MMSys'18

Quickly Starting Media Streams Using QUIC

Packet Video Workshop 2018

Şevket Arısu and Ali C. Begen

ÖZYEĞİN UNIVERSITY

TURKCELL

Agenda

- Motivation and our goal
- Previous work and our contributions
- Approach, setup and evaluation
- Results
- Conclusions

Motivation

QUIC: Multiplexed transport protocol over UDP

- **Reduced** handshake latency
- Improved **congestion control**
- Improved **loss recovery**
- **Multiplexing** streams

Can QUIC help improve viewer experience in HTTP adaptive streaming?

Can QUIC help us

- Start media streams more quickly
- Reduce seeking latency
- Cope better with frequent connection changes?

Previous Research

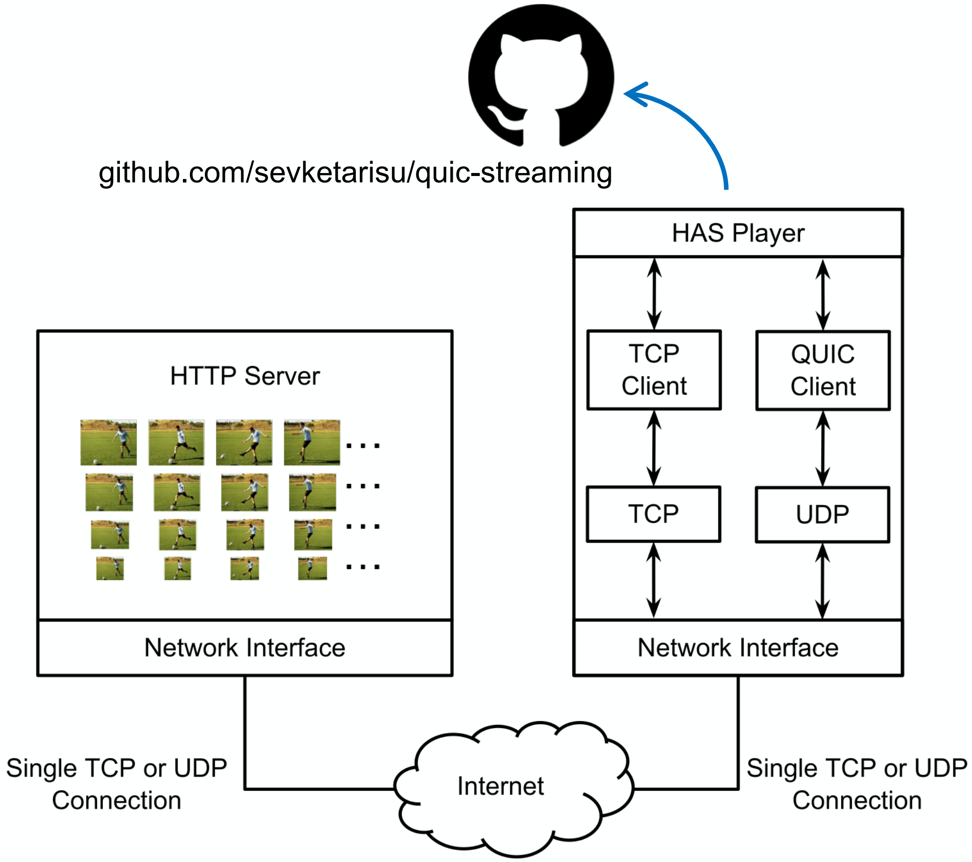
- Some reported
  - QUIC does not impact streaming performance
  - QUIC does not provide a boost to HAS
- Others reported
  - QUIC's 0-RTT performed better than the other protocols
  - QUIC provided better streaming, but only for high-quality video
- Google said QUIC reduced YouTube rebuffer rates by 15%

QUIC code evolves rapidly and 3rd-party implementations may not necessarily reflect protocol's real performance

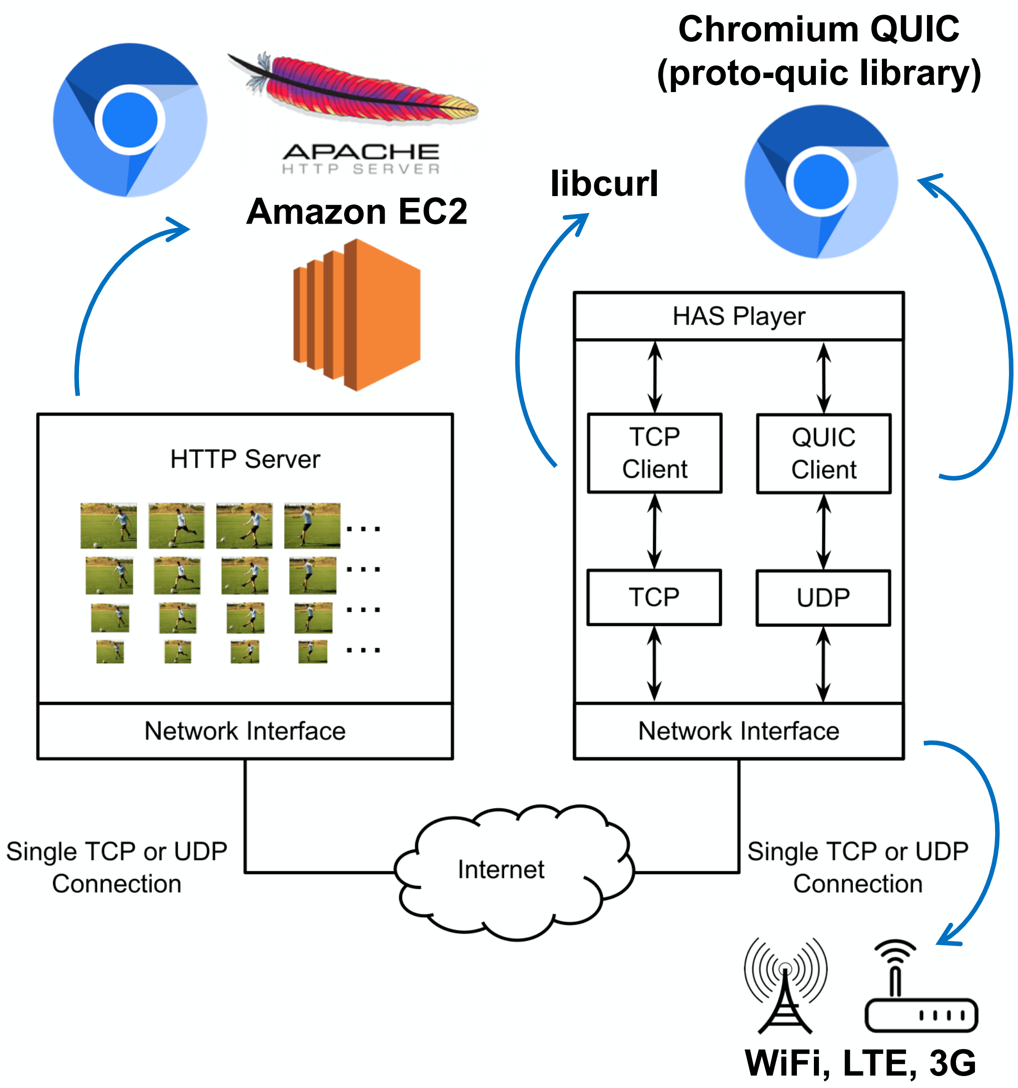
Comparison between Our and Prior Work

	Tested QUIC Version	Used Official Google Server?	Wireless Networks[1]	Tested Different Algorithms?	Evaluated Frame Seeking?	Evaluated Conn. Switches?	Tested Live Video?
Timmerer [27]	v19	✓	✗	✗	✗	✗	✗
Szabo [26]	Latest [2]	✗	Only WiFi	✗	✗	✗	✗
Li [22]	Latest [3]	✓	✗	✗	✗	✗	✗
Bhat [13]	Latest [2]	✗	Only WiFi	✓	✗	✗	✗
Zinner [29]	Latest [3]	✓	✗	✗	✗	✗	✗
Kakhki [20]	v37	✓	All	✗	✗	✗	✗
Ayad [10]	Latest [3]	✓	✗	✓	✗	✗	✗
Our work	v39 [3]	✓	All	✓	✓	✓	✓

[1] WiFi, 4G/LTE and 3G, [2] Based on the third-party implementation version at the time of research, [3] Latest at the time of research.



- Player Features
  - Python based
  - TCP and QUIC support integrated as subprocess
  - Scripted frame-seek (fast-forward) ability
  - BASIC, SARA, BBA-2 adaptation algorithms
- Source Code
  - github.com/sevketarisu/quic-streaming



- Environment Setup
  - Public Internet
  - No traffic shaping
  - Each test repeated 10 times
- Server
  - Apache HTTP server on Amazon EC2
  - Located in Frankfurt, DE
- Client
  - Runs Google's proto-quic library
  - Located in Istanbul, TR
  - Connected via WiFi, LTE or 3G

Evaluated Metrics

- Average playback bitrate
  - Average of the bitrates of the downloaded segments
- Average wait time after seeking
  - Time from the frame-seek request to the playback of the requested media
  - A rule of thumb is to keep this time under two seconds
- Rebuffer rate

$$\text{Rebuffer Rate} = \frac{\text{Rebuffer Time}}{\text{Rebuffer Time} + \text{Media Play Time}}$$

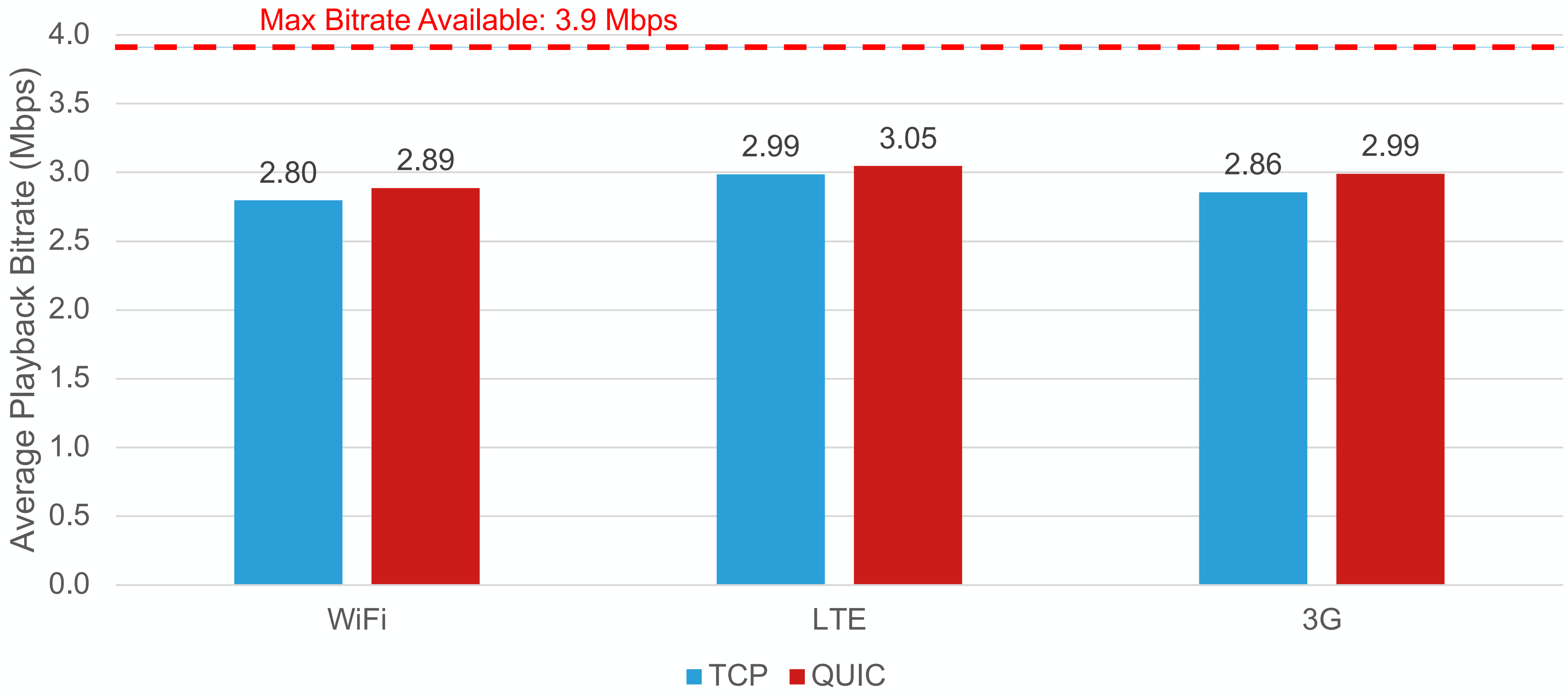
Frame-Seek Scenarios

Frame-Seek (Fast-Forward) Feature Implemented in the Player

Viewer Action	Seek at	Seek to	Play Duration
Start at 0 s	-	-	40 s
Seek #1	40 s	100 s	50 s
Seek #2	150 s	200 s	80 s
Seek #3	280 s	350 s	70 s
Seek #4	420 s	500 s	100 s
Finish at 600 s	-	-	-
Total Viewed			340 s

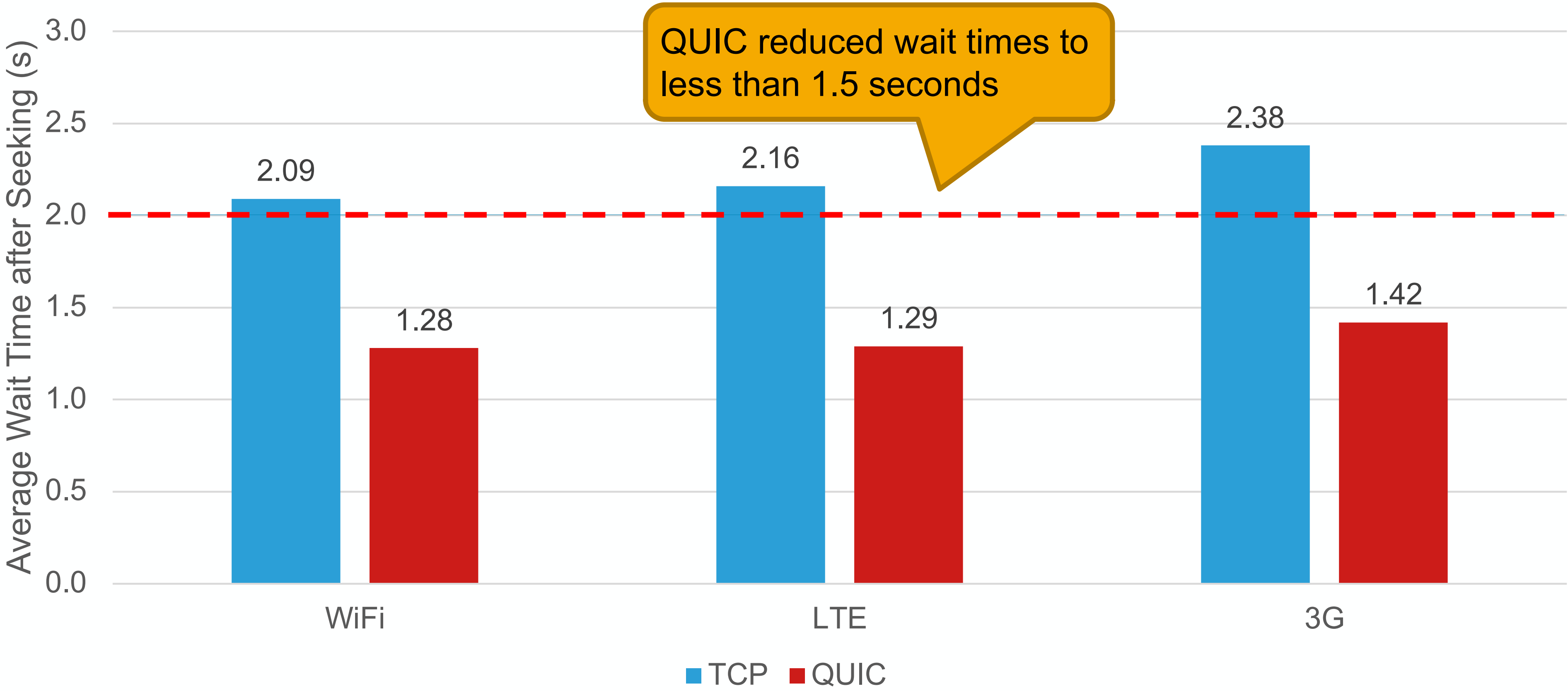
Frame-Seek Scenario Results

QUIC provided a higher (or at least an equal) average playback bitrate in all cases and for all algorithms



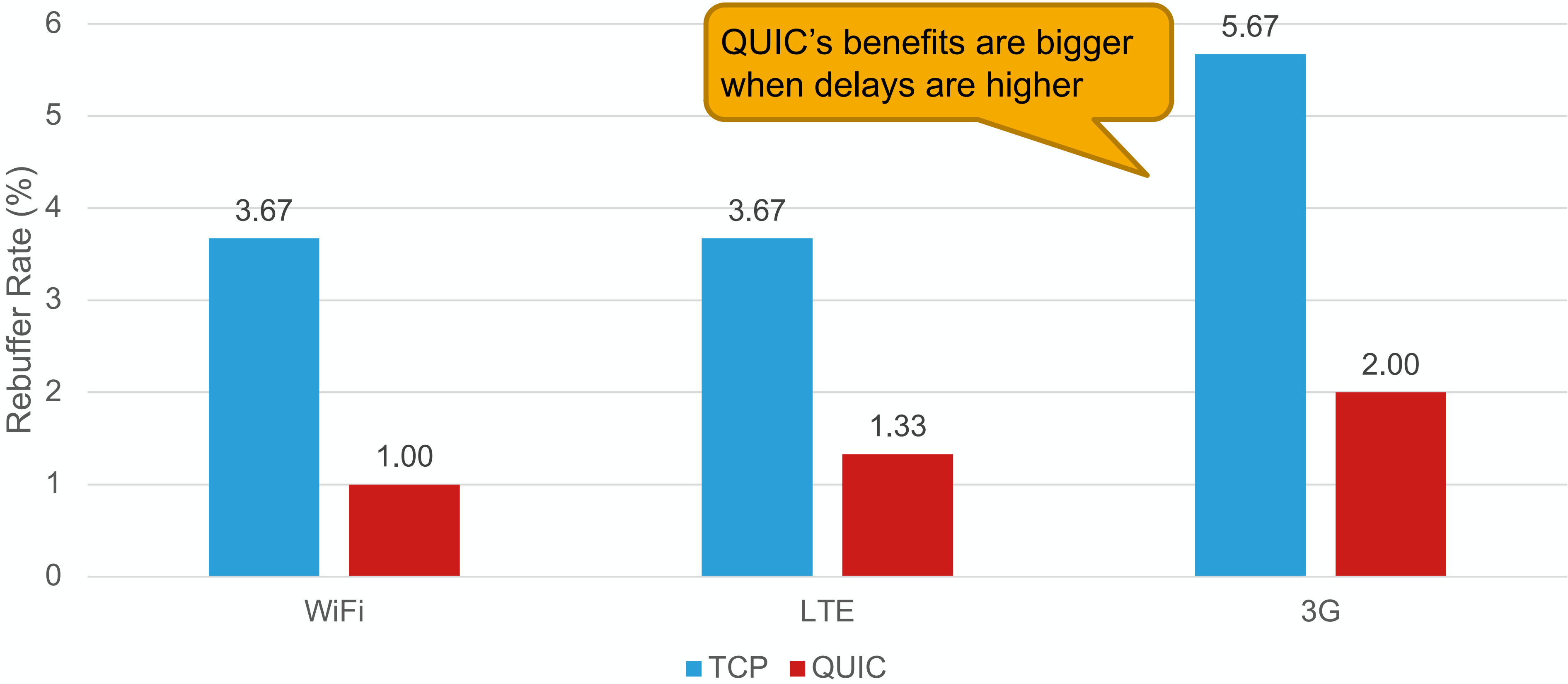
Results are averaged for BASIC, SARA and BBA-2 algorithms.

Frame-Seek Scenario Results



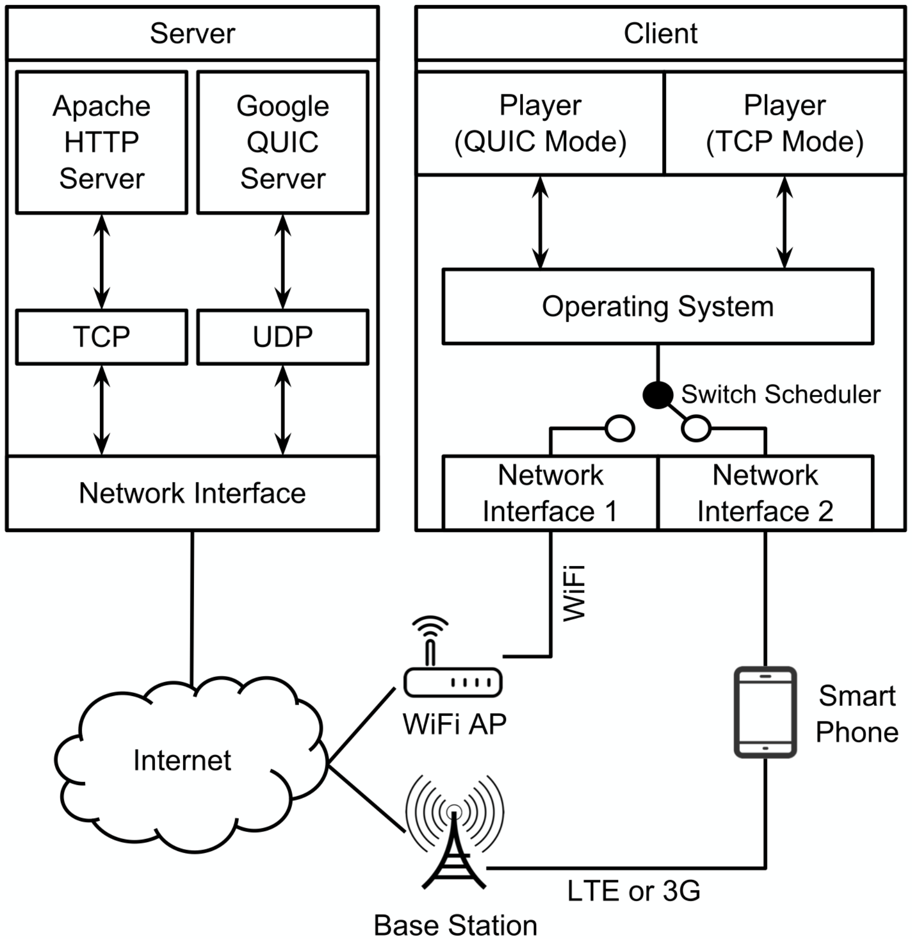
Results are averaged for BASIC, SARA and BBA-2 algorithms.

Frame-Seek Scenario Results



Results are averaged for BASIC, SARA and BBA-2 algorithms.

Connection-Switch Simulation

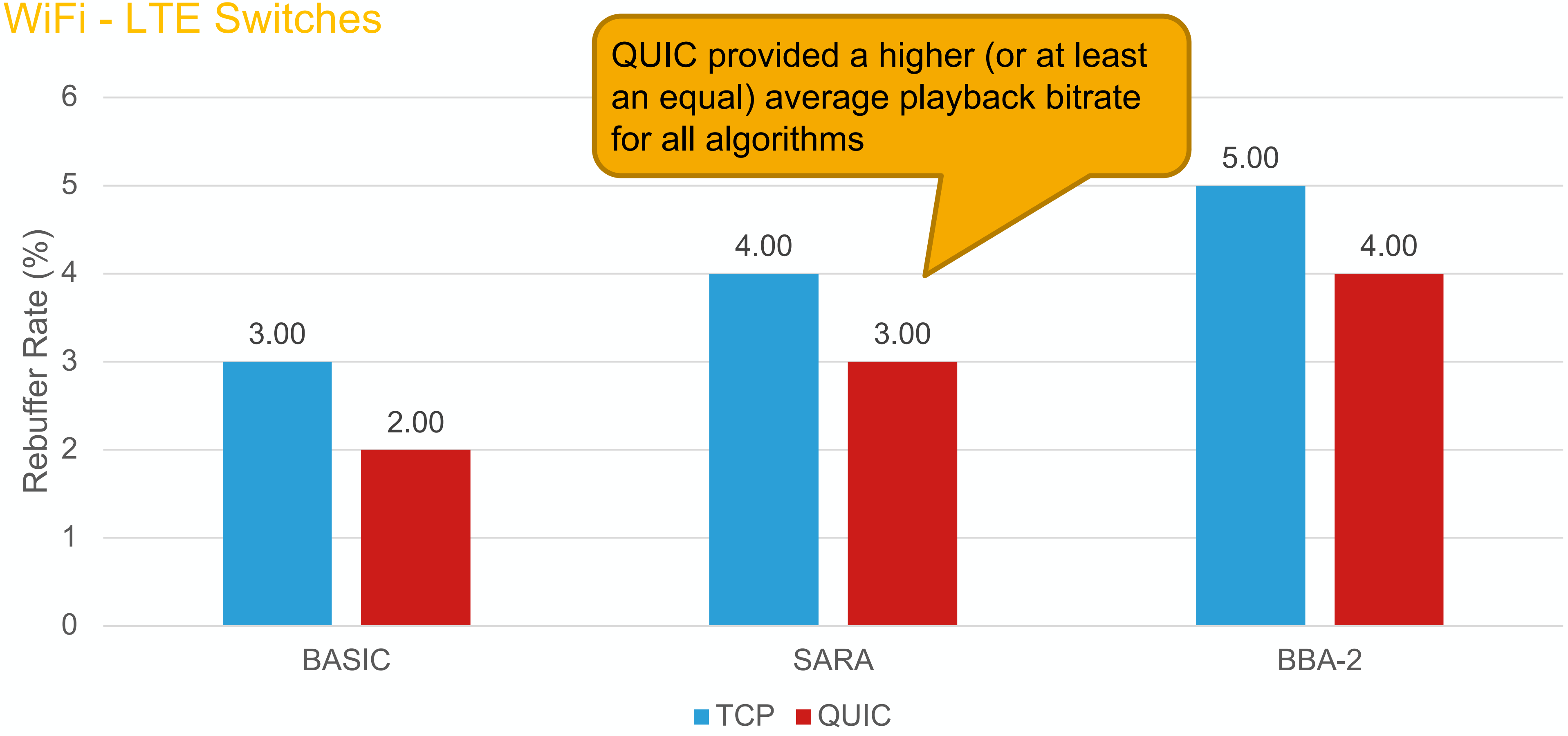


Connection-Switch Scenario

From Second	To Second	Connection Type
0	60	WiFi
60	180	LTE or 3G
180	300	WiFi
300	420	LTE or 3G
420	480	WiFi
480	540	LTE or 3G
540	600	WiFi

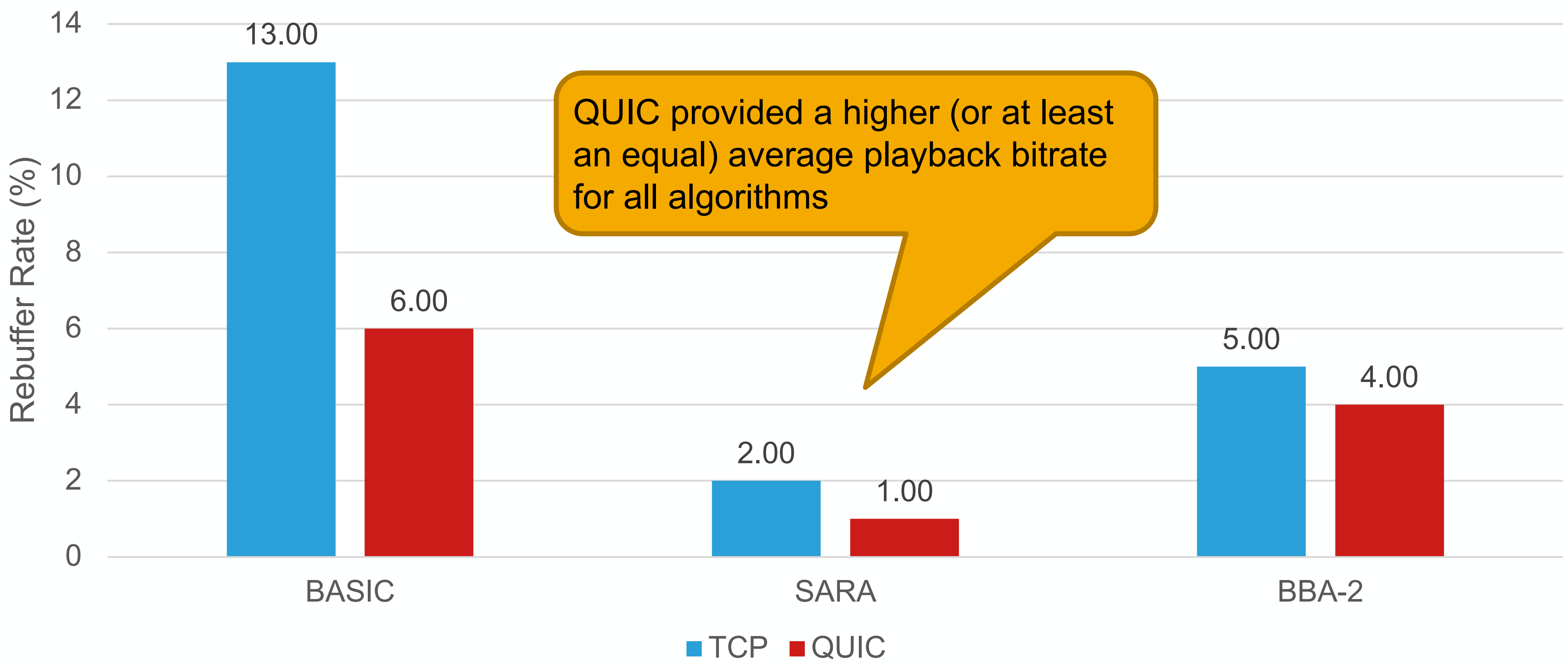
Connection-Switch Results

WiFi - LTE Switches



Connection-Switch Results

WiFi - 3G Switches



Conclusions

- QUIC reduces wait times and rebuffer rates without reducing playback bitrate
- QUIC outperforms TCP when frequent network changes occur
- QUIC's benefits are greater in networks with larger delay (e.g., early generation 3G networks)

QUIC is still evolving in the IETF; should there be significant changes, the tests will need to be repeated

- https://quicwg.org/
- https://github.com/quicwg

MMSys'18

Thank You

Download and test our code at github.com/sevketarisu/quic-streaming

ÖZYEĞİN UNIVERSITY

TURKCELL