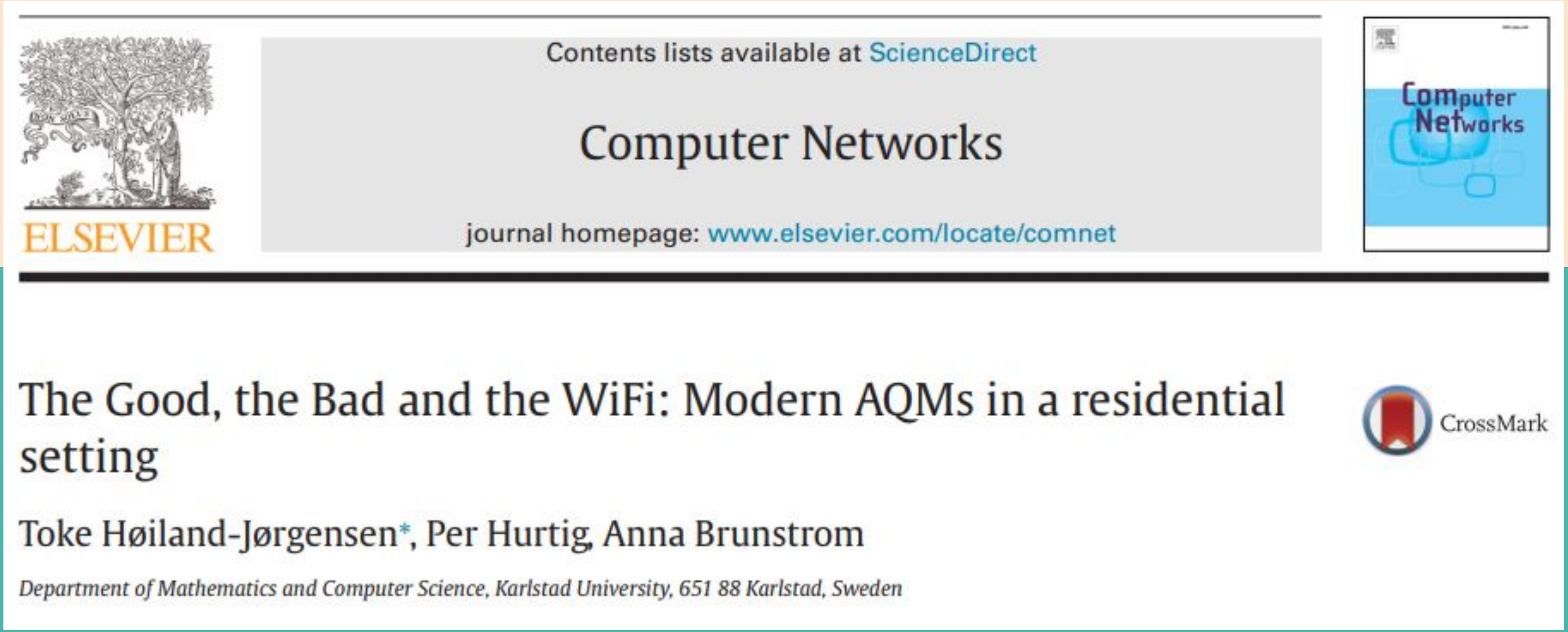
Active Queue Managements In Residential Setting

Ish Kumar Jain
Supervisor: Prof. Shiv Panwar and Fraida Fund

Goal: Reproducing the results of-



Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

The Good, the Bad and the WiFi: Modern AQMs in a residential setting

Toke Høiland-Jørgensen*, Per Hurtig, Anna Brunstrom

Department of Mathematics and Computer Science, Karlstad University, 651 88 Karlstad, Sweden

Problem:

Linux Default: FIFO Queuing

BufferBloat
Network Bottleneck is congested.
Large buffer fill up and do not drain.

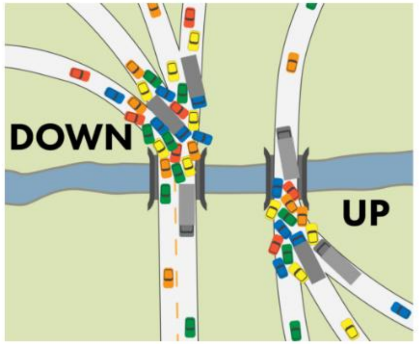


Image Source: https://forum.peplink.com/t/bufferbloat/3331/21

Solution: 3 AQMs

Adaptive RED
Adjust the RED max dropping probability based on observed queue length.

CoDel
Controlled Delay: Directly measures the time pkt spent in a controlled queue.

PIE
Proportional Integral Controller: Infers Delay from instantaneous Queuing occupancy

But,

Can we do better than AQMs?

Fairness Queuing
SFQ: Hashing in sub-queues Served as Round Robin
Fq_codel: FQ with CoDel
Fq_nocodel: FQ without AQM

Performance Metrics

Good

Steady State Behavior: Throughput and Latency behavior.

Real-time Response under load (**RRUL Test**): 4 concurrent TCP with UDP and ICMP packets.

Bad

Transient Behavior: AQMs gives high latency spikes.

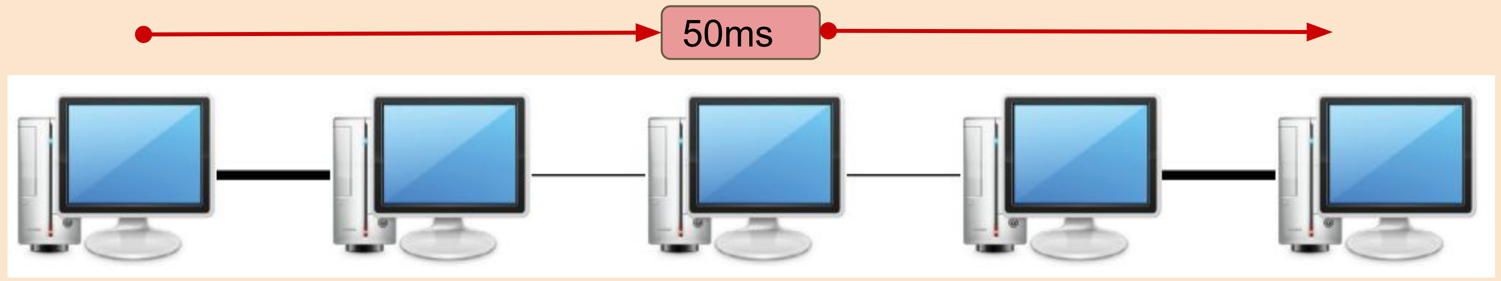
Fairness among flows: AQMs are not fairer

WiFi

Behavior on WiFi Link:

Performance drop as compared to the Wired link in all properties.

Implementation: GENI Testbed



Client:

```
flent rrul
-H server
-o img.png
```

Router1:

```
tc qdisc
add dev
eth2 ared
```

Router2:

```
tc qdisc
add netem
delay 50ms
```

Router3:

```
tc qdisc
add dev
eth1 ared
```

Server:

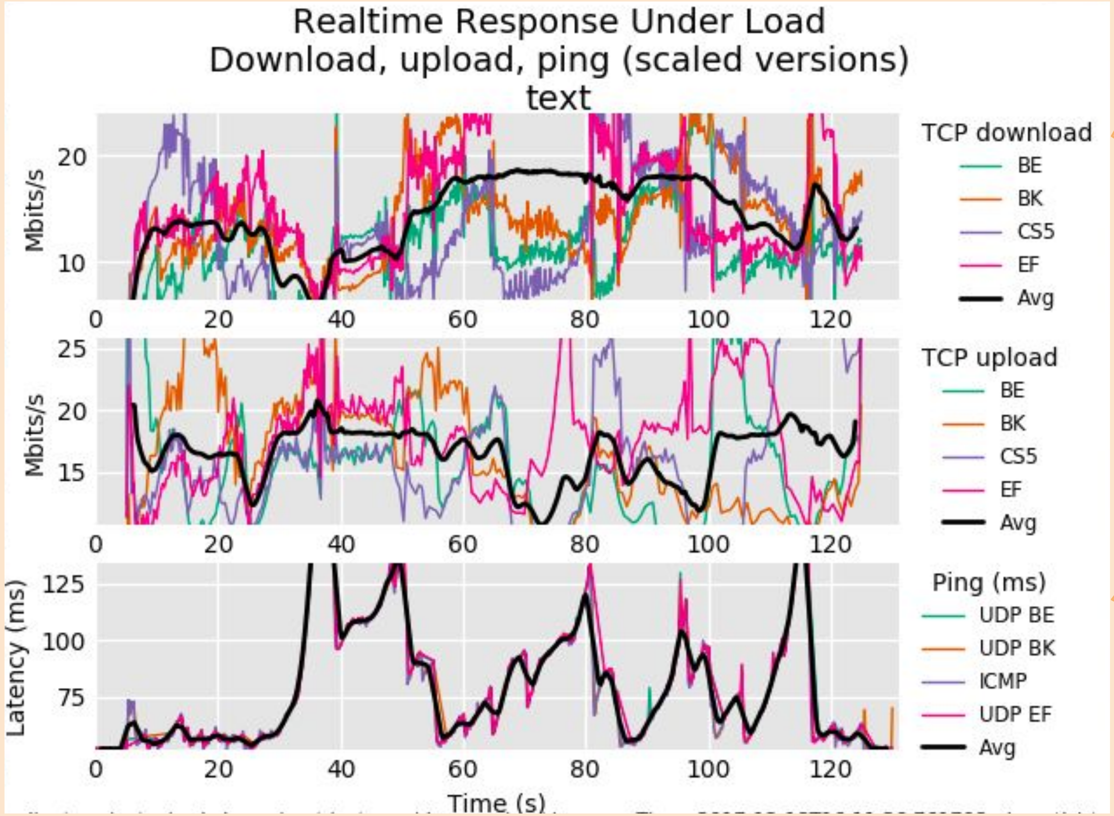
```
netperf
-s
```

Run my Experiment:

Install flent and netperf at client and server | Change ssh and scp commands in the code | Run `bash maincode.sh 10` | Find data in a local directory | Analyze it on Matlab.

Image Source: Toke et.al. 2015 (original paper)

Flent: RRUL Test details



Four priority queues in WiFi

BE: Best Effort

BK: Background,

CS5:Class Selector 5 (video)

EF: Expedient Forwarding(voice)

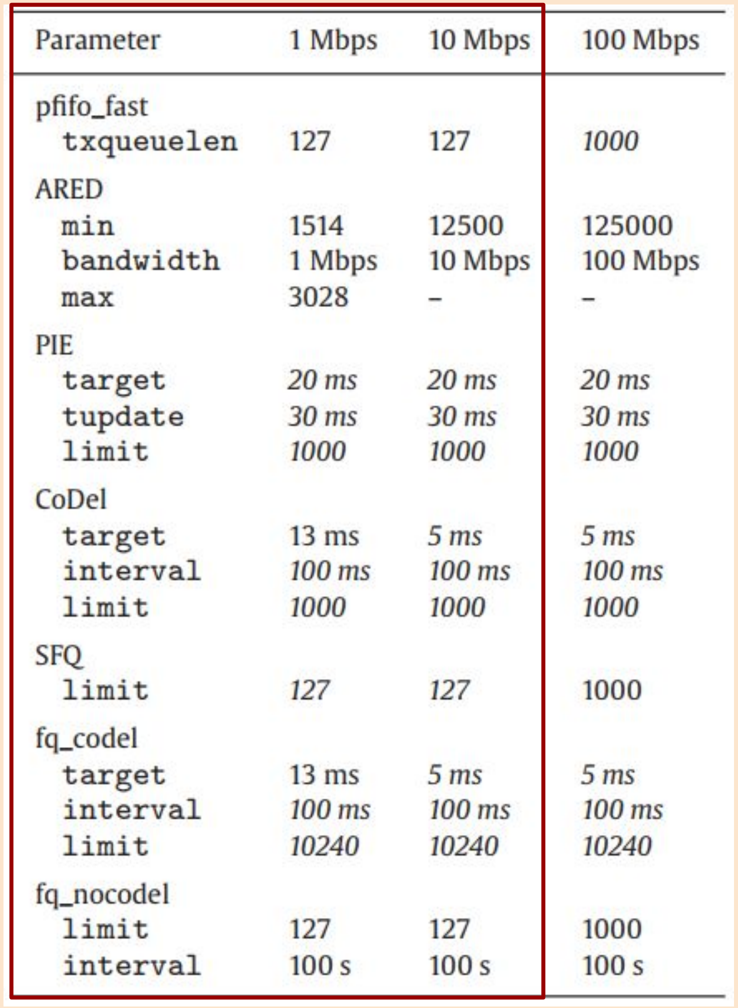
Avg: Average of four flows.

Latency Measurement

Measure the average latency of four concurrent flows (UDP and ICMP)

AQM Parameters

- Many AQM parameters can be tuned.
- But, we use the default parameters for AQMs that have generally shown good results.
- Strict buffer size: 127 packets for most algorithms.
- Left: default parameters in *italic*



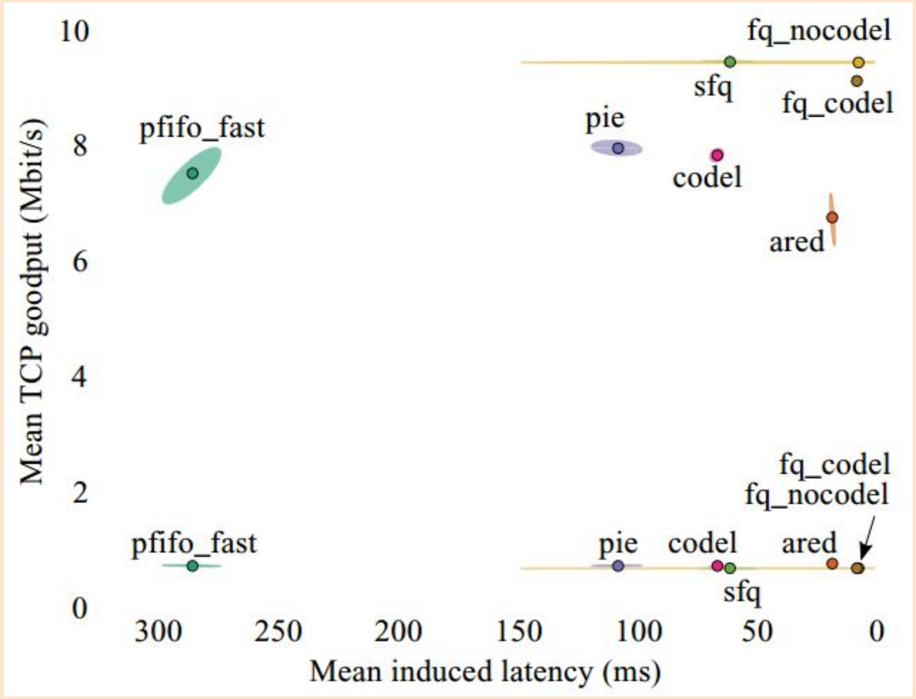
Parameter	1 Mbps	10 Mbps	100 Mbps
pfifo_fast			
`txqueuelen`	127	127	*1000*
ARED			
`min`	1514	12500	125000
`bandwidth`	1 Mbps	10 Mbps	100 Mbps
`max`	3028	–	–
PIE			
`target`	*20 ms*	*20 ms*	*20 ms*
`tupdate`	*30 ms*	*30 ms*	*30 ms*
`limit`	*1000*	*1000*	*1000*
CoDel			
`target`	13 ms	*5 ms*	*5 ms*
`interval`	*100 ms*	*100 ms*	*100 ms*
`limit`	*1000*	*1000*	*1000*
SFQ			
`limit`	*127*	*127*	1000
fq_codel			
`target`	13 ms	*5 ms*	*5 ms*
`interval`	*100 ms*	*100 ms*	*100 ms*
`limit`	*10240*	*10240*	*10240*
fq_nocodel			
`limit`	127	127	1000
`interval`	100 s	100 s	100 s

Implementation details

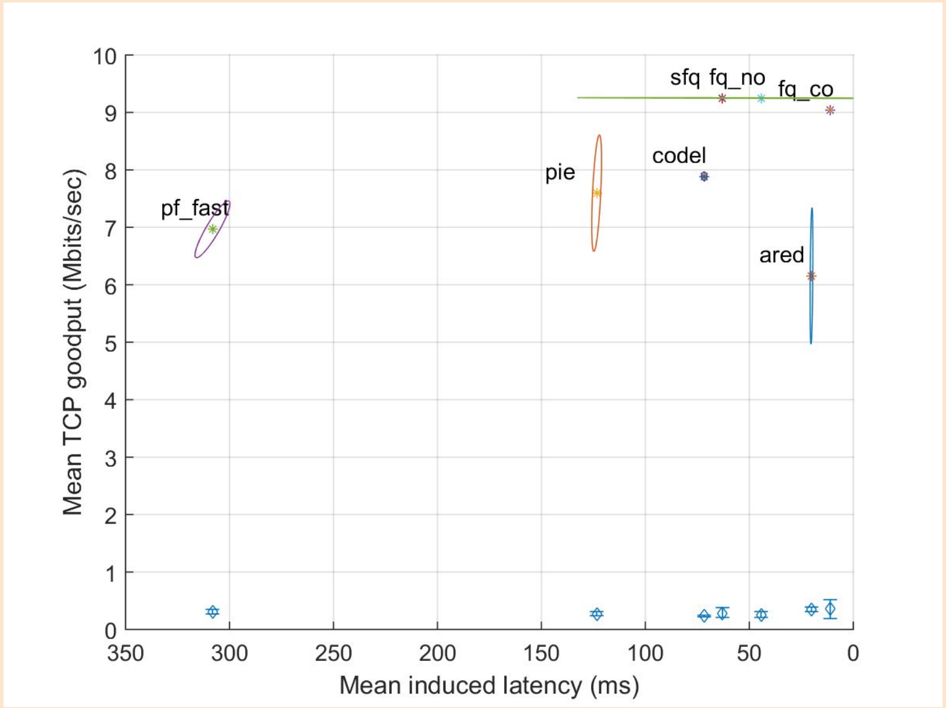
Required by Paper	My Implementation	Description
Use Dummynet to generate delay	Used netem instead	Dummynet not compatible
Run Flent and netperf for both RRUL and fairness test.	Flent used for RRUL test. But, iperf for fairness test	Due to difficulties in implementing multiple flows with Flent.
Repeat 30 times	Repeated 5 times	It takes a lot of time.
Hardware offload feature turn off	Turned off	On client: `ethtool -K interface tso off gso off`
Kernel clock frequency 1000 Hz	250 Hz	Caused issue at 100 Mbps
Tcp default cubic algorithm	Used cubic	Other TCP algos give same results
Fairness test on 4 concurrent flows	Implemented only 3 flows	Netem didn’t allow >3 firewall rule

Results: RRUL Test 10/1

Our results are comparable to that of the paper. A small difference in magnitude can be attributed to some of the implementation issues discussed previously.

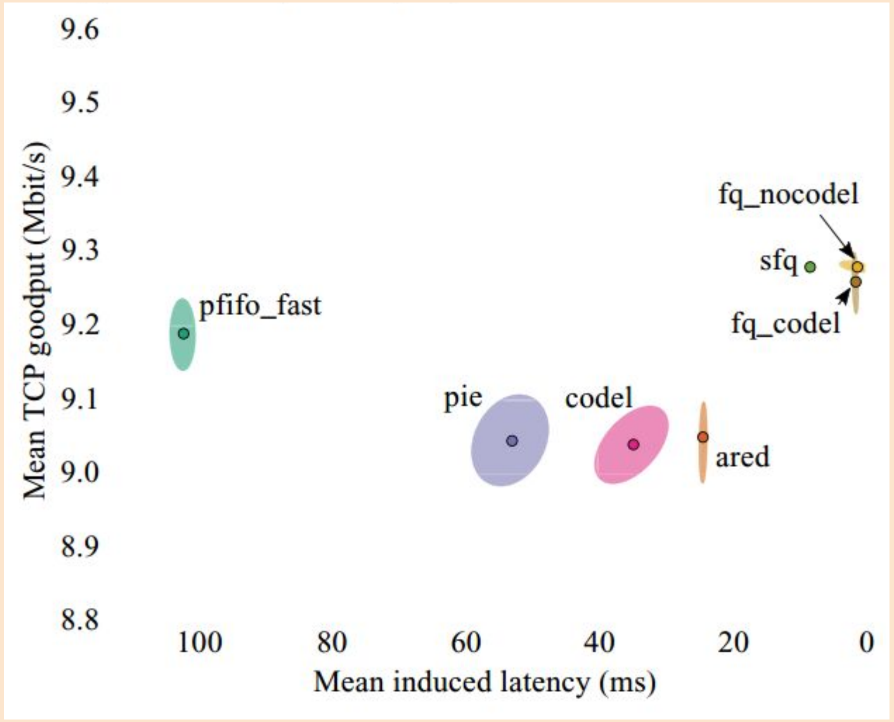


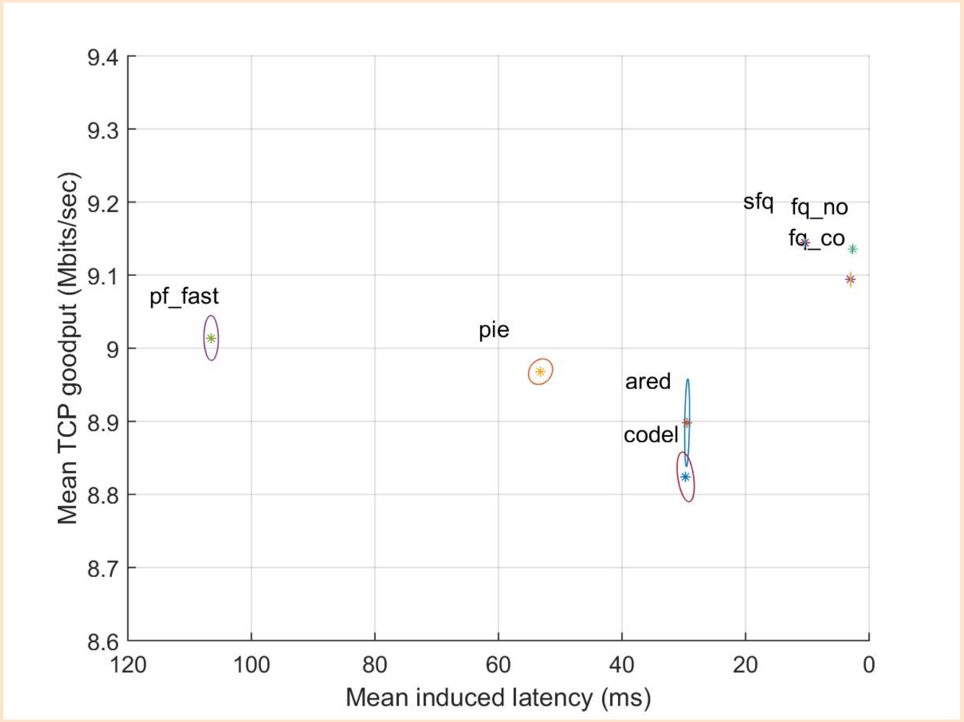
Toke et.al.



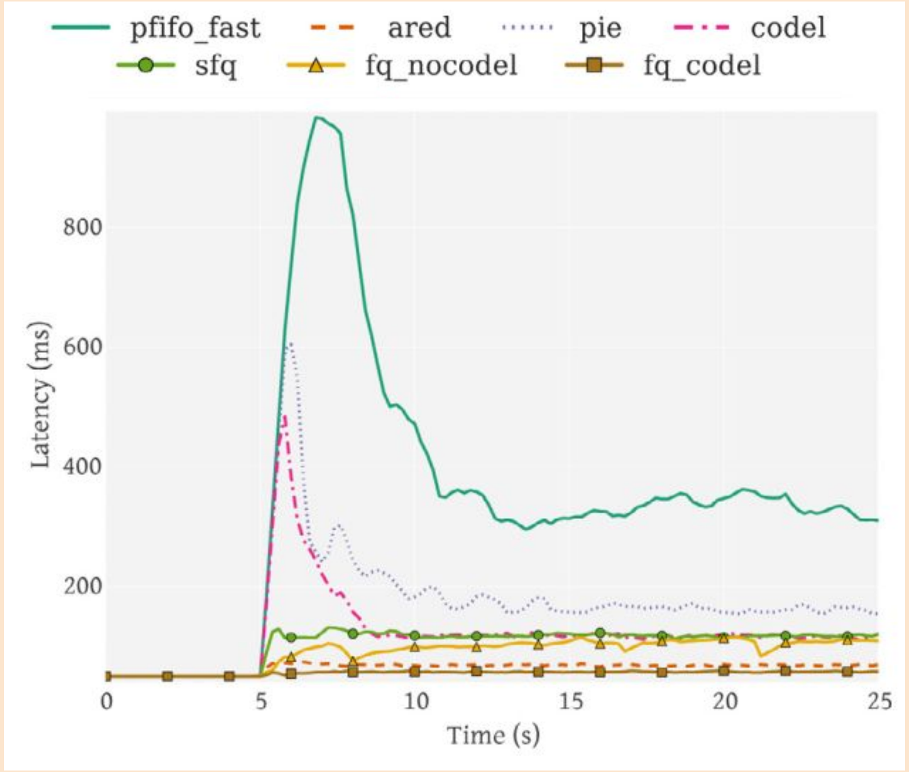
Our Results

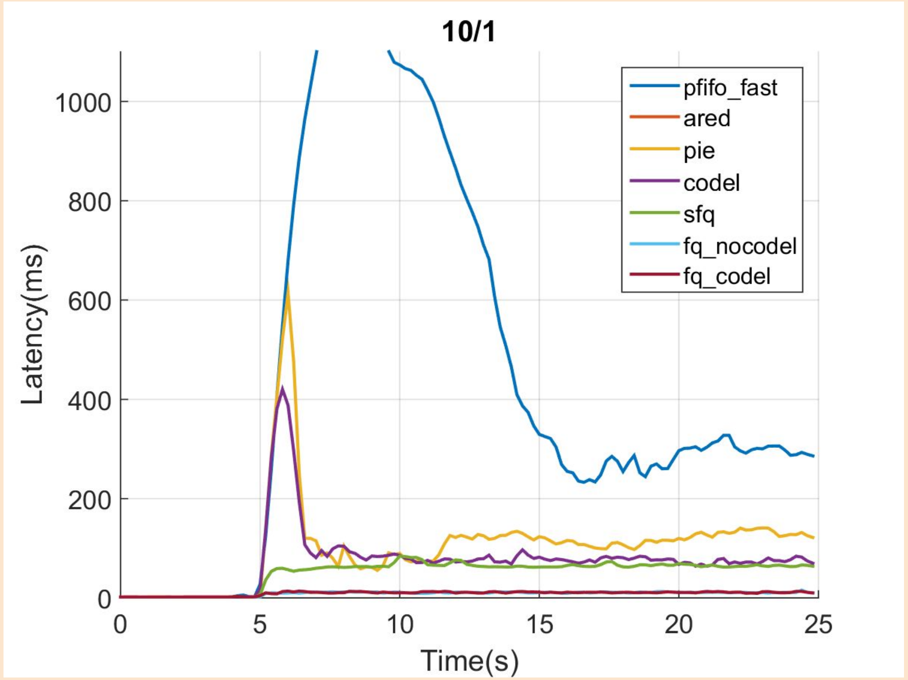
RRUL 10/10


9.6
9.5
9.4
9.3
9.2
9.1
9.0
8.9
8.8
Mean TCP goodput (Mbit/s)
fq_nocodel
sfq
fq_codel
pfifo_fast
pie
codel
ared
100
80
60
40
20
0
Mean induced latency (ms)

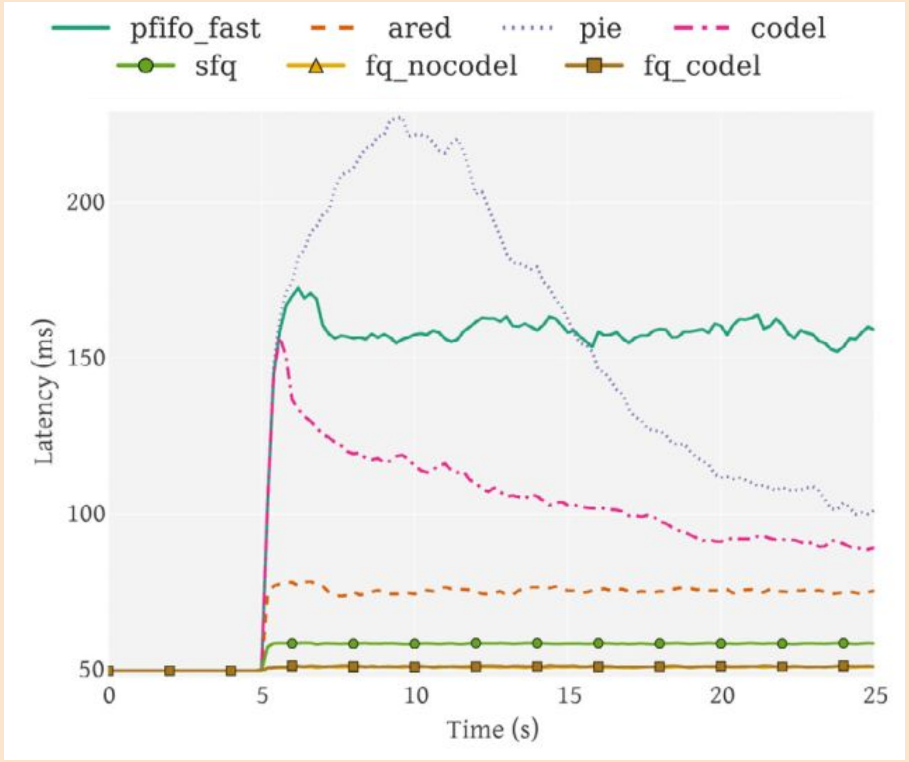

9.4
9.3
9.2
9.1
9
8.9
8.8
8.7
8.6
Mean TCP goodput (Mbits/sec)
sfq
fq_no
fq_co
pf_fast
pie
ared
codel
120
100
80
60
40
20
0
Mean induced latency (ms)

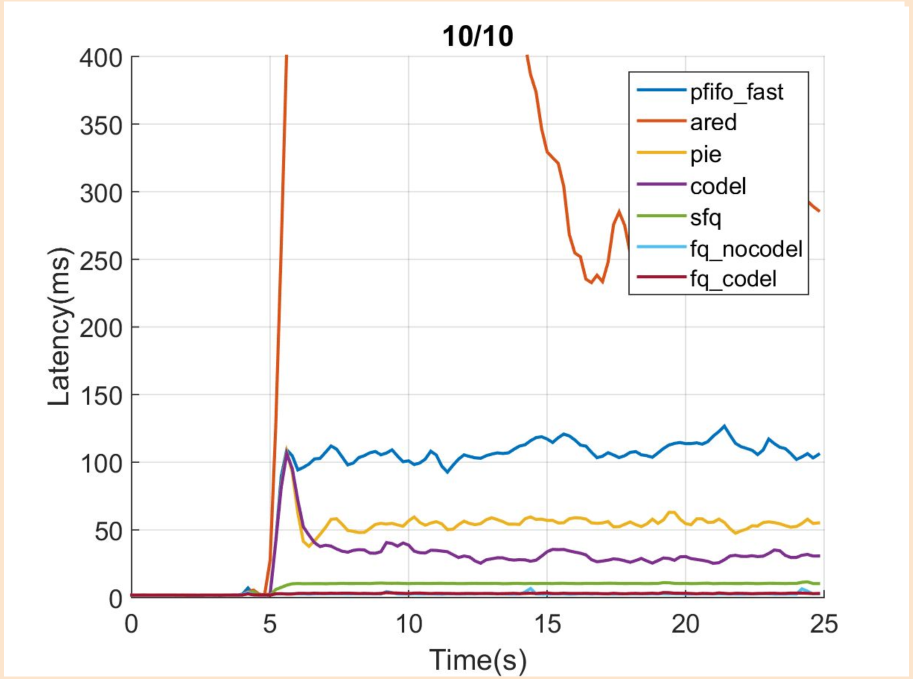
Transient 10/1



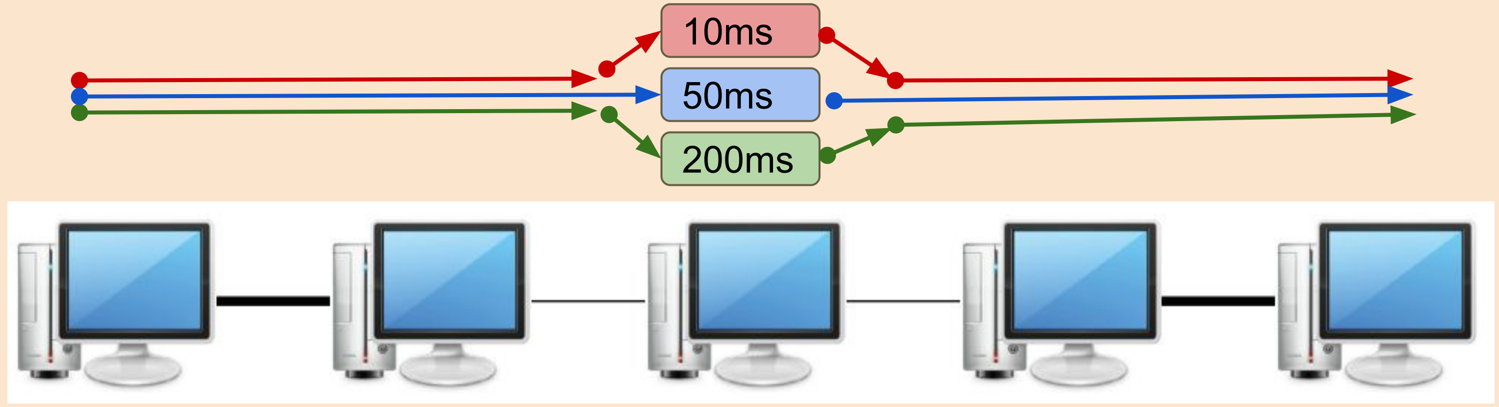


Transient 10/10





Fairness Test Implementation



Client:

Iperf -c server on 3 ports. Measure per-flow throughput.

Router1:

```
tc qdisc
add dev
eth2 ared
```

Router2:

Use netem for diff delays.

Router3:

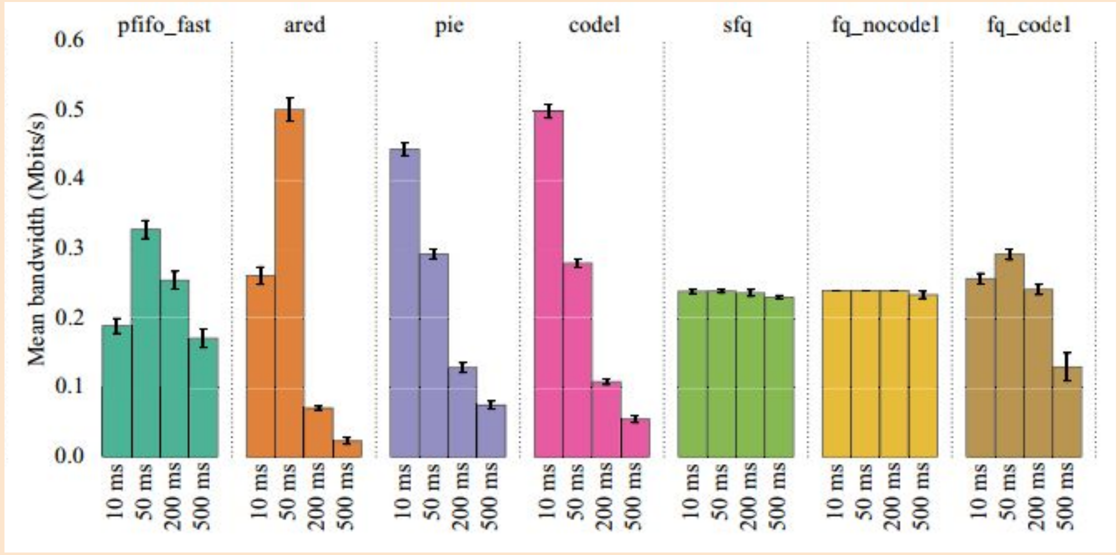
```
tc qdisc
add dev
eth1 ared
```

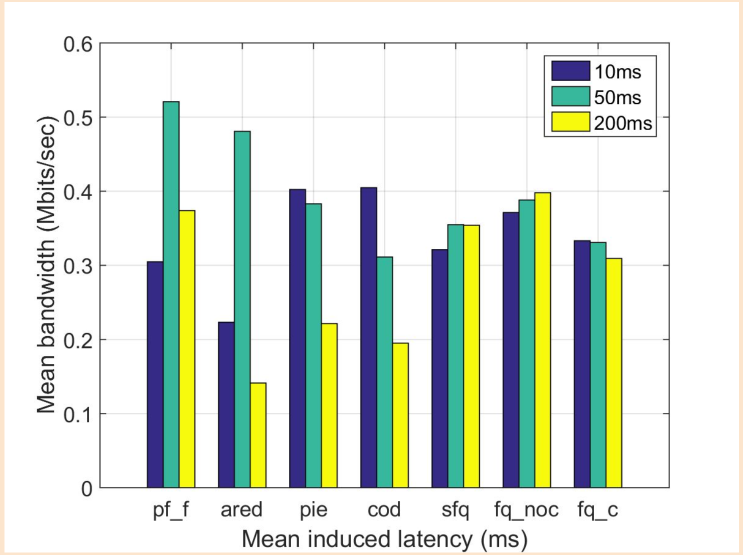
Server:

Iperf -s on 3 ports

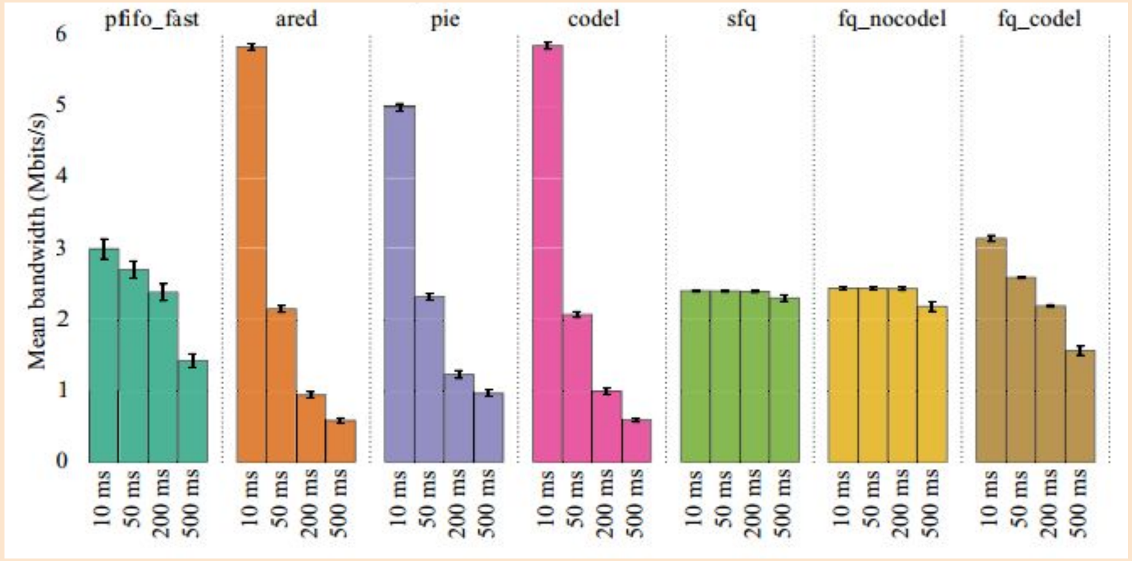
Image Source: Toke et.al. 2015 (original paper)

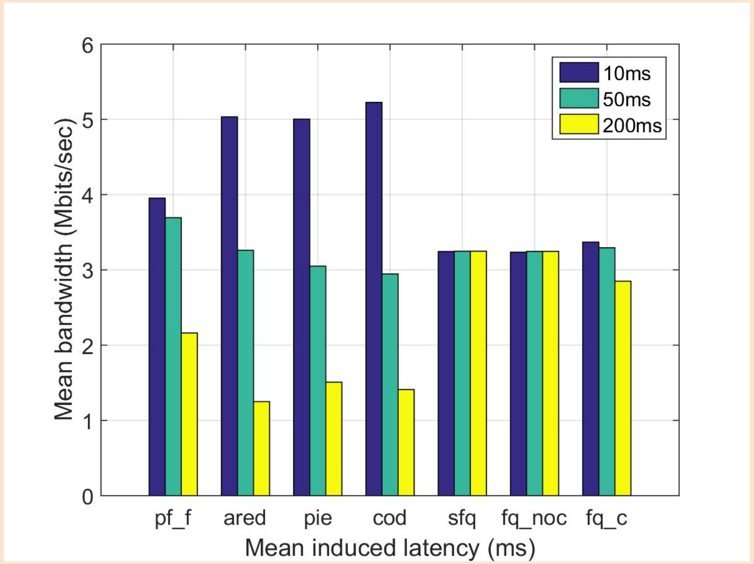
Fairness 10/1





Fairness 10/10



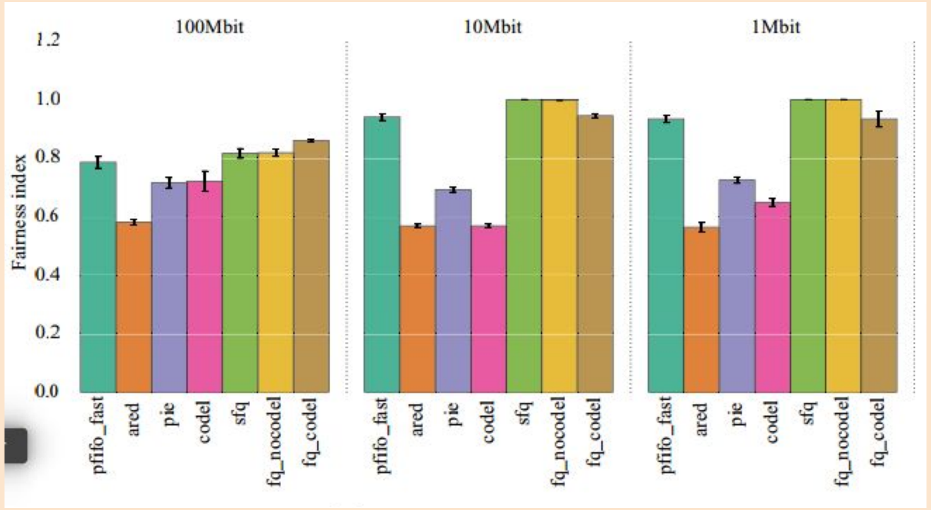


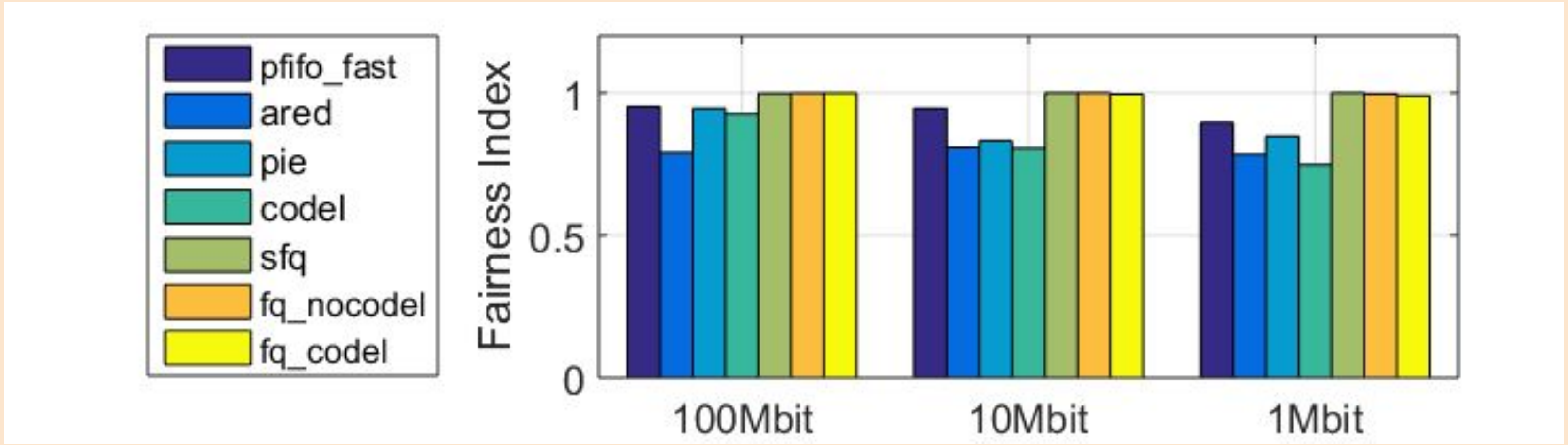
Fairness Index

Fairness Index for 3 flows:

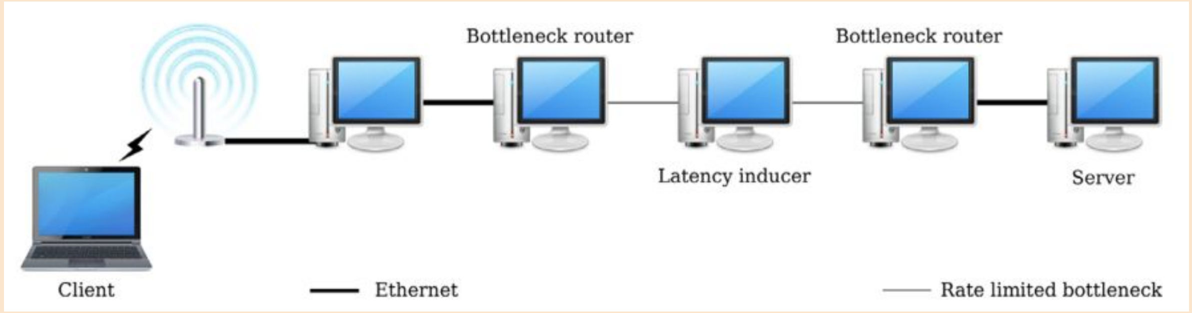
$$\mathcal{L} = \frac{(x_1 + x_2 + x_3)^2}{3(x_1^2 + x_2^2 + x_3^2)}$$

Value between 0 and 1.

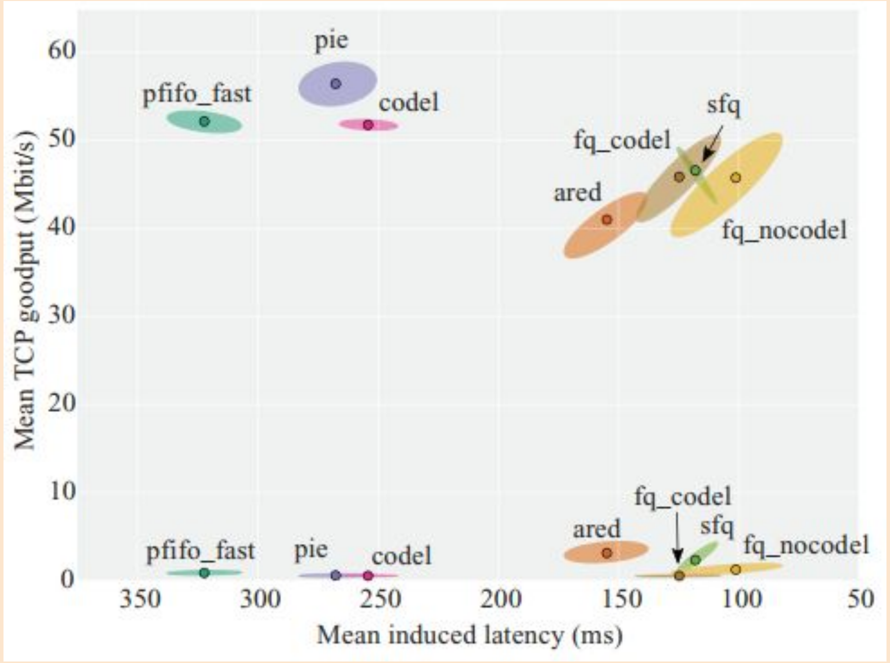




WiFi Test



- Throughput Latency graph for 100 Mbps flows
- Result: same ordering of latency behavior.
  But, magnitude is different.
  (Minimum 100ms)
- Reason: Queuing at lower layers (no control in WiFi).



Conclusion

- We compared the performance of three AQMs and fairness queueing.
- Good: AQMs perform better than FIFO in terms of throughput, and latency.
- Bad: AQMs give a spike in latency at transient phase and they have a tendency to exacerbate the unfairness issue.
- WiFi: AQMs failed to perform better.

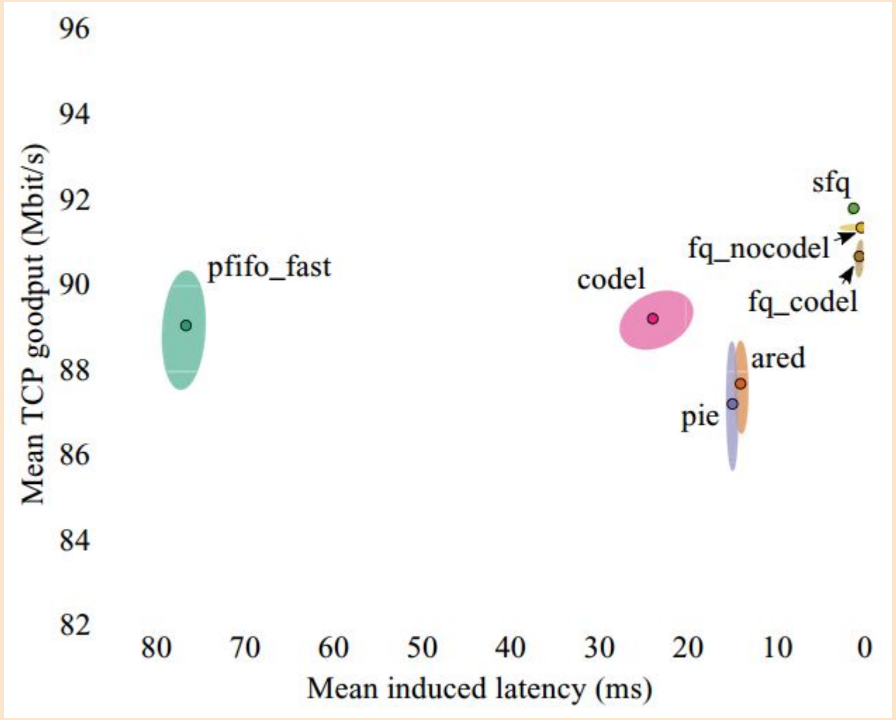
Thank You

Questions!!

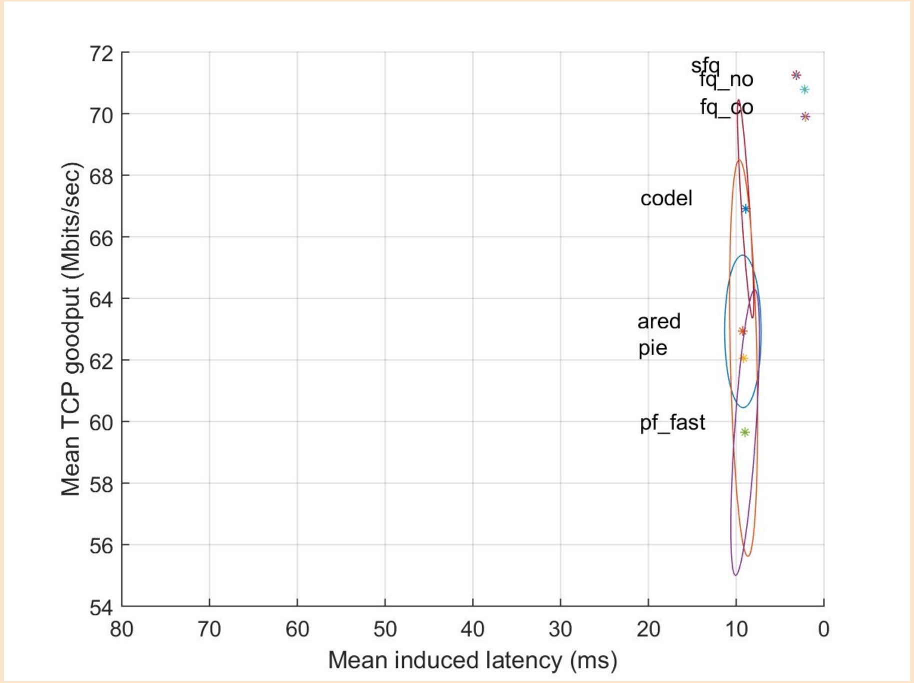
BackUp Slides

100/100 Mbps Flow

RRUL 100/100

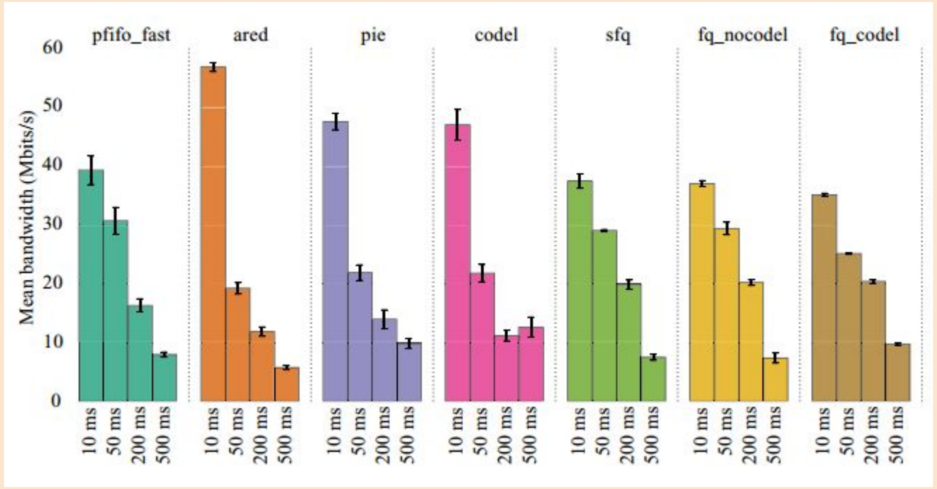


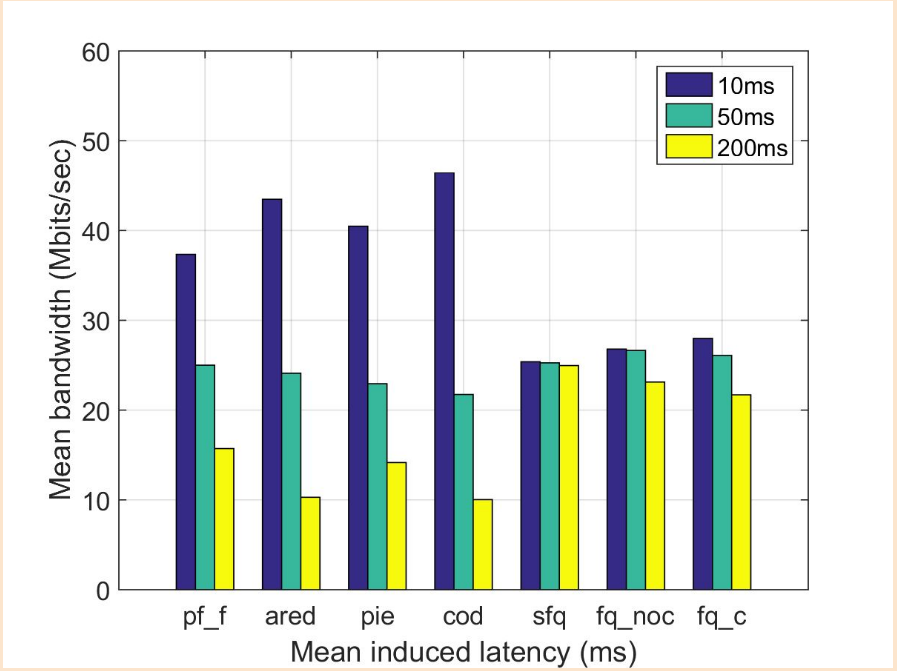
Mean TCP goodput (Mbit/s)
96
94
92
90
88
86
84
82
80
70
60
50
40
30
20
10
0
Mean induced latency (ms)
pfifo_fast
codel
sfq
fq_nocodel
fq_codel
ared
pie



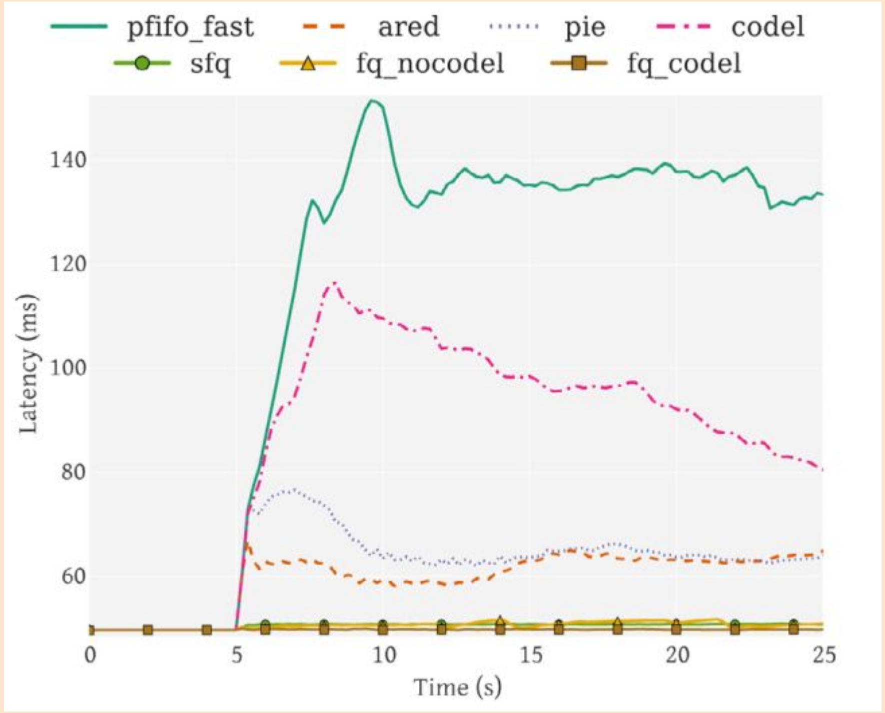
Mean TCP goodput (Mbits/sec)
72
70
68
66
64
62
60
58
56
54
80
70
60
50
40
30
20
10
0
Mean induced latency (ms)
sfq
fq_no
fq_co
codel
ared
pie
pf_fast

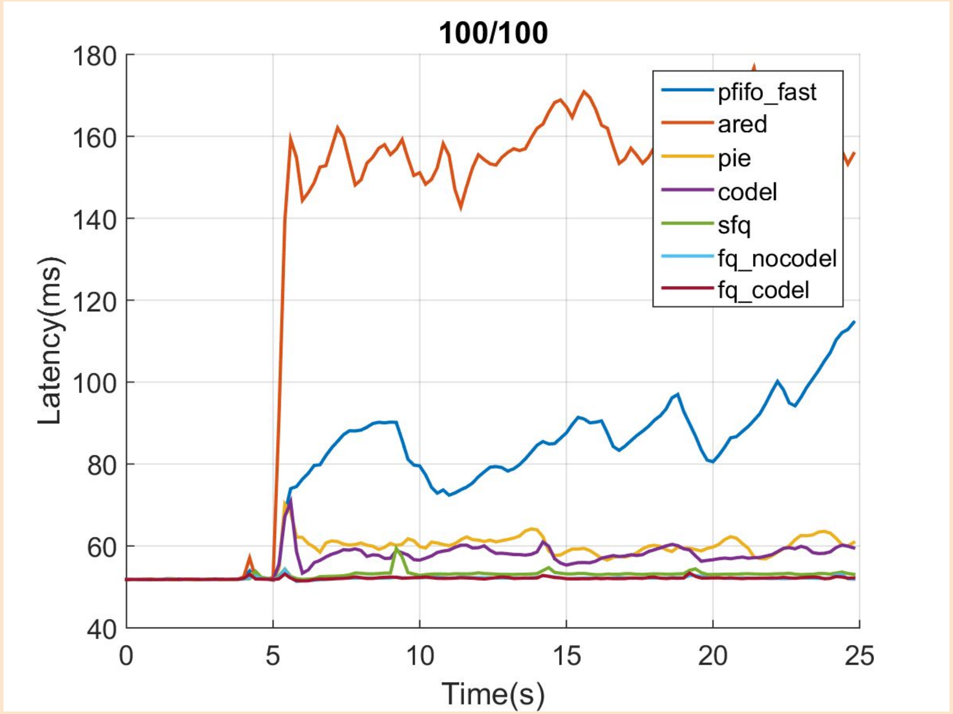
Fairness 100/100





Transient 100/100


pfifo_fast
ared
pie
codel
sfq
fq_nocodel
fq_codel
Latency (ms)
Time (s)


100/100
pfifo_fast
ared
pie
codel
sfq
fq_nocodel
fq_codel
Latency(ms)
Time(s)