Latency-Based WiFi Congestion Control in the Air for Dense WiFi Networks

Changhua Pei, Youjian Zhao, Yunxin Liu, Kun Tan,

Jiansong Zhang, Yuan Meng, Dan Pei

IWQOS 2017

Motivation

Algorithm

Implementation

Evaluation

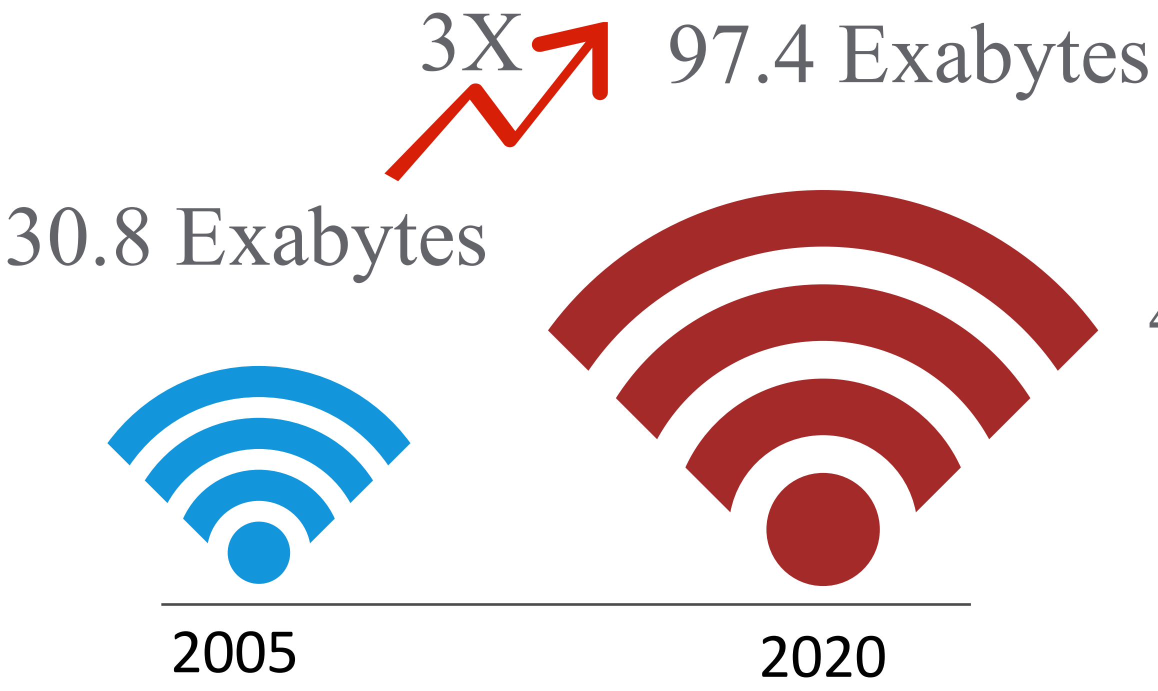
Motivation

Algorithm

Implementation

Evaluation

WiFi is indispensable in our daily lives

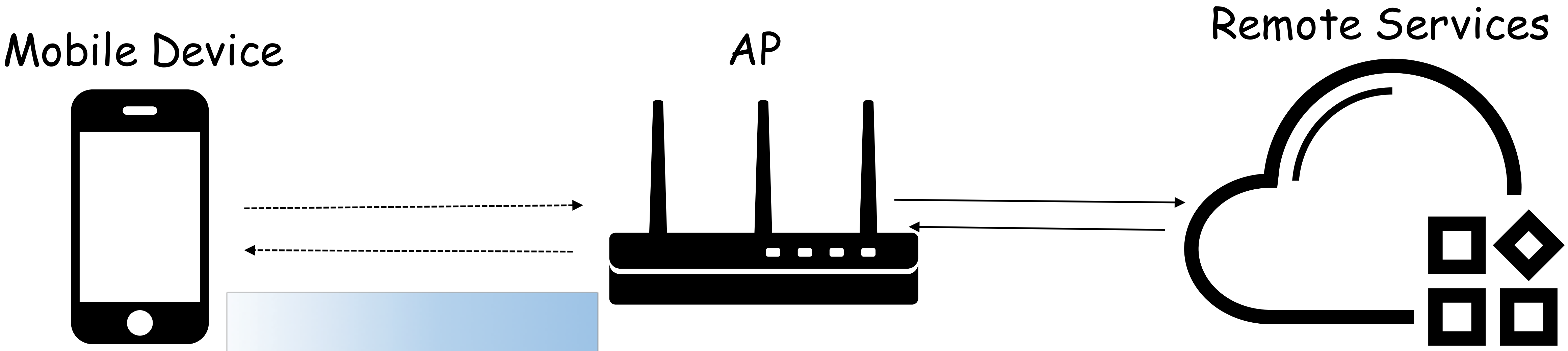


Overall WiFi Traffic Growth

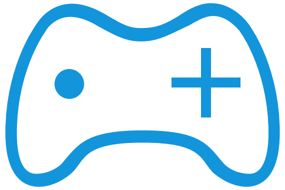
49% of overall traffic is WiFi traffic in 2020.

Source: Cisco VNI Mobile, 2017

Experience of WiFi Network



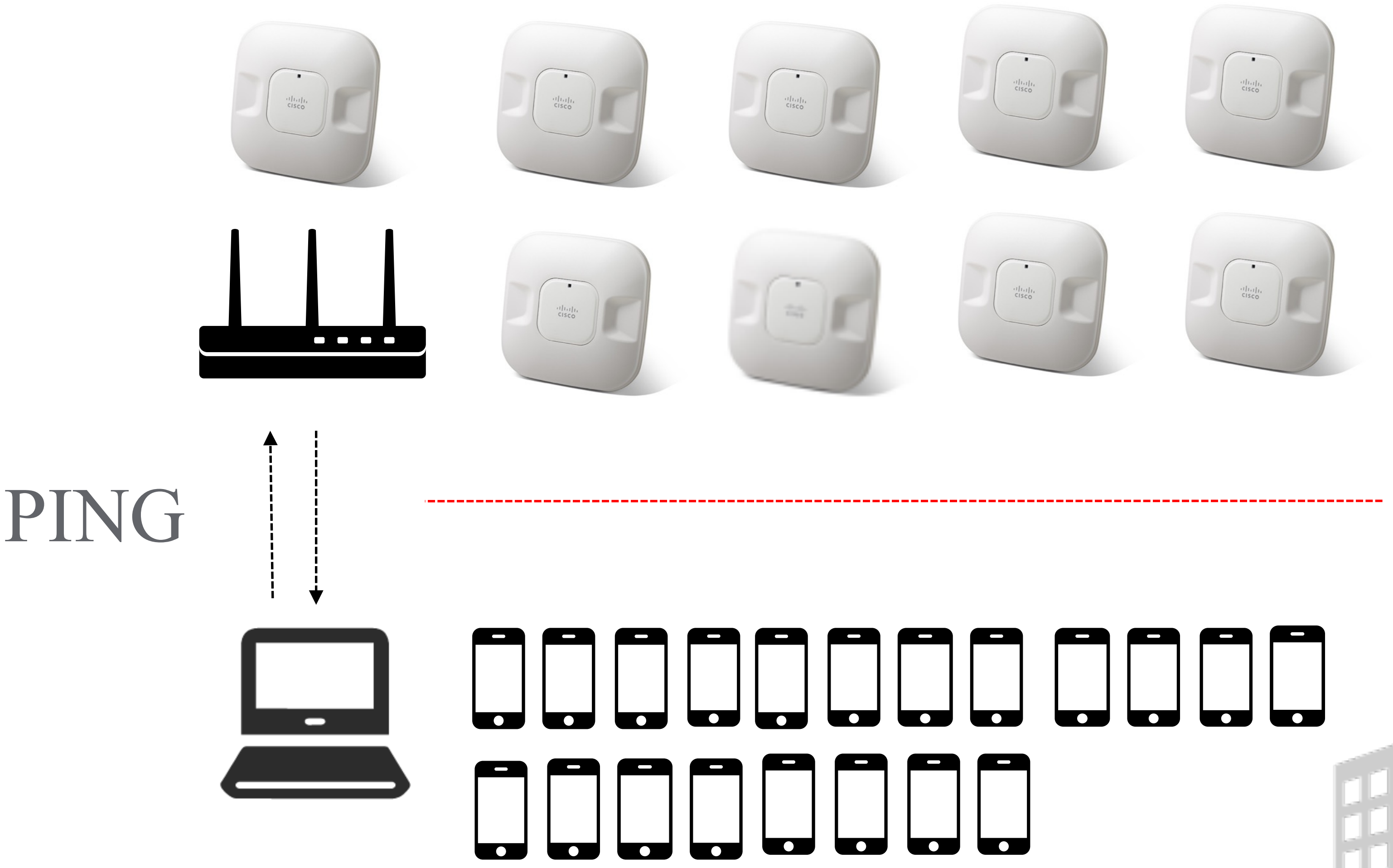
WiFi Hop Latency



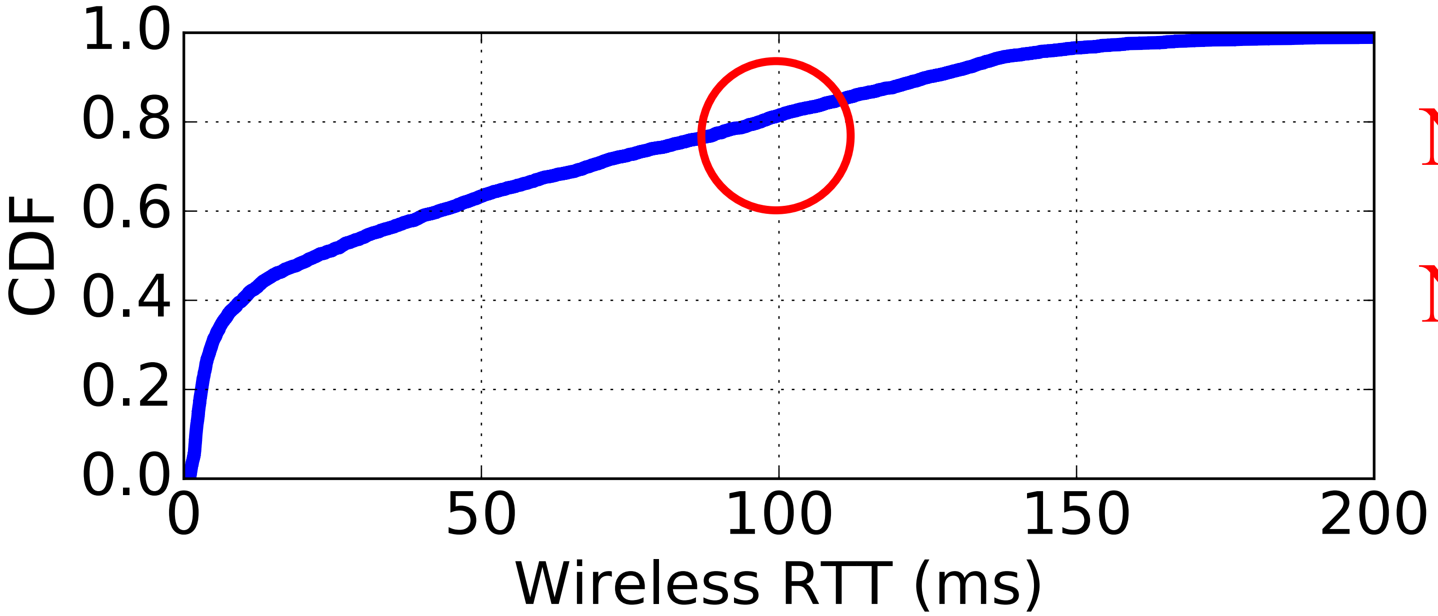
Online Gaming

0.2 billion downloads
50 million active users every day

WiFi Hop Latency



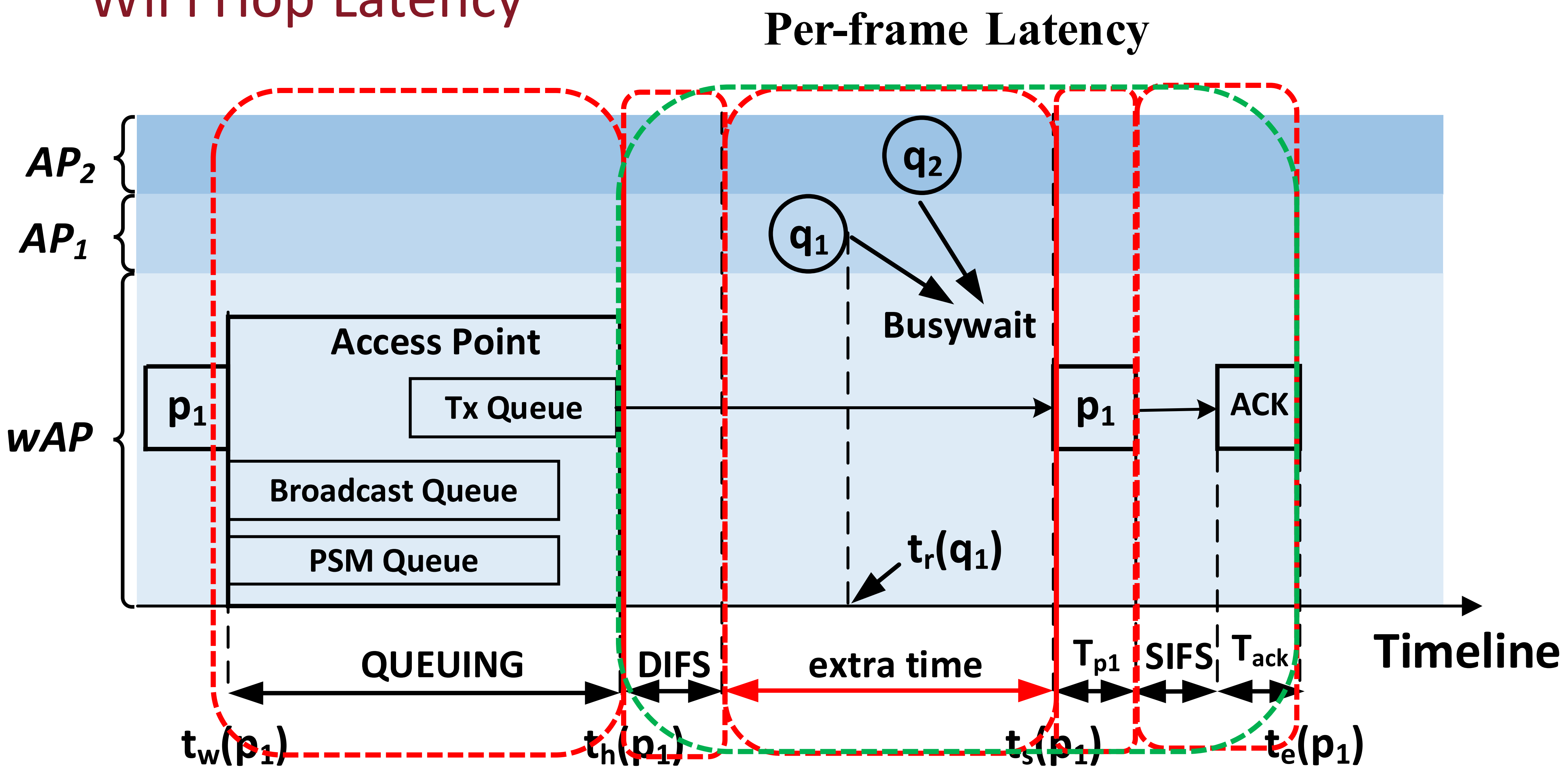
WiFi Hop Latency



NO local contention

NO low SNR problem

WiFi Hop Latency



Per-frame Latency

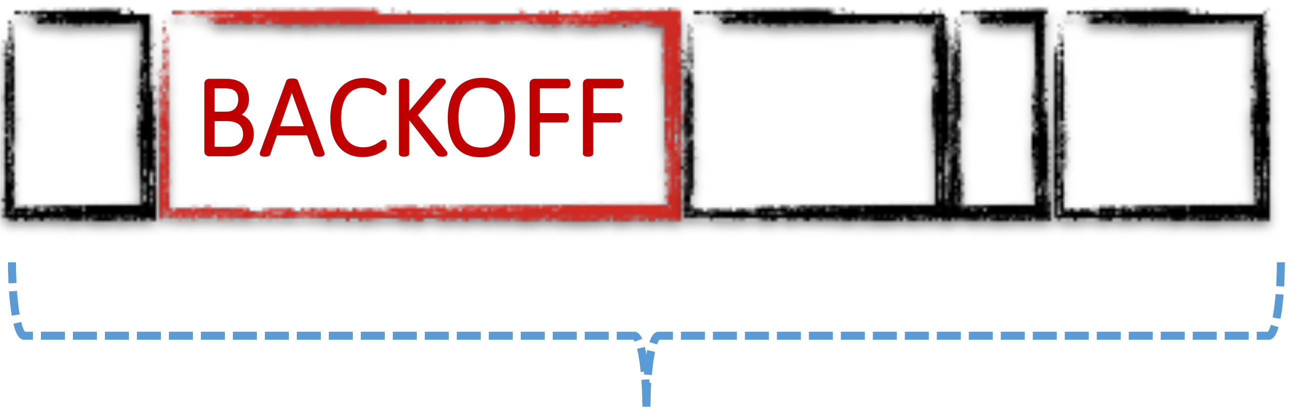
$$t_{const} = DIFS + SIFS + t_{ack}$$

$$t_{tr} = framesize/phyrate$$

BACKOFF

$$t_{cont}^{max} = SLOT * CW_{max} + \sum_{1}^{\beta-1} \frac{framesize}{phyrate}$$

```
N = CWmin
1. Wait DIFS until channel becomes idle.
2. If channel is not busy, go to step 4.
3. Perform a regular backoff
   bo = rand[0, N]
   While bo > 0
      probe channel for 20us
      if busy wait DIFS until idle.
      --bo
4. Send the frame; if no ACK is received,
   double N and retry from step 1.
5. N = CWmin, perform a mandatory backoff
   as in step 3.
```



$$t^{max} \approx t_{const} + SLOT * CW_{max} + \beta * \frac{avgframesize}{avgphyrate}$$

Motivation

- Experience of delay sensitive applications depends on the WiFi hop latency.
- 20% of packets' WiFi hop latency is larger than 100ms.
- The latency increases linearly with the number of contenders because of the current DCF mac-layer protocol.

There is urgent need to revisit the mac protocol as the increasing number of contenders.

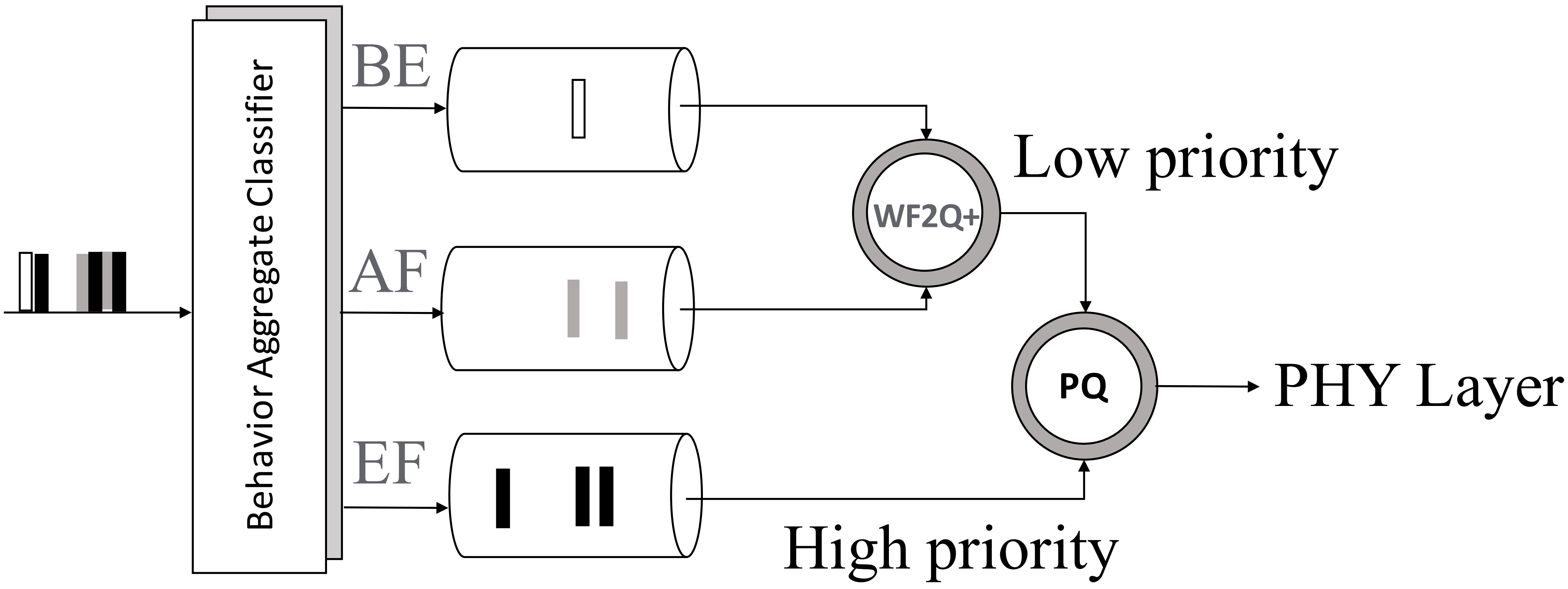
Motivation

Algorithm

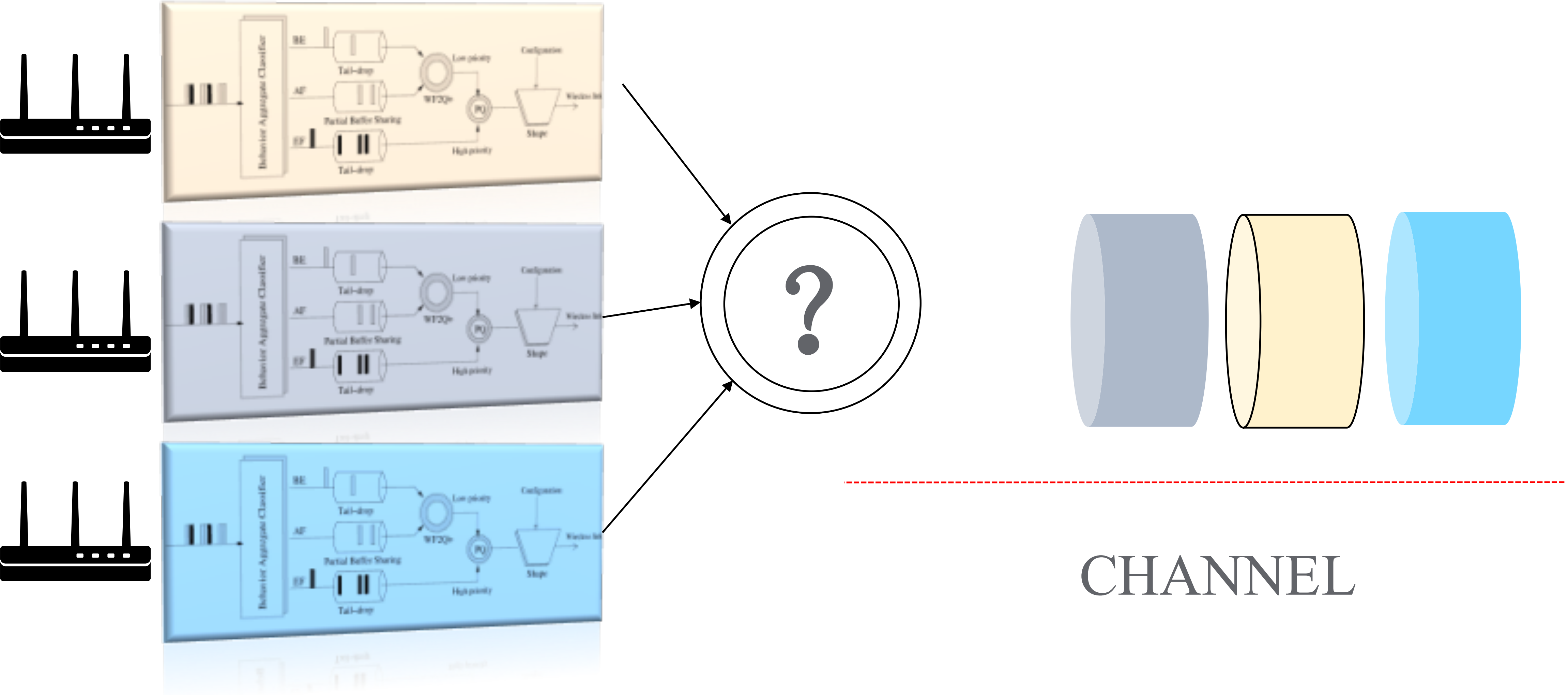
Implementation

Evaluation

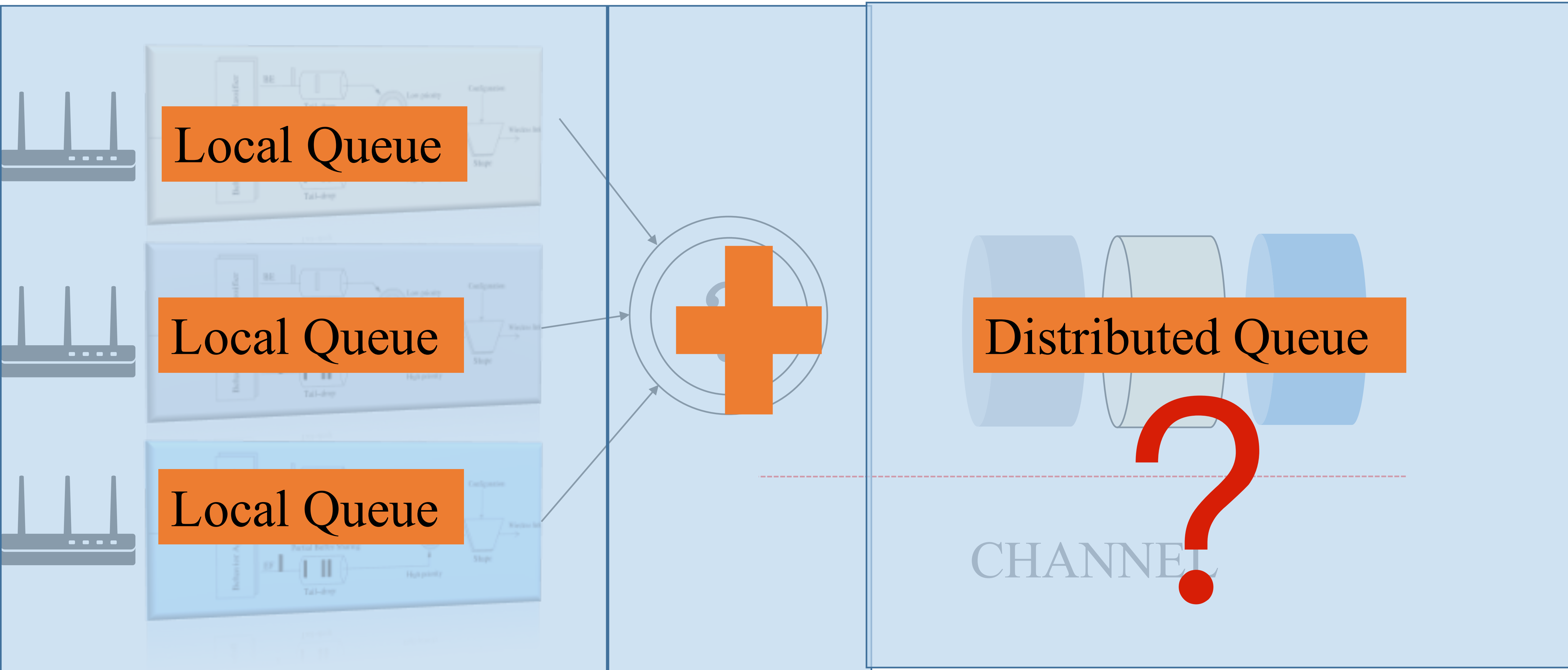
End-host QoS Method



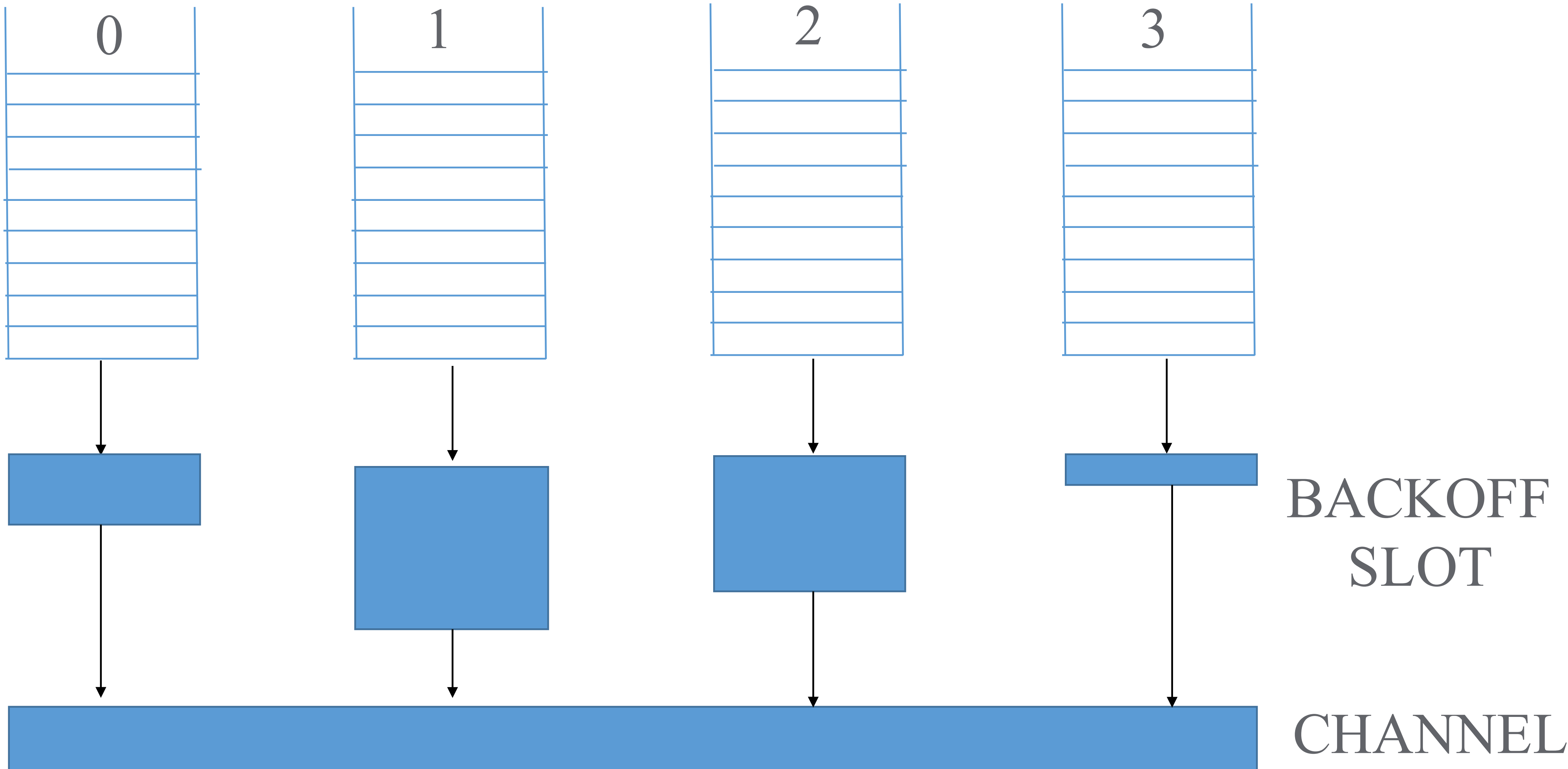
End-host QoS Method



End-host QoS Method



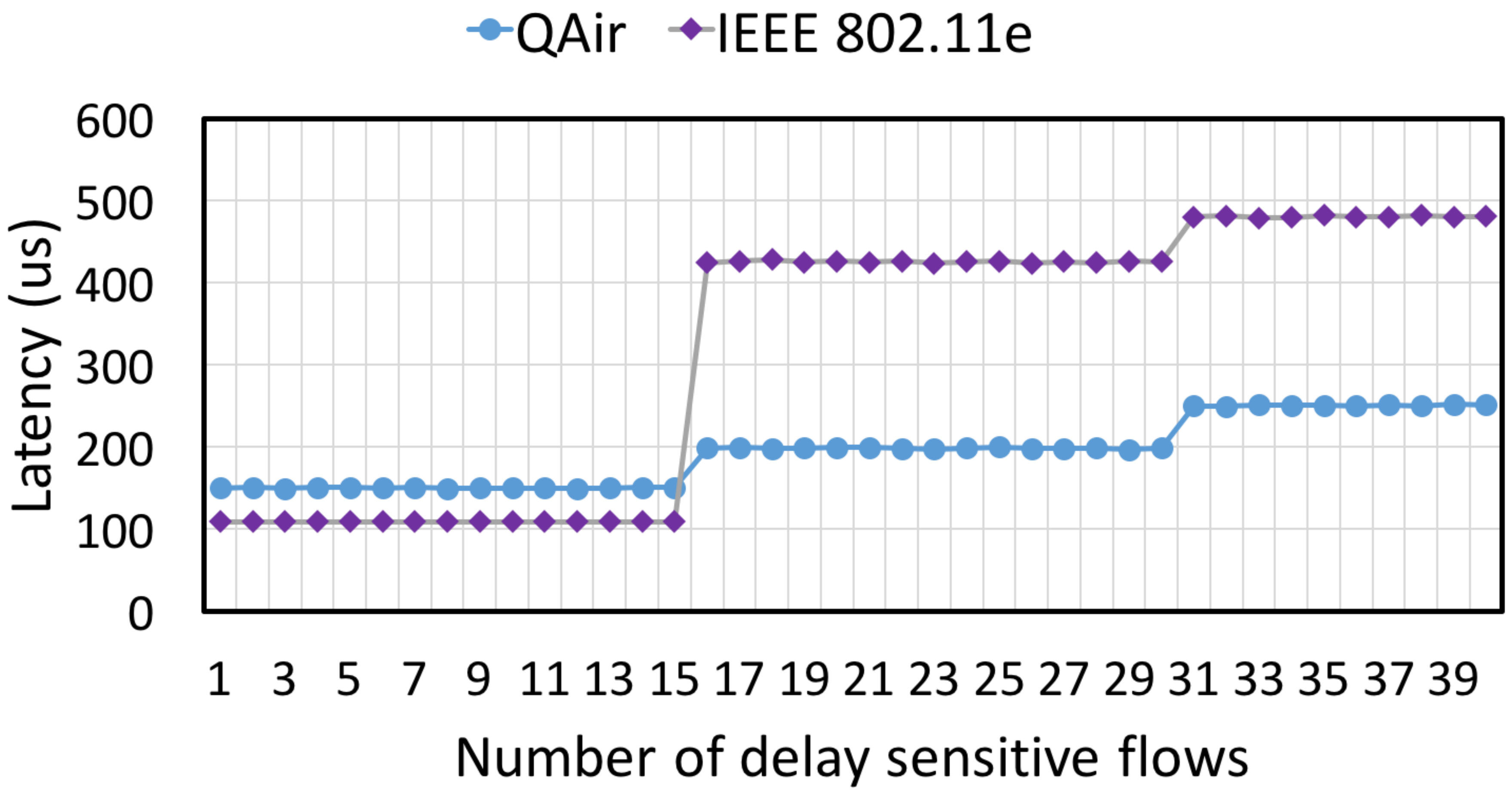
IEEE 802.11e



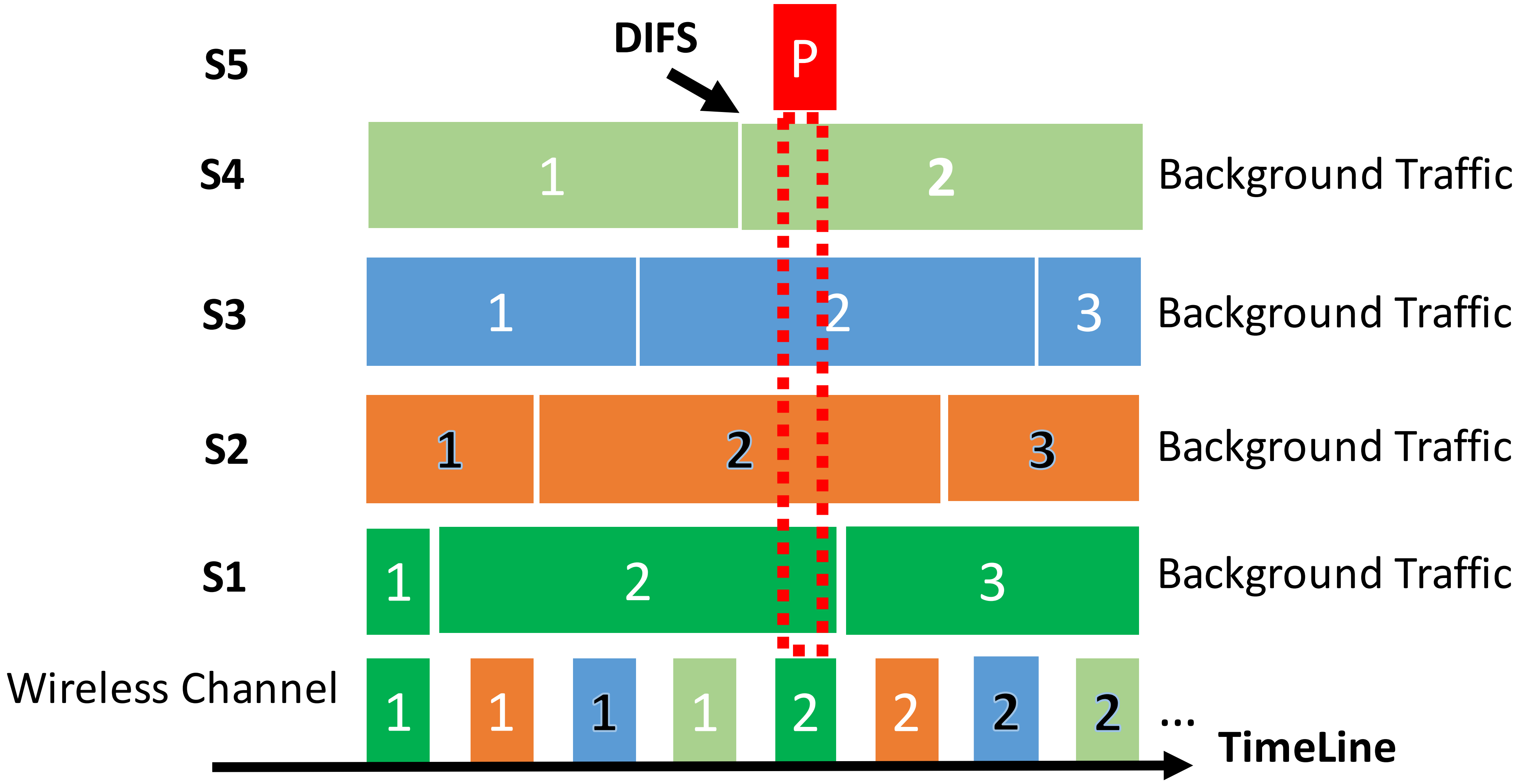
IEEE 802.11e

- Smaller window
  - For dense environment: higher collision
  - Enlarge the smallest window: low utilization
- Cross-layer configuration
- Mixed traffic flows on one IP port, e.g., HTTP traffic on 80

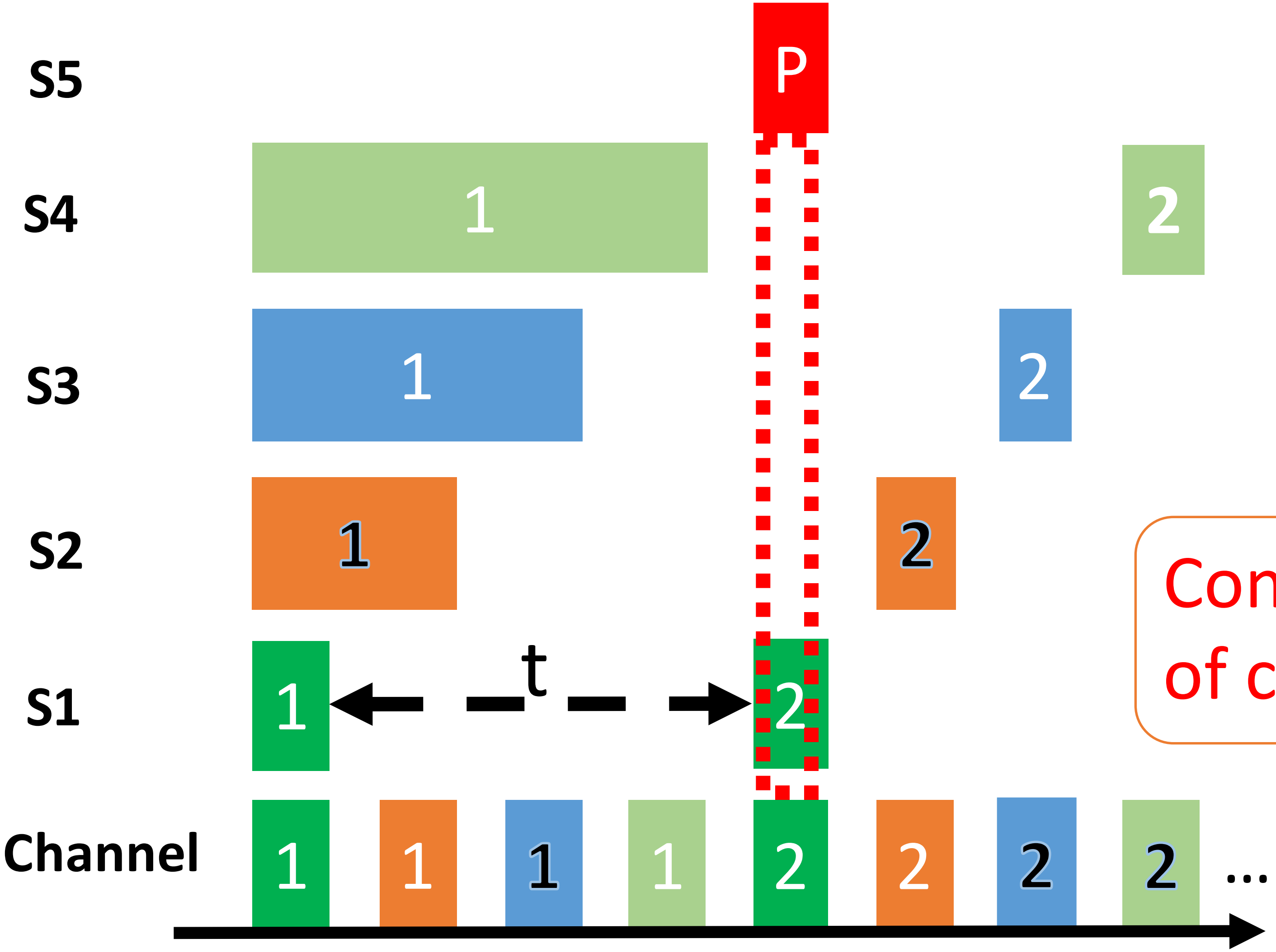
IEEE 802.11e


QAir
IEEE 802.11e
Latency (us)
0
100
200
300
400
500
600
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39
Number of delay sensitive flows

Legacy 802.11 DCF

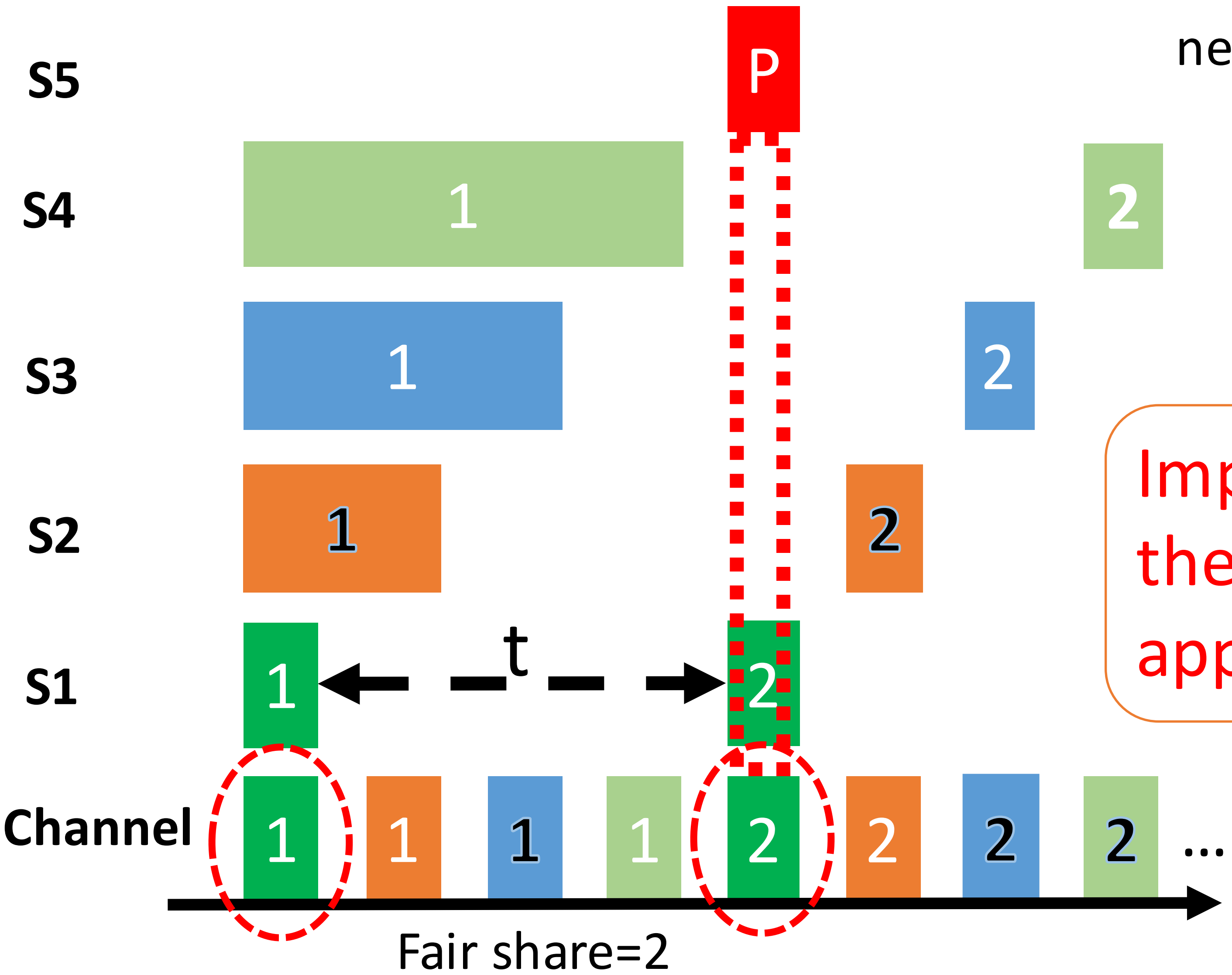


QAir Algorithm



Control the number of contenders.

QAir Algorithm



Why S5's packets do not need to be deferred?

Implicit priority to the delay-sensitive applications.

Core idea of QAir:

1. Control the number of concurrent contenders to reduce the length of distributed queue.
2. Based on assigning different flows fair shares, QAir assigns implicit priority to the delay-sensitive flows.

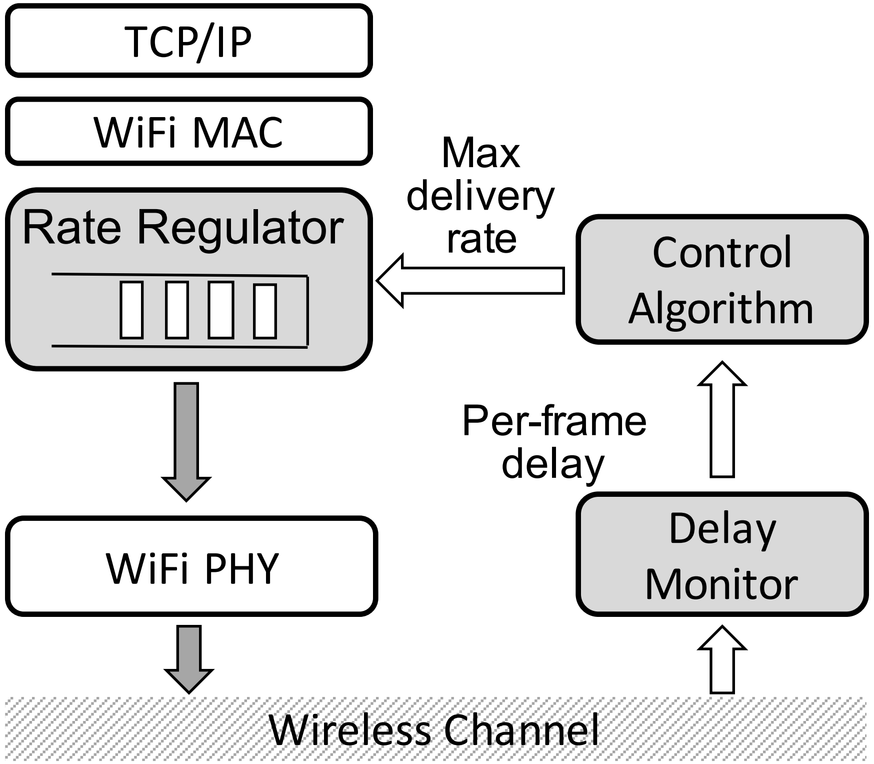
Motivation

Algorithm

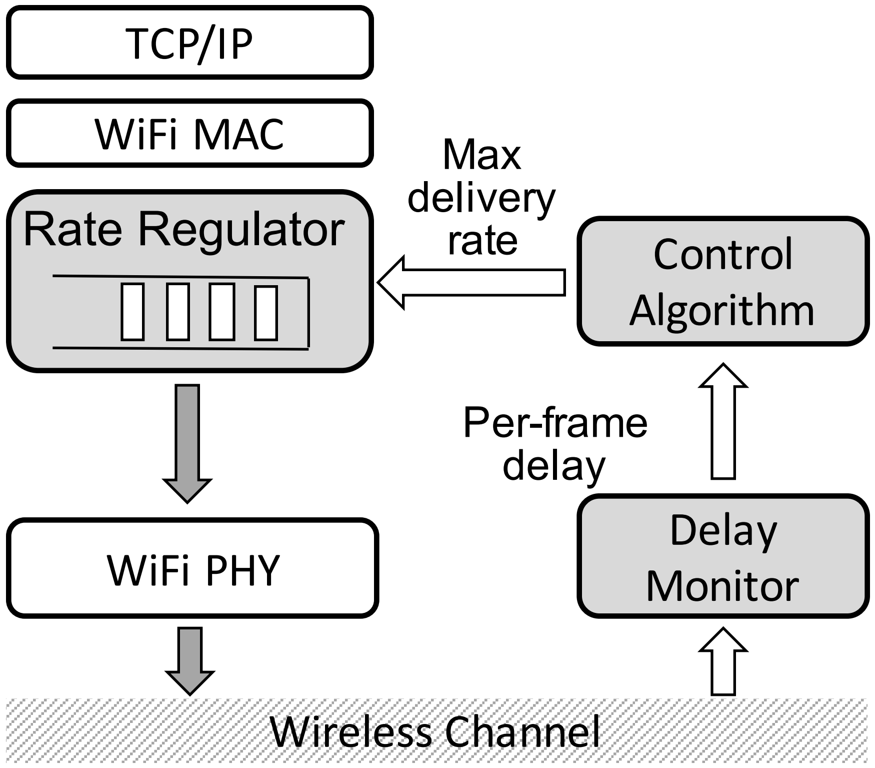
Implementation

Evaluation

QAir Architecture

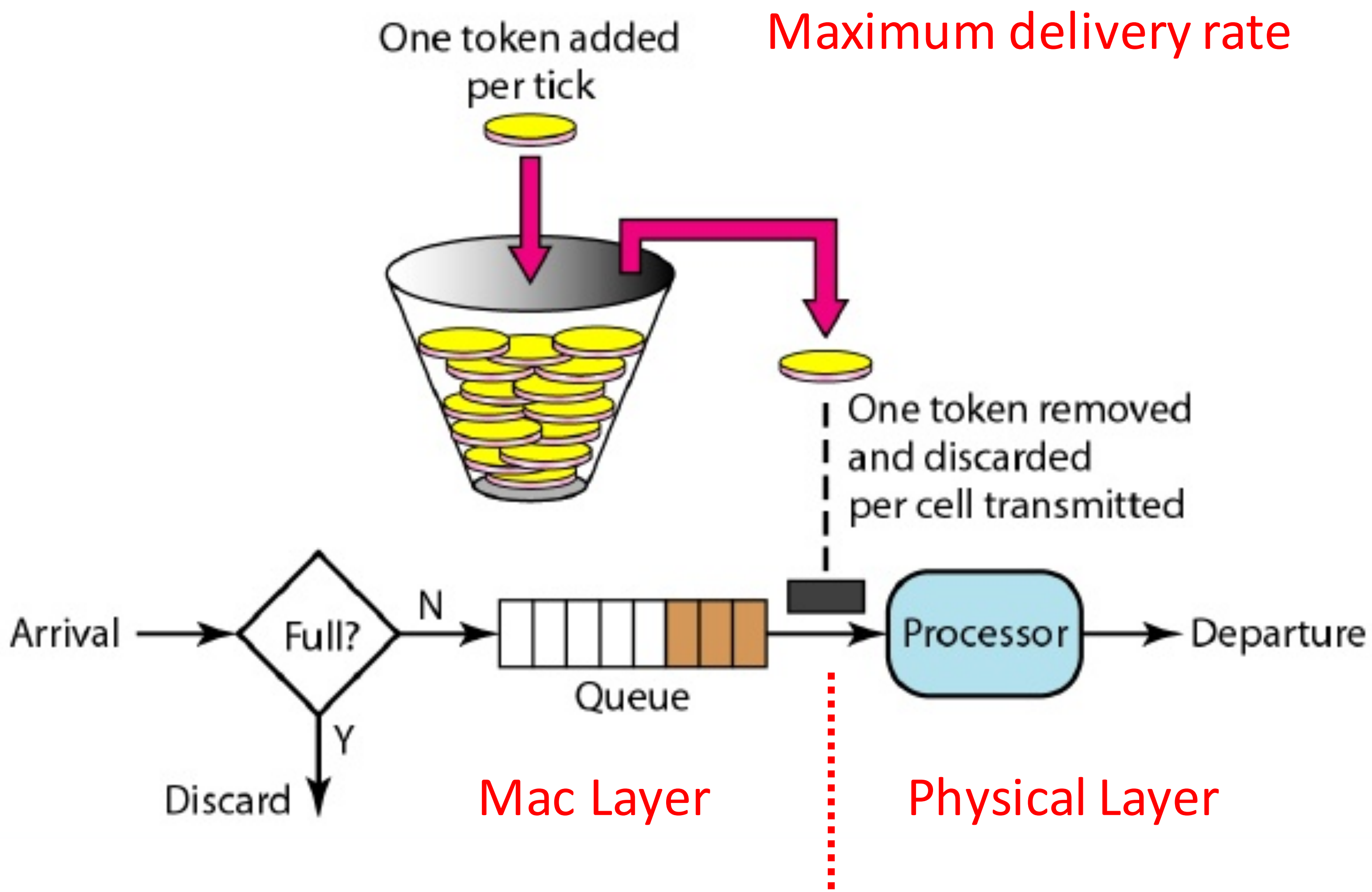


Rate Regulator

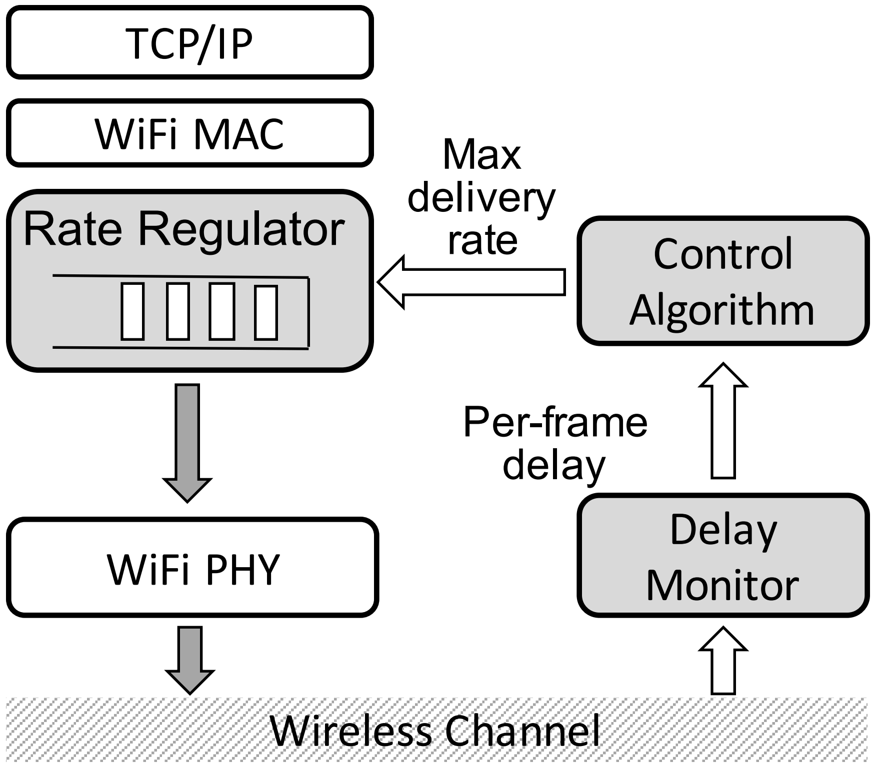


- Input: max delivery rate
- Target: control the delivery rate to the WiFi PHY
- Token bucket

Token bucket



Control Algorithm



- Input: per-frame delay
- Output: calculate a max delivery rate to each flow equally
- Latency based congestion control

Control Algorithm

Algorithm 1 Decide maximum delivery rate.

t_k : measured delay of frame k.

T_{target} : the delay constraint.

R_{tc} : *maximum delivery rate.*

procedure F(t_k)

  $sum \leftarrow sum + t_k$

  $ntrans \leftarrow ntrans + 1$

  if $(now\_time - last\_time) > checkInterval$ **then**

    $\hat{T} \leftarrow \alpha * \hat{T} + (1 - \alpha) * \frac{sum}{ntrans}$

    $last\_time \leftarrow now\_time$

    $sum \leftarrow 0$

    $ntrans \leftarrow 0$

    if $\hat{T} < T_{target}$ **then**

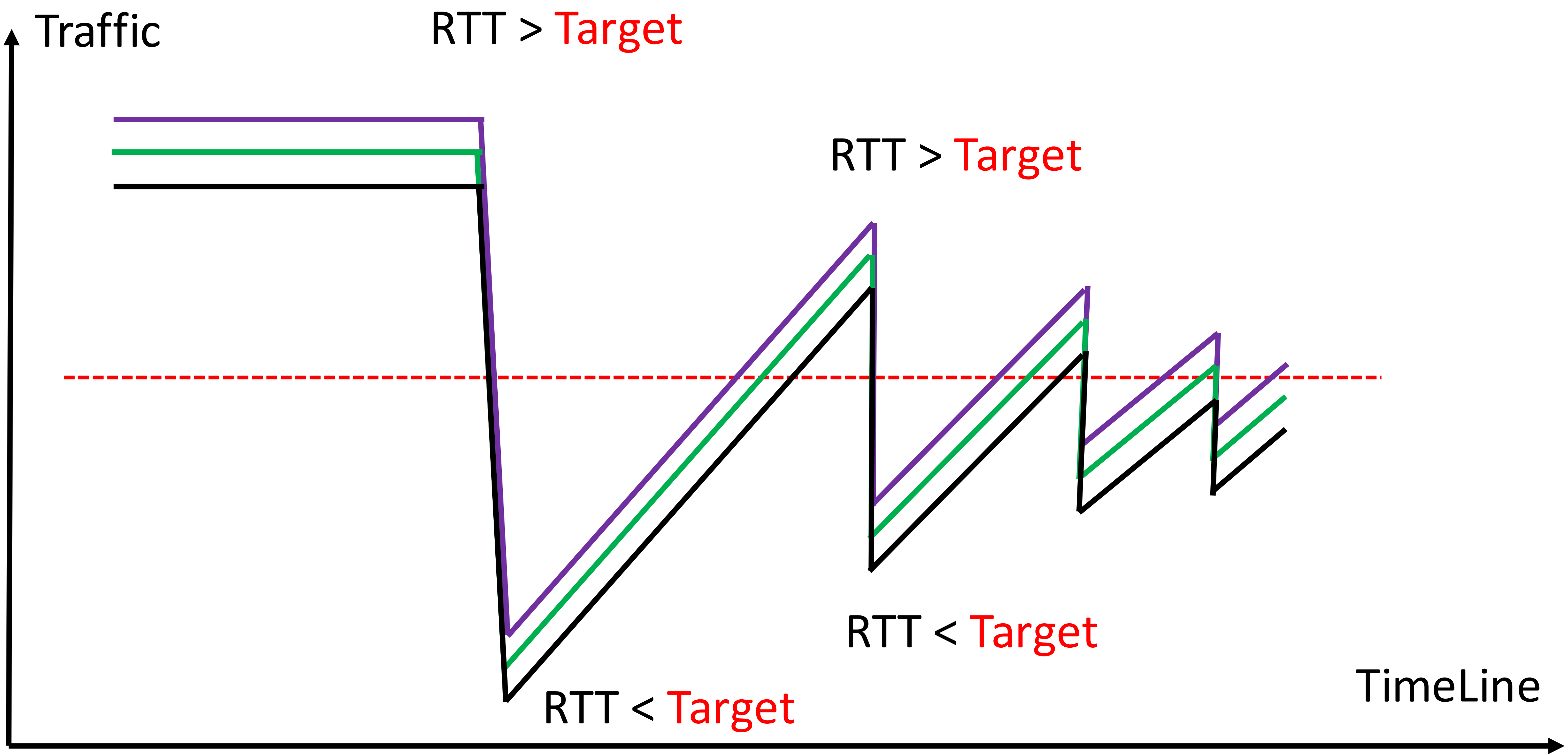
      $R_{tc} \leftarrow minimum(R_{tc} + \delta, R_{max})$

    else

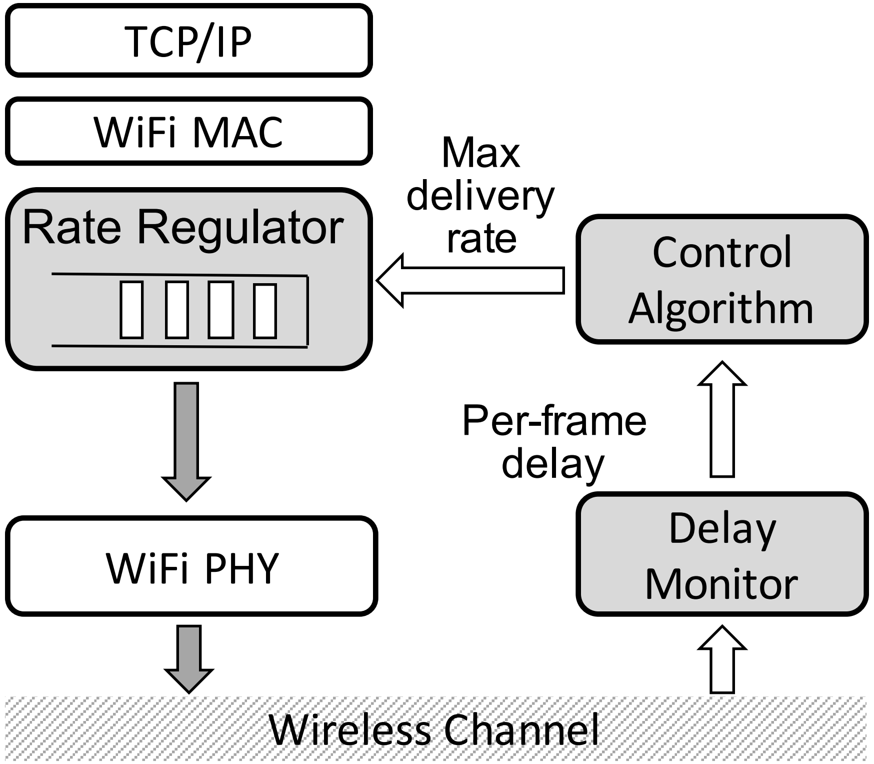
      $R_{tc} \leftarrow maximum(\frac{T_{target}}{\hat{T}_i} * R_{tc}, R_{min})$

  return R_{tc}

Control path of QAir

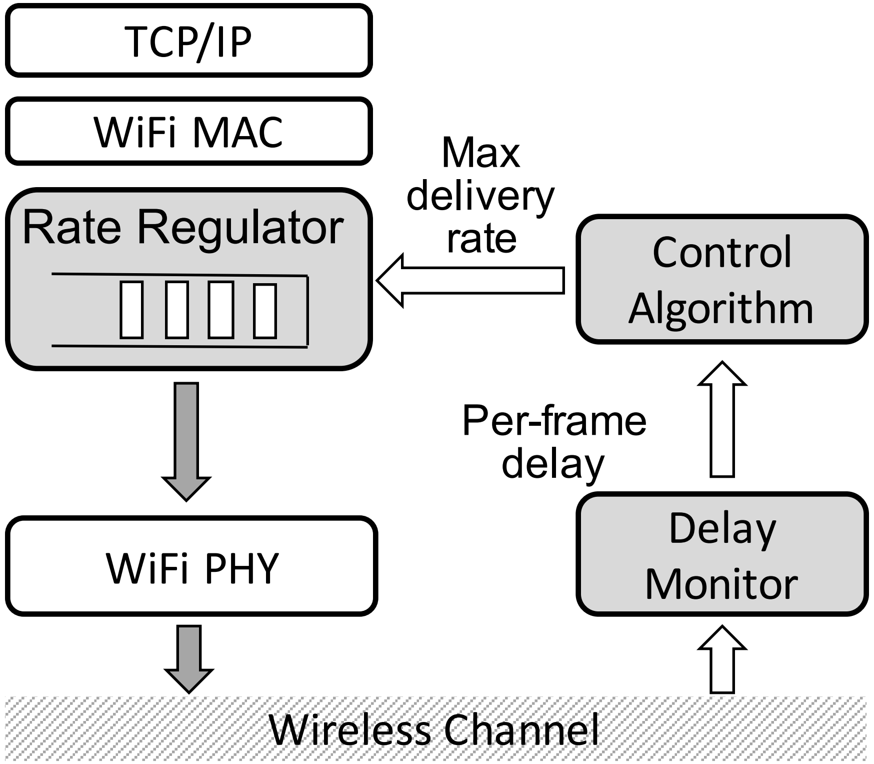


Delay Monitor



- Input: real traffic
- Output: per-frame latency
- Tradeoff:
  - Eliminate the variance
  - Reflect the level of contention

Delay Monitor



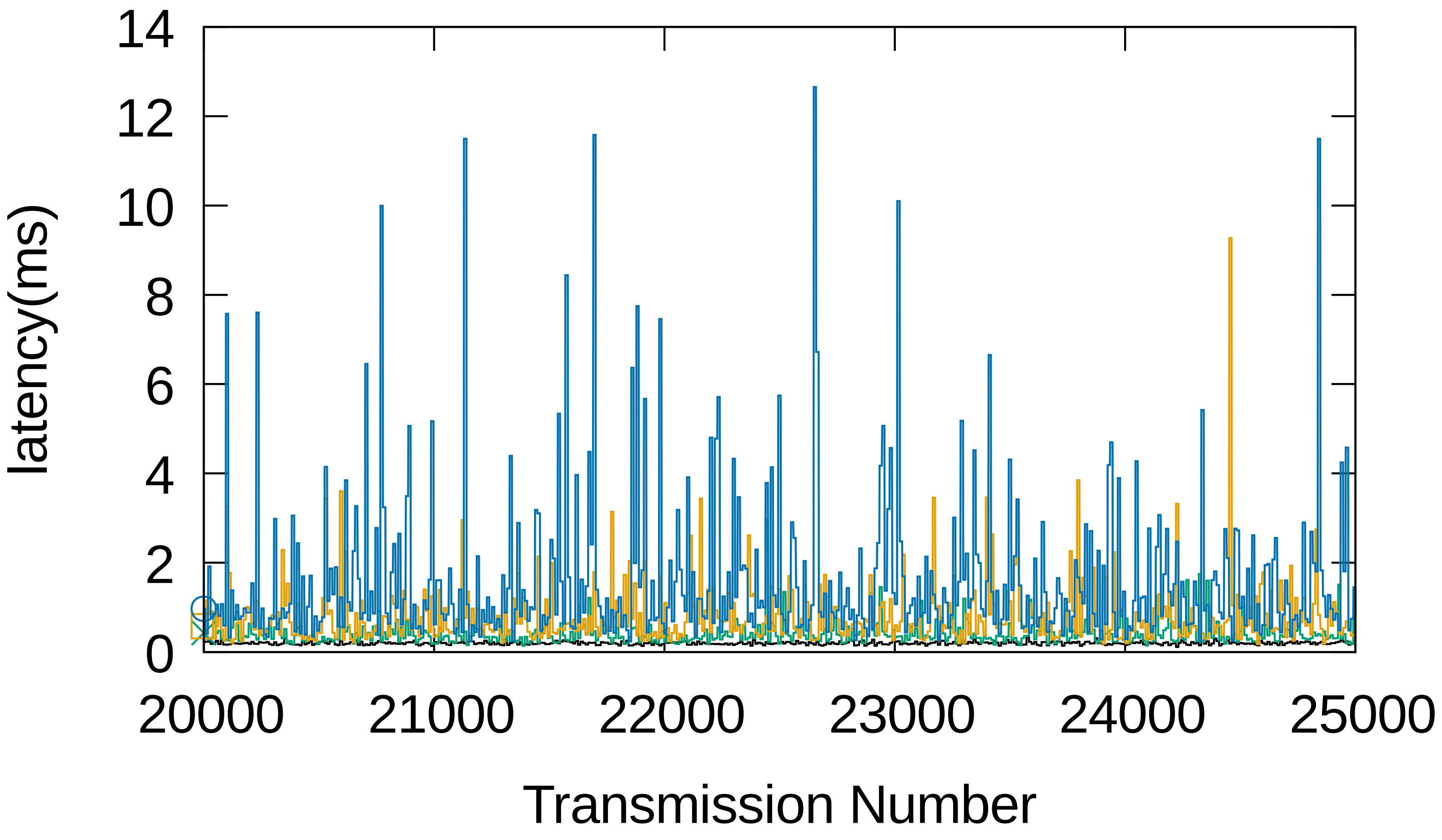
Arithmetic average

$$T_i = \frac{1}{N}\sum_{k=1}^{N} t_k$$

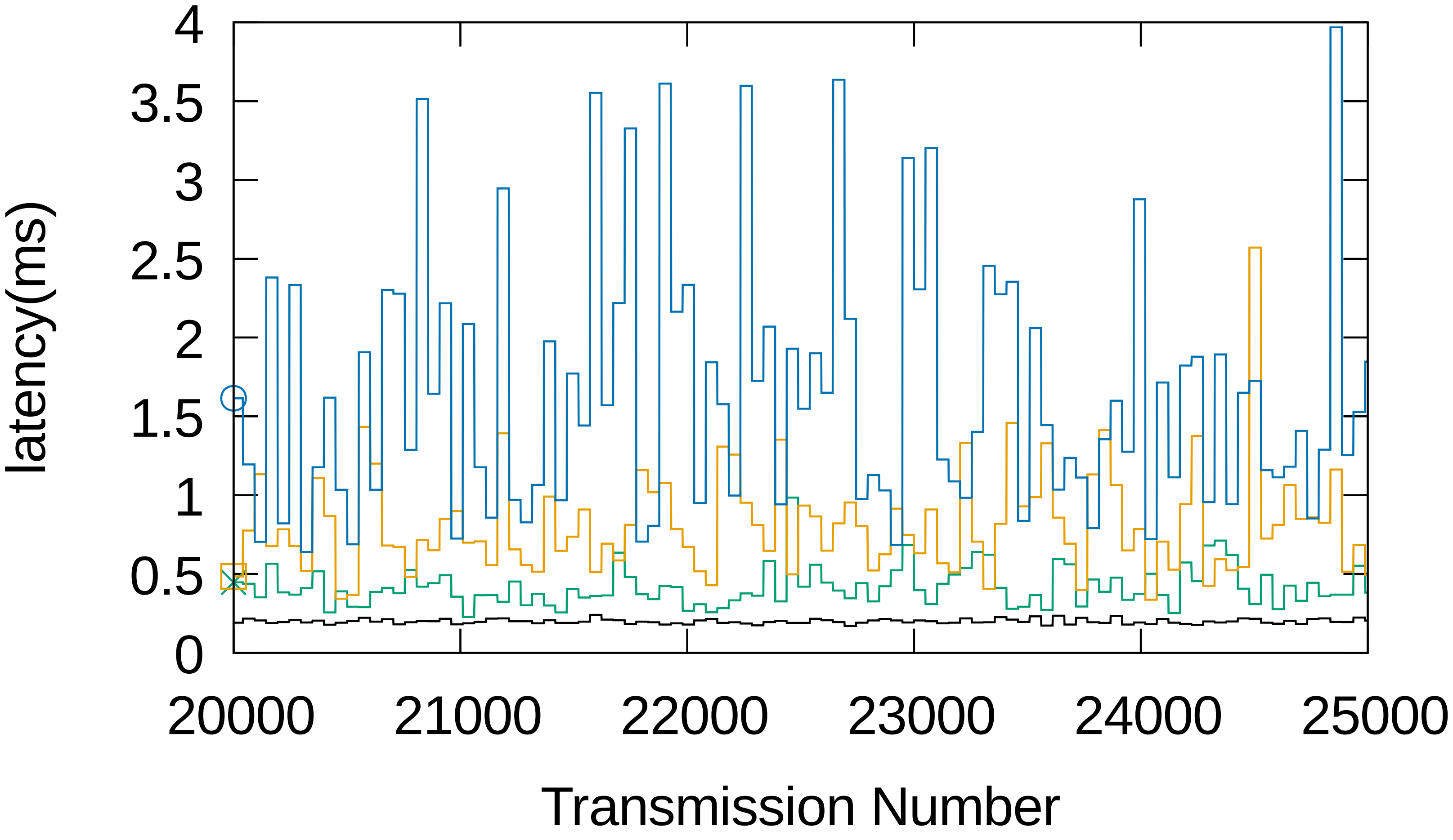
Moving average between different slots

$$\hat{T}(new) = \alpha * \hat{T}(old) + (1-\alpha) * T_i$$

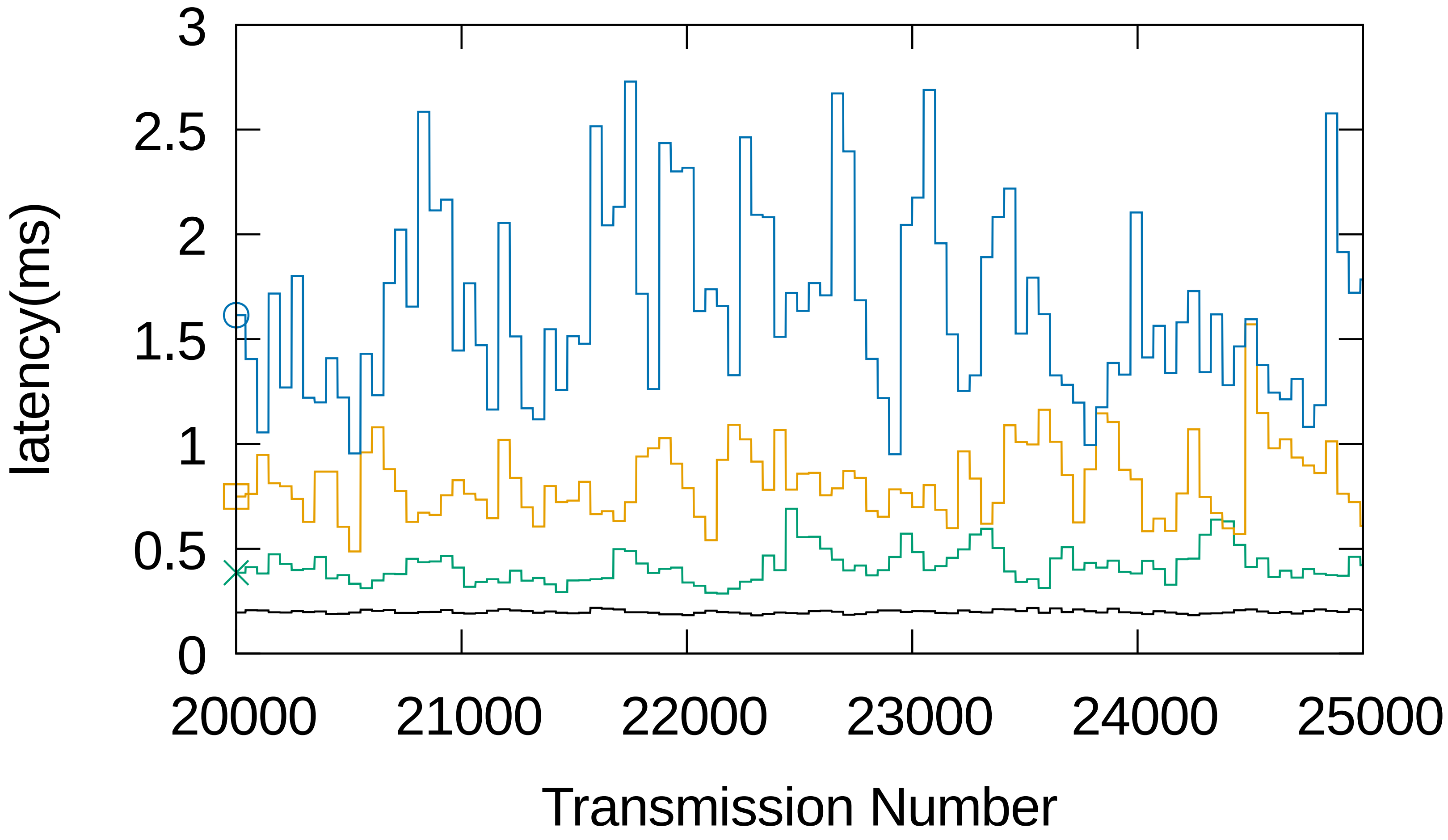
Delay Monitor-Raw Data



Delay Monitor-Arithmetic average



Delay Monitor-Moving average



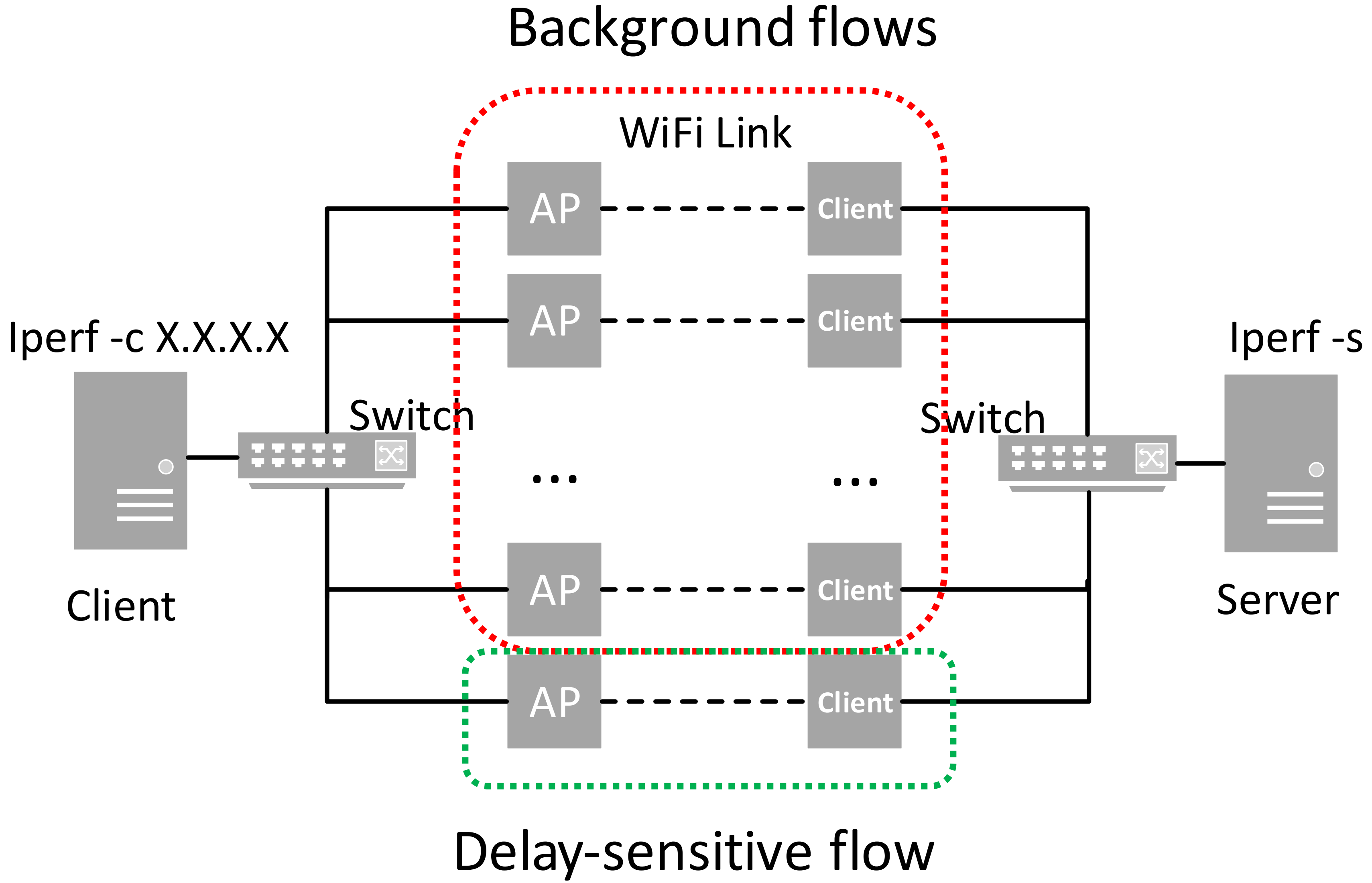
Motivation

Algorithm

Implementation

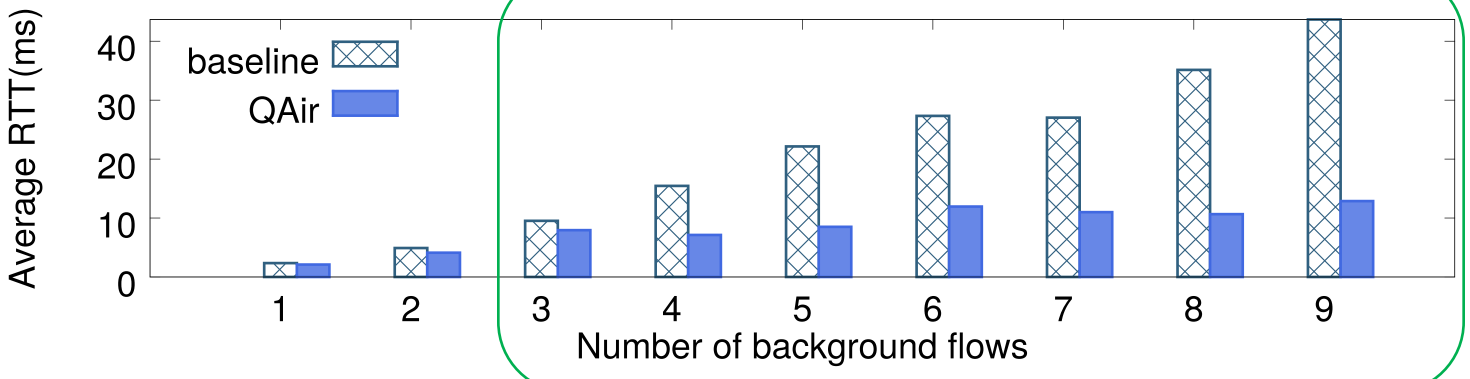
Evaluation

Testbed


Background flows
WiFi Link
AP
Client
AP
Client
Iperf -c X.X.X.X
Iperf -s
Switch
Switch
...
...
AP
Client
Client
Server
AP
Client
Delay-sensitive flow

UDP Traffic

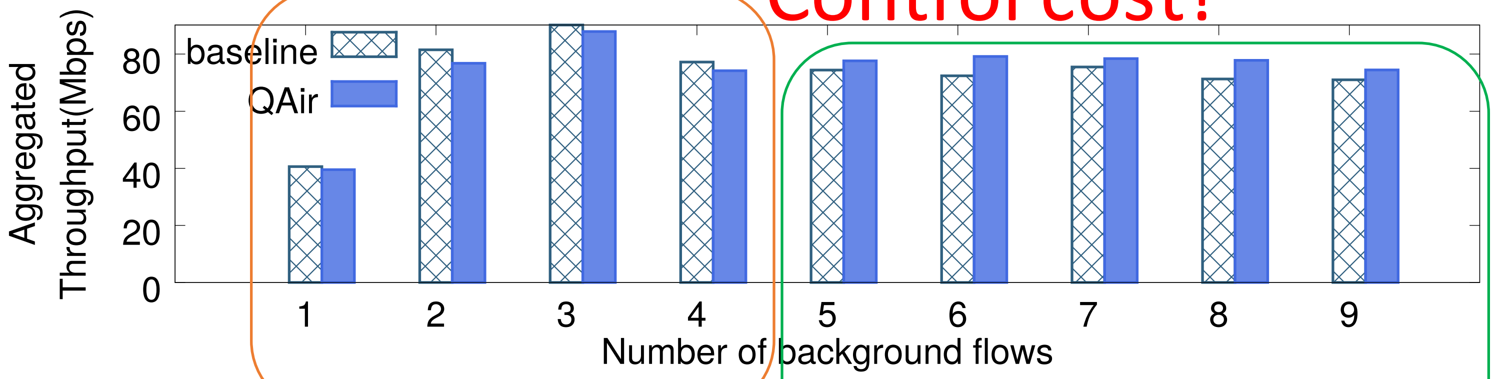
Latency (ms)



Reduce the length of distributed queue!

Control cost!

Throughput (Mbps)

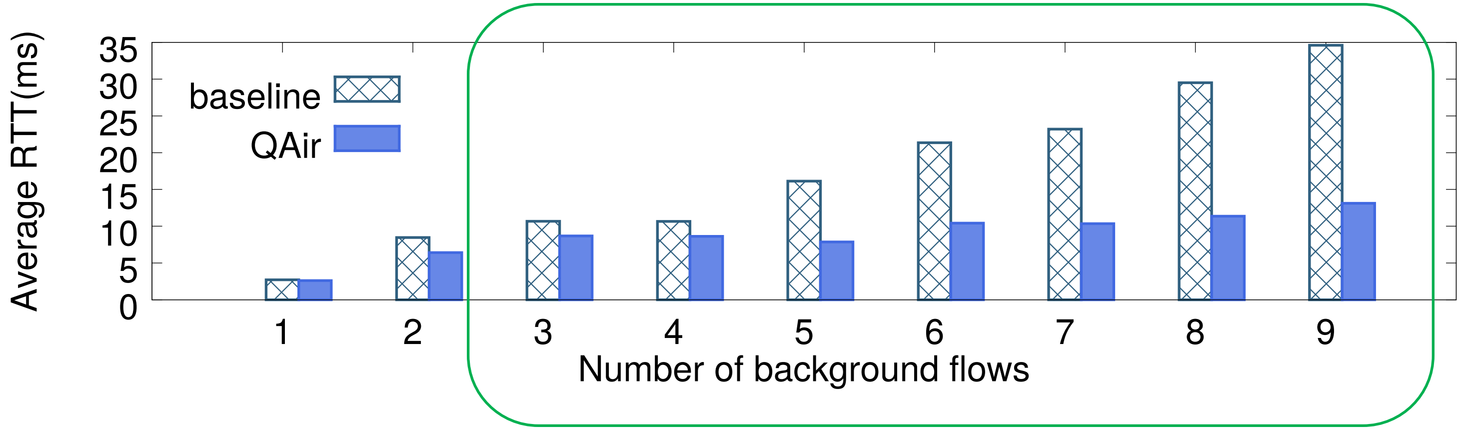


Reduce the Collision rate!

Number of background flows

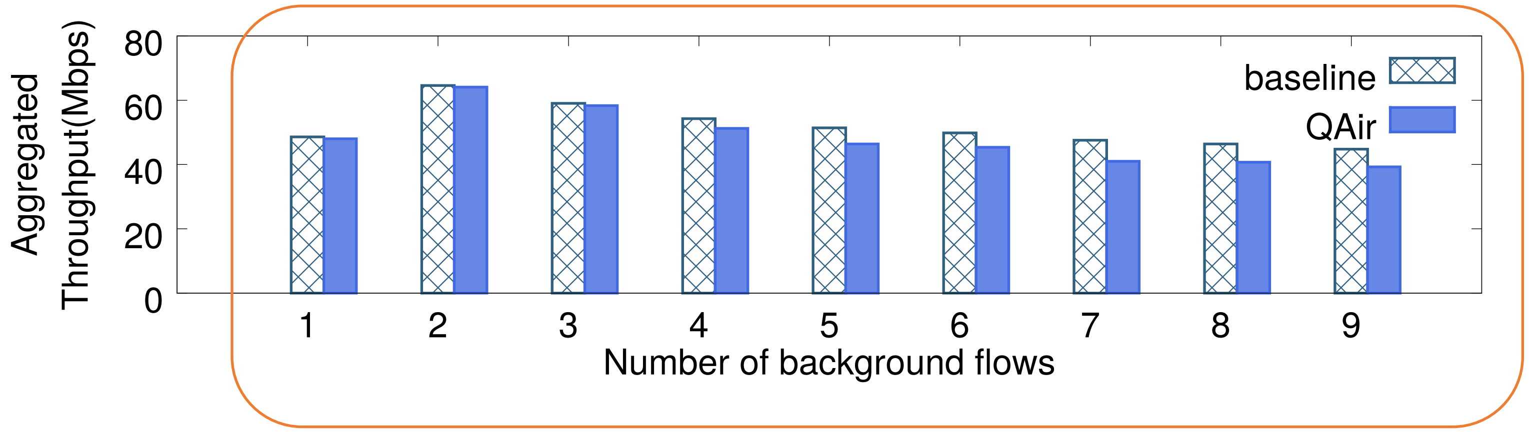
TCP Traffic

Latency (ms)



Coexist with TCP level congestion control.

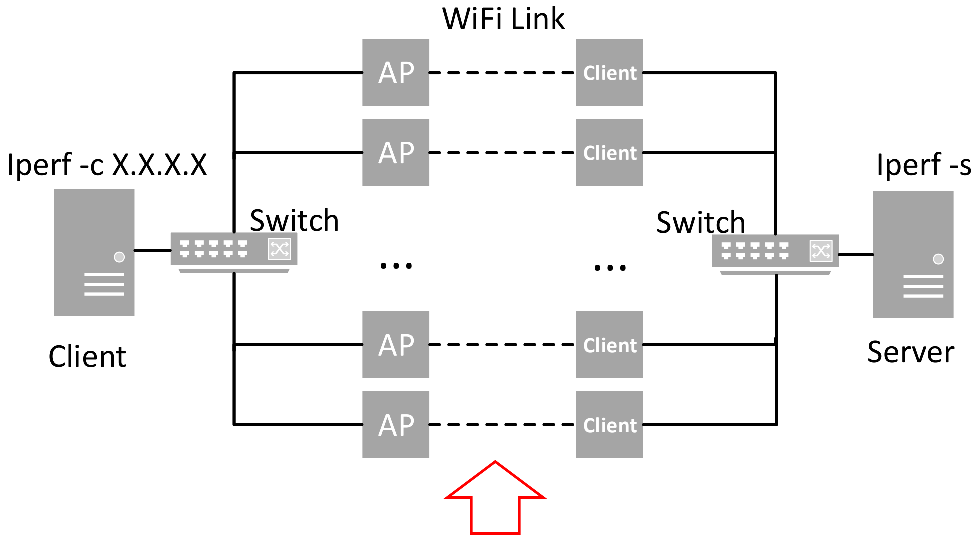
Throughput (Mbps)



Control cost! < 5Mbps

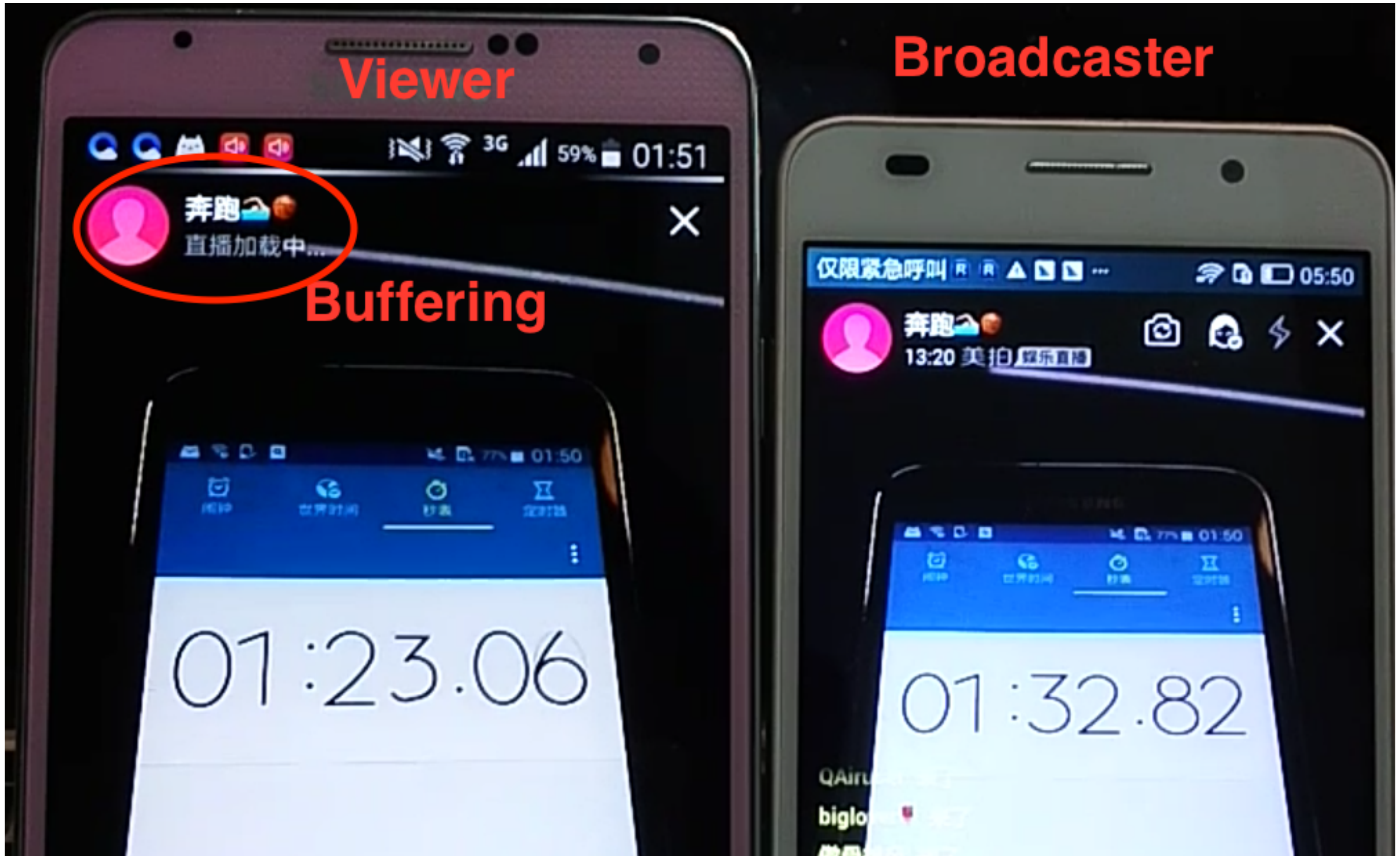
Number of background flows

Personalized Live Streaming



Live Streaming

PING



QoE Metrics

- QAir works well for real applications whose traffic demand range from 1Mbps~8Mbps.

TABLE III: QoE of live streaming with/without QAir.

	#Buffering events	Minimum e2e Delay (s)	Maximum e2e Delay (s)
QAir disable	7	2.15	9.76
QAir enable	0	2.15	2.23

Conclusion

- Propose a practical solution to control the contention level of the WiFi channel
- Assign implicit priority using the traffic volume of different flows.
- Through real experiments, QAir can significantly reduce WiFi latency of delay-sensitive flows without sacrificing the network throughput.

IWQoS

2017

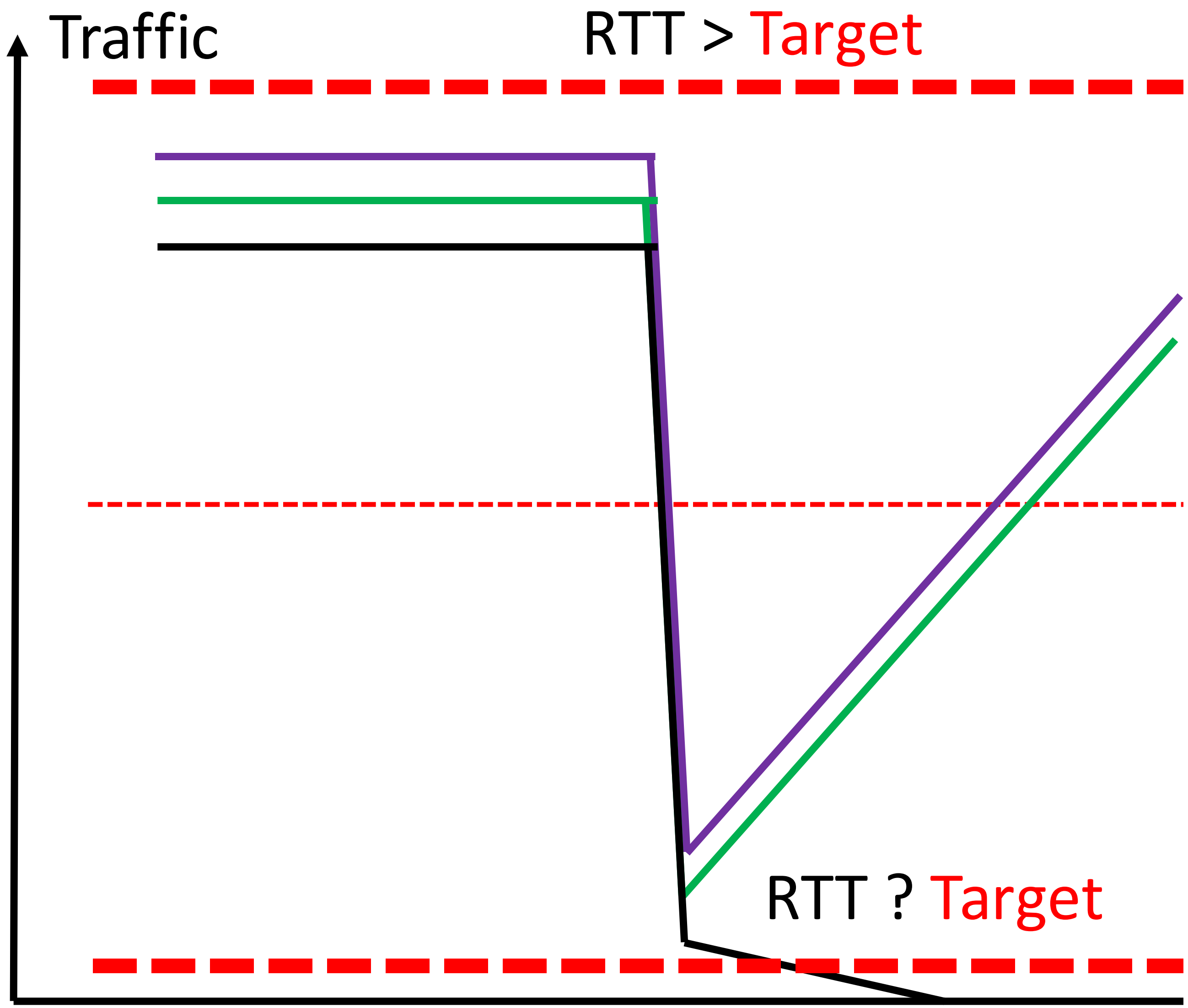
Thank you! Q&A?

changhuapei@gmail.com

Backup

IWQOS

Differences between wireless and wired



Algorithm 1 Decide maximum delivery rate.

t_k : measured delay of frame k.
T_{target} : the delay constraint.
R_{tc} : *maximum delivery rate.*

procedure F(t_k)
  $sum \leftarrow sum + t_k$
  $ntrans \leftarrow ntrans + 1$

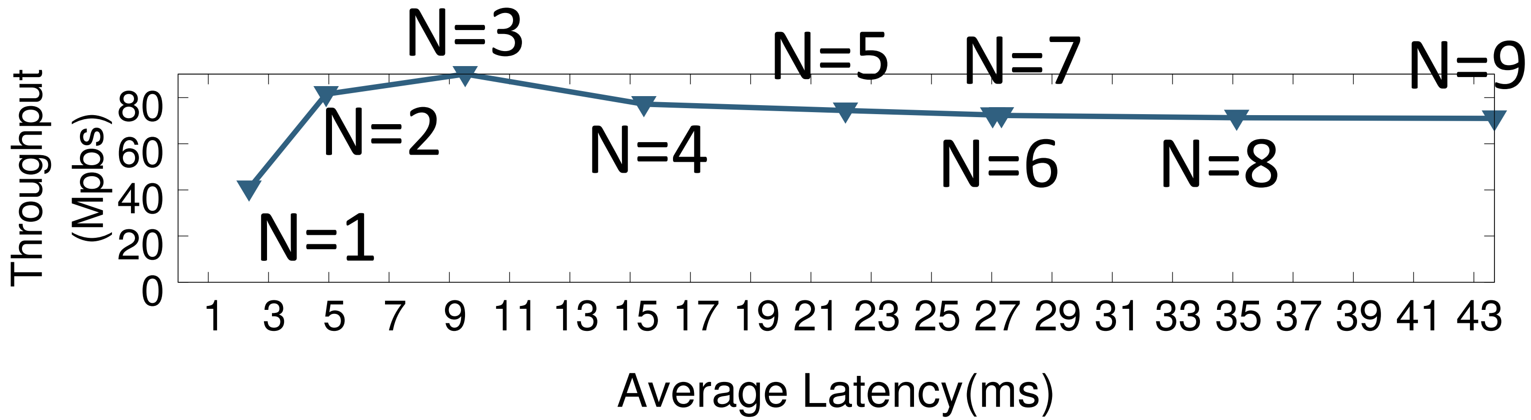
  if $(now\_time - last\_time) > checkInterval$ **then**
    $\hat{T} \leftarrow \alpha * \hat{T} + (1 - \alpha) * \frac{sum}{ntrans}$
    $last\_time \leftarrow now\_time$
    $sum \leftarrow 0$
    $ntrans \leftarrow 0$

    if $\hat{T} < T_{target}$ **then**
      $R_{tc} \leftarrow minimum(R_{tc} + \delta, R_{max})$
    else
      $R_{tc} \leftarrow maximum(\frac{T_{target}}{\hat{T}_i} * R_{tc}, R_{min})$

  return R_{tc}

Control Target: number of contenders

- Trade off
  - Higher target: can tolerate more concurrent flows but higher contention
  - Smaller target: may sacrifice the throughput



The optimal point: N=3