TECHNISCHE UNIVERSITÄT WIEN

Cocoa: Congestion Control Aware Queuing

Maximilian Bachl, Tanja Zseby, Joachim Fabini

Technische Universität Wien, Vienna, Austria

"Bufferbloat"

- If queues in routers/switches too small: **Underutilization**
- **Solution:** Make queues very **large** for maximum throughput!
- **New problem:** Packets **wait** a long time (several seconds) in the queue: Bufferbloat

CoDel Active Queue Management against Bufferbloat

- Moving time window of 100 ms
- Queuing Delay **must be** $<$ **5 ms** once in each window
- Otherwise: Drop packet(s)

Fair Queuing

- Separate each flow in separate queue
- No flow can "steal" bandwidth
- Popular implementation for Linux: `fq`

Fair Queuing

- Separate each flow in separate queue
- No flow can "steal" bandwidth
- Popular implementation for Linux: `fq`

Static buffer size:

Buffer often too large or too small

`fq_codel` Combining fair queuing with CoDel

- State-of-the-art
- Separate queues
- Each queue managed by CoDel
- Keeps each flow's queue $< 5\,\text{ms}$
- Linux implementation: `fq_codel`

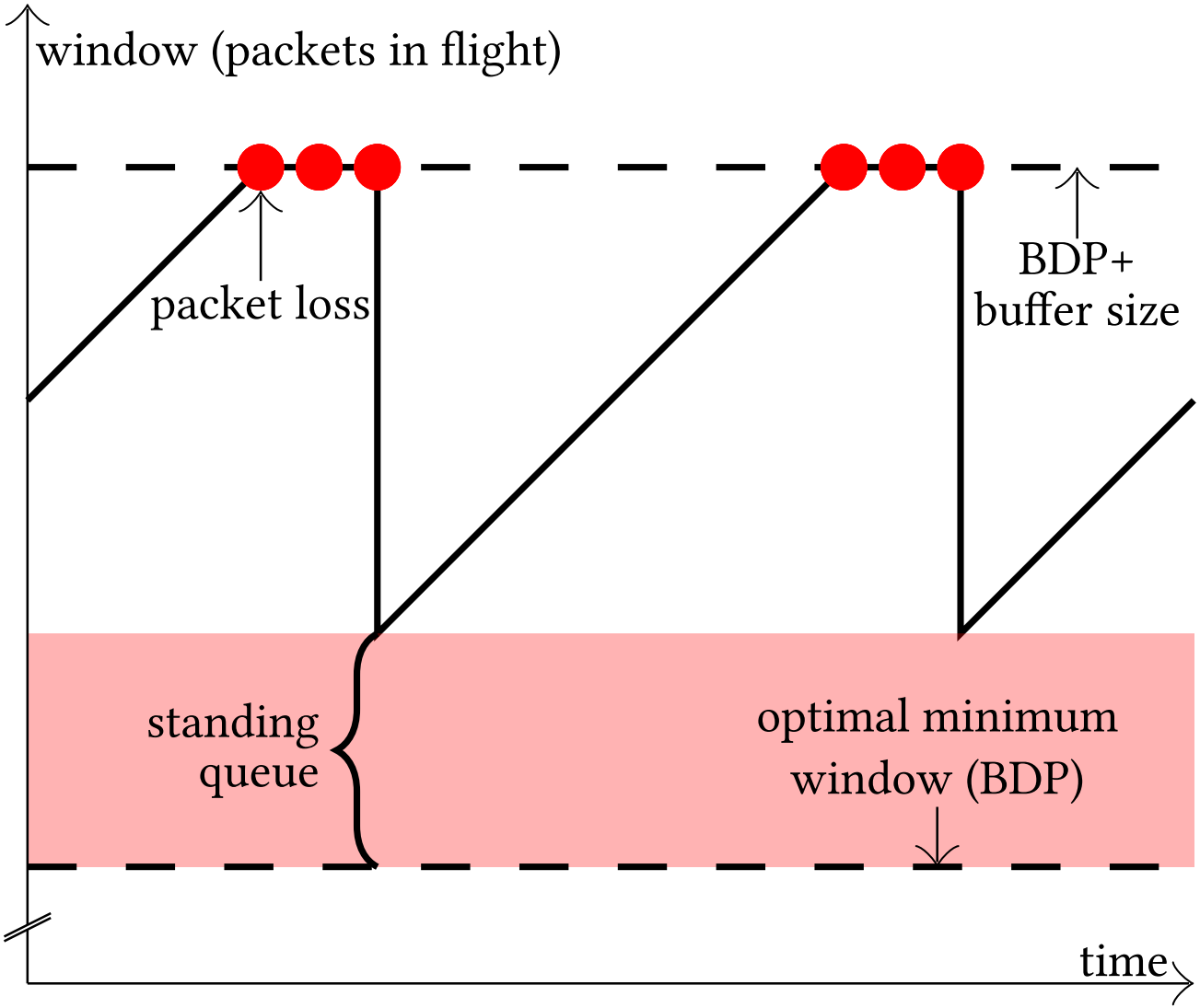
`fq_codel` Combining fair queuing with CoDel

- State-of-the-art
- Separate queues
- Each queue managed by CoDel
- Keeps each flow's queue $< 5\,\text{ms}$
- Linux implementation: `fq_codel`

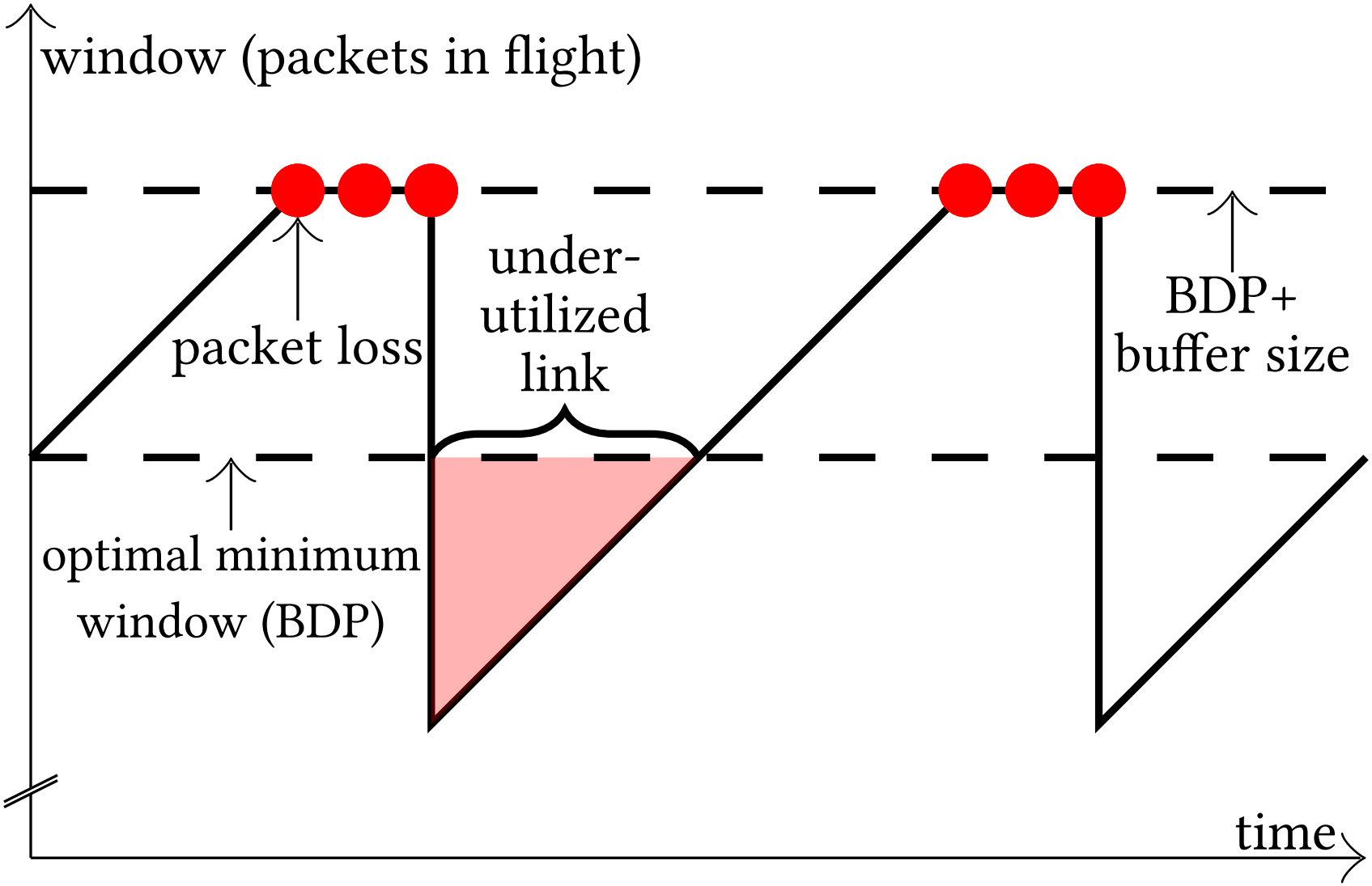
Inadequate interaction with Congestion Control:

Cubic doesn't achieve full throughput or keeps standing queue

Buffer too large



Buffer too small



Measuring a flow's congestion control

- Get period between reductions of the congestion window
- If there's a standing queue:
  - Reduce buffer
- If the link was idle:
  - Increase buffer

Measuring a flow's congestion control

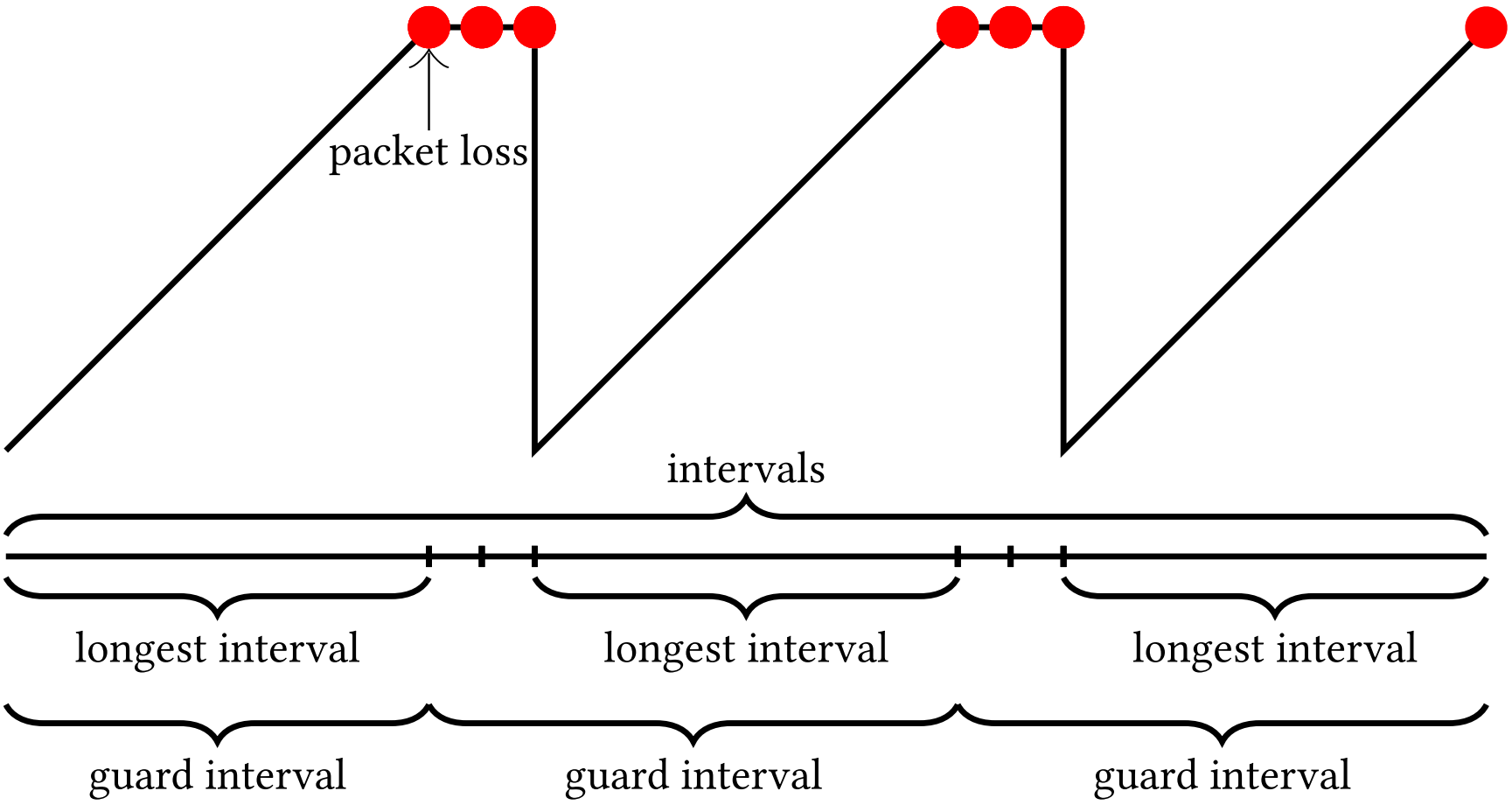
- Get period between reductions of the congestion window
- If there's a standing queue:
  - Reduce buffer
- If the link was idle:
  - Increase buffer

→ **Problem**: How to get the interval between reductions of the congestion window?

Interval mechanism – Concept

1. Wait for previous longest interval between two packet losses times 0.5 and continue waiting until the next packet loss
2. Take longest interval between two packet losses since previous longest interval ended
3. Adjust buffer accordingly
4. Repeat

Interval mechanism – Illustration



Parameters

- Multiplier (default 1.25)
  - Needed because of BBR, for Cubic and Reno 0.5 is sufficient
- Maximum buffer increase (default 2)
  - Otherwise huge increase if link speed suddenly dramatically increases
- Maximum guard interval (default 1 s)
  - Otherwise giant guard intervals possible and slow adaptation

Parameters

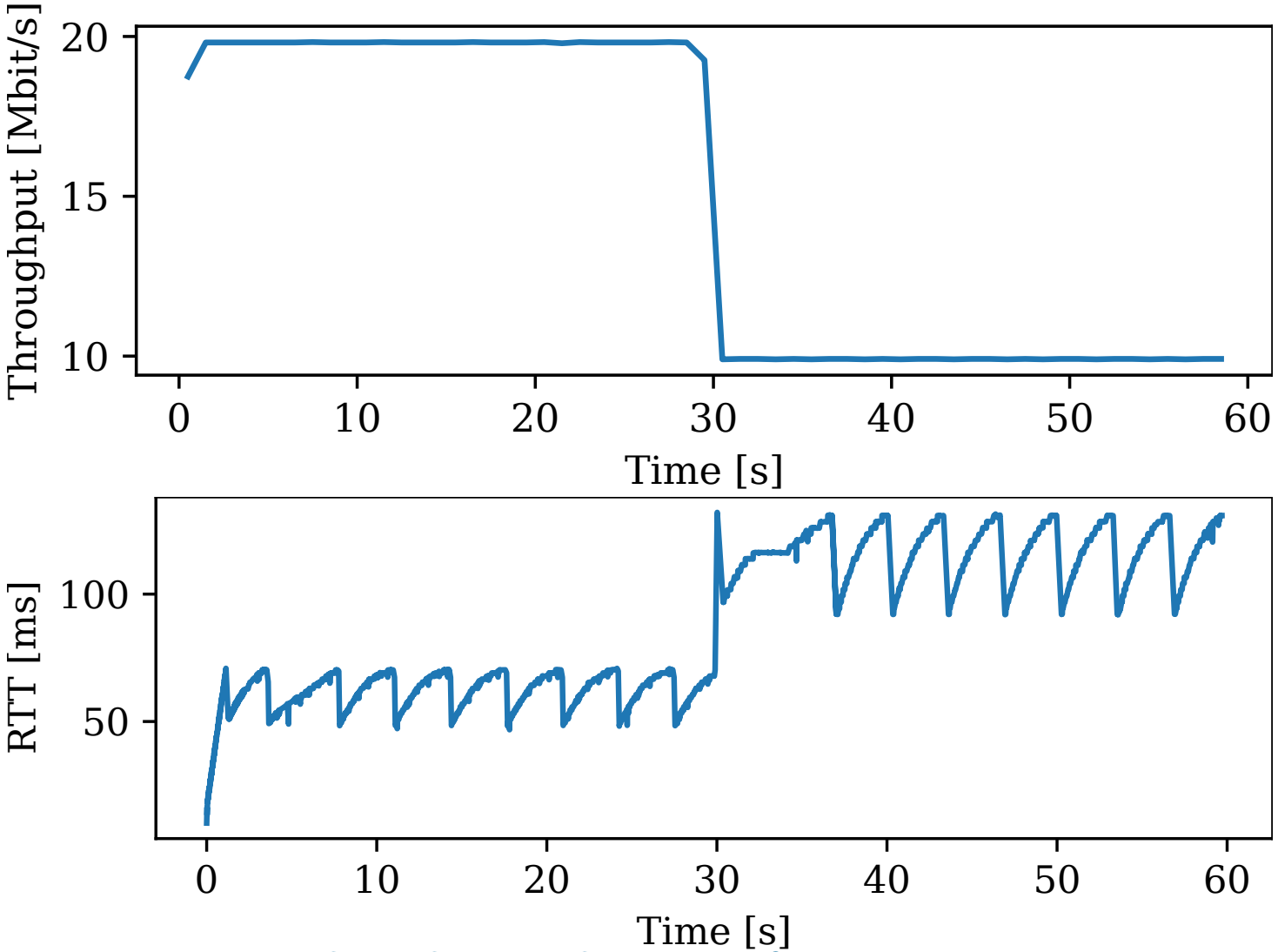
- Multiplier (default 1.25)
  - Needed because of BBR, for Cubic and Reno 0.5 is sufficient
- Maximum buffer increase (default 2)
  - Otherwise huge increase if link speed suddenly dramatically increases
- Maximum guard interval (default 1 s)
  - Otherwise giant guard intervals possible and slow adaptation

→ Worked for all tested scenarios

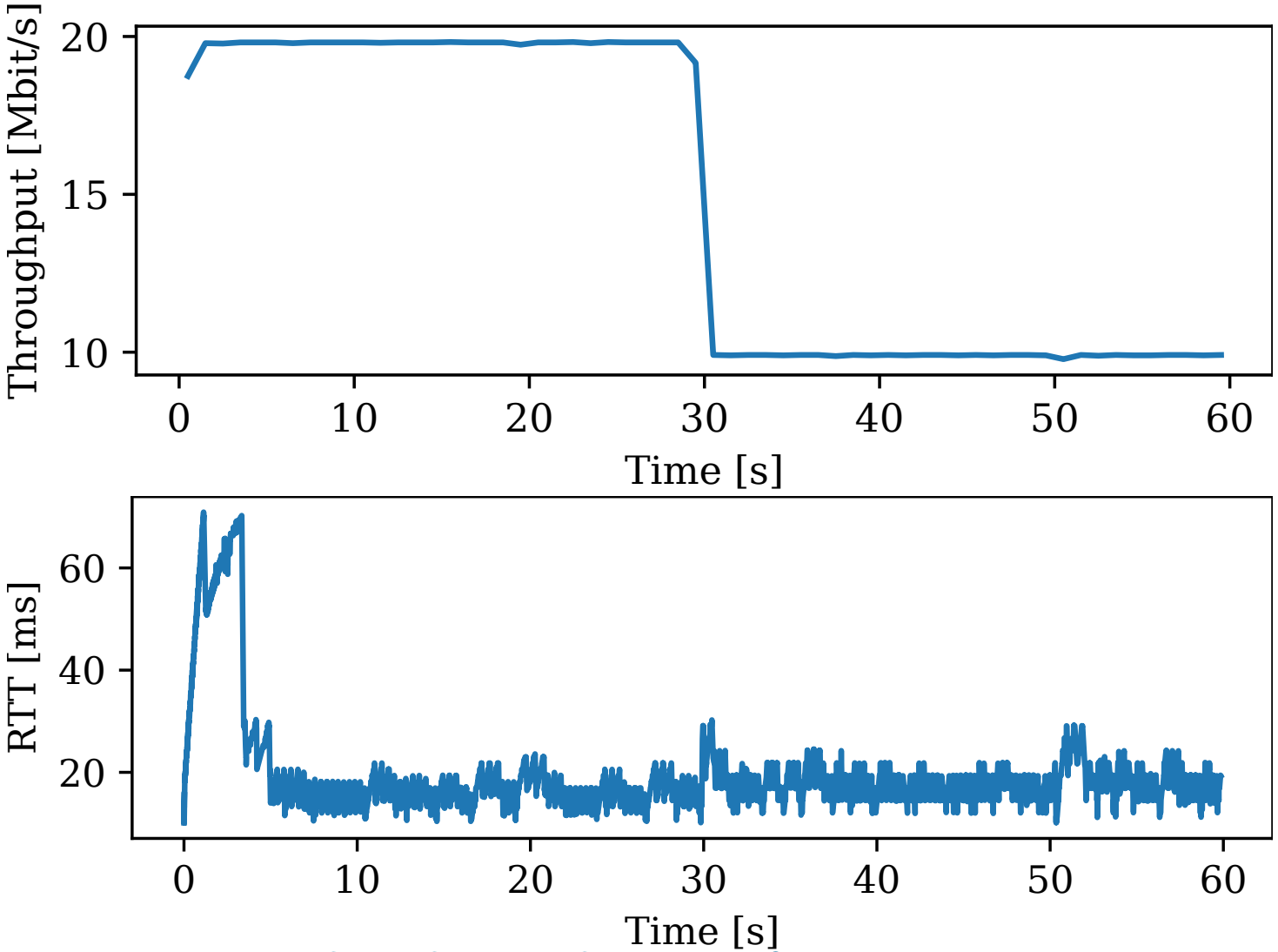
Setup

- Implemented on Linux
- Extension of the `fq` kernel module
- Virtual network with `py-virtnet`
- All other queuing disciplines for comparison with default parameters

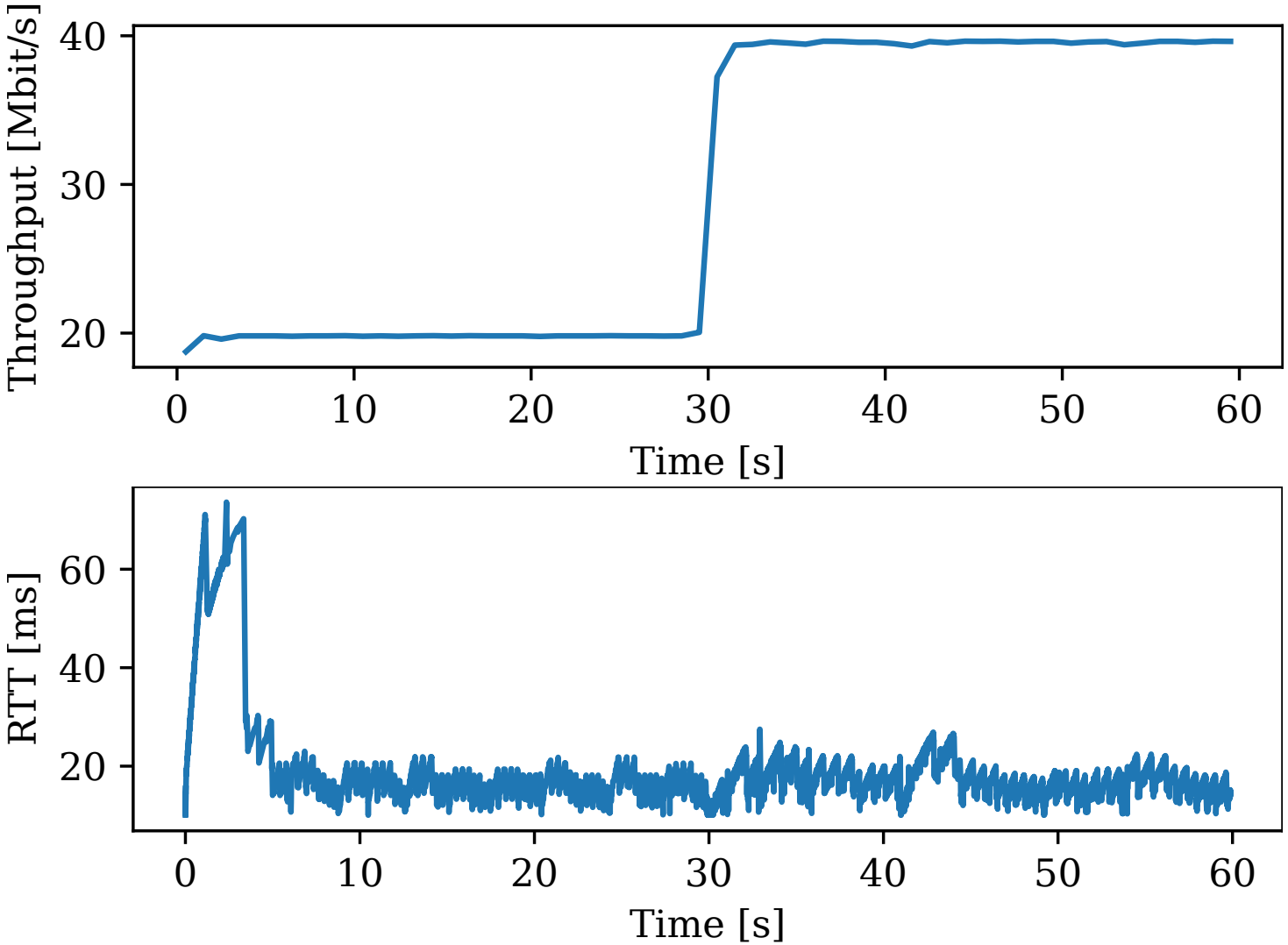
Cubic, `fq` with 20 Mbit/s, 10 ms



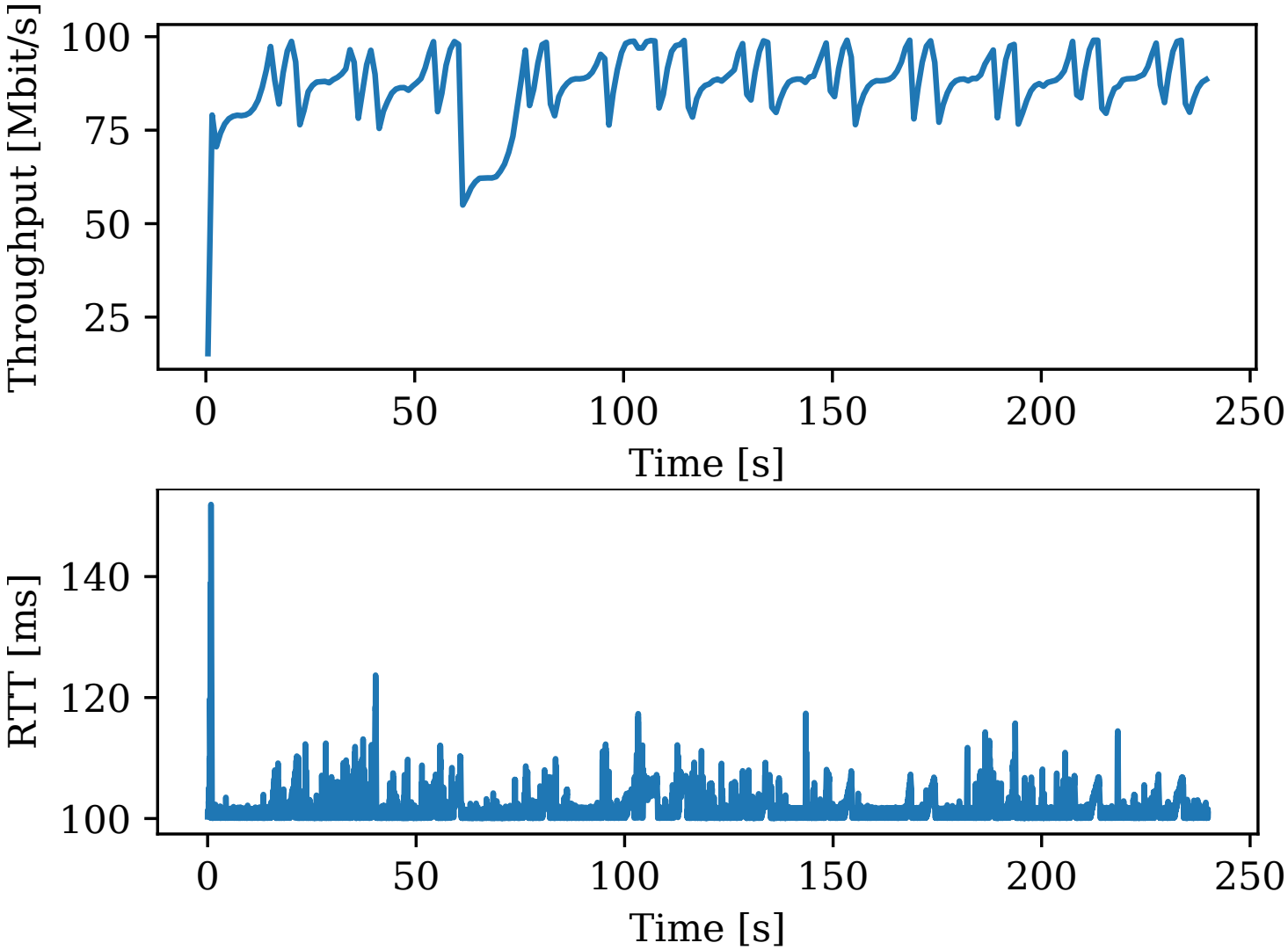
Cubic, cocoa with 20 Mbit/s, 10 ms



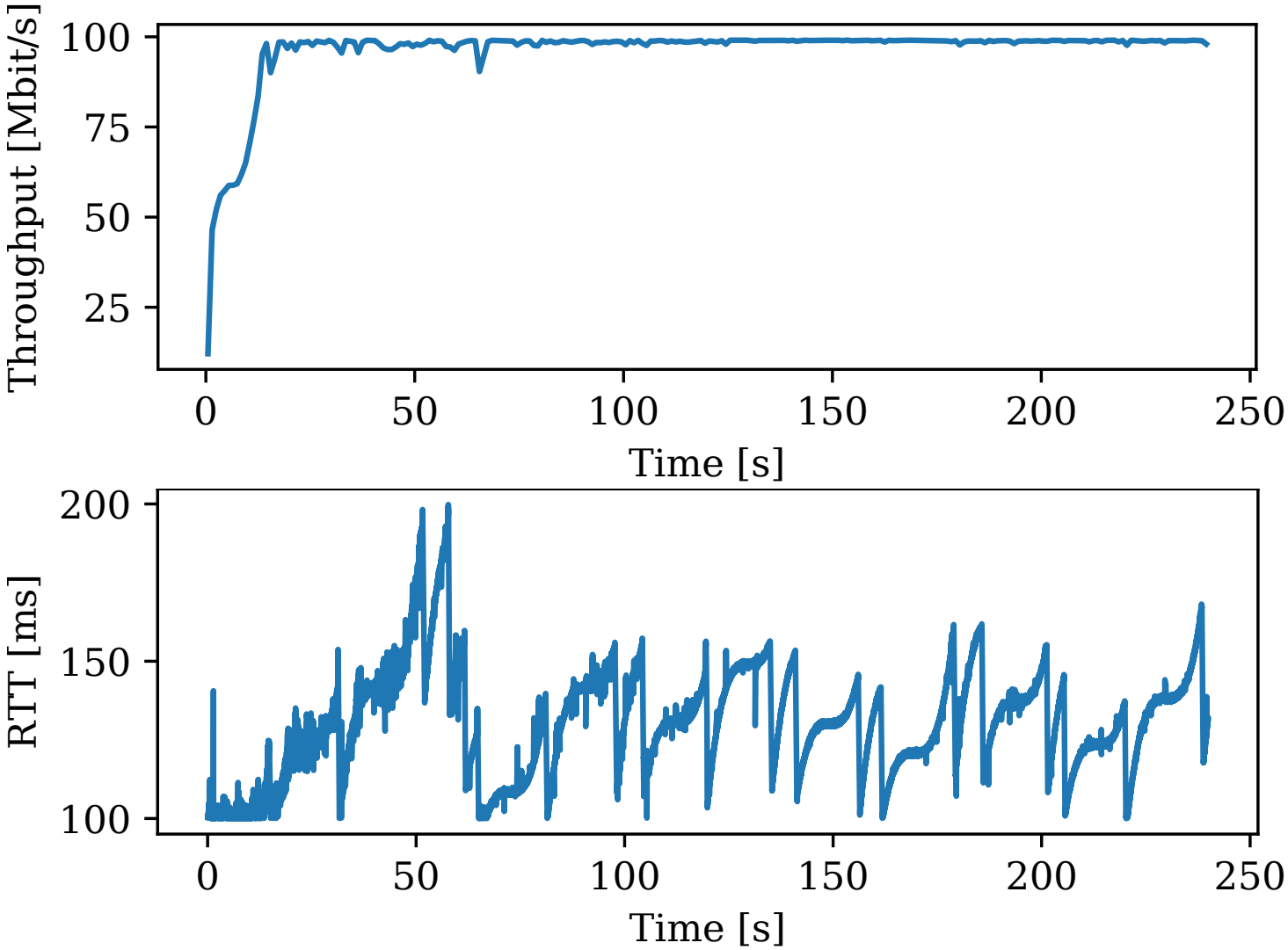
Cubic, cocoa with 20 Mbit/s, 10 ms



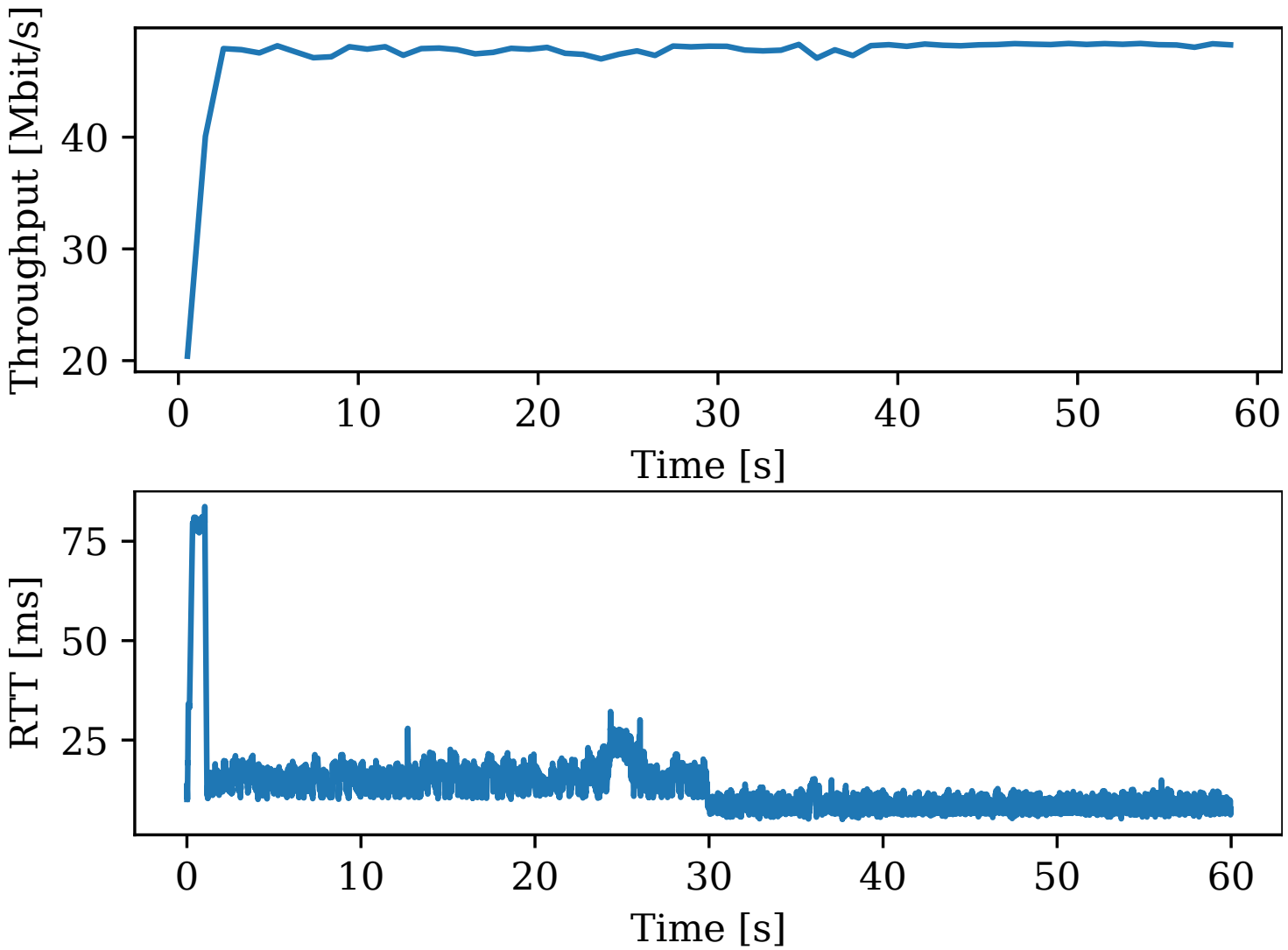
Cubic, `fq_codel` with 100 Mbit/s, 100 ms



Cubic, cocoa with 100 Mbit/s, 100 ms



BBR, cocoa with 50 Mbit/s, 10 ms



Conclusions

- Throughput increase of $\sim 10\%$ for Cubic and $\sim 25\%$ for Reno compared to `fq_codel` and `fq` for high BDP connections
- Same throughput and lower delay for low BDP connections
- `cocoa` finds sweet spot of maximal throughput/minimal queue
- Envisioned deployment close to the Internet's edge

Outlook

- Use previous flows' optimal queue for initialization of new flows' buffers
- More thorough testing with real-world flows
- Reinforcement learning to fingerprint flows and adapt queue dynamically using previously learned experience

TECHNISCHE UNIVERSITÄT WIEN

Cocoa: Congestion Control Aware Queuing

Maximilian Bachl, maximilian.bachl@tuwien.ac.at
Tanja Zseby, tanja.zseby@tuwien.ac.at
Joachim Fabini, joachim.fabini@tuwien.ac.at

Technische Universität Wien, Vienna, Austria