Chorus: Coordinating Mobile Multipath Scheduling and Adaptive Video Streaming

Gerui Lv, Qinghua Wu, Yanmei Liu, Zhenyu Li, Qingyue Tan, Furong Yang, Wentao Chen, Yunfei Ma, Hongyu Guo, Ying Chen, Gaogang Xie

Institute of Computing Technology, Chinese Academy of Sciences (ICT, CAS)
University of Chinese Academy of Sciences (UCAS)
Purple Mountain Laboratories
Computer Network Information Center, Chinese Academy of Sciences (CNIC, CAS)

Adaptive Bitrate (ABR) Streaming

Raw video (YUV format)

Adaptive Bitrate (ABR) Streaming

Encoded in multiple bitrate (resolution) versions

720p

1080p

2K

Adaptive Bitrate (ABR) Streaming

Divided into chunks of equal duration (e.g., 4s)

720p

1080p

2K

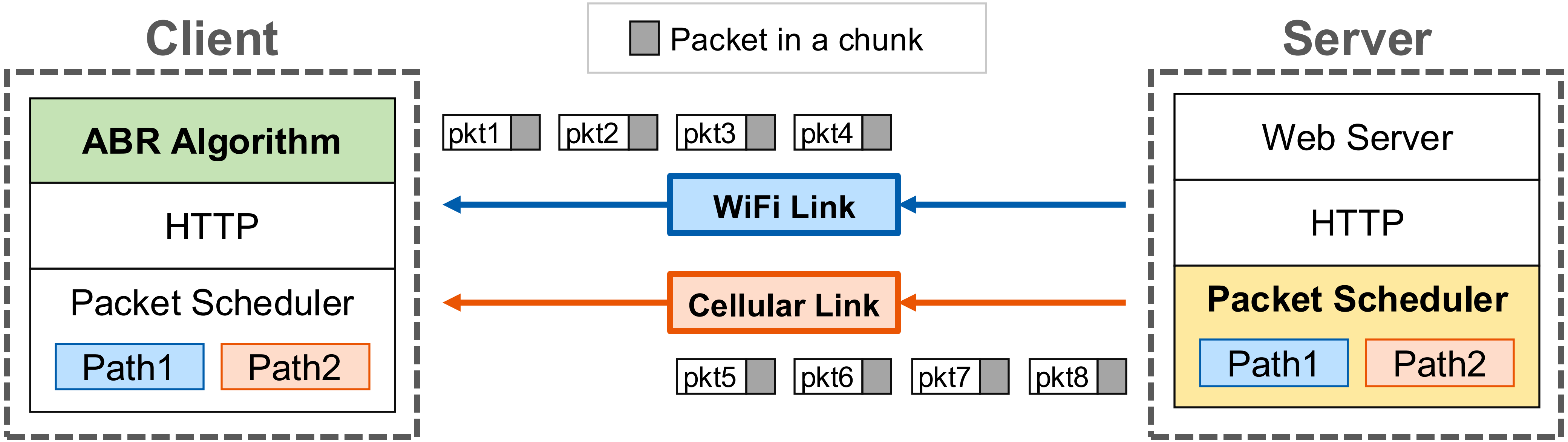
Adaptive Bitrate (ABR) Streaming

The client player runs ABR algorithms,
dynamically determining bitrate based on throughput prediction.

Goal: Optimize the quality of experience (QoE) – bitrate, rebuffering time, etc.

Mobile Multipath Transmission

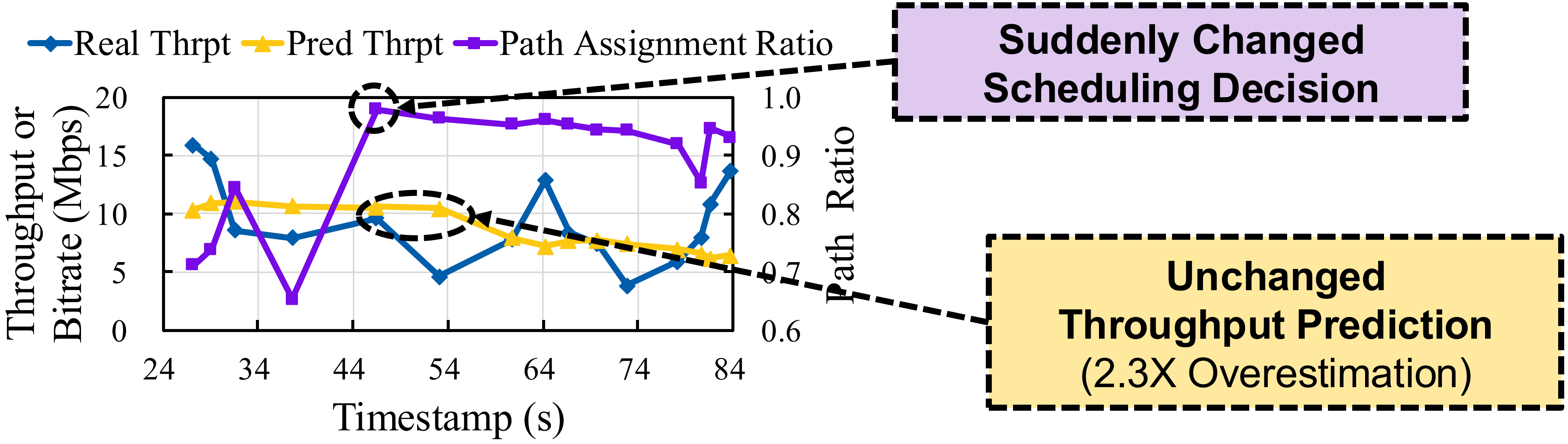
- Multipath transmission promises higher bandwidth for adaptive streaming
  - Mainstream protocol: Multipath TCP (MPTCP), Multipath QUIC (MPQUIC)
- Core component: **Packet scheduler**
  - Determines **path assignment ratio** (how many packets are assigned to each path)
  - **Goal: Optimize the quality of service (QoS)** – throughput, transmission time, etc.



The goal of ABR algorithms is different from that of multipath scheduling.

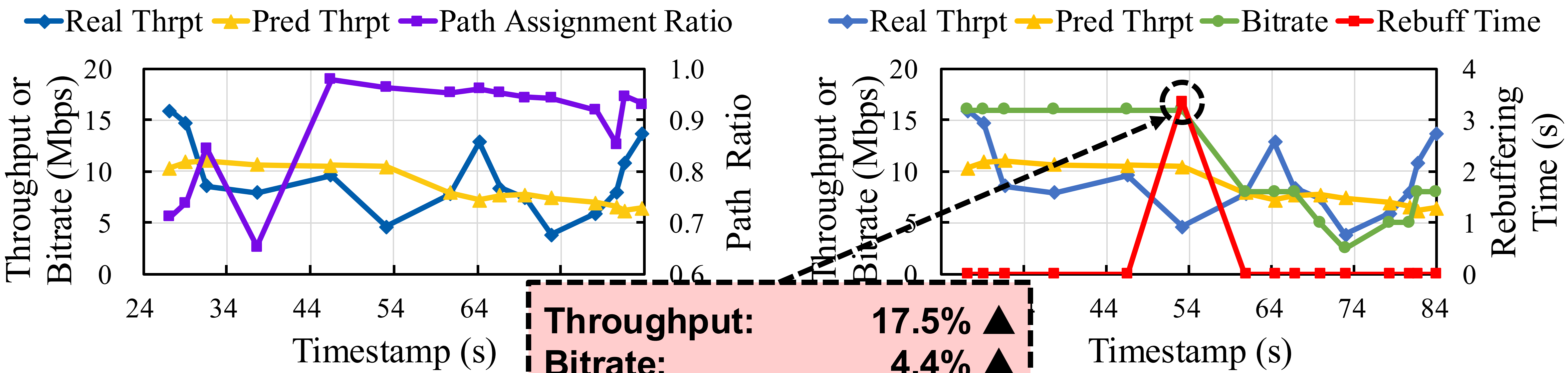
Motivation: QoS ≠ QoE

- Common logic: Optimize multipath mechanism → Optimize **QoS** → Optimize **QoE**
  - E.g., ECF [CoNEXT '17], DEMS [MobiCom '17], STMS [ATC '18], XLINK [SIGCOMM '21]
- Issue: **Better multipath scheduling can lead to lower QoE performance**
  - MinRTT+RI (Upper bound performance of XLINK [SIGCOMM '21]) vs. SP (Single path)



Motivation: QoS ≠ QoE

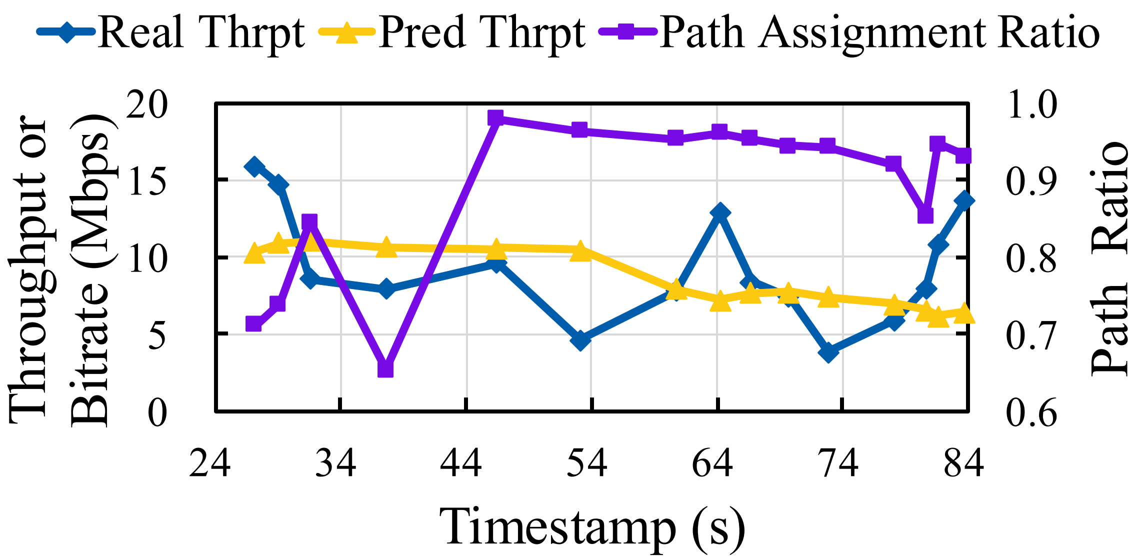
- Common logic: Optimize multipath mechanism → Optimize **QoS** → Optimize **QoE**
  - E.g., ECF [CoNEXT '17], DEMS [MobiCom '17], STMS [ATC '18], XLINK [SIGCOMM '21]
- Issue: **Better multipath scheduling can lead to lower QoE performance**
  - MinRTT+RI (Upper bound performance of XLINK [SIGCOMM '21]) vs. SP (Single path)

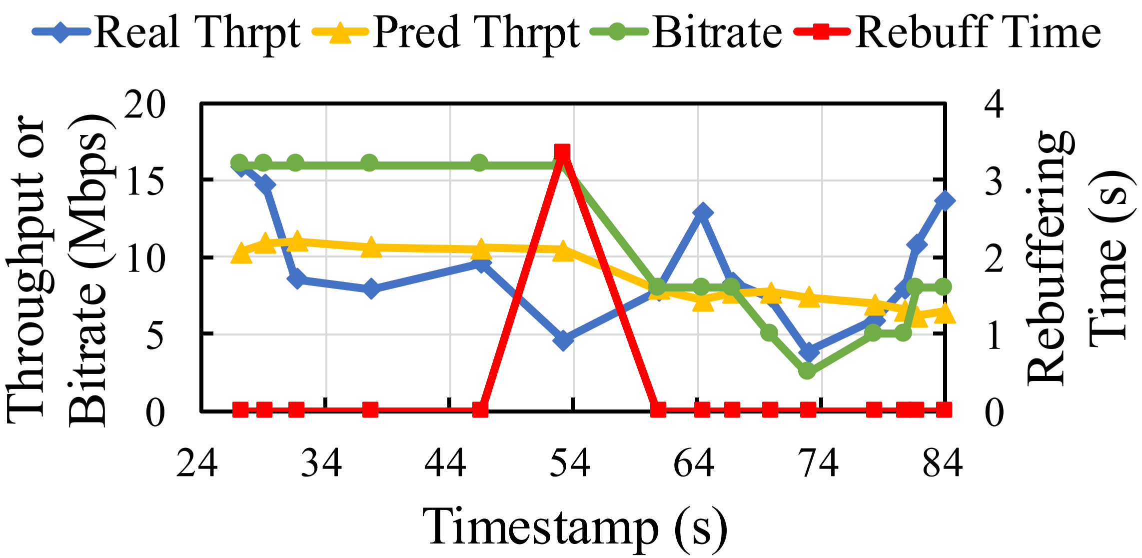


Throughput: 17.5% ▲
Bitrate: 4.4% ▲
Rebuffering Time: 4.7x ▲
Overall QoE: 20.6% ▼

Motivation: QoS ≠ QoE

- Common logic: Optimize multipath mechanism → Optimize **QoS** → Optimize **QoE**
  - E.g., ECF [CoNEXT '17], DEMS [MobiCom '17], STMS [ATC '18], XLINK [SIGCOMM '21]
- Issue: **Better multipath scheduling can lead to lower QoE performance**
  - MinRTT+RI (Upper bound performance of XLINK [SIGCOMM '21]) vs. SP (Single path)





Root cause: Adaptive streaming is uncoordinated with multipath scheduling.

Solution

❖ Idea: Coordinating multipath scheduling and ABRs to optimize QoE jointly

❖ Goal: Meeting **two necessary conditions** for ABRs to optimize QoE

Goal 1
Ensure appropriate bitrate selection

Challenge 1
How to improve multipath throughput prediction?

↓

Solution 1
Incorporating scheduling information in throughput prediction

Goal 2
Provide transport performance that meets QoE requirements

Challenge 2
How to satisfy QoE requirements while minimizing costs?

↓

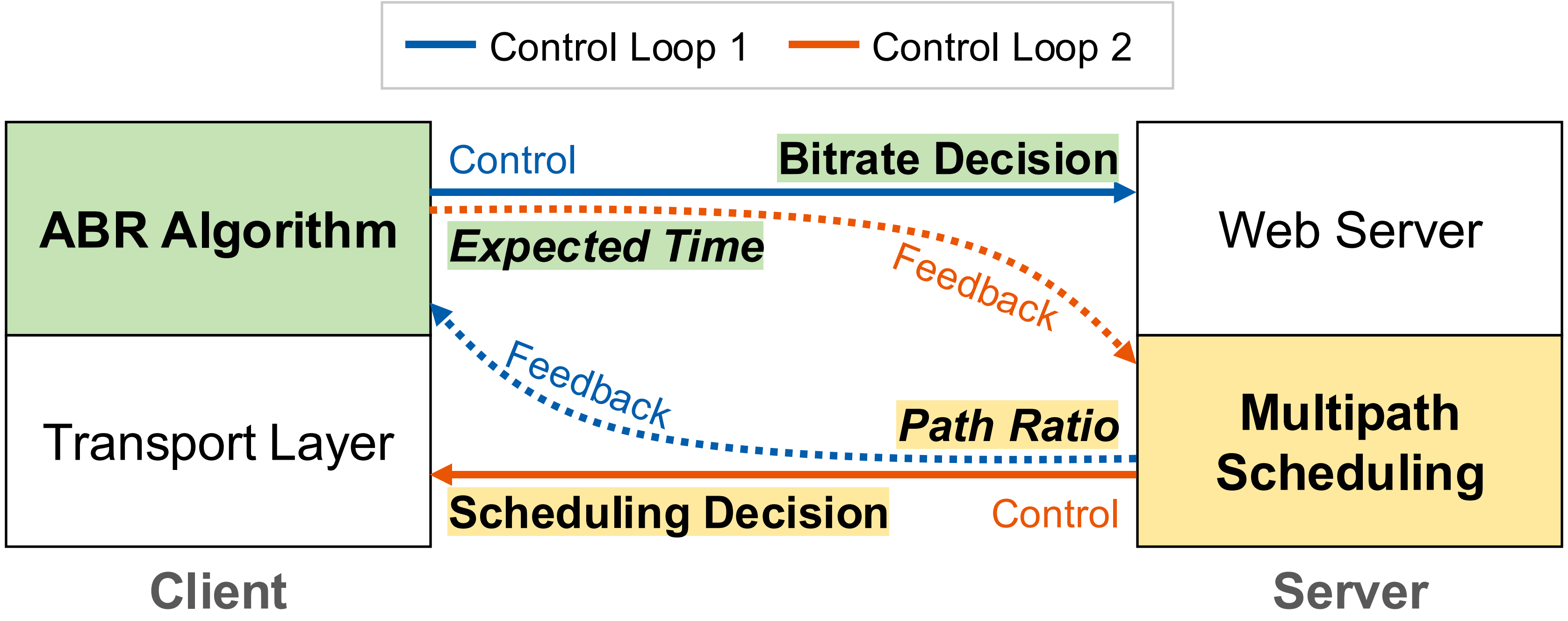
Solution 2
Knowing the expected time of ABR algorithms

Chorus: Coordination framework for multipath adaptive streaming

Chorus Overview

❖ Two-way Feedback Control Loops

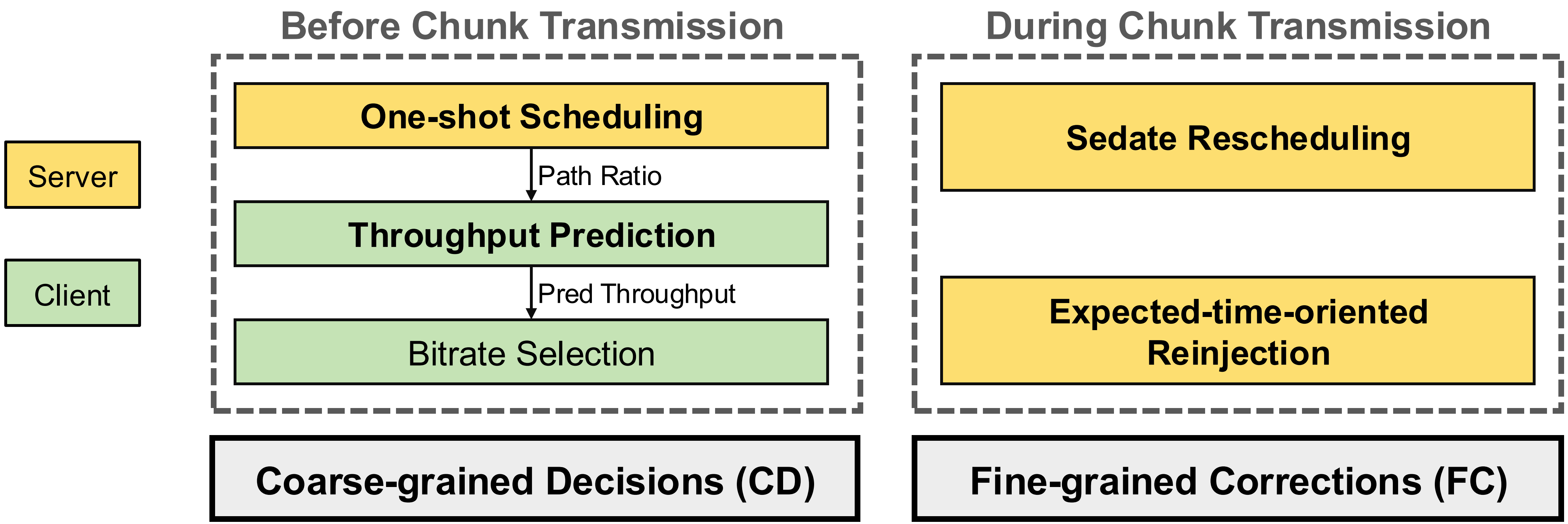
- *via QOE_CONTROL_SIGNAL frame in MPQUIC



Chorus Design: CD & FC

❖ Coarse-grained Decisions (CD) & Fine-grained Corrections (FC)

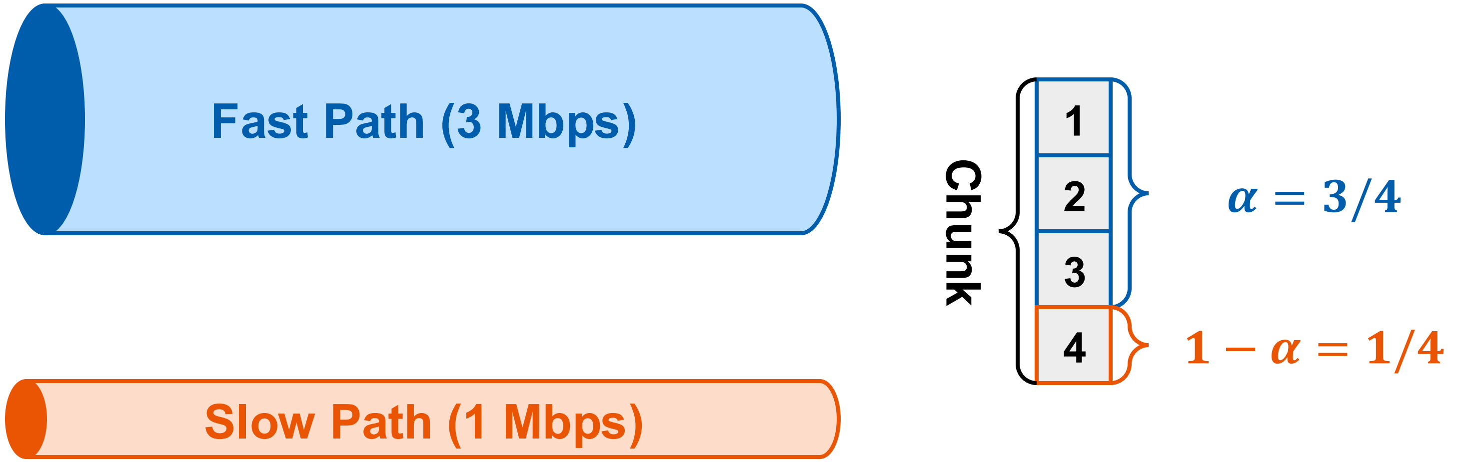
- **CD**: Appropriate bitrate selection
  - Predetermine the scheduling decision **at the chunk level** to reduce prediction uncertainty
- **FC**: Adequate transport performance
  - Adjust the scheduling decision **at the packet level** to meet the predicted throughput



CD: One-shot Scheduling

- **Server: One-shot Packet Scheduling**
  - Assign $\underline{\alpha}$ and $\underline{1\text{-}\alpha}$ of the packets in a chunk to the fast and slow paths, respectively.
  - α is determined by the ratio of path bandwidths:

$$\alpha = \frac{B_f}{B_f + B_s}$$

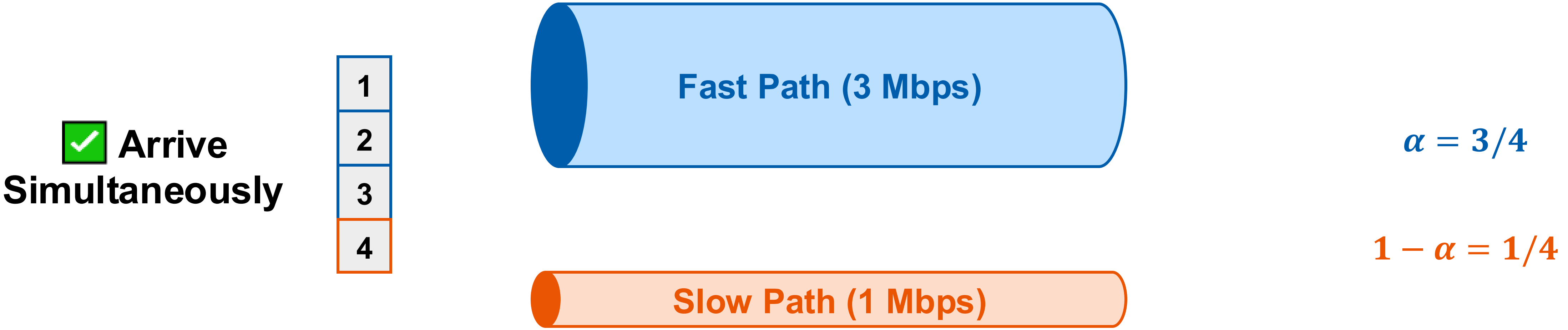


CD: One-shot Scheduling

❖ Server: One-shot Packet Scheduling

- Assign α and $1-\alpha$ of the packets in a chunk to the fast and slow paths, respectively.
- α is determined by the ratio of path bandwidths:

$$\alpha = \frac{B_f}{B_f + B_s}$$



CD: Throughput Prediction

❖ Server: One-shot Packet Scheduling

- Assign α and $1\text{-}\alpha$ of the packets in a chunk to the fast and slow paths, respectively.
- α is determined by the ratio of path bandwidths:

$$\alpha = \frac{B_f}{B_f + B_s}$$

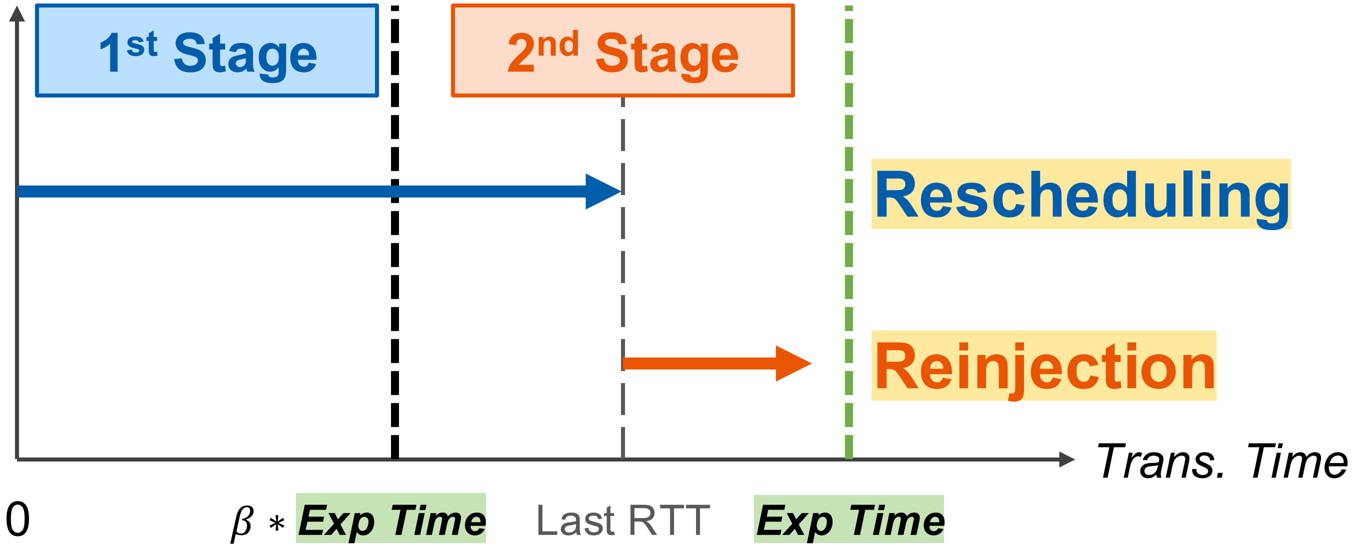
❖ Client: Multipath Throughput Prediction

- Chunk throughput depends on the minimum transmission rate of each path
- Transmission Rate = Path Bandwidth / ***Path Ratio*** α

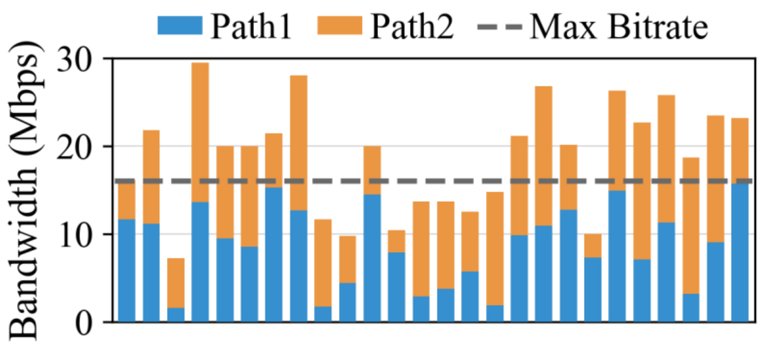
$$\hat{C}_k = \min\{\frac{\hat{B}_f}{\alpha}, \frac{\hat{B}_s}{1-\alpha}\}$$

FC: Two-stage Corrections

- **Goal:** Transmission Time ≤ ***Expected Time***
  - Expected Time = Chunk Size / Predicted Throughput
- 1st Stage: Sedate **Rescheduling** – Fully utilize bandwidth
  - Reschedule unsent packets on all paths to adapt to network dynamics
- 2nd Stage: Expected-time-oriented **Reinjection** – Meet QoE needs
  - Retransmit inflight packets of one path (e.g., slow path) on other paths



Trace-Driven Emulation



❖ Video Settings

- Bitrate levels: [1, 2.5, 5, 8, 16] Mbps
- Resolutions: [360p, 480p, 720p, 1080p, 1440p (2K)]

❖ Baselines

- Multipath QUIC: **XLINK** [SIGCOMM '21] / **MinRTT** / **MinRTT+RI**
- Single-path QUIC: **SP**

❖ Network Traces

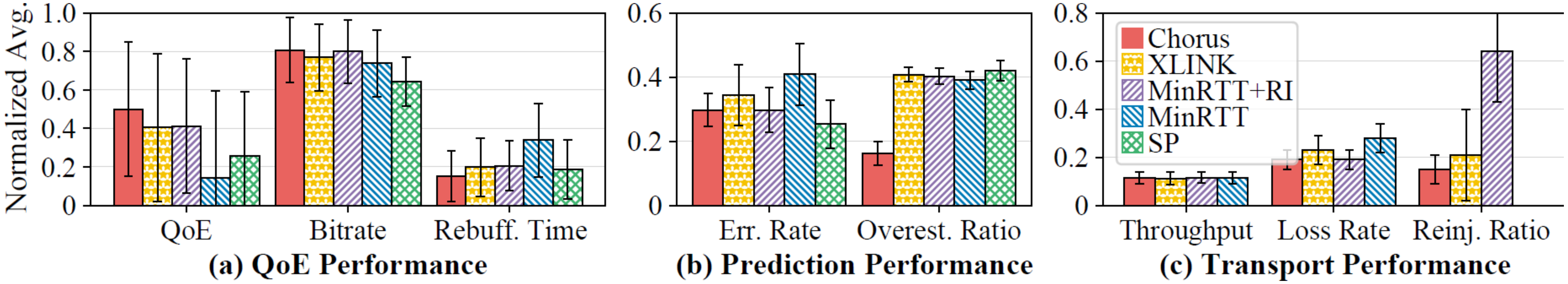
- Type: 5 WiFi traces + 47 Cellular traces
- Mobility: 50% Stationary + 50% Movement
- Statistics: Average downlink bandwidth 1.5 Mbps~15.9 Mbps
- Emulation Testbed: Mahimahi (mpshell) + Virtual player

❖ QoE Metrics

- Linear QoE from MPC (bitrate, rebuffering time, and smoothness)

Emulation Results

- Chorus achieves the **best overall QoE performance**
  - Average QoE performance: **21.1%~247.3%** ▲
- Chorus provides **better throughput prediction** for ABR algorithms
- Chorus delivers the **best transport performance** at the **lowest cost**
  - vs. XLINK: Reinjection ratio **28.6%** ▼, Bitrate **5%** ▲, Rebuffering time **24%** ▼



Chorus has successfully implemented its two design principles.

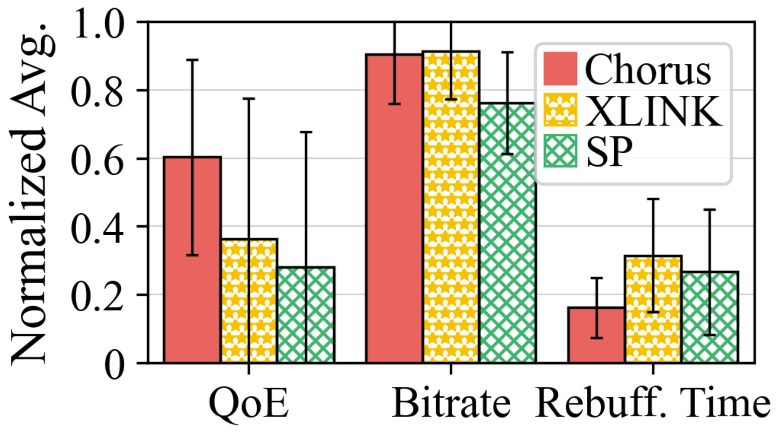
Real-world Deployment

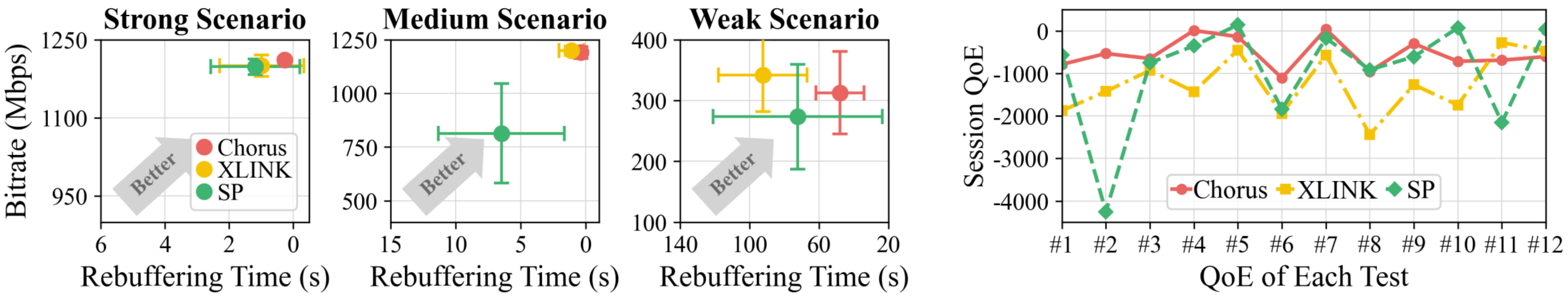
- **Implementation: User-space MPQUIC** (XQUIC library)
  - Server: Tengine Web Server
  - Client: MediaPlayer-Extended, running on 3 Android phones
- **Test Environment: Real-world Mobile Networks**
  - Baselines: XLINK / SP
  - Access Network: WiFi (WiFi4 / WiFi5 / WiFi6) + Cellular (4G / 5G)
  - Mobility: 50% Stationary + 50% Movement (by walking)
  - Setting: 3 scenarios of 12 test units each; 108 sessions in total

Strong Scenario	Medium Scenario	Weak Scenario
WiFi BW ≥ Highest Bitrate	WiFi BW ≤ Highest Bitrate Cellular BW ≥ Highest Bitrate	WiFi BW ≤ Highest Bitrate Cellular BW ≤ Highest Bitrate

Real-world Results

- Overall QoE of Chorus: **65.7%~114.4%** ▲
- Strong and medium scenarios: Near-optimal
- Weak scenario: Performs well in the heavy tail
  - Rebuffering time of Chorus: **33.7%~48.1%**▼
  - XLINK performs worse than SP in most cases
    - Severe stalling events: Inaccurate throughput prediction; Failure to meet QoE requirements





Chorus shows consistent performance advantage in all scenarios.

Contributions

- Reported the **discoordination issue** of adaptive streaming and multipath scheduling; revealed the root cause and the fundamental solution.
- Designed **Chorus**, a closed-loop coordination framework that ensures effective bitrate control for multipath adaptive streaming.
- Implemented **Chorus** based on multipath QUIC and integrated it into a real-world mobile video system.
- Confirmed **Chorus**'s consistently high performance through extensive evaluations in mobile networks.

Thanks!

Gerui Lv (ICT, CAS)

lvgerui@ict.ac.cn
https://greenlv.github.io/