEE Access*, 9:141611–141625, 2021.

[103] Harald Ott, Konstantin Miller, and Adam Wolisz. Simulation framework for HTTP-based adaptive streaming applications. In *Proceedings of the Workshop on ns-3*, pages 95–102, 2017.

[104] Christoph Paasch, Simone Ferlin, Ozgu Alay, and Olivier Bonaventure. Experimental evaluation of multipath TCP schedulers. In *Proceedings of the 2014 ACM SIGCOMM workshop on Capacity sharing workshop*, pages 27–32, 2014.

[105] Jitendra Padhye, Victor Firoiu, Don Towsley, and Jim Kurose. Modeling TCP throughput: A simple model and its empirical validation. In *Proceedings of the ACM SIGCOMM'98 conference on Applications, technologies, architectures, and protocols for computer communication*, pages 303–314, 1998.

[106] Jitendra Padhye, Victor Firoiu, Don Towsley, and Jim Kurose. Modeling TCP throughput: A simple model and its empirical validation. In *Proceedings of the ACM SIGCOMM'98 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, pages 303–314, 1998.

[107] Jianli Pan, Raj Jain, Subharthi Paul, and Chakchai So-In. MILSA: A new evolutionary architecture for scalability, mobility, and multihoming in the future internet. *IEEE Journal on Selected Areas in Communications*, 28(8):1344–1362, 2010.

[108] Jianping Pan, Jinwei Zhao, and Lin Cai. Measuring a low-earth-orbit satellite network. *2023 International Symposium on Personal, Indoor and Mobile Radio Communications*, 2023.

[109] Qiuyu Peng, Anwar Walid, Jaehyun Hwang, and Steven H. Low. Multipath TCP: Analysis, Design, and Implementation. *IEEE/ACM Transactions on Networking*, 24(1):596–609, 2016.

[110] Shiva Raj Pokhrel and Michel Mandjes. Improving multipath TCP performance over WiFi and cellular networks: An analytical approach. *IEEE Transactions on Mobile Computing*, 18(11):2562–2576, 2018.

[111] Otilia Popescu. Power budgets for cubesat radios to support ground communications and inter-satellite links. *IEEE Access*, 5:12618–12625, 2017.

[112] Alexander Rabitsch, Per Hurtig, and Anna Brunstrom. A stream-aware multipath QUIC scheduler for heterogeneous paths. In *Proceedings of the Workshop on the Evolution, Performance, and Interoperability of QUIC*, pages 29–35, 2018.

[113] Costin Raiciu and Gianni Antichi. NDP: Rethinking datacenter networks and stacks two years after. *ACM SIGCOMM Computer Communication Review*, 49(5):112–114, 2019.

[114] Costin Raiciu, Christoph Paasch, Sebastien Barre, Alan Ford, Michio Honda, Fabien Duchene, Olivier Bonaventure, and Mark Handley. How hard can it be? Designing and implementing a deployable multipath TCP. In *9th USENIX symposium on networked systems design and implementation (NSDI 12)*, pages 399–412, 2012.

[115] Murali R Rajamani and James B Rawlings. Estimation of the disturbance structure from data using semidefinite programming and optimal weighting. *Automatica*, 45(1):142–148, 2009.

[116] Dipankar Raychaudhuri, Kiran Nagaraja, and Arun Venkataramani. Mobilityfirst: a robust and trustworthy mobility-centric architecture for the future internet. *ACM SIGMOBILE Mobile Computing and Communications Review*, 16(3):2–13, 2012.

[117] Haakon Riiser, Paul Vigmostad, Carsten Griwodz, and Pål Halvorsen. Commute path bandwidth traces from 3G networks: analysis and applications. In *Proceedings of the 4th ACM Multimedia Systems Conference*, page 114–118, New York, NY, USA, 2013. Association for Computing Machinery.

[118] Mattia Sandri, Matteo Pagin, Marco Giordani, and Michele Zorzi. Implementation of a channel model for non-terrestrial networks in ns-3. *arXiv preprint arXiv:2305.05544*, 2023.

[119] Golam Sarwar, Roksana Boreli, Emmanuel Lochin, Ahlem Mifdaoui, and Guillaume Smith. Mitigating receiver's buffer blocking by delay aware packet scheduling in multipath data transfer. In *2013 27th International Conference on Advanced Information Networking and Applications Workshops*, pages 1119–1124. IEEE, 2013.

[120] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal Policy Optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[121] Amir Sepahi, Lin Cai, Wenjun Yang, and Jianping Pan. Meta-DAMS: Delay-aware multipath scheduler using hybrid meta reinforcement learning. In *2023 IEEE 98th Vehicular Technology Conference (VTC2023-Fall)*, pages 1–5, 2023.

[122] Rishad Ahmed Shafik, Md Shahriar Rahman, and AHM Razibul Islam. On the extended relationships among EVM, BER and SNR as performance metrics. In *2006 International Conference on Electrical and Computer Engineering*, pages 408–411. IEEE, 2006.

[123] Hang Shi, Lei Zhang, Xutong Zuo, Qian Wu, Hewu Li, and Yong Cui. Multipath Deadline-Aware Transport Proxy for Space Network. *IEEE Internet Computing*, 25(6):51–57, 2021.

[124] Shengjie Shu, Wenjun Yang, Jianping Pan, and Lin Cai. A multipath extension to the QUIC module for ns-3. In *Proceedings of the 2023 Workshop on ns-3*, pages 86–93, 2023.

[125] Michael Simon, Ebenezer Kofi, Louis Libin, and Mark Aitken. ATSC 3.0 broadcast 5G unicast heterogeneous network converged services starting release 16. *IEEE Transactions on Broadcasting*, 66(2):449–458, 2020.

[126] Congxi Song, Biao Han, Xiaolan Ji, Yahui Li, and Jinshu Su. AI-driven multipath transmission: Empowering UAV-based live streaming. *IEEE Network*, 2023.

[127] Wei Song and Weihua Zhuang. Performance analysis of probabilistic multipath transmission of video streaming traffic over multi-radio wireless devices. *IEEE Transactions on Wireless Communications*, 11(4):1554–1564, 2012.

[128] Thomas Stockhammer. Dynamic adaptive streaming over HTTP–standards and design principles. In *Proceedings of the second annual ACM conference on Multimedia systems*, pages 133–144, 2011.

[129] Srikanth Sundaresan, Nick Feamster, and Renata Teixeira. Home network or access link? locating last-mile downstream throughput bottlenecks. In *International Conference on Passive and Active Network Measurement*, pages 111–123. Springer, 2016.

[130] Suramya Tomar. Converting video formats with FFmpeg. *Linux journal*, 2006(146):10, 2006.

[131] Tobias Viernickel, Alexander Froemmgen, Amr Rizk, Boris Koldehofe, and Ralf Steinmetz. Multipath QUIC: A deployable multipath transport protocol. In *2018 IEEE International Conference on Communications (ICC)*, pages 1–7. IEEE, 2018.

[132] Vu Anh Vu and Brenton Walker. On the latency of multipath-QUIC in real-time applications. In *2020 16th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, pages 1–7. IEEE, 2020.

[133] Jing Wang, Yunfeng Gao, and Chenren Xu. A multipath QUIC scheduler for mobile HTTP/2. In *Proceedings of the 3rd Asia-Pacific Workshop on Networking 2019*, pages 43–49, 2019.

[134] Jingyu Wang, Jianxin Liao, and Xiaomin Zhu. Latent Handover: A flow-oriented progressive handover mechanism. *Computer Communications*, 31(10):2319–2340, 2008.

[135] Ren Wang, Massimo Valla, MY Sanadidi, and Mario Gerla. Adaptive bandwidth share estimation in TCP Westwood. In *Global Telecommunications Conference, 2002. GLOBECOM'02. IEEE*, volume 3, pages 2604–2608. IEEE, 2002.

[136] Yanmin Wang, Wei Feng, Jue Wang, and Tony QS Quek. Hybrid satellite-UAV-terrestrial networks for 6G ubiquitous coverage: A maritime communications perspective. *IEEE Journal on Selected Areas in Communications*, 39(11):3475–3490, 2021.

[137] Wenjia Wei, Kaiping Xue, Jiangping Han, David SL Wei, and Peilin Hong. Shared bottleneck-based congestion control and packet scheduling for multipath TCP. *IEEE/ACM Transactions on Networking*, 28(2):653–666, 2020.

[138] Wikipedia. Transponder (satellite communications), 2023.

[139] Hongjia Wu, Özgü Alay, Anna Brunstrom, Simone Ferlin, and Giuseppe Caso. Peekaboo: Learning-based multipath scheduling for dynamic heterogeneous environments. *IEEE Journal on Selected Areas in Communications*, 38(10):2295–2310, 2020.

[140] Jiyan Wu, Bo Cheng, Chau Yuen, Ngai-Man Cheung, and Junliang Chen. Trading delay for distortion in one-way video communication over the internet. *IEEE*

Transactions on Circuits and Systems for Video Technology, 26(4):711–723, 2015.

[141] Jiyan Wu, Bo Cheng, Chau Yuen, Yanlei Shang, and Junliang Chen. Distortion-aware concurrent multipath transfer for mobile video streaming in heterogeneous wireless networks. *IEEE Transactions on Mobile Computing*, 14(4):688–701, 2014.

[142] Jiyan Wu, Chau Yuen, Bo Cheng, Yanlei Shang, and Junliang Chen. Goodput-aware load distribution for real-time traffic over multipath networks. *IEEE Transactions on Parallel and Distributed Systems*, 26(8):2286–2299, 2014.

[143] Jiyan Wu, Chau Yuen, Bo Cheng, Ming Wang, and Junliang Chen. Streaming high-quality mobile video with multipath TCP in heterogeneous wireless networks. *IEEE Transactions on Mobile Computing*, 15(9):2345–2361, 2015.

[144] Jiyan Wu, Chau Yuen, Ming Wang, and Junliang Chen. Content-aware concurrent multipath transfer for high-definition video streaming over heterogeneous wireless networks. *IEEE Transactions on Parallel and Distributed Systems*, 27(3):710–723, 2015.

[145] Han Xiao, Changqiao Xu, Zichen Feng, Renjie Ding, Shujie Yang, Lujie Zhong, Jie Liang, and Gabriel-Miro Muntean. A transcoding-enabled 360 VR video caching and delivery framework for edge-enhanced next-generation wireless networks. *IEEE Journal on Selected Areas in Communications*, 40(5):1615–1631, 2022.

[146] Chao Xie, Hefei Hu, and Yuanan Liu. Shared bottleneck detection for multipath transmission in high latency satellite network. In *2019 IEEE 7th International Conference on Computer Science and Network Technology (ICCSNT)*, pages 38–42. IEEE, 2019.

[147] Lan Xie, Zhimin Xu, Yixuan Ban, Xinggong Zhang, and Zongming Guo. 360probdash: Improving qoe of 360 video streaming using tile-based http adaptive streaming. In *Proceedings of the 25th ACM international conference on Multimedia*, pages 315–323, 2017.

[148] Min Xing, Siyuan Xiang, and Lin Cai. A real-time adaptive algorithm for video streaming over multiple wireless access networks. *IEEE Journal on Selected Areas in communications*, 32(4):795–805, 2014.

[149] Yitao Xing, Kaiping Xue, Yuan Zhang, Jiangping Han, et al. A stream-aware MPQUIC scheduler for HTTP traffic in mobile networks. *IEEE Transactions on Wireless Communications*, 22(4):2775–2788, 2023.

[150] Yitao Xing, Kaiping Xue, Yuan Zhang, Jiangping Han, Jian Li, Jianqing Liu, and Ruidong Li. A low-latency MPTCP scheduler for live video streaming in mobile networks. *IEEE Transactions on Wireless Communications*, 20(11):7230–7242, 2021.

[151] Yitao Xing, Kaiping Xue, Yuan Zhang, Jiangping Han, Jian Li, and David SL Wei. An online learning assisted packet scheduler for MPTCP in mobile networks. *IEEE/ACM Transactions on Networking*, 31(5):2297–2312, 2023.

[152] Huihui Xu, Deshi Li, Mingliu Liu, Guangjie Han, Wei Huang, and Chan Xu. Qoe-driven intelligent handover for user-centric mobile satellite networks. *IEEE Transactions on Vehicular Technology*, 69(9):10127–10139, 2020.

[153] Jianfeng Xu, Liming Wang, Chen Song, Ding Tang, and Zhen Xu. Toward Bandwidth-efficient Data Distribution in Satellite Networks. In *Proceedings of the 2018 International Conference on Transportation & Logistics, Information & Communication, Smart City (TLICSC 2018)*, pages 393–399. Atlantis Press, 2018/11.

[154] Kaiping Xue, Jiangping Han, Dan Ni, Wenjia Wei, Ying Cai, Qing Xu, and Peilin Hong. DPSAF: forward prediction based dynamic packet scheduling and adjusting with feedback for multipath TCP in lossy heterogeneous networks. *IEEE Transactions on Vehicular Technology*, 67(2):1521–1534, 2017.

[155] Francis Y Yan, Hudson Ayers, Chenzhi Zhu, Sadjad Fouladi, James Hong, Keyi Zhang, Philip Levis, and Keith Winstein. Learning in situ: A randomized experiment in video streaming. In *17th USENIX Symposium on Networked Systems Design and Implementation (NSDI 20)*, pages 495–511, 2020.

[156] Li Yan, Xuming Fang, and Yuguang Fang. A Novel Network Architecture for C/U-Plane Staggered Handover in 5G Decoupled Heterogeneous Railway

Wireless Systems. *IEEE Transactions on Intelligent Transportation Systems*, 18(12):3350–3362, 2017.

[157] Fan Yang, Qi Wang, and Paul D Amer. Out-of-order transmission for in-order arrival scheduling for multipath TCP. In *2014 28th International Conference on Advanced Information Networking and Applications Workshops*, pages 749–752. IEEE, 2014.

[158] Kemeng Yang, Iqbal Gondal, Bin Qiu, and Laurence S Dooley. Combined SINR based vertical handoff algorithm for next generation heterogeneous wireless networks. In *IEEE GLOBECOM 2007-IEEE Global Telecommunications Conference*, pages 4483–4487. IEEE, 2007.

[159] Wenjun Yang, Lin Cai, Shengjie Shu, and Jianping Pan. Scheduler design for mobility-aware multipath QUIC. In *GLOBECOM 2022-2022 IEEE Global Communications Conference*, pages 2849–2854. IEEE, 2022.

[160] Wenjun Yang, Lin Cai, Shengjie Shu, and Jianping Pan. Scheduler design for mobility-aware multipath QUIC. In *GLOBECOM 2022 - 2022 IEEE Global Communications Conference*, pages 2849–2854, 2022.

[161] Wenjun Yang, Lin Cai, Shengjie Shu, and Jianping Pan. Mobility-aware congestion control for multipath QUIC in integrated terrestrial satellite networks. *IEEE Transactions on Mobile Computing*, 2024.

[162] Wenjun Yang, Lin Cai, Shengjie Shu, Jianping Pan, and Zhiming Huang. Qos-driven contextual MAB for MPQUIC supporting video streaming in mobile networks. Submitted to *IEEE TMC*, major revision.

[163] Wenjun Yang, Lin Cai, Shengjie Shu, Jianping Pan, and Amir Sepahi. MAMS: Mobility-aware multipath scheduler for MPQUIC. *IEEE/ACM Transactions on Networking*, pages 1–16, 2024.

[164] Wenjun Yang, Pingping Dong, Lin Cai, and Wensheng Tang. Loss-aware throughput estimation scheduler for multi-path TCP in heterogeneous wireless networks. *IEEE Transactions on Wireless Communications*, 20(5):3336–3349, 2021.

[165] Wenjun Yang, Pingping Dong, Wensheng Tang, Xiaoping Lou, Hangjun Zhou, Kai Gao, and Haodong Wang. A mptcp scheduler for web transfer. *Comput. Mater. Continua*, 57(2):205–222, 2018.

[166] Wenjun Yang, Shengjie Shu, Lin Cai, and Jianping Pan. MM-QUIC: Mobility-aware multipath QUIC for satellite networks. In *17th International Conference on Mobility, Sensing and Networking (MSN)*, pages 608–615. IEEE, 2021.

[167] Xinlei Yang, Hao Lin, Zhenhua Li, Feng Qian, Xingyao Li, Zhiming He, Xudong Wu, Xianlong Wang, Yunhao Liu, Zhi Liao, et al. Mobile access bandwidth in practice: Measurement, analysis, and implications. In *Proceedings of the ACM SIGCOMM 2022 Conference*, pages 114–128, 2022.

[168] Xiaoqi Yin, Abhishek Jindal, Vyas Sekar, and Bruno Sinopoli. A control-theoretic approach for dynamic adaptive video streaming over HTTP. In *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication*, pages 325–338, 2015.

[169] Han Zhang, Wenzhong Li, Shaohua Gao, Xiaoliang Wang, and Baoliu Ye. ReLeS: A neural adaptive multipath scheduler based on deep reinforcement learning. In *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*, pages 1648–1656. IEEE, 2019.

[170] Lixia Zhang, Alexander Afanasyev, Jeffrey Burke, Van Jacobson, KC Claffy, Patrick Crowley, Christos Papadopoulos, Lan Wang, and Beichuan Zhang. Named data networking. *ACM SIGCOMM Computer Communication Review*, 44(3):66–73, 2014.

[171] Xingjun Zhang, Xuan Zhang, Zhe Fu, Bocheng Yu, and Scott Fowler. Joint distortion estimation and layer selection of unequal error protection for svc video transmission over fso networks. In *IEEE 21st International Conference on High Performance Computing and Communications; IEEE 17th International Conference on Smart City; IEEE 5th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*, pages 726–733, 2019.

[172] Yuanxing Zhang, Pengyu Zhao, Kaigui Bian, Yunxin Liu, Lingyang Song, and Xiaoming Li. DRL360: 360-degree video streaming with deep reinforcement learning. In *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*, pages 1252–1260. IEEE, 2019.

[173] Zhenning Zhang, Baokang Zhao, Zhenqian Feng, Wanrong Yu, and Chunqing Wu. Msn: A mobility-enhanced satellite network architecture: Poster. In *Proceedings of the 22nd Annual International Conference on Mobile Computing and Networking*, MobiCom '16, page 465–466, New York, NY, USA, 2016. Association for Computing Machinery.

[174] Zhilong Zheng, Yunfei Ma, Yanmei Liu, Furong Yang, Zhenyu Li, Yuanbo Zhang, Jiuhai Zhang, Wei Shi, Wentao Chen, Ding Li, et al. Xlink: Qoe-driven multi-path quic transport in large-scale video services. In *Proceedings of the 2021 ACM SIGCOMM 2021 Conference*, pages 418–432, 2021.

[175] Di Zhou, Min Sheng, Jiandong Li, and Zhu Han. Aerospace integrated networks innovation for empowering 6G: A survey and future challenges. *IEEE Communications Surveys & Tutorials*, 2023.

[176] Dizhi Zhou, Wei Song, and Minghui Shi. Goodput improvement for multipath TCP by congestion window adaptation in multi-radio devices. In *2013 IEEE 10th Consumer Communications and Networking Conference (CCNC)*, pages 508–514. IEEE, 2013.

[177] Xiaoqing Zhu, Rong Pan, Mythili S Prabhu, Nandita Dukkipati, Vijay Subramanian, and Flavio Bonomi. Layered internet video adaptation (LIVA): Network-assisted bandwidth sharing and transient loss protection for video streaming. *IEEE Transactions on Multimedia*, 13(4):720–732, 2011.

[178] Yibo Zhu, Haggai Eran, Daniel Firestone, Chuanxiong Guo, Marina Lipshteyn, Yehonatan Liron, Jitendra Padhye, Shachar Raindel, Mohamad Haj Yahia, and Ming Zhang. Congestion control for large-scale RDMA deployments. *ACM SIGCOMM Computer Communication Review*, 45(4):523–536, 2015.

[179] Xutong Zuo, Yong Cui, Xin Wang, and Jiayu Yang. Deadline-aware multipath transmission for streaming blocks. In *IEEE INFOCOM 2022-IEEE Conference on Computer Communications*, pages 2178–2187. IEEE, 2022.