Operating ranges, tunability and performance of CoDel and PIE

Nicolas Kuhn, David Ros, Amadou Baba Bagayoko,
Chamil Kulatunga, Gorry Fairhurst, Naeem Khademi

PII: S0140-3664(16)30271-7
DOI: 10.1016/j.comcom.2016.07.013
Reference: COMCOM 5360

To appear in: *Computer Communications*

Received date: 31 July 2015
Revised date: 14 March 2016
Accepted date: 24 July 2016

Please cite this article as: Nicolas Kuhn, David Ros, Amadou Baba Bagayoko, Chamil Kulatunga, Gorry Fairhurst, Naeem Khademi, Operating ranges, tunability and performance of CoDel and PIE, *Computer Communications* (2016), doi: 10.1016/j.comcom.2016.07.013

Operating ranges, tunability and performance of CoDel and PIE

Nicolas Kuhn[1], David Ros[2], Amadou Baba Bagayoko[3], Chamil Kulatunga[4], Gorry Fairhurst[4], Naeem Khademi[5]

[1] *Centre National d'Etudes Spatiales (CNES), Toulouse, France*
[2] *Simula Research Laboratory, Fornebu, Norway*
[3] *IMT Télécom Bretagne / IRISA, France*
[4] *School of Engineering, University of Aberdeen, United Kingdom*
[5] *Department of Informatics, University of Oslo, Norway*

Abstract

Bufferbloat is excessive delay due to the accumulation of packets in a router's oversized queues. CoDel and PIE are two recent Active Queue Management (AQM) algorithms that have been proposed to address bufferbloat by reducing the queuing delay while trying to maintain a high bottleneck utilization. This paper fills a gap by outlining what are the operating ranges, that is the network characteristics (in terms of round-trip times and bottleneck capacity), for which these algorithms achieve their design goals. This new approach to the problem lets us identify deployment scenarios where both AQM schemes result in poor performance when used with default parameters. Because PIE and CoDel have been proposed with RED's deployment issues in mind, it was essential to evaluate to what extent we can tune them to achieve various trade-offs and let them control the queuing delay outside their default operating range. We find that, by appropriate tuning (1) the amount of buffering can easily be controlled with PIE, (2) the Round Trip Time (RTT) sensitivity of CoDel can be reduced. Also, we observe there is more correlation between the congestion level, the achieved queuing delay and the targeted delay with CoDel than with PIE. This paper therefore concludes there is no single overall best AQM scheme, as each scheme proposes a specific trade-off.

Keywords: AQM; Bufferbloat; CoDel; PIE; congestion control

1. Introduction

The combination of large buffers and loss-based congestion control mechanisms can result in persistently full buffers and increased end-to-end delay; this issue, known as bufferbloat [1], may be a serious hindrance to the increasing number of latency sensitive applications. Even though there is some discussion on whether bufferbloat is a widespread problem [2, 3], its presence has been observed in mobile networks [4, 5].

Active Queue Management (AQM) has been proposed as a key way to alleviate bufferbloat, i.e., AQM should be able to reduce end-to-end latency and avoid lock-out phenomena within the Internet [6]. Widespread deployment of AQM has not happened since the first proposals dating back more than a decade, mainly due to their sensitivity to the operating conditions in the network and to the characteristics of the low layers, and especially the difficulty of tuning their parameters. Indeed, even though the hard-to-tune RED [7] was quickly followed by the Adaptive Random Early Detection (ARED) [8] algorithm, and even though many other mechanisms have since been proposed to ease setting parameters, AQM schemes have been reported to be usually turned off. Controlled Delay (CoDel) [9] and Proportional Integral controller Enhanced (PIE) [10] are two recently proposed AQM schemes that aim at tackling bufferbloat by controlling the queuing delay while attempting to address the problems that (A)RED encountered in terms of deployment and stability. The authors of PIE state in [10] that PIE "self-tunes its parameters to optimize system performance" depending on network conditions, and the authors of CoDel claim in [9] that CoDel "controls delay, while being insensitive to round-trip delays, link rates, and traffic loads."

This paper provides the first evaluation of the limits of PIE's self-tuning ability and CoDel's sensitivity to network conditions by defining their operating ranges (in terms of congestion levels, round-trip times or link rates). Our work is based on ns-2 simulations, as this lets us assess a wide range of network characteristics. CoDel has been found to be sensitive to the congestion level [11,

12], and PIE and CoDel have difficulties controlling the queuing delay both in high Round Trip Time (RTT) and high capacity paths [13] and in low capacity paths [14]. This paper goes beyond the previously cited works by assessing, with a new approach, the range of conditions in which the default parameters of PIE and CoDel fail to perform well.

We then focus on cases where the default parameters of CoDel and PIE are not suitable to assess whether CoDel and PIE can be easily tuned to (1) achieve any desired trade-off between queuing delay and bottleneck utilization and (2) let CoDel and PIE improve their performance outside their "default" operating ranges.

This paper finally compares how CoDel and PIE react to a traffic mix consisting of applications that have different characteristics, and how this specific traffic impacts their tunability.

The remainder of this paper is structured as follows: in Section 2, we briefly describe the CoDel and PIE algorithms. We present the simulation setup in Section 3. Section 4 looks into the operating ranges of the AQM schemes. In Section 5, we study the tunability of CoDel and PIE, that is, how their parameters can be set to achieve different trade-offs. Section 6 assesses the performance and the tunability of CoDel and PIE when the traffic in the network consists in a traffic mix where the different flows have various characteristics. Section 7 presents and sums up the related work to better highlight the contributions of this paper. Section 8 concludes the paper.

2. CoDel and PIE Algorithms

In this section, we sketch how CoDel and PIE work, focusing on how these algorithms use similarly-named *target delay* (denoted by τ) and *update interval* (denoted by λ) parameters in a different manner. When needed for clarity, we use the notation τ_x, λ_x to denote the parameters of AQM algorithm x.

2.1. CoDel

CoDel keeps track of enq_i, the *enqueue* time of each packet p_i. At the *dequeue* time deq_i, CoDel measures the queuing delay of each dequeued packet, $\delta_i = deq_i - enq_i$. The algorithm has two states, a dropping state and a non-dropping state, and the initial state is non-dropping. CoDel keeps track of the number of consecutive drops in a variable n_{drop}, that is initialized as: $n_{drop} = 1$. We denote by $\mu_{n_{drop}}$ the interval after which CoDel may enter (or stay in) the dropping state and drop one packet; $\mu_{n_{drop}}$ depends on the number of consecutive drops, as follows:

$$\mu_{n_{drop}} = \frac{\lambda_{CoDel}}{\sqrt{n_{drop}}} \tag{1}$$

Let t_1 denote the time of the last drop. When a packet p_i is about to be dequeued at time t:

- if $\delta_i > \tau_{CoDel}$ and $t > t_2 = t_1 + \mu_{n_{drop}}$, then: (a) CoDel enters (or stays in) the dropping state, (b) p_i is dropped, (c) n_{drop} is increased by 1;
- else: (a) CoDel enters the non dropping state, (b) p_i is not dropped, (c) n_{drop} is set to 1.

Then, at t_2, $\mu_{n_{drop}}$ is updated as per Eq. (1).

2.2. PIE

PIE estimates the current queuing delay, $E[T]$, by considering the current queue length, *qlen*, and the draining rate of the queue, *depart_rate* as follows: $E[T] = qlen/depart\_rate$. $E[T]_{old}$ represents the previous estimation of the queuing delay. PIE updates its drop probability, p_{drop}, every λ_{PIE} according to Eq. (2). When a packet is enqueued, it is dropped with probability p_{drop}.

$$p_{drop} = p_{drop} + \alpha \times (E[T] - \tau_{PIE}) + \beta \times (E[T] - E[T]_{old}) \tag{2}$$

The α parameter determines how the deviation of current latency from the target value affects the drop probability. The β term exerts additional adjustments depending on whether the latency is trending up or down. PIE scales up these parameters to make p_{drop} adapt more rapidly when $p_{drop} \notin [0.01, 0.1]$.

2.3. Default parameterizations

Since PIE [10] and CoDel [9] are intended to be mainly deployed with default parameters [6], we first evaluate their operating range with these default parameterizations. The default values of τ and λ are: $\tau_{CoDel} = 5\,\text{ms}$, $\lambda_{CoDel} = 100\,\text{ms}$, $\tau_{PIE} = 20\,\text{ms}$ and $\lambda_{PIE} = 30\,\text{ms}$.

The default settings of CoDel and PIE, together with their different ways of deciding whether to drop a packet, achieve different trade-offs between the bottleneck utilization and the queuing delay. Intuitively, this can be explained as follows:

- PIE adapts its drop probability every λ_{PIE}, whereas CoDel may drop a packet only every $\mu_{n_{drop}}$ in order to give end-points time to react to a drop.
- PIE tries to maintain the average queuing delay around τ_{PIE}, whereas CoDel would tend to act as a delay limiter, since drops are only applied if delays grow above τ_{CoDel}.
- PIE allows more buffering than CoDel which would result in higher bottleneck utilization.

2.4. Scheduling and AQM schemes

A recent proposal, Flow-Queuing CoDel (FQ-CoDel) [15] combines Stochastic Fair Queuing (SFQ), priority queuing and CoDel. The IETF recommendations on AQM [6] note that "queue management" and "scheduling schemes" are closely related, but address different performance issues. They may be used in combination, but they should not be confused [6]. This article focuses on queue management alone and the operating ranges of CoDel and PIE, and does not consider FQ-CoDel or any other combination of AQM and scheduling.

3. Network characteristics for the sensitivity and tunability studies

Figure 1 presents the topology for the ns-2 simulations. We do not pretend that this topology perfectly reflect actual topologies, however it is commonly

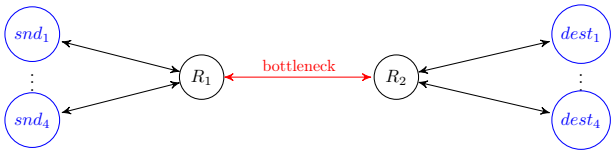


Figure 1: Dumbbell simulation topology.

used to assess the performance of TCP and AQM schemes, such as advised in standard documents [16, 17]. We define three congestion levels, *light*, *moderate* and *heavy*, where respectively 4, 16 and 64 long-lived TCP flows are associated with 4 sender/receiver pairs sharing the bottleneck.

The capacity of the bottleneck is 10 Mbps, and the capacities of other links $snd_{1\ldots n} \leftrightarrow R_1$ and $dest_{1\ldots n} \leftrightarrow R_2$ are 100 Mbps, unless otherwise noted. The size of buffers for the simulations are set according to the bandwidth–delay product, where the bandwidth is the capacity of the bottleneck and the delay is the base RTT (i.e., in the absence of queuing).

Figure 2 serves as example of the figures used in this paper: the boxes encompass the first to third quartile (25 % to 75 % of the samples) and the median value is shown as a point. On the x axis, the distance Δ to the target delay is the difference between the queuing delay δ and the target delay τ:

$$\Delta = \delta - \tau \tag{3}$$

We represent the bottleneck utilization as a function of Δ. Figure 2 illustrates ideal design goals of trying to maximise bottleneck utilisation, while keeping the queuing delay under control with respect to the desired target delay.

The bottleneck utilization is averaged every 5 s and we measure the per-packet queuing delay. Each simulation lasts 300 s and has been repeated with 10 different random seeds, so each figure summarizes 600 link utilization samples and several thousands of per packet delay samples.

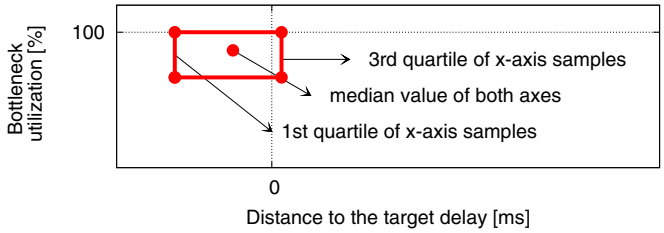


Figure 2: Plot style used in Figures 3 to 8.

4. Sensitivity of CoDel and PIE to network conditions

4.1. Impact of congestion level

Traffic and congestion levels are higher during peak hours than off-peak hours. When an AQM scheme is deployed in such a non-stationary traffic environment, its ability to actually maintain its control on the buffer occupancy is challenged. Thus, this section assesses the sensitivity of CoDel and PIE to different congestion levels.

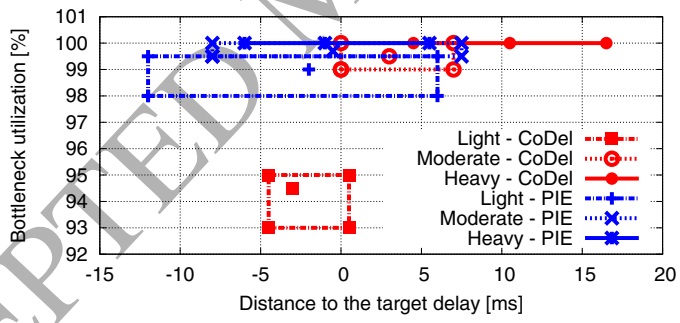


Figure 3: Impact of the congestion level.

For the tests discussed in this section, the base RTT was set to 100 ms, and the one-way delay (OWD) of each link is RTT/6. The base RTT was set to 100 ms because a common value for terrestrial inter-continental Internet access and also because it is the RTT assumed by the algorithm of CoDel: the rationale for this choice is thus to evaluate the capability of CoDel to carry out various

congestion levels without considering the impact of CoDel's answer to wrong RTT assumptions.

Figure 3 shows that PIE's ability to control the median queuing delay does not depend strongly on the level of congestion. Under the three levels of congestion considered, the median queuing delay is kept close to τ_{PIE}. We notice that the spread decreases as the congestion level increases. PIE's algorithm enables it to have an median queuing delay very close to τ_{PIE} with the traffic considered. However, probably because PIE considers previous queue states, adaptation of drop probability to changes of the queue occupancy can be slow, resulting in non-negligible variations in queuing delay.

The figure also shows that the median queuing delay with CoDel is lower than, but close to, τ_{CoDel}, which is consistent with claims in [18]. However, we can see that higher loads can result in delays higher than 20 ms; under heavy load, median queuing delay is $\approx$ 15 ms (i.e., $\Delta \approx 10$ ms).

As discussed in § 2.3, by default PIE should allow more buffering than CoDel. This is verified by the results presented in this section (remember that $\Delta = 0$ means $\delta = 20$ ms for PIE but $\delta = 5$ ms for CoDel).

To summarize, with PIE, the achieved median queuing delay is close to τ_{PIE} for the tested congestion levels. However the variation of the queuing delay may be high. With CoDel, we noted a stronger correlation between the congestion level, the queuing delay and τ.

4.2. RTT sensitivity

A different base RTT—that is, the minimum RTT, without any queuing delays—can impact the behavior of congestion controllers and, as a result, the ability of an AQM to control the queue. Thus, this section examines the sensitivity of CoDel and PIE to different base RTTs.

The analysis below considers the following edge link characteristics: $RTT_{(snd_{1\ldots N} \leftrightarrow R_1)} = RTT_{(dest_{1\ldots N} \leftrightarrow R_2)} = 1.25\,\text{ms}$. The RTT of the bottleneck varies so that the base RTT is one of: 100 ms, 200 ms and 300 ms. We do not

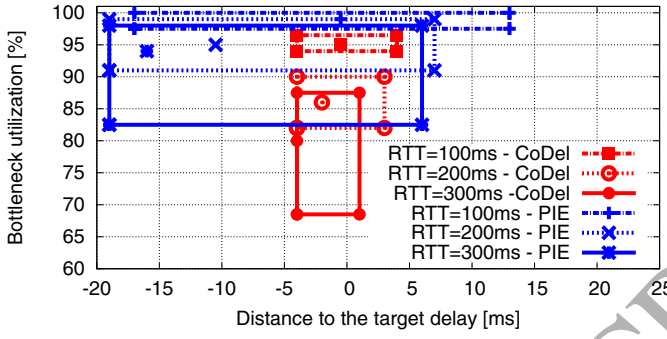


Figure 4: Impact of the base RTT.

consider a lower RTT: since CoDel has shown poorer performance with larger RTTs [9, Figure 8], we prefer focusing on an RTT higher than 100 ms.

Delays of 200 and 300 ms can be considered as common for intercontinental paths. We consider a scenario where the likely congestion level in such a network is light.

The results presented in Figure 4 show that both CoDel and PIE are sensitive to the base RTT value. For both algorithms, when the RTT increases:

- the median bottleneck utilization is lower—down to 80 % with CoDel and down to 94 % with PIE when the RTT is 300 ms;
- the median queuing delay decreases;
- there is more variation in bottleneck utilization and less variation in queuing delay.

To better understand the impact of higher RTTs on the behavior of AQM schemes, Figure 5 presents the queuing delay (averaged every second) and the dropping probability of PIE for two RTT values during a representative time interval. When the RTT is 100 ms, the queuing delay tends to oscillate around τ_{PIE}. When the RTT is 300 ms, the maximum queuing delay and the maximum dropping probability are not higher than when the RTT is 100 ms. The main difference observed when the RTT increases, is wider oscillations in buffer

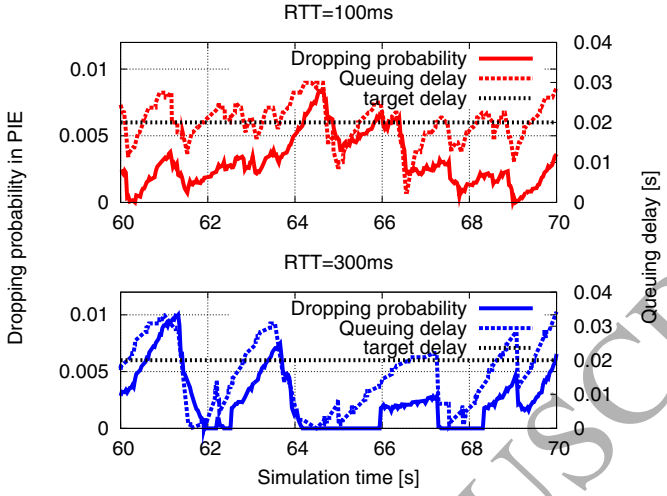


Figure 5: Behavior of PIE with RTTs of 100 and 300 ms.

occupancy and queuing delay: when the queuing delay is higher than τ_{PIE}, the dropping probability increases in order to maintain a queuing delay around τ_{PIE}. However, when the RTT increases, the increase of the dropping probability does not result in a queuing delay around τ_{PIE} but in a momentarily empty buffer.

4.3. Capacity-limited networks

There is no single "typical" bottleneck speed and there is a wide variety of access technologies; fiber deployments in urban areas may offer around 100 Mbps, but long-haul DSL used in many developed countries for rural access networks may offer only 1 Mbps and some links may still have lower capacities [19]. Thus, this section looks at the sensitivity of CoDel and PIE to low link capacities.

The following analysis considers the following edge link characteristics: $RTT_{(snd_{1\ldots N} \leftrightarrow R_1)} = RTT_{(dest_{1\ldots N} \leftrightarrow R_2)} = 1.25\,\text{ms}$. The base RTT is 100 ms. The capacity of the bottleneck is set to one of 500 kbps, 1 Mbps or 2 Mbps. We consider a scenario where the congestion level is light, as defined in § 3.

Table 1 shows the time needed to transmit 1500 B (the packet size used in this paper), the Bandwidth-Delay Product (BDP) and the resulting size of

the average or maximum queue size that PIE or CoDel, respectively, tries to maintain. In these scenarios τ_{CoDel} and τ_{PIE} are so low that the targeted queuing delays are by default not achievable as they mean extremely short queues: this can be seen in Figure 6 with queuing delays way above the targets when the bottleneck is set to 500 kbps.

Table 1: Bottleneck capacity, BDP and τ.

	bottleneck capacity		
	500 kbps	1 Mbps	2 Mbps
transmission time of 1500 B (ms)	24	12	6
BDP (in 1500 B packets)	4.2	8.3	16.6
PIE: avg. queue size aimed (in pkts)	≈ 1	≈ 2	≈ 3
CoDel: max. queue size aimed (in pkts)	< 1	< 1	≈ 1

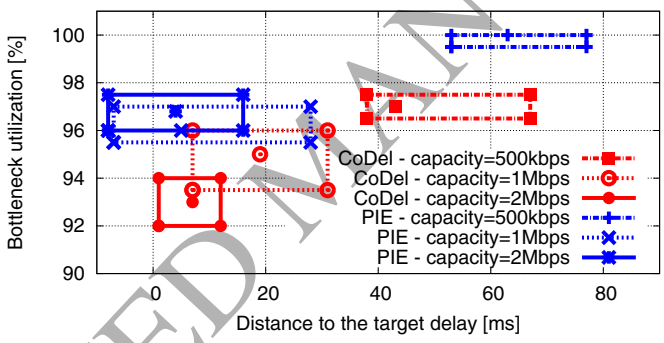


Figure 6: Impact of low bottleneck capacities.

4.4. Discussion

In § 4.1, we saw that CoDel is sensitive to the traffic load; in § 4.2, that CoDel and PIE are sensitive to the base RTT; and in § 4.3, that CoDel and PIE are sensitive to link speeds and "break" when link capacities are very low: the default parameters of PIE and CoDel may not suit every situation, making these AQM schemes to drastically reduce the bottleneck utilization or to allow a queuing delay that is far from τ.

Moreover, in light of the results presented so far, we can conclude that:

- CoDel may not work well with RTTs higher than 100 ms and is sensitive to the traffic load;
- the self-tuning of PIE, supposed to make it robust and optimized for various network conditions, shows some limits.

Therefore, suggesting that no knobs need to be turned may be misleading.

The AQM characterization guidelines detail that the assessments "are not bound to a particular evaluation tool-set" [16]. Our high-level conclusions, based on ns-2 simulations, are consistent not only with other works that used ns-2 as well [11], but also with studies based on IKR SimLib and its Linux Virtual Machine (VM) integration [13] or emulated networks [20] using the Linux implementations of the algorithms.

5. Tunability of CoDel and PIE

We focus on cases where the default parameters are not suitable, and τ and λ were changed to assess whether CoDel and PIE can be easily tuned to (1) achieve any desired trade-off between queuing delay and bottleneck utilization and (2) let CoDel and PIE improve their performance outside their "default" operating ranges.

5.1. Operating ranges of CoDel and PIE

We define the operating range as the range of network characteristics (bottleneck capacity, RTT) for which the AQM schemes work as desired, i.e., maintaining a high bottleneck utilization and a low queuing delay.

Based on our results and those of [13, 20, 14, 11], we outline below the operating ranges of PIE and CoDel. Considering only the RTT and the capacity of the bottleneck, denoted $base_{RTT}$ and C_{bot}, PIE and CoDel, with their default parameters, have trouble with controlling the queuing delay and maintaining a high bottleneck utilization if:

- $base_{RTT} \geq 200$ ms (§ 4.2 and [20, 13]);

- $C_{bot} < 2\,\text{Mbps}$ (§ 4.3 and [14]);
- $C_{bot} > 100\,\text{Mbps}$ ([13]).

We have observed that these AQM algorithms do not fulfill their design goal outside these bounds, but we cannot guarantee that they work as expected within them. If we consider other parameters than $base_{RTT}$ and C_{bot}, our results and those presented in [11] illustrate that there is a high correlation between the traffic load and the performance of CoDel.

The limited operating range of CoDel and PIE prevents them from working as expected with the specific characteristics of e.g. rural broadband ($base_{RTT} \approx 300\,\text{ms}$ and $C_{bot} \approx 1\,\text{Mbps}$) and data-center networks ($base_{RTT} \approx 10^{-5}\,\text{ms}$ and $C_{bot} \approx 10^{4}\,\text{Mbps}$).

PIE and CoDel have had to be updated for cable-modems networks [21]: in the context of networks with characteristics that make them fit into the operating ranges, the adequate performance of CoDel and PIE may not be guaranteed as other parameters may have an impact, such as lower layers characteristics. This shows that both schemes can be tuned to achieve another trade-off between queuing delay and bottleneck utilization. However, it is not clear how changes on τ and λ would affect the resulting queuing delay and bottleneck utilization.

We therefore turn to evaluating the "tunability" of PIE and CoDel, that is, to assess to what extent we can obtain different trade-offs than the default ones by changing only τ and λ; this would be a straightforward way to adapt their operating ranges to specific network conditions.

PIE has already been updated for data-centers [22] by changing the α and β parameters (see Eq. (2)). However, the IETF recommendations on AQM [6] advocate the selection of default parameters appropriate to the general Internet with auto-tuning to avoid RED's deployment issues.

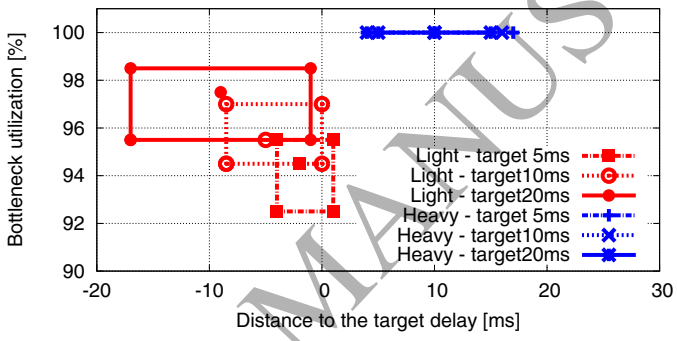
In [18], the authors say that λ_{CoDel} should be set to the RTT and τ_{CoDel} to 5% of λ_{CoDel}. The objective is to make CoDel less sensitive to different base RTTs, but the viability of these settings was not shown by the authors of [18]. We should also verify whether the queuing delay is actually closer to τ_{CoDel}

when it is set to another value than 5% of λ_{CoDel}.

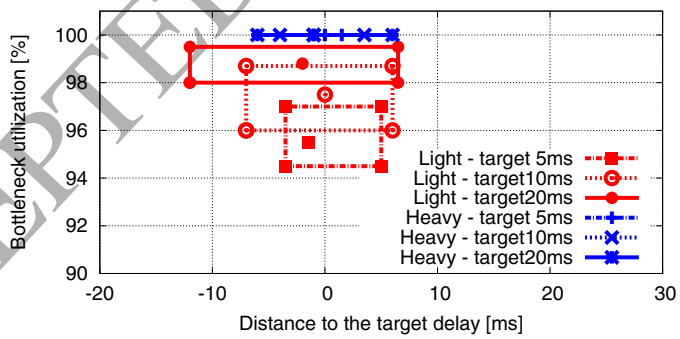
5.2. Impact of the target delay

We first evaluate the sensitivity of CoDel and PIE to changes of τ for various congestion levels. The network characteristics (RTT, link capacity, traffic generation) are the same as in § 4.1.

In our simulations, we also tested with other values of τ but results were qualitatively similar, so for the sake of clarity we only present those corresponding to $\tau = \{5, 10, 20\}$ ms.



(a) CoDel.



(b) PIE.

Figure 7: Sensitivity to τ; λ is set to its default value for each AQM.

Figure 7 shows that, as τ increases, both CoDel and PIE allow more buffering, resulting in higher bottleneck utilization. Also, when τ increases, the queu-

ing delay δ, that is $\Delta+\tau$, increases. For both schemes, when the congestion level is heavy, an increase in τ does not significantly change the bottleneck utilization or the queuing delay.

With CoDel (Figure 7a), when the congestion level is light and for all the target delays considered, 75 % of the queuing delay samples are lower than τ_{CoDel}, which is in line with what CoDel has been designed for. When the congestion level is heavy, the median and maximum queuing delays are higher than τ_{CoDel}, no matter the value of τ_{CoDel}. Thus, CoDel does act as a delay limiter as long as the traffic load is not high.

With PIE (Figure 7b), under the two levels of congestion considered, and for the various target delays considered, the median queuing delay is kept close to τ_{PIE}. Increasing τ_{PIE} results in higher bottleneck utilization when the congestion level is light. Therefore, different trade-offs can be achieved with PIE: by changing τ_{PIE}, one can accurately tune the median queuing delay to any desired value, bearing in mind that increasing τ_{PIE} results in a higher bottleneck utilization and more variability in queuing delay.

5.3. Impact of the update interval

In this section, we evaluate the sensitivity of CoDel and PIE to changes of λ when the congestion level is light. The network characteristics (RTT, link capacity, traffic generation) are the same as in § 4.1 with a base RTT of 300 ms.

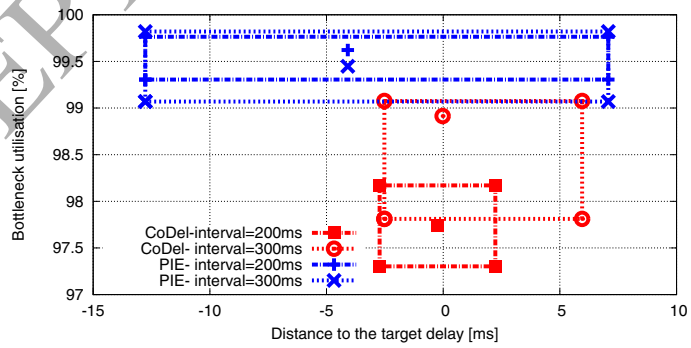


Figure 8: Sensitivity to λ. $RTT = 300$ ms, default τ.

The performance issues that had been observed in § 4.2 are alleviated when λ_{CoDel} is higher than 100 ms and set closer to the RTT, as illustrated in Figure 8; contrary to what is seen Figure 4, bottleneck utilization is very close to 100 %. The fact that, in most deployment cases, the path RTT is not known at the router deploying the AQM is a deployment issue for CoDel.

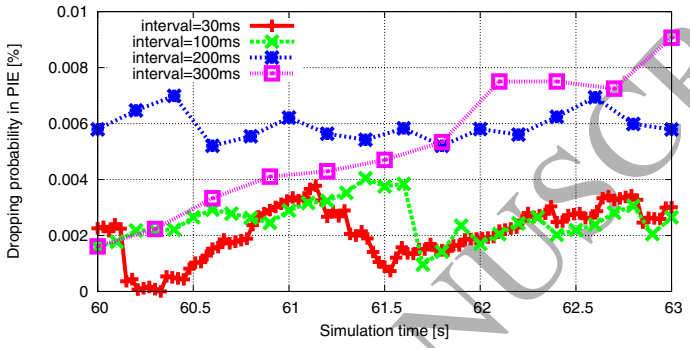


Figure 9: PIE's dropping probability for various values of λ_{PIE}.

By increasing λ_{PIE} when the RTT increases, the median bottleneck utilization can be increased (compare with the results in Figure 4), but the high variability of the queuing delay cannot be reduced. To visualize the impact of λ_{PIE} on PIE's behavior, we represent in Figure 9 the evolution of the drop probability, during a representative time interval, for various values of λ. This figure shows that increasing λ_{PIE} tends to result in a smoother evolution of the drop probability, which does not actually reduce the RTT sensitivity of PIE in terms of stabilizing queuing delay.

6. Performance with mixed traffic

Following the section "8.1. Traffic mix" of the IETF AQM characterization guidelines [16], this section further assesses the tunability of CoDel and PIE when the traffic is composed of a mix of applications which have different characteristics.

This section compares the performance of CoDel and PIE when they are not

stressed by a high congestion level, as CoDel has been found to be sensitive to the congestion level (§ 4.1 and [11, 12]). The scope of this section to evaluate how suitable both schemes limit the queuing delay and we suggest can guarantee good user experience for latency sensitive applications.

6.1. Topology and traffic characteristics

We present in Figure 10 the topology and how the traffic mix is transferred between the nodes.

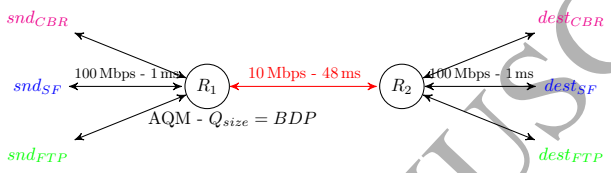


Figure 10: Topology and traffic mix

This evaluation is made with network conditions within the operating range of these schemes, that are defined in § 5.1. Therefore, we set the capacity of the bottleneck (between R_1 and R_2) to 10 Mbps and its one-way delay to 48 ms. For the other links (between snd_X and R_1, or between R_2 and $dest_X$), the capacity is set to 100 Mbps and the one-way delay to 1 ms. All the links are symmetric and the base RTT is 100 ms.

CoDel or PIE may be introduced at R_1 and the queue size is set to the BDP. To further evaluate the tunability of both schemes and validate the conclusions derived in § 5.2, we will consider two target values for both CoDel and PIE ($\tau_{CoDel} = \tau_{PIE} = 5$ ms or $\tau_{CoDel} = \tau_{PIE} = 20$ ms). As shown in § 5.3, the interval has been shown to (1) be related to the RTT with CoDel and this scenario considers a unique RTT, and (2) only "smooth" the dropping probability in PIE; therefore, we only consider one value for this parameter that will be set to its default value for each algorithm.

The different kinds of traffic considered are the following:

- Between snd_{CBR} and $dest_{CBR}$, N_{CBR} Constant Bit-Rate (CBR) flows use User Datagram Protocol (UDP) with a sending rate of 87 kbps and a

packet size of 218 B. This models Voice-over-IP (VoIP) or gaming traffic, as in [21, p. 17].

- Between snd_{SF} and $dest_{SF}$, N_{SF} flows transfer files of S kB ($S \in \{15; 44; 73; 102\}$). For every new download, a value of S is randomly picked from the set $\{15; 44; 73; 102\}$ and every new download starts after T seconds, generated with an exponential law with an average of 9.5 s. This traffic enables assesment of the benefits of using AQM for short flows.

- Between snd_{FTP} and $dest_{FTP}$, N_{FTP} TCP bulk flows are generated. TCP flows use the CUBIC congestion control policy. The TCP options are the same as those specified in § 3.

The following traffic mixes are generated:

- Traffic case 1: $N_{CBR} = 1$, $N_{SF} = 6$ and $N_{FTP} = 1$;

- Traffic case 2: $N_{CBR} = 2$, $N_{SF} = 12$ and $N_{FTP} = 2$.

All flows randomly start between 0 s and 1 s. Each run lasts 100 s and is repeated 10 times. The metrics are sampled each second (except for the queuing delay and the one way delay which are sampled per-packet).

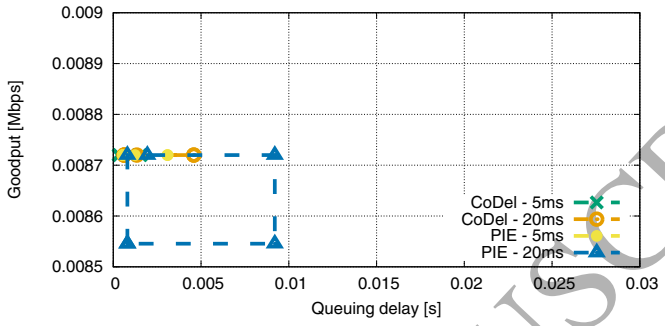
In Figures 11, 12 and 13, "CoDel - 5ms" (resp. "PIE - 20ms") denotes the case where the AQM is CoDel (resp. "PIE") with a target value of 5 ms (resp. 20 ms).

Each following subsections 6.2, 6.3 and 6.4 separately present the results for each class of traffic but the results have been obtained with every class generating traffic. As one example, when subsection 6.2 presents the results for the CBR traffic with traffic case 1, the traffic been carried out in the network is: 1 CBR flow, 6 small files downloads and 1 FTP bul transfer.

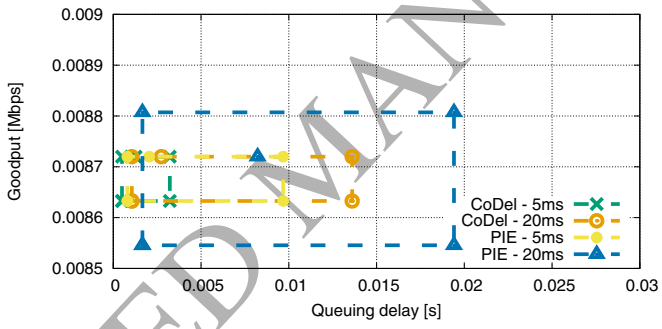
6.2. CBR traffic - between snd_{SF} and $dest_{SF}$

The performance for CBR traffic is shown in Figure 11. For the sake of simplicity, the cumulative goodput measured at node $dest_{CBR}$ has been divided

by N_{CBR}. This figure has been generated following the example in Figure 2. For the sake of clarity, the results with DropTail are not shown on this graph. These results show that the measured queuing delay is close to 100 ms.



(a) Traffic case 1



(b) Traffic case 2

Figure 11: Average goodput and queuing delay for the CBR traffic

With CoDel as an AQM, changing the target value does not affect the queuing delay experienced by the CBR flow for traffic case 1. The queuing delay is never above the target delay, whether it is set to 5 ms or 20 ms. With the traffic case 2, the queuing delay is increased and the meaning of the target delay is consistent with the conclusions of § 5.2. These results further provide evidence that there is a correlation between the traffic characteristics and the performance of CoDel.

When the PIE AQM algorithm is used, the queuing delay increases when the

target delay increases, which is closer to what is expected. This confirms our conclusions proposed in § 5.2 that PIE can easily be tuned to achieve different trade-offs. With the traffic case 2, the median queuing delay is notably lower than the target, because the congestion level is low: PIE allows buffering, but on average, the buffer occupancy is low. As PIE allows more buffering than CoDel, more packets might accumulate in the buffer, resulting in data arriving in bursts at the receiver, which explains why the cumulative goodput may be above 87 kbps.

When the traffic consists of at least two bulk flows, our conclusions of § 5.2 are confirmed: with they have the same target delays, PIE would allow more buffering than CoDel, as PIE would tend to set the average queuing delay to its target, whereas CoDel acts as a queuing delay limiter.

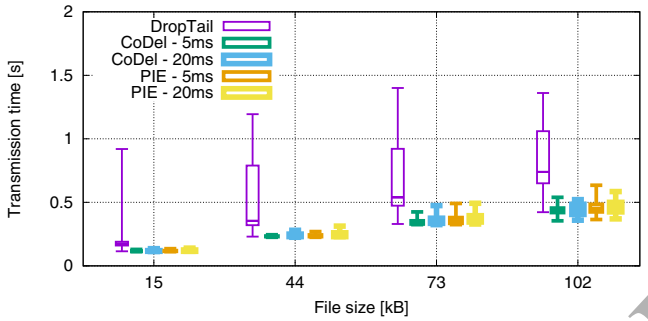
Furthermore, the results presented in this section let us conclude that both CoDel and PIE enable a significant latency reduction that can achieve an increase in the quality of experience for this kind of traffic. These results also confirm the correlation between the traffic characteristics and CoDel's performance and the fact that PIE is easy to tune.

6.3. Small files download - between snd_{SF} and $dest_{SF}$

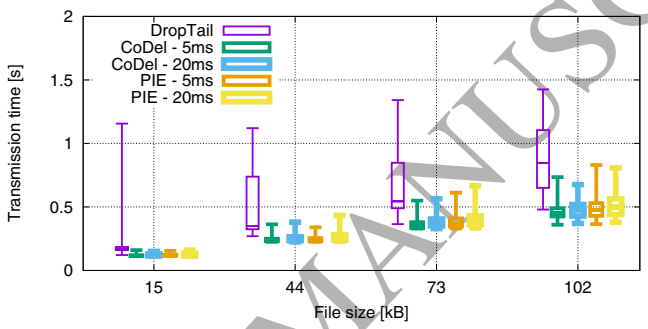
Figure 12 shows the transfer time of small files as a function of the AQM chosen and the file size. The box-plots show the 5^{th}, the 25^{th}, the 75^{th} and the 95^{th} percentiles and the median value.

With DropTail, this transfer time is much longer than with any AQM. Indeed, even if we can expect a better bottleneck utilization without any AQM, such gain comes at the cost of high queuing delay. As one example, with a queuing delay of 100 ms, the experienced RTT with DropTail is 200 ms, whereas it is ≤ 120 ms with both CoDel and PIE. This can be achieved without causing a slower congestion window progression at the beginning of the communication, but also in a slower reaction to congestion losses. These results clearly indicate that AQM is needed to guarantee a lower web page download time.

The transfer time with the different parameterizations of CoDel and PIE



(a) Traffic case 1



(b) Traffic case 2

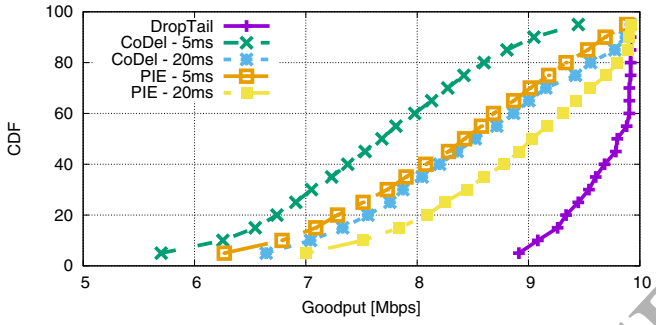
Figure 12: Small file download time

are similar. We attribute the small difference to come from (1) PIE allowing more buffering (and a higher queuing delay) than CoDel and (2) the bottleneck utilization being momentarily higher with PIE than with CoDel (see § 6.4). As the state of the queue evolves during one simulation (full, empty, queuing delay close to the target, etc.), the current state impact the download time of a given small flow.

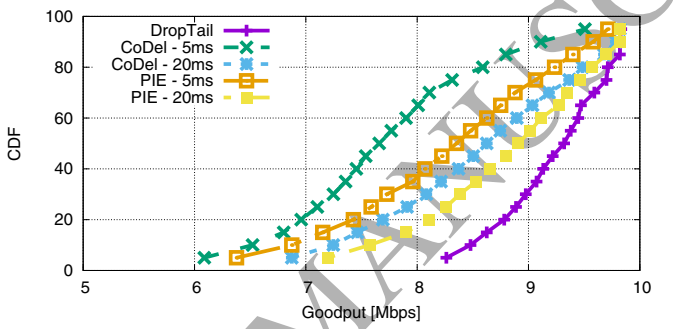
6.4. Bulk flows - between snd_{FTP} and $dest_{FTP}$

We show in Figure 13 the CDF of the goodput measured at node $dest_{FTP}$ with and without AQM and for both traffic cases 1 and 2.

Consistent with intuition, the goodput with DropTail is higher than with



(a) Traffic case 1



(b) Traffic case 2

Figure 13: Goodput of the bulk flow

any AQM. The goodput reduction resulting from the introduction of AQM is the cost that enables the latency reduction shown in § 6.2 and § 6.3.

The results shown in this figure confirm our earlier analysis that identified the different behaviours for both CoDel and PIE: when these schemes have the same target value, the goodput is higher with PIE than with CoDel. PIE acts as an average queuing delay controller and CoDel as a queuing delay limiter. Our results also show a relationship between target delay and goodput. Both schemes achieve a higher goodput when their target delay is increased.

7. Related work

Over the last few years, academic research on AQM has been quite active. It is thus important to assess how the work presented in this article fits in the past and current efforts in this area.

7.1. Early AQMs in the 90's

In 1993, RED [7] was designed mostly to keep the queue short while allowing occasional bursts of packets. The rationale behind such approach was to limit the global synchronization that results in low channel utilization, a bad use of the network resources and low quality of experience.

However, the internal parameters of RED made it hardly deployable by default without specific parameterizations that are linked to the deployment use case. Thus, there are many studies that (1) update RED for specific use cases, or (2) trying to optimize its parameterization, or (3) its algorithm to achieve specific goals [23, 24, 25, 26, 8, 27, 28]. An extensive list of these variants can be found in [29].

Also, based on the concept of dropping packets before the queue is full, many AQM have been proposed in the last decade, such as BLUE [30], CHOKe [31], PI [32] or RIO [33]. A survey [29] proposes a classification where these AQMs are classified between heuristic approaches, control-theoretic approaches and deterministic optimization approach.

7.2. The bufferbloat: an issue that raises the interest for new AQM schemes

Despite this amount of academic activities on the interest on deploying AQMs, the AQMs have been reported to be usually turn off, even though they can be available in network components. Indeed, a vendor may hardly sell an hardware with AQM enabled by default when the parameterization can depend on the deployment use-case or the characteristics of the lower layer. However, the need for deployed AQMs becomes essential, with the increasing number of latency sensitive applications and the non neglectable amount of buffering in queues.

In the light of these deployment issues and the need for parameterless AQM, CoDel and PIE have been proposed to tackle bufferbloat. While PIE extends the past work on control-theoric approaches, CoDel's deterministic approach is quite newer and based on the distinction between "good" and "bad" queues. On top of comparing their performance with previous AQMs, the actual performance of CoDel and PIE shall be looked at with this deployment issue in mind, which is what is assessed in this paper. This is essential information since these schemes are about to be standardized at IETF.

7.3. Evaluations of CoDel and PIE

As soon as their code was available, CoDel and PIE have been evaluated in distinct papers that use various evaluation toolset [34, 11, 12, 35]. The objective of this paper is slightly different, since the scope is to identify the network configuration under which CoDel and PIE have struggles with working by default. Moreover, it is worth pointing out our results are consistent with those of [13, 20, 14, 11], and this let us have confidence in the working areas delimited in this paper.

This work also identifies that there is no AQM that is optimized to work on every use-case. This confirms the results of [36] and joins the discussion provided in [34]. The working areas that have been identified in this paper have let us improve CoDel to work on capacity limited-networks [37] or PIE on long RTT networks [38].

8. Conclusion

Deploying AQM is an essential step in reducing end-to-end latency [6]. Widespread deployment of early AQM proposals has not happened yet, mainly due to their sensitivity to the network characteristics. Two recently proposed AQM schemes attempt to tackle bufferbloat while addressing these issues. Before considering actual deployment of CoDel and PIE, it is essential to identify their operating ranges (in terms of RTT, bottleneck capacity and congestion

level). We provide the first evaluation of the limits of PIE's self-tuning ability and the sensitivity of CoDel to network conditions. Our conclusions, based on ns-2 simulations, are consistent with recently published studies that emulates networks using Linux implementations of the algorithms. We found that both schemes show performance issues when the RTT is higher than 200 ms, or the bottleneck capacity is lower than 2 Mbps. By considering a traffic mix, we also analyze how the default parameterization of PIE and CoDel lets PIE allow more queuing delay than CoDel and achieve higher bottleneck utilization.

Because the use of CoDel and PIE with their default parameterizations results in poor performance for some deployment use-cases, such as data-centers or rural broadband networks, and this shows that manual tuning can hardly be avoided. Also, within the operating ranges, the desirable performance may differ from the trade-off that can be obtained with the default parameters of CoDel and PIE. As a result, we have also studied to what extent these algorithms can be manually tuned for these use-cases. We saw that the median queuing delay can easily be modified with PIE and CoDel by changing the target delay only; however, this works for CoDel only when the congestion level is low. Setting the update interval to the RTT reduces the RTT sensitivity of CoDel, but in many deployment cases, the person configuring the router will not be aware of the path RTT. Increasing the update interval tends to result in smoother evolution of the drop probability for PIE. The paper therefore identifies the need to tune parameters in these network scenarios, and has identified which parameters need to be tuned to achieve acceptable performance.

Acknowledgment

This work was part-funded by the European Community under its Seventh Framework Programme through the Reducing Internet Transport Latency (RITE) project (ICT-317700). The views expressed are solely those of the authors.

References

[1] J. Gettys, Bufferbloat: Dark Buffers in the Internet, Internet Computing, IEEE 15 (3) (2011) 96–96. doi:10.1109/MIC.2011.56.

[2] M. Allman, Comments on Bufferbloat, SIGCOMM Comput. Commun. Rev. 43 (1) (2012) 30–37. doi:10.1145/2427036.2427041.
URL http://doi.acm.org/10.1145/2427036.2427041

[3] O. Hohlfeld, E. Pujol, F. Ciucu, A. Feldmann, P. Barford, A QoE Perspective on Sizing Network Buffers, in: Proceedings of the 2014 Conference on Internet Measurement Conference, IMC '14, ACM, New York, NY, USA, 2014, pp. 333–346. doi:10.1145/2663716.2663730.
URL http://doi.acm.org/10.1145/2663716.2663730

[4] H. Jiang, Z. Liu, Y. Wang, K. Lee, I. Rhee, Understanding Bufferbloat in Cellular Networks, in: Proceedings of the 2012 ACM SIGCOMM Workshop on Cellular Networks: Operations, Challenges, and Future Design, CellNet '12, ACM, New York, NY, USA, 2012, pp. 1–6. doi:10.1145/2342468.2342470.
URL http://doi.acm.org/10.1145/2342468.2342470

[5] S. Alfredsson, G. Del Giudice, J. Garcia, A. Brunstrom, L. De Cicco, S. Mascolo, Observations of Bufferbloat in Swedish Cellular Networks, in: Proceedings of the 9th Swedish National Computer Networking Workshop (SNCNW 2013), 2013.

[6] F. Baker, G. Fairhurst, IETF Recommendations Regarding Active Queue Management, no. 7567, RFC Editor, 2015.

[7] S. Floyd, V. Jacobson, Random Early Detection Gateways for Congestion Avoidance, IEEE/ACM Trans. Netw. 1 (4) (1993) 397–413. doi:10.1109/90.251892.
URL http://dx.doi.org/10.1109/90.251892

[8] S. Floyd, R. Gummadi, S. Shenker, Adaptive RED: An Algorithm for Increasing the Robustness of RED's Active Queue Management, Tech. rep. (2001).

[9] K. Nichols, V. Jacobson, Controlling Queue Delay, Queue 10 (5) (2012) 20:20–20:34. doi:10.1145/2208917.2209336.
URL http://doi.acm.org/10.1145/2208917.2209336

[10] R. Pan, P. Natarajan, C. Piglione, M. Prabhu, V. Subramanian, F. Baker, B. VerSteeg, PIE: A lightweight control scheme to address the Bufferbloat problem, in: IEEE HPSR, 2013.

[11] I. Jarvinen, M. Kojo, Evaluating CoDel, PIE, and HRED AQM techniques with load transients, in: Local Computer Networks (LCN), 2014 IEEE 39th Conference on, 2014, pp. 159–167. doi:10.1109/LCN.2014.6925768.

[12] N. Khademi, D. Ros, M. Welzl, The new AQM kids on the block: An experimental evaluation of CoDel and PIE, in: Computer Communications Workshops (INFOCOM WKSHPS), 2014 IEEE Conference on, 2014, pp. 85–90. doi:10.1109/INFCOMW.2014.6849173.

[13] J. Schwardmann, D. Wagner, M. Kühlewind, Evaluation of ARED, CoDel and PIE, 20th Eunice Open European Summer School and Conference (2014).

[14] E. Grigorescu, C. Kulatunga, G. Fairhurst, Evaluation of the impact of packet drops due to AQM over capacity limited paths, in: Network Protocols (ICNP), 2013 21st IEEE International Conference on, 2013, pp. 1–6. doi:10.1109/ICNP.2013.6733658.

[15] T. Hoeiland-Joergensen, P. McKenney, D. Taht, J. Gettys, E. Dumazet, FlowQueue-Codel, 2015, IETF.
URL `https://tools.ietf.org/html/draft-ietf-aqm-fq-codel-01`

[16] N. Kuhn, P. Natarajan, D. Ros, N. Khademi, AQM Characterization Guidelines, 2015, IETF.
URL `https://tools.ietf.org/html/draft-ietf-aqm-eval-guidelines-11`

[17] D. Hayes, D. Ros, L. Andrew, S. Floyd, Common TCP Evaluation Suite, 2014, IRTF.
URL `https://tools.ietf.org/html/draft-irtf-iccrg-tcpeval-01`

[18] K. Nichols, V. Jacobson, A. McGregor, J. Iyengar, Controlled Delay Active Queue Management, 2015, IETF.
URL `https://tools.ietf.org/html/draft-ietf-aqm-codel-01`

[19] SamKnows - Commission for Rural Communities, Mind the Gap: Digital England – A Rural Perspective, 2014.

[20] N. Khademi, D. Ros, M. Welzl, The New AQM Kids on the Block: Much Ado About Nothing?, no. 434, 2014.

[21] Greg White, CableLabs, Active Queue Management Algorithms for DOCSIS 3.0: A Simulation Study of CoDel, SFQ-CoDel and PIE in DOCSIS 3.0 Networks, Cable Television Laboratories.
URL http://www.cablelabs.com/wp-content/uploads/2013/11/Active_Queue_Management_Algorithms_DOCSIS_3_0.pdf

[22] R. Pan, P. Natarajan, C. Piglione, M. S. Prabhu, V. Subramanian, F. Baker, B. VerSteeg, PIE: A lightweight control scheme to address the bufferbloat problem. Further Studies– PIE for Data Centers, in: IETF 86, 2013.

[23] O. Yelbasi, E. Germen, A new approach to estimate red parameters using global congestion notification, in: Network Computing and Information Security (NCIS), 2011 International Conference on, Vol. 2, 2011, pp. 15–19. doi:10.1109/NCIS.2011.101.

[24] C. Wang, J. Liu, B. Li, K. Sohraby, Y. T. Hou, Lred: A robust and responsive aqm algorithm using packet loss ratio measurement, IEEE Transactions on Parallel and Distributed Systems 18 (1) (2007) 29–43. doi:http://doi.ieeecomputersociety.org/10.1109/TPDS.2007.14.

[25] D. Lin, R. Morris, Dynamics of random early detection, in: Proceedings of the ACM SIGCOMM '97 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication, SIGCOMM '97, ACM, New York, NY, USA, 1997, pp. 127–137. doi:10.1145/263105.263154.
URL http://doi.acm.org/10.1145/263105.263154

[26] H. J. Chao, X. Guo, Quality of Service Control in High-Speed Networks, John Wiley & Sons, Inc., New York, NY, USA, 2001.

[27] S. Liu, T. Basar, R. Srikant, Exponential-red: a stabilizing aqm scheme for low- and high-speed tcp protocols, IEEE/ACM Transactions on Networking 13 (5) (2005) 1068–1081. doi:10.1109/TNET.2005.857110.

[28] E. Lochin, B. Talavera, Managing network congestion with a kohonen-based red

queue, in: Communications, 2008. ICC '08. IEEE International Conference on, 2008, pp. 5586–5590. doi:10.1109/ICC.2008.1047.

[29] R. Adams, Active queue management: A survey, IEEE Communications Surveys Tutorials 15 (3) (2013) 1425–1476. doi:10.1109/SURV.2012.082212.00018.

[30] W.-c. Feng, K. G. Shin, D. D. Kandlur, D. Saha, The blue active queue management algorithms, IEEE/ACM Trans. Netw. 10 (4) (2002) 513–528. doi:10.1109/TNET.2002.801399.
URL http://dx.doi.org/10.1109/TNET.2002.801399

[31] R. Pan, B. Prabhakar, K. Psounis, Choke - a stateless active queue management scheme for approximating fair bandwidth allocation, in: INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE, Vol. 2, 2000, pp. 942–951 vol.2. doi:10.1109/INFCOM.2000.832269.

[32] Y. Hong, O. W. W. Yang, Self-tuning pi rate controller for aqm router supporting best-effort traffic, in: Communications, 2005. ICC 2005. 2005 IEEE International Conference on, Vol. 3, 2005, pp. 1539–1543 Vol. 3. doi:10.1109/ICC.2005.1494602.

[33] D. D. Clark, W. Fang, Explicit allocation of best-effort packet delivery service, IEEE/ACM Transactions on Networking 6 (4) (1998) 362–373. doi:10.1109/90.720870.

[34] N. Kuhn, E. Lochin, O. Mehani, Revisiting Old Friends: Is CoDel Really Achieving What RED Cannot?, in: ACM SIGCOMM Workshop CSWS, 2014.

[35] J. Schwardmann, D. Wagner, M. Kühlewind, Advances in Communication Networking: 20th EUNICE/IFIP EG 6.2, 6.6 International Workshop, Rennes, France, September 1-5, 2014, Revised Selected Papers, Springer International Publishing, Cham, 2014, Ch. Evaluation of ARED, CoDel and PIE, pp. 185–191.
URL http://dx.doi.org/10.1007/978-3-319-13488-8_17

[36] A. Sivaraman, K. Winstein, S. Subramanian, H. Balakrishnan, No Silver Bullet: Extending SDN to the Data Plane, in: HotNets-XII, 2013.

[37] C. Kulatunga, N. Kuhn, G. Fairhurst, D. Ros, Tackling bufferbloat in capacity-limited networks, in: Networks and Communications (EuCNC), 2015 European Conference on, 2015, pp. 381–385. doi:10.1109/EuCNC.2015.7194103.

[38] N. Kuhn, D. Ros, Improving pie's performance over high-delay paths, CoRR abs/1602.00569.
URL http://arxiv.org/abs/1602.00569