PI^2: A Linearized AQM for both Classic and Scalable TCP

Koen De Schepper
Olga Bondarenko
Ing-Jyh Tsang
Bob Briscoe

NOKIA Bell Labs
[simula . research laboratory]

Use DCTCP on the Internet ?

- ☺ consistently low queuing delay
- ☺ full link utilization with very small queues
- ☺ very low loss
- ☺ more stable throughput between competing flows
- ☺ scalable with higher link rates
- ☺ available in Windows 10 and Linux 3.18
- 😐 not yet optimal (for high RTTs, …)

Can we use DCTCP on the Internet ?

Unfortunately (currently) not:

☹ starves the classic TCP-friendly flows

☹ keeps big tail drop queues full

☹ needs ECN, so high loss (or fallback to Reno)

😐 only used where everything can change at once

→ gradual & safe migration strategy

Challenges

How to:

make DCTCP and TCP-Reno rate compatible

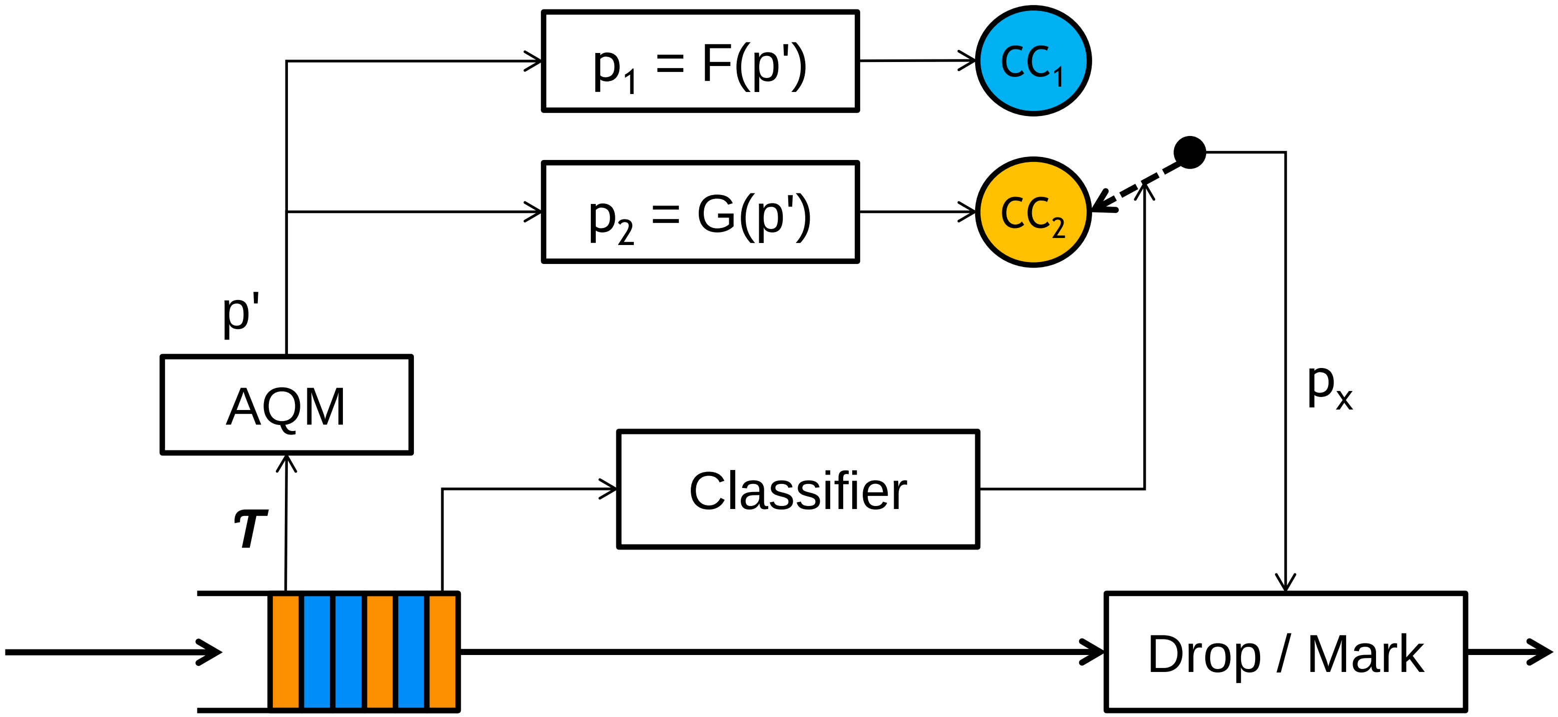
→ this paper: PI² AQM

preserve low latency for DCTCP

→ next papers: DualQ-PI² & TCP-Prague

→ IETF: L4S BoF successful, drafts in tsvwg

AQM for multiple congestion controls



F, G adapt the probability to align the
steady state congestion window of both CCs

Equal congestion window

Classic: Reno

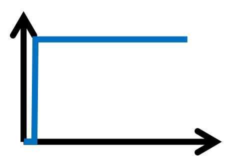
$$w_{reno} = \frac{1.22}{\sqrt{p_{reno}}}$$

Cubic

$$w_{cubic-reno} = \frac{1.68}{\sqrt{p_{cubic-reno}}}$$

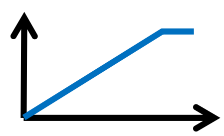
DCTCP – step

on-off marking



$$w_{dco} = \frac{2}{p_{dco}^2}$$

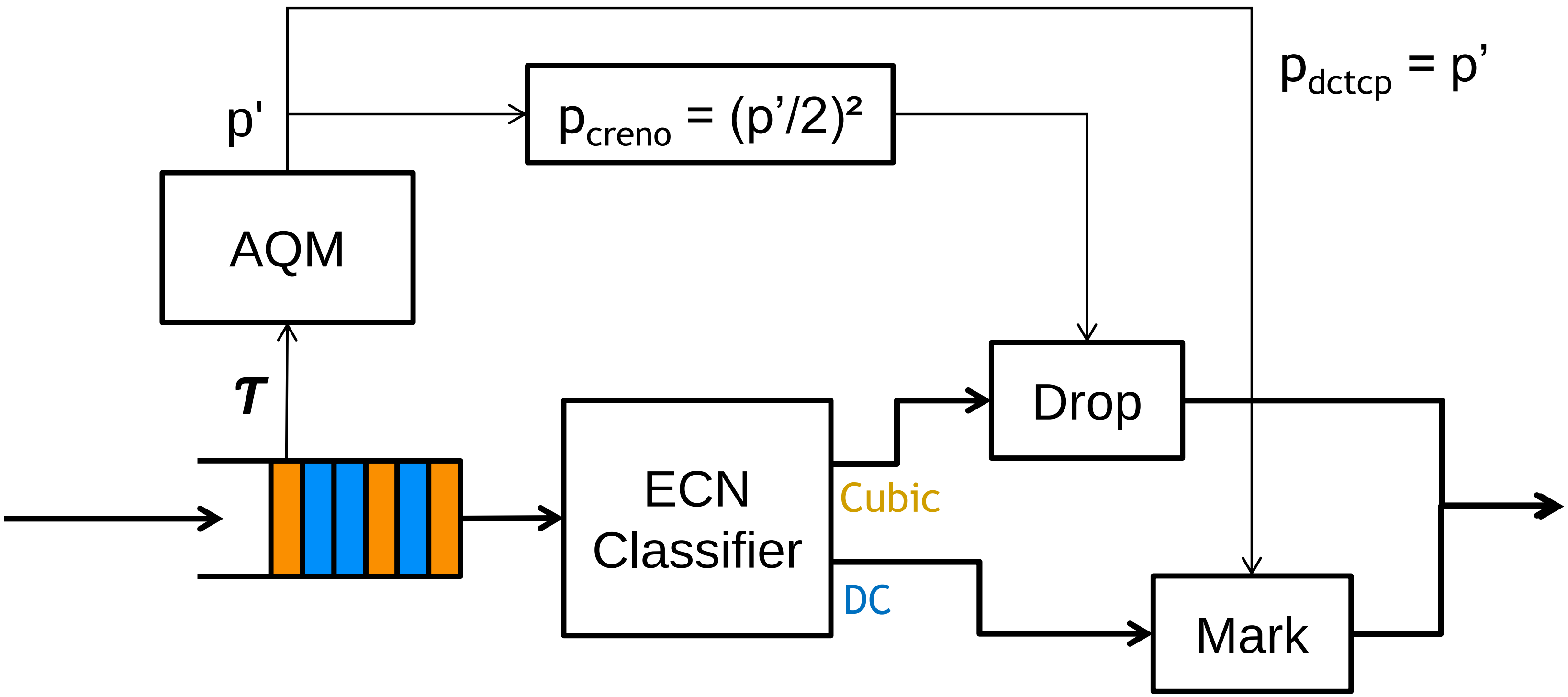
DCTCP – slope



$$w_{dc} = \frac{2}{p_{dc}}$$

$$p_{creno} = \left(\frac{p_{dc}}{2}\right)^2$$

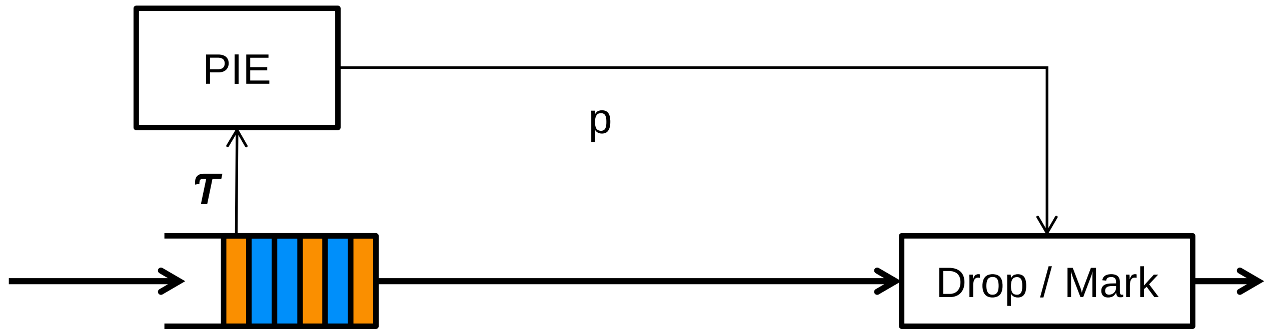
AQM for DCTCP and Cubic



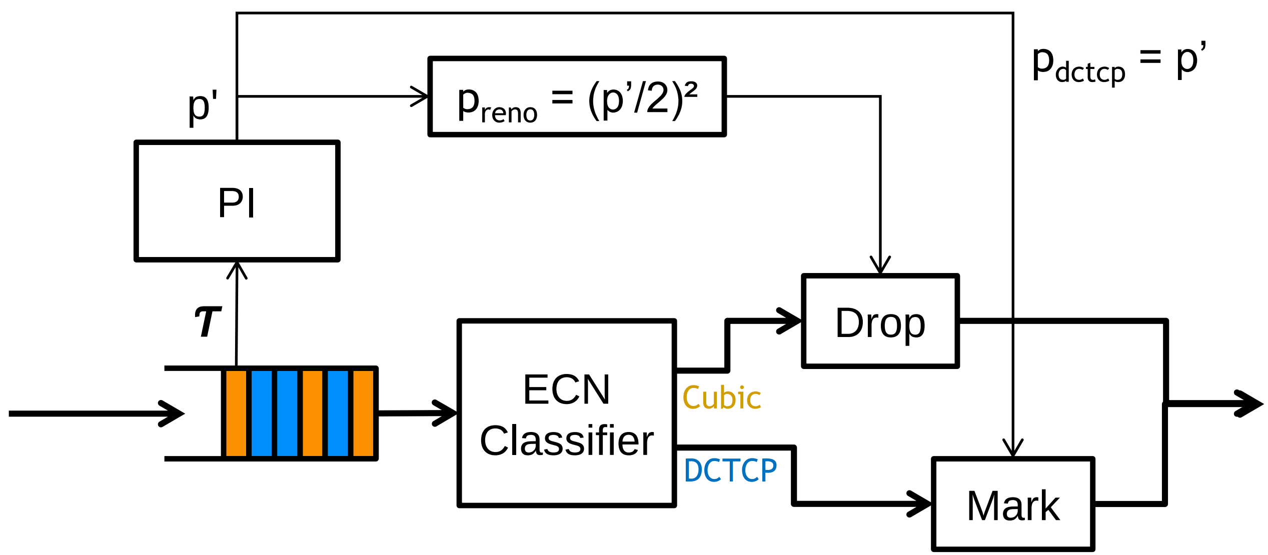
Implemented as a Linux tc qdisc: https://github.com/olgabo/dualpi2
Evaluated on a real testbed

AQMs for steady state test results

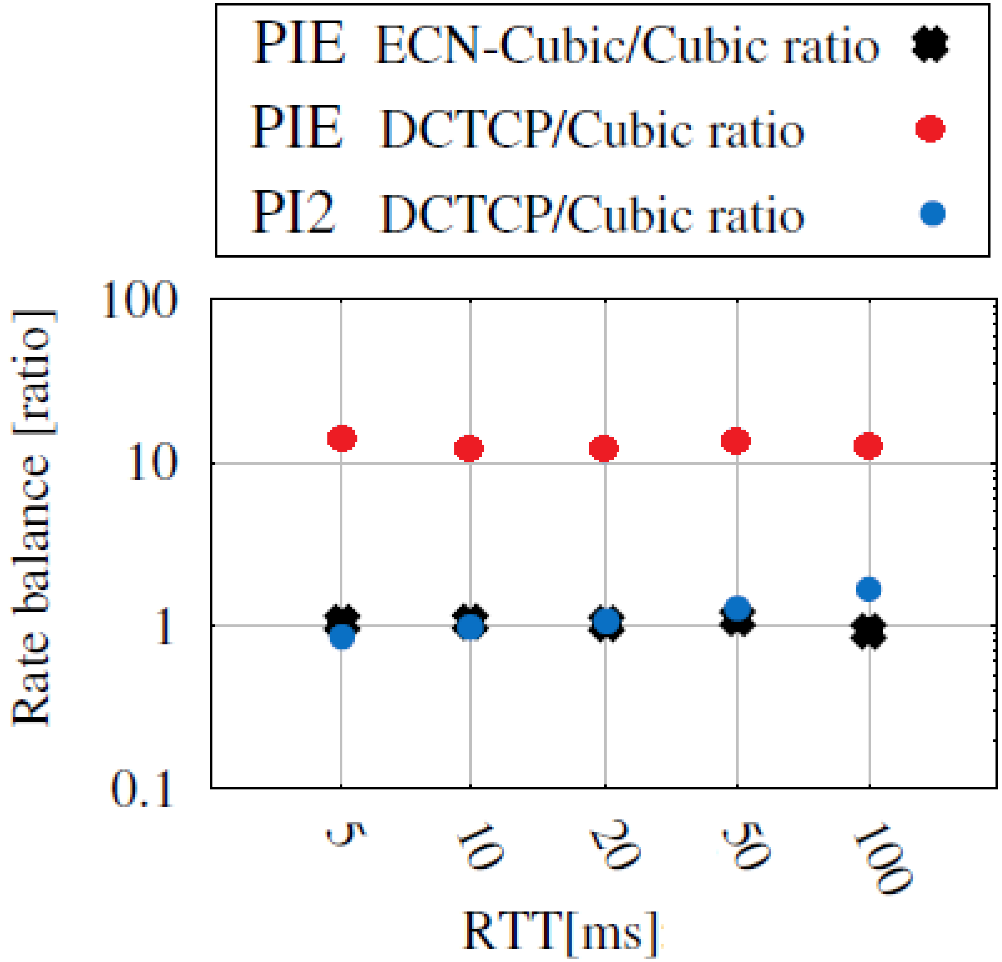
PIE:



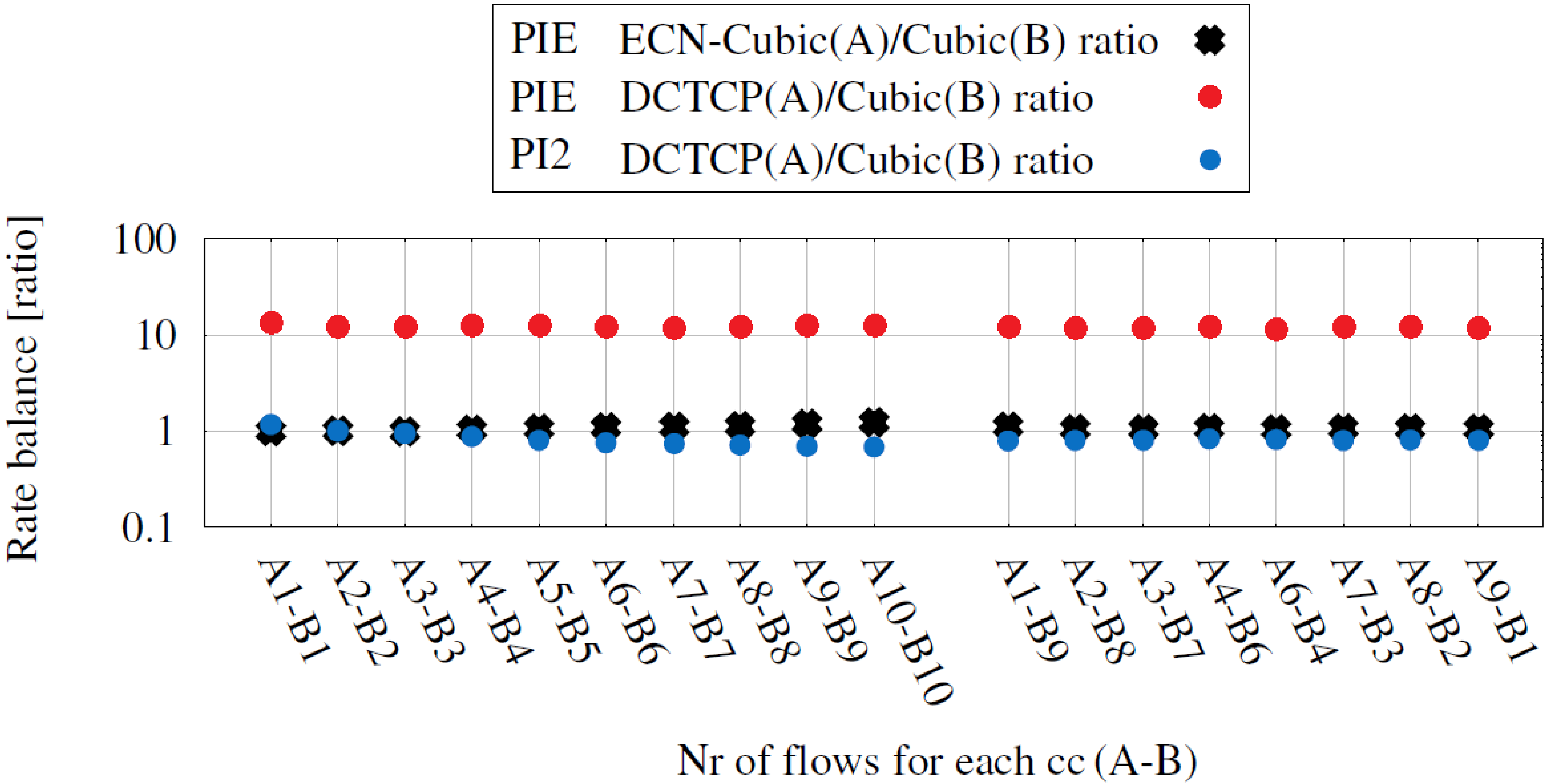
PI²:



Equal rate at different RTTs



Equal rate with different flow nbrs



✓ Equal window for steady state

? Dynamic behavior

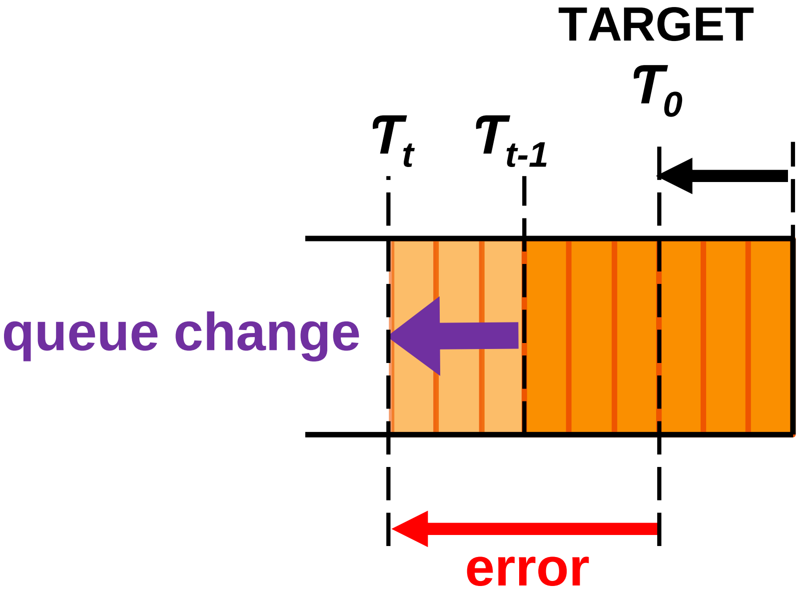
? Stability PI

PI-AQM recap

Every T_{update} interval do:

$\Delta p = \alpha * (\text{error}) + \beta * (\text{queue change})$

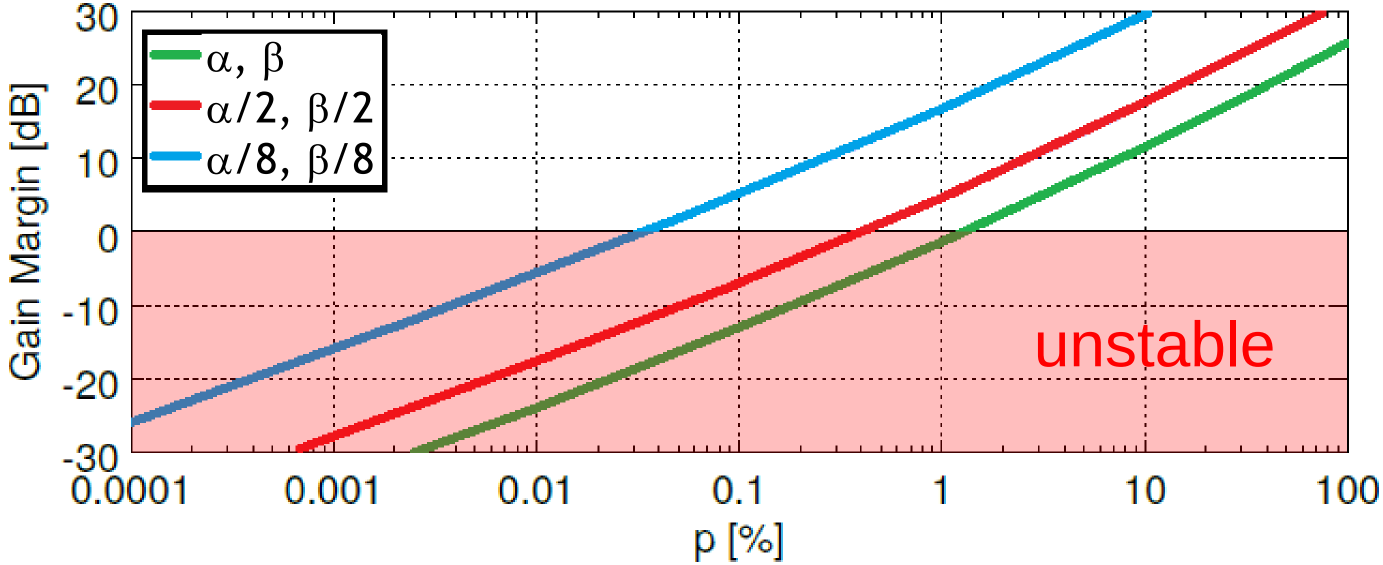
$p = p + \Delta p$



Choosing α and β

Larger α and β values give faster response

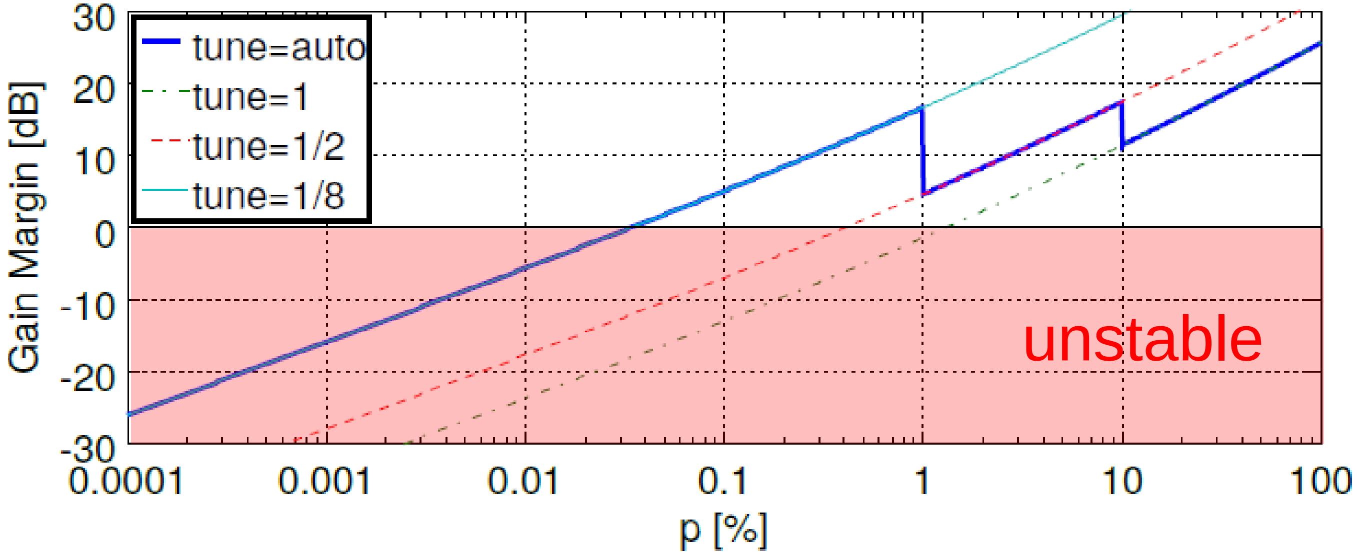
Stability analysis: stable if gain margin > 0



Gain margin evolves diagonally with p → problem!

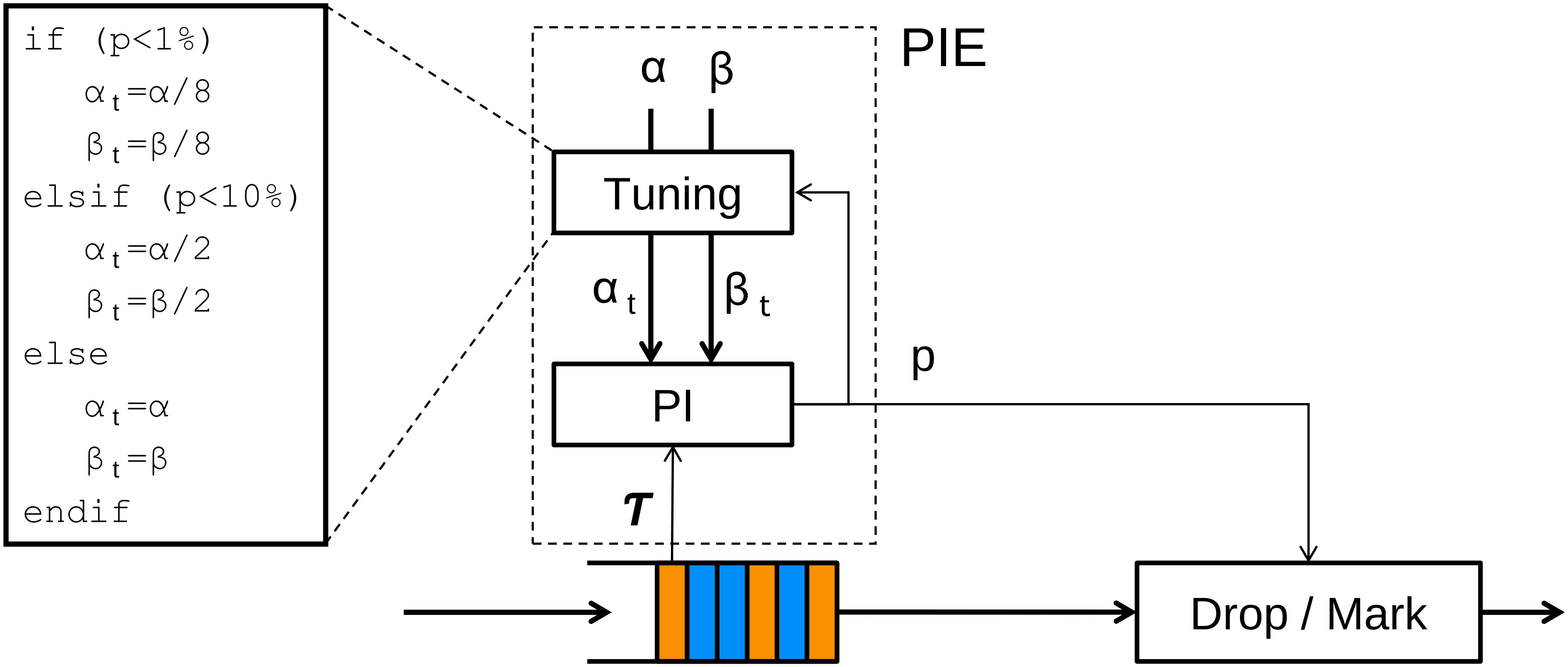
PIE solution: α and β tuning

Adapt (tune) α and β based on p



PIE solution: α and β tuning

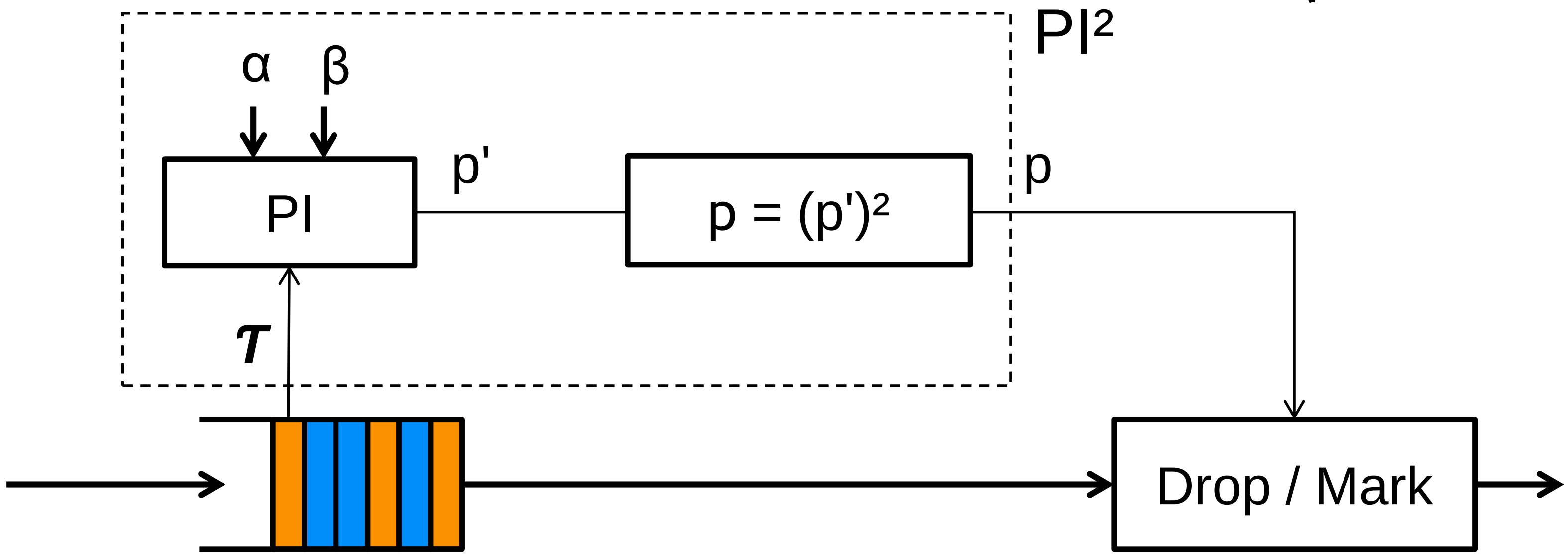
Tune α and β based on previous p:



Works well. Tuning is required improvement ! Curing the symptoms

PI² solution: remove the $\sqrt{}$

Reason diagonals is the $\sqrt{}$ in $w_{reno} = \dfrac{1.22}{\sqrt{p_{reno}}}$



$$w = \frac{1.22}{\sqrt{p}} = \frac{1.22}{\sqrt{(p')^2}} = \frac{1.22}{p'}$$

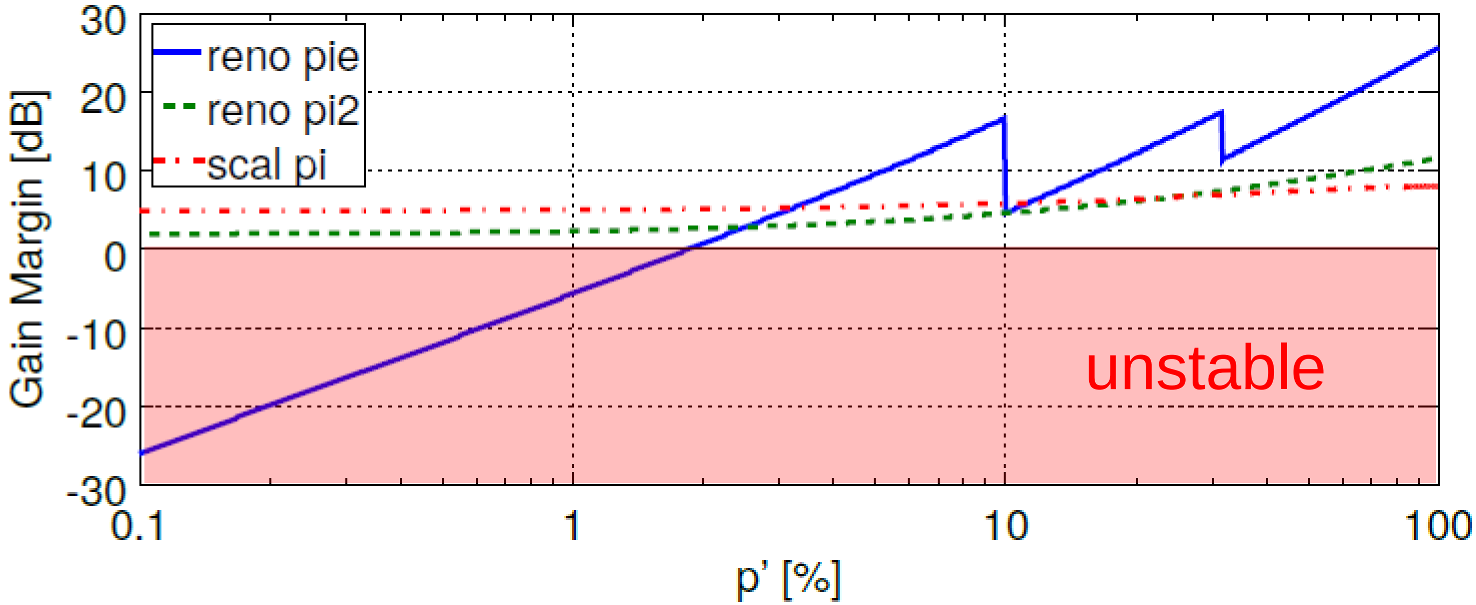
→ makes Reno controllable like a scalable CC

PI² solution: remove the $\sqrt{}$

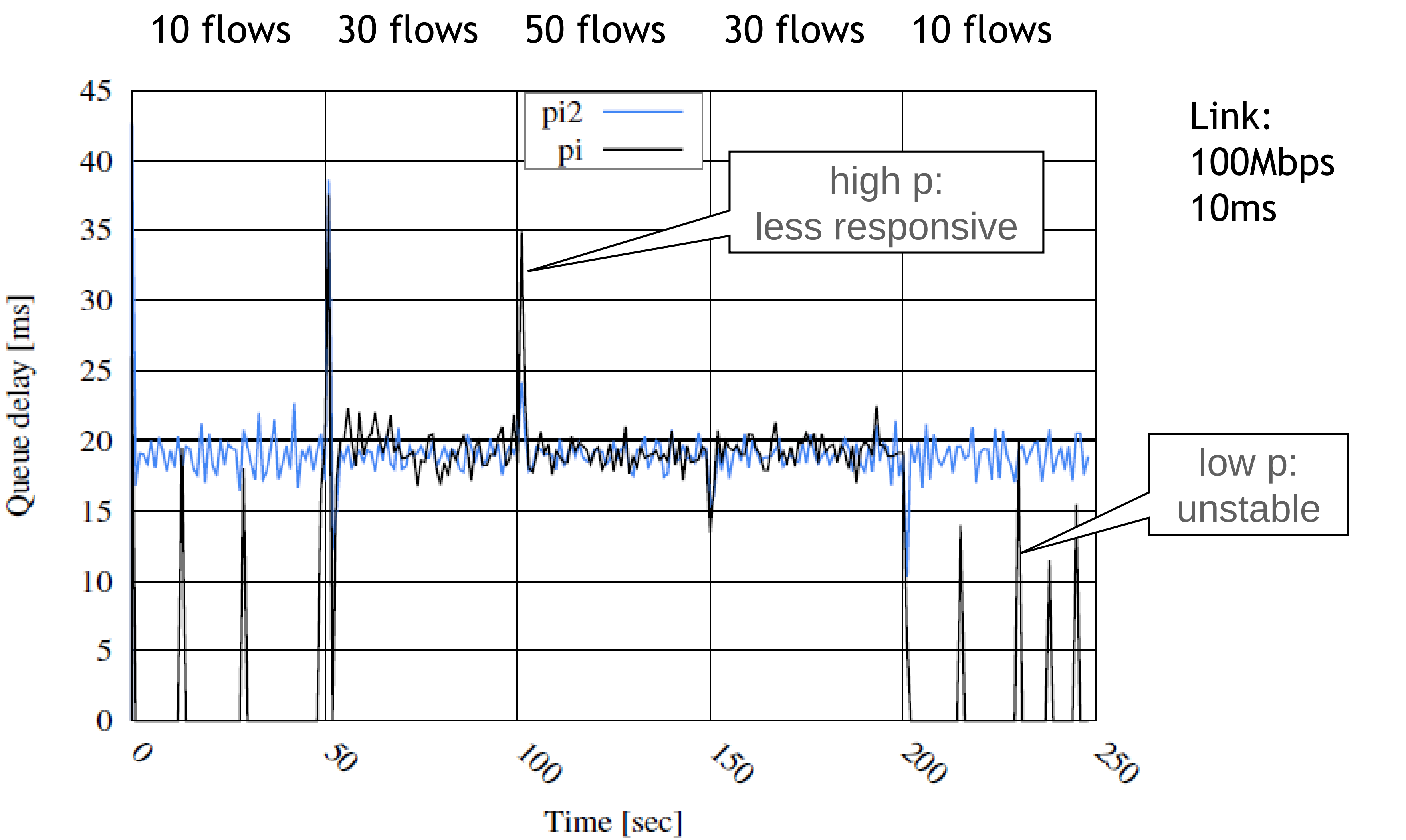
Stability models used for:

TCP Reno on PI: $$\frac{dW(t)}{dt} = \frac{1}{R(t)} - 0.5\frac{W(t)W(t-R(t))}{R(t-R(t))}p(t-R(t))$$

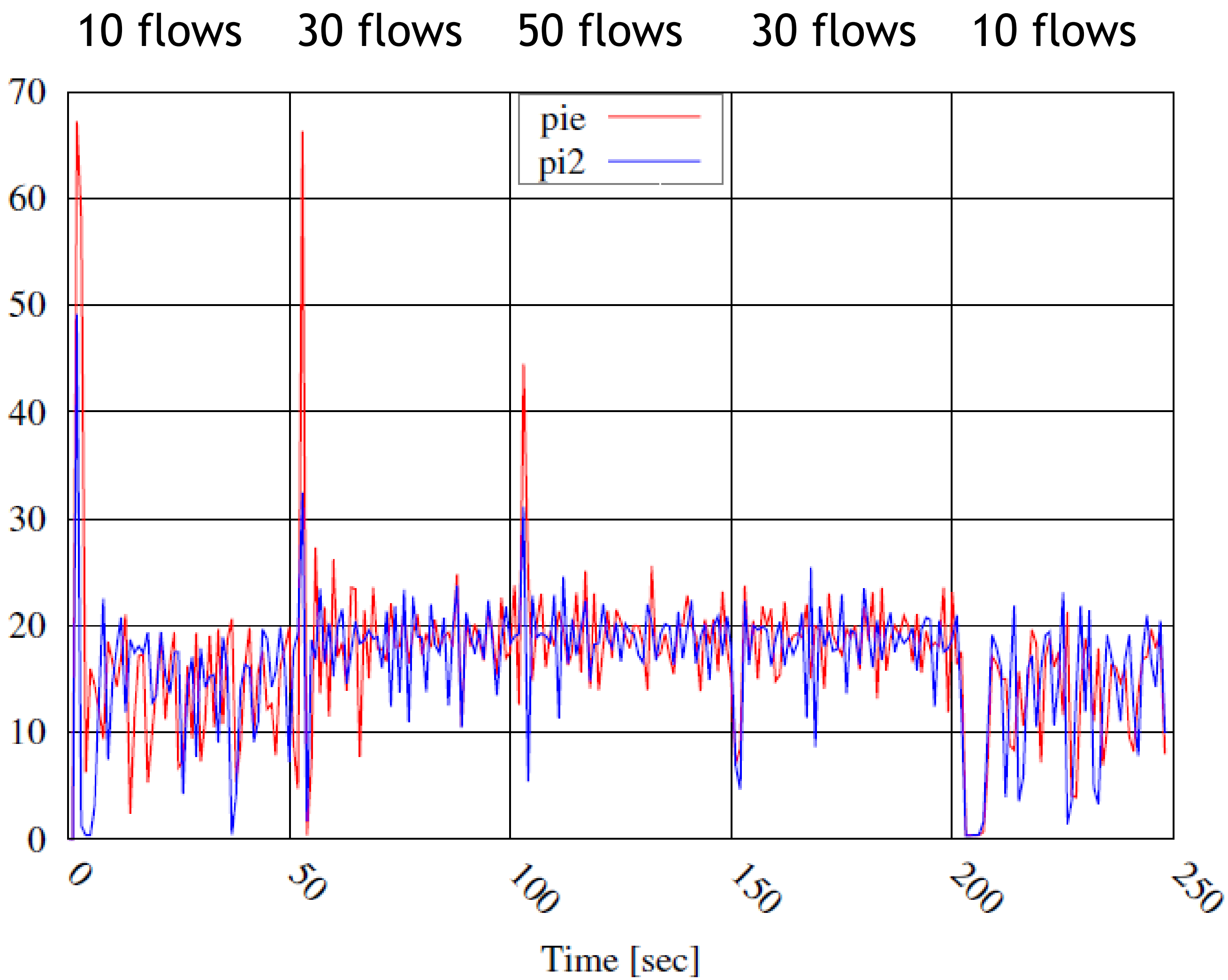
TCP Reno on PI2: $$\frac{dW(t)}{dt} = \frac{1}{R(t)} - 0.5\frac{W(t)W(t-R(t))}{R(t-R(t))}\left(p'(t-R(t))\right)^2$$



Effect on not squaring PI for Reno

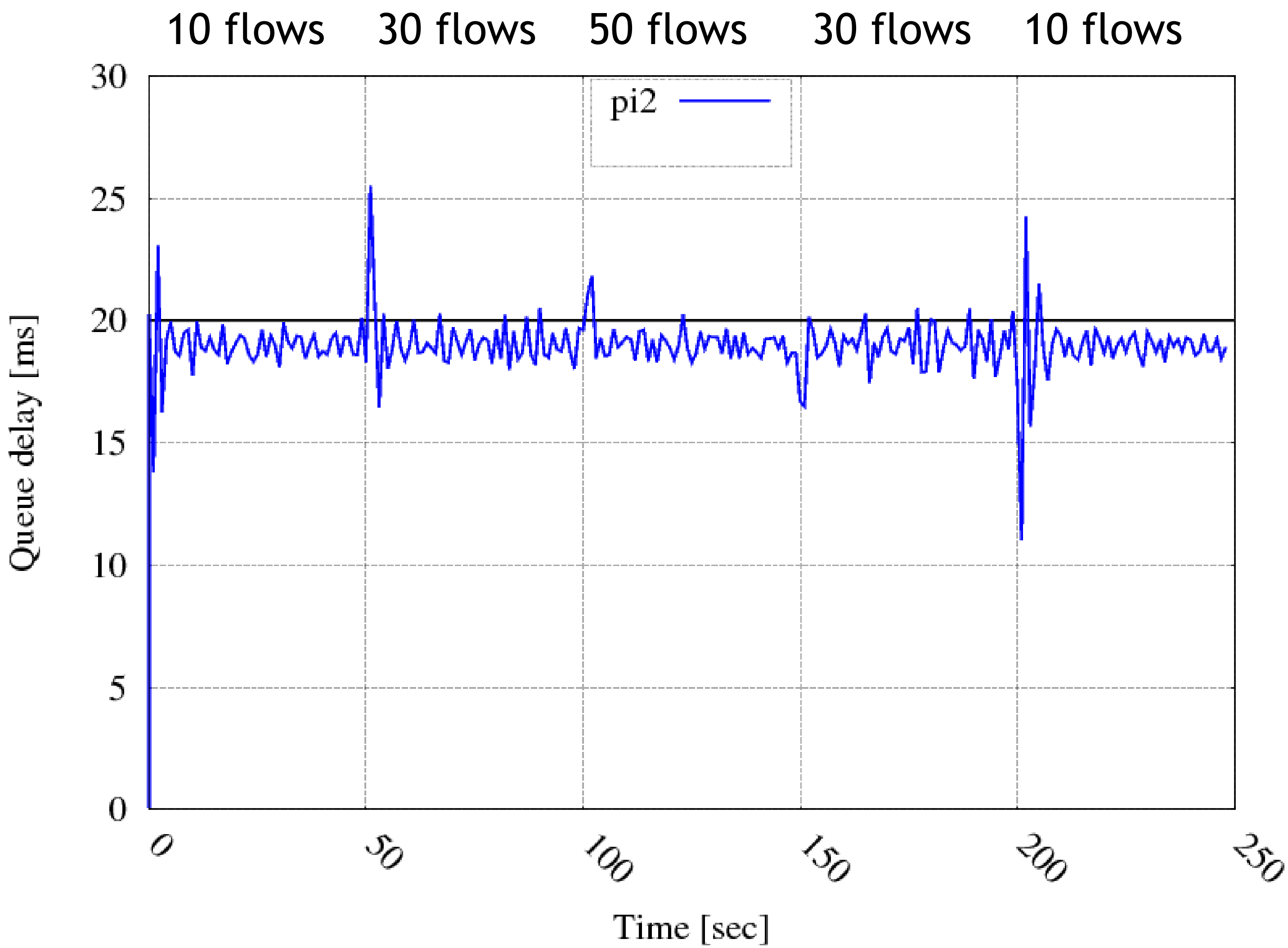


PI² similar to PIE for Reno



Link:
10Mbps
100ms

PI(2) controls DCTCP



Link:
100Mbps
10ms

Conclusions

PI² is simpler than PIE, performs not worse and supports scalable CCs (without the square)

PI controls natively scalable CCs, use adaption function to convert any CC to a scalable

Future work:

Single Q deployment is not recommended for low latency

→ DualQ to preserve low latency

→ TCP-Prague to improve DCTCP for Internet

Questions?

koen.de_schepper@nokia-bell-labs.com

https://github.com/olgabo/dualpi2
http://riteproject.eu/dctth

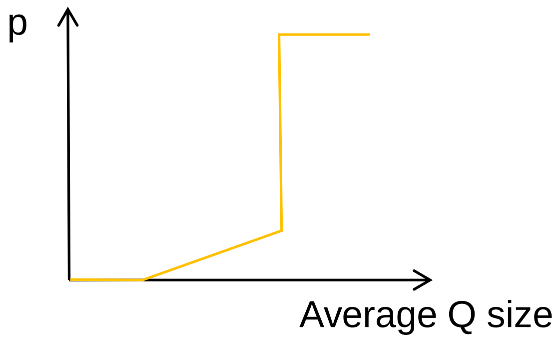
Backup

DCTCP recap

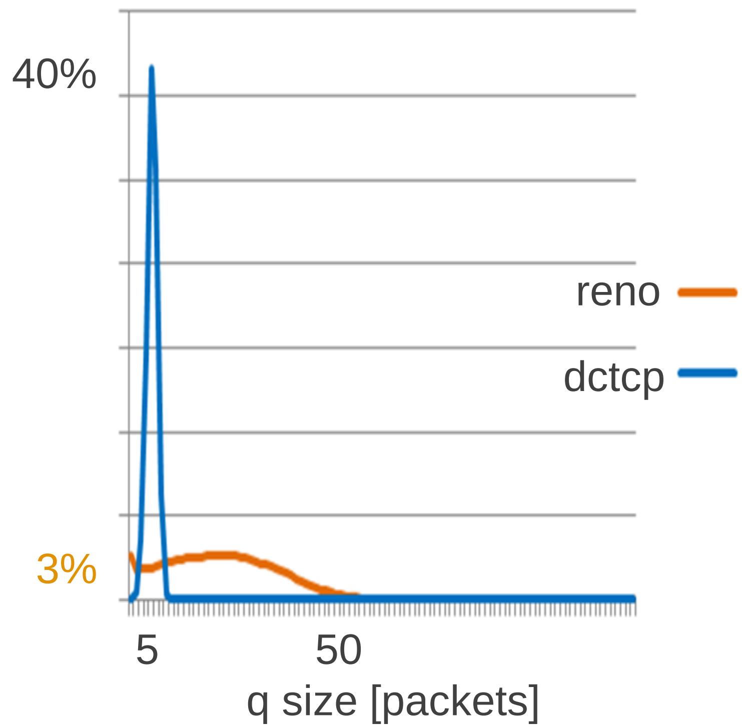
TCP (Reno)	← →	**DCTCP**
	Response to congestion in sender	
Half the congestion window		On average half a packet per ECN mark ➔ React to level of congestion
	ECN feedback in receiver	
Echo once per RTT		Echo every mark / non-mark ➔ accurate ECN feedback
	ECN marking in network	
Smooth and delay a drop or mark to allow bursts		Don’t smooth or delay queue size ➔ immediate ECN marking

DCTCP recap

RED for Reno



Queue size
distribution



Reconfigured RED for DCTCP