RITE

REDUCING INTERNET TRANSPORT LATENCY

# Evaluating CoDel, FQ_CoDel and PIE: how good are they really?

Naeem Khademi <naeemk@ifi.uio.no>
David Ros <David.Ros@telecom-bretagne.eu>
Michael Welzl <michawe@ifi.uio.no>

ICCRG – IETF 88
Vancouver, BC, Canada

# Outline

# The New AQM Kids on the Block...

- Two very recent proposals:
  - (FQ_)CoDel (IETF 84)
  - PIE (IETF 85)
- Some older AQMs dating back to early 90's/00's (*RED, REM, BLUE, CHOKe,...)
  - Designed to be better than RED, just like CoDel and PIE
- Little academic literature available on CoDel and PIE

## Literature *(bold = peer-reviewed)*

| | CoDel | PIE | FQ_CoDel |
|---|---|---|---|
| Wired, sim | **[NJ12][GRT+13]**[WP12] [Whi13] | [Whi13] | [Whi13] |
| Wired, real-life | **[GRT+13]** ✓ | ✓ | ✓ |
| Wireless (any) | ✓ | ✓ | - |

NOTE: [WP12] and [Whi13] are on DOCSIS 3.0 while [GRT+13] has tests with LP CC.

# The New AQM Kids on the Block (cont.)

## AQM Deployment Status

- (W)RED is available on plenty of HW but mostly "turned off"

## Mentioned Reasons for Lack of Deployment

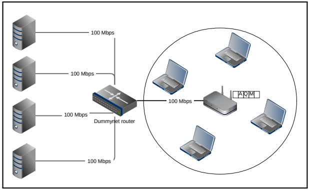
- Bad implementation (?)
- Hard to tune RED params

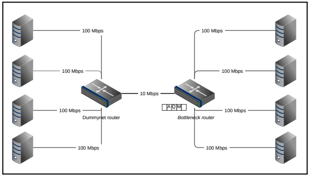
- Sally Floyd's ARED (2001 draft, available in Linux) adaptively tunes RED params aiming for a certain target queuing => with fixed BW maps to a "target delay"
- Target delay can be set in ARED, CoDel and PIE

# Experimental Setup

- **Traffic:** 60/180/300 sec (wired/wireless/RTT=500 ms) of *iperf*, repeated for 10 runs
- **AQM iface:** GSO TSO off, BQL=1514, txqueuelen=1000
- **TCP:** Linux default with *reno*
- **Topology:** Dumbbell with 4 sender-receiver pairs

| | |
|---|---|
| Model | Dell OptiPlex GX620 |
| CPU | Intel(R) Pentium(R) 4 CPU 3.00 GHz |
| RAM | 1 GB PC2-4200 (533 MHz) |
| Ethernet | Broadcom NetXtreme BCM5751<br>RTL-8139 (*AQM interface*)<br>RTL8111/8168B (*Dummynet* router) |
| Ethernet driver | tg3<br>8139too (*AQM* interface)<br>r8168 (*Dummynet* router) |
| 802.11 b/g | D-Link DWL-G520 AR5001X+ |
| 802.11 driver | ath5k |
| OS kernel | Linux 3.8.2 (FC14)<br>Linux 3.10.4 (*AQM* router) (FC16) |



# Experimental Setup (cont.)

- AQM parameters used *unless* otherwise noted.

## CoDel

interval=100 ms
target=5 ms

## PIE

parameters in [pie].

| PIE Parameter | Default value |
|---|---|
| $t_{update}$ | 30 ms |
| $T_{target}$ | 20 ms |
| $\alpha$ | 0.125 |
| $\beta$ | 1.25 |

## ARED

parameters in [FGS01].

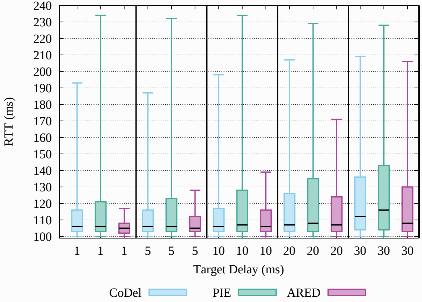
| ARED Parameter | Default value |
|---|---|
| *interval* | 500 ms |
| $\alpha$ | $min(0.01, p_{max}/4)$ |
| $\beta$ | 0.9 |

# Experimental Setup (cont.)

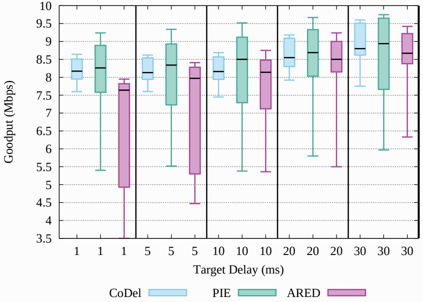
- RTT is measured on per-packet basis using *Synthetic Packet Pairs (SPP)* tool [spp]
  - Gives a very precise distribution of perceived RTT on the path
- Goodput is measured per 5-sec intervals
  - long-term throughput/goodput does not reflect AQM performance *over time* (e.g. bursts of packet drops are not desired)

# A Basic Test

## Single TCP Flow ($RTT_{base}$=100 ms)




(a) Per-packet RTT



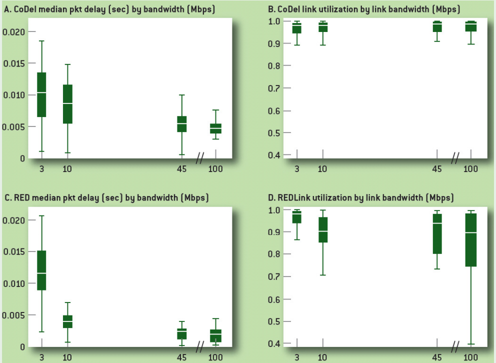

(b) Goodput

Per-packet RTT and goodput. Bottom and top of whisker-box plots show 10th and 90th percentiles respectively.

# A Basic Test (cont.)

- A similar trend can be observed between CoDel and RED in a different test in [NJ12]

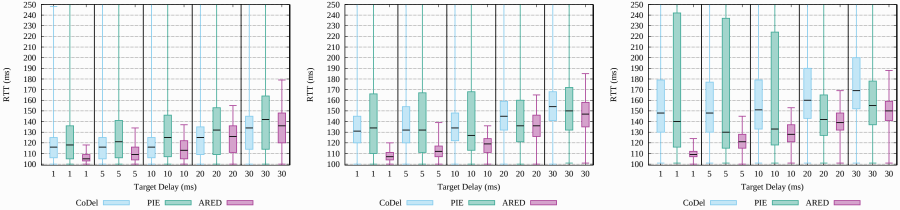
**CoDel vs. RED from K. Nichols, "Controlling Queue Delay" [NJ12]**




FTP traffic mix w/ and w/o web-browsing and CBR applications and RTTs from 10∼500 ms.

# Parameter Sensitivity (cont.)

## Target Delay (Per-packet RTT)



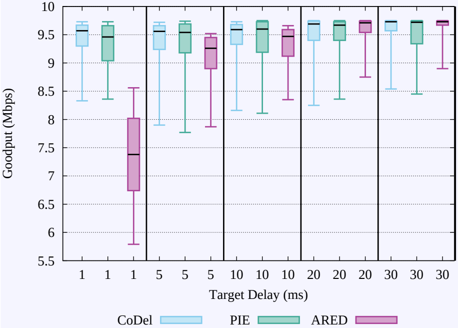

(c) Light (d) Moderate (e) Heavy

Per-packet RTT. Light, moderate and heavy congestion scenarios (4 senders and $RTT_{base}$=100 ms).

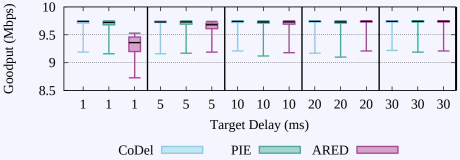
*Light*, *moderate* and *heavy* congestion correspond to 4, 16 and 64 concurrent TCP flows respectively.

# Parameter Sensitivity (cont.)

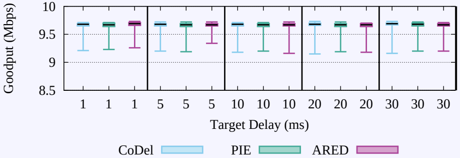
## Target Delay (Goodput)




(f) Light




(g) Moderate




(h) Heavy

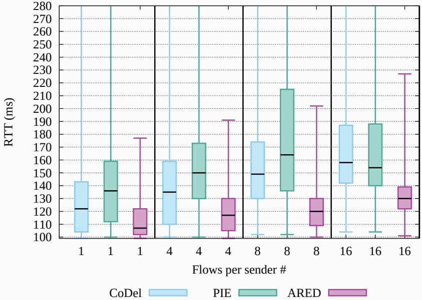
Goodput. Light, Moderate and Heavy congestion scenarios (4 senders and $RTT_{base}$=100 ms).

# AQM on 802.11 WLANs

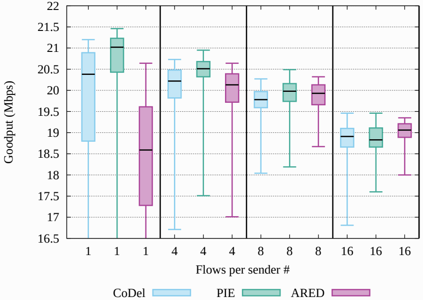
- 802.11 is a challenging environment for AQM deployment
  - Varying MCS (BW) by RA and MAC retries
    - Some drivers still use *SampleRate* instead of *Minstrel* (e.g. in FBSD)
  - Shared channel with various active STAs
    - Direction-based unfairness (uplink vs. downlink)
- Hard to predict the BW as input to ARED
  - We use TCP max achievable BW for testing (e.g. $\sim$27 Mbps in .11g)
- CoDel and PIE use delay
  - CoDel: queuing delay by timestamping
  - PIE: estimated queuing delay (queue_length / departure_rate)
- Public Wi-Fi e.g. at airports, hotels, corporations with bottleneck on *wlanX* interface

# AQM on 802.11 WLANs (cont.)

## 802.11 Downlink Traffic Scenario – Target Delay=5 ms




(i) Per-packet RTT



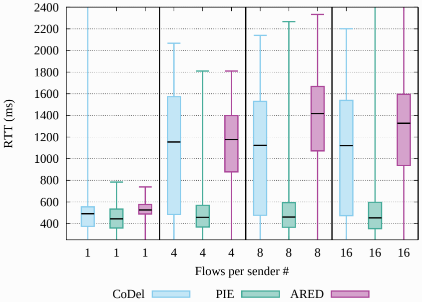

(j) Goodput

4 senders, $RTT_{base}$=100 ms.

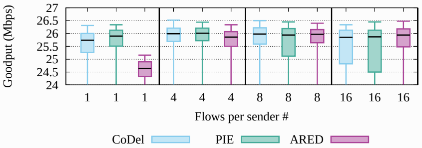
# AQM on 802.11 WLANs (cont.)

- AQM on AP's *wlanX* interface
- **~65%/~70% ACK loss** at AQM (ARED) router for 16/32 flows.

## 802.11 Mixed Traffic – Target Delay=5 ms (Uplink's Stats)




(k) Per-packet RTT



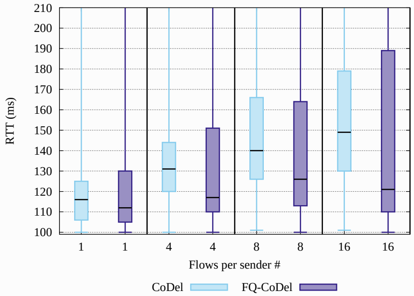

(l) Goodput

4 senders, $RTT_{base}$=100 ms.

# FQ_CoDel: Blending SFQ and AQM

- SFQ is highly likely to improve the performance when combined with any AQM
  - **Flow isolation/protection** with non-responsive traffic
  - Close to 100% flow-level **fairness** on the edge
  - Significantly Lower 10th percentile and *median* RTTs but with longer (upper) distribution tail

Comparison between CoDel and FQ_CoDel – *target_delay*=5 ms

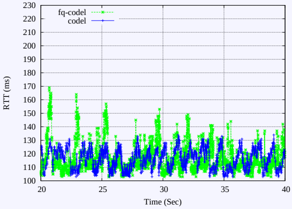



Per-packet RTT. 4 senders and $RTT_{base}$=100 ms.

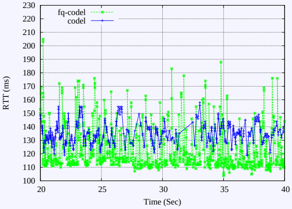
# FQ_CoDel: Blending SFQ and AQM

- **FQ_CoDel:** Lower median latency at the expense of higher jitter than CoDel (with the increase of congestion level)

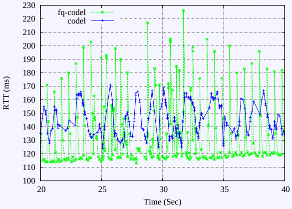
## Per-packet RTT Samples of a Single Flow – *target_delay*=5 ms




(m) Light Congestion




(n) Moderate Congestion




(o) Heavy Congestion

4 senders, $RTT_{base}$=100 ms.

# ECN

- Proposed two decades ago
- Still turned off in clients
- Can give a lot of benefits (out of this talk's scope)

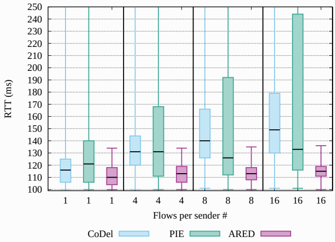
**RFC 3168**

*For a router, the CE codepoint of an ECN- Capable packet SHOULD only be set if the router would otherwise have dropped the packet as an indication of congestion to the end nodes.*

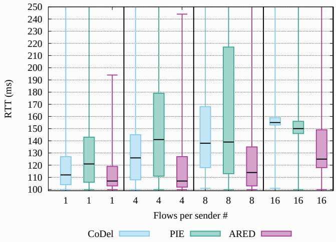
- RFC 3168 is the only guideline for general use
- **Marking affects the marking/dropping probability since it changes the metrics AQMs use (e.g. queue length and delay)**

# ECN (cont.)

## Per-packet RTT. 4 senders, *target_delay*=5 ms, $RTT_{base}$=100 ms

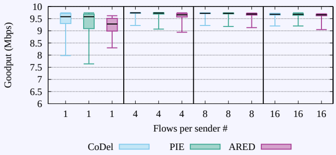



(p) ECN disabled

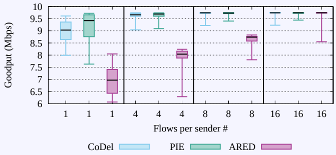



(q) ECN enabled

## Goodput




(r) ECN disabled




(s) ECN enabled

# ECN (cont.)

- Constant CE-marking under heavy congestion
- with ECN, ARED's goodput drops significantly as congestion level *decreases*
  - Aggressive reaction to the increase in average queue length
  - CE-marking (w/ ECN) 3.5∼6.6 times more than dropping (w/o ECN)

$p_{marking|ecn} / p_{drop|noecn}$

| Flows | CoDel | PIE | ARED |
|---|---|---|---|
| 4 | 1.256 | 1.156 | 6.621 |
| 16 | 1.356 | 1.106 | 3.465 |
| 32 | 1.719 | 1.591 | 4.303 |
| 64 | 6.117 | 6.569 | 3.873 |

# ECN (cont.)

## Our Recommendation

- AQMs should modify their dropping/marking decision process to incorporate the impact of CE-marked packets on their measurements.

## Possible Solutions

- Exclude CE-marked packets from queue size (ARED), or exclude delay caused by CE-marked packets (PIE)
- Set lower ECN thresholds for AQMs
  (fairness against non-ECN flows? dynamic threshold?)

# Conclusive Remarks

- ARED *only* performed worse than CoDel or PIE in few scenarios
  1. With very small number of flows
  2. Public Wi-Fi with mixed (up- and down-link) traffic
  3. With the common ECN implementation that CE-marks only when it would otherwise drop
- ARED outperforms CoDel and PIE in *all* other studied scenarios, most notably regarding delay

## Future Work

- More realistic traffic types (here, only bulk TCP traffic)
- Implementing and testing SFQ_ARED
- Delay-based ARED

# Bibliography

[FGS01] Sally Floyd, Ramakrishna Gummadi, and Scott Shenker.
Adaptive RED: An Algorithm for Increasing the Robustness of RED's Active Queue Management.
Technical report, 2001.

[GRT+13] Y. Gong, D. Rossi, C. Testa, S. Valenti, and D. Taht.
Fighting the Bufferbloat: On the Coexistence of AQM and Low Priority Congestion Control.
2013.

[NJ12] Kathleen Nichols and Van Jacobson.
Controlling Queue Delay.
*Queue*, 10(5):20:20–20:34, May 2012.

[pie] PIE Linux code (Cisco).
`ftp://ftpeng.cisco.com/pie/linux_code/`.

[spp] Synthetic Packet Pairs.
`http://caia.swin.edu.au/tools/spp/`.

[Whi13] Greg White.
A Simulation Study of CoDel, SFQ-CoDel and PIE in DOCSIS 3.0 Networks.
Technical Report, CableLabs, April 2013.

[WP12] Greg White and Joey Padden.
Preliminary Study of CoDel AQM in a DOCSIS Network.
Technical Report, CableLabs, November 2012.

# Q&A

**More on experimental results:** N. Khademi, D. Ros, and M. Welzl, *"The New AQM Kids on the Block: Much Ado About Nothing?"*, Technical Report 434, Department of Informatics, University of Oslo, 23 October 2013, available at http://urn.nb.no/URN:NBN:no-38868