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
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)*

<table>
<thead>
<tr>
<th></th>
<th>CoDel</th>
<th>PIE</th>
<th>FQ_CoDel</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wired, sim</td>
<td>[NJ12][GRT+13][WP12]</td>
<td>[Whi13]</td>
<td></td>
</tr>
<tr>
<td>Wired, real-life</td>
<td>[GRT+13] ✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Wireless (any)</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Model</th>
<th>Dell OptiPlex GX620</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td>Intel(R) Pentium(R) 4 CPU 3.00 GHz</td>
</tr>
<tr>
<td>RAM</td>
<td>1 GB PC2-4200 (533 MHz)</td>
</tr>
<tr>
<td>Ethernet</td>
<td>Broadcom NetXtreme BCM5751</td>
</tr>
<tr>
<td></td>
<td>RTL-8139 (<em>AQM interface</em>)</td>
</tr>
<tr>
<td></td>
<td>RTL8111/8168B (<em>Dummynet router</em>)</td>
</tr>
<tr>
<td>Ethernet driver</td>
<td>tg3</td>
</tr>
<tr>
<td></td>
<td>8139too (<em>AQM interface</em>)</td>
</tr>
<tr>
<td></td>
<td>r8168 (<em>Dummynet router</em>)</td>
</tr>
<tr>
<td>OS kernel</td>
<td>Linux 3.8.2 (FC14)</td>
</tr>
<tr>
<td></td>
<td>Linux 3.10.4 (<em>AQM router</em>) (FC16)</td>
</tr>
</tbody>
</table>
Experimental Setup (cont.)

- AQM parameters used *unless* otherwise noted.

<table>
<thead>
<tr>
<th>CoDel</th>
<th>PIE</th>
<th>ARED</th>
</tr>
</thead>
<tbody>
<tr>
<td>interval=100 ms</td>
<td>parameters in [pie].</td>
<td>parameters in [FGS01].</td>
</tr>
<tr>
<td>target=5 ms</td>
<td>PIE Parameter</td>
<td>Default value</td>
</tr>
<tr>
<td></td>
<td>$t_{update}$</td>
<td>30 ms</td>
</tr>
<tr>
<td></td>
<td>$T_{target}$</td>
<td>20 ms</td>
</tr>
<tr>
<td></td>
<td>$\alpha$</td>
<td>0.125</td>
</tr>
<tr>
<td></td>
<td>$\beta$</td>
<td>1.25</td>
</tr>
</tbody>
</table>
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

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

(b) Goodput

N. Khademi, D. Ros, M. Welzl () Evaluating CoDel, FQ_CoDel and PIE November 5, 2013 8 / 23
A similar trend can be observed between CoDel and RED in a different test in [NJ12]
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.
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 \texttt{SampleRate} instead of \texttt{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 wlan\textit{X} interface
AQM on 802.11 WLANs (cont.)

802.11 Downlink Traffic Scenario – Target Delay=5 ms

(i) Per-packet RTT

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

(j) Goodput
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

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)

<table>
<thead>
<tr>
<th>Flows</th>
<th>CoDel</th>
<th>PIE</th>
<th>ARED</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>1.256</td>
<td>1.156</td>
<td>6.621</td>
</tr>
<tr>
<td>16</td>
<td>1.356</td>
<td>1.106</td>
<td>3.465</td>
</tr>
<tr>
<td>32</td>
<td>1.719</td>
<td>1.591</td>
<td>4.303</td>
</tr>
<tr>
<td>64</td>
<td>6.117</td>
<td>6.569</td>
<td>3.873</td>
</tr>
</tbody>
</table>
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.

Fighting the Bufferbloat: On the Coexistence of AQM and Low Priority Congestion Control.
2013.

Controlling Queue Delay.

[pie] PIE Linux code (Cisco).

[spp] Synthetic Packet Pairs.

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

Preliminary Study of CoDel AQM in a DOCSIS Network.