Computer Networks 89 (2015) 90–106

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

# The Good, the Bad and the WiFi: Modern AQMs in a residential setting

Toke Høiland-Jørgensen*, Per Hurtig, Anna Brunstrom

*Department of Mathematics and Computer Science, Karlstad University, 651 88 Karlstad, Sweden*

ARTICLE INFO

*Article history:*
Received 16 February 2015
Revised 4 July 2015
Accepted 28 July 2015
Available online 31 July 2015

*Keywords:*
Active queue management
Fairness queueing
Bufferbloat
Latency
Performance measurement
Wireless networks

ABSTRACT

Several new active queue management (AQM) and hybrid AQM/fairness queueing algorithms have been proposed recently. They seek to ensure low queueing delay and high network goodput without requiring parameter tuning of the algorithms themselves. However, extensive experimental evaluations of these algorithms are still lacking. This paper evaluates a selection of bottleneck queue management schemes in a test-bed representative of residential Internet connections of both symmetrical and asymmetrical bandwidths as well as WiFi. Latency under load and the performance of VoIP and web traffic patterns are evaluated under steady state conditions. Furthermore, the impact of the algorithms on fairness between TCP flows with different RTTs, and also the transient behaviour of the algorithms at flow startup is examined. The results show that while the AQM algorithms can significantly improve steady state performance, they exacerbate TCP flow unfairness. In addition, the evaluated AQMs severely struggle to quickly control queueing latency at flow startup, which can lead to large latency spikes that hurt the perceived performance. The fairness queueing algorithms almost completely alleviate the algorithm performance problems, providing the best balance of low latency and high throughput in the tested scenarios. However, on WiFi the performance of all the tested algorithms is hampered by large amounts of queueing in lower layers of the network stack inducing significant latency outside of the algorithms' control.



## 1. Introduction

Ensuring low latency, and in particular *consistently* low latency, in modern computer networks has become increasingly important over the last several years. As more interactive applications are deployed over the general Internet, this trend can be expected to continue. Several factors can contribute to unnecessary latency (for a survey of such factors, see [1]); in this paper we focus on the important factor of excessive queueing delay, particularly when the network is congested.

Recent re-emergence of interest in the problem of congestion-induced excessive queueing latency has, to a large extent, been driven by the efforts of the bufferbloat community [2,3], which has also worked to develop technical solutions to mitigate it. In short, bufferbloat is a term used to describe the effect that occurs when a network bottleneck is congested and large buffers fill up *and do not drain*, thus inducing a persistent queueing delay that can be much larger than the path round-trip time. Since the inception of the bufferbloat community effort, more and more people in both academia and industry are becoming aware of the problem; and several novel queue management schemes have been proposed to combat the problem.

These new queue management schemes seek to provide both low latency and high goodput, without requiring

* Corresponding author. Tel.: +46547001611.
*E-mail addresses:* toke.hoiland-jorgensen@kau.se (T. Høiland-Jørgensen), per.hurtig@kau.se (P. Hurtig), anna.brunstrom@kau.se (A. Brunstrom).

http://dx.doi.org/10.1016/j.comnet.2015.07.014
1389-1286/

the extensive parameter tuning that was needed for earlier schemes like Random Early Detection (RED) [4]. The schemes include new active queue management (AQM) algorithms, such as Controlled Delay (CoDel) [5] and Proportional Integral Controller Enhanced (PIE) [6]. In addition, the older Adaptive RED (ARED) [7] algorithm has seen revival attempts for this use.

Most previous evaluations of these algorithms have been based on simulation studies. We extend this by comparing more algorithms (seven in total), both pure AQM algorithms and fairness queueing scheduling algorithms. In addition, we examine more traffic scenarios and application behaviours. Finally, we provide an updated examination of actual running code (the Linux kernel, version 3.14), which, due to the wide availability and open nature of the code, can be considered a real-world reference implementation for the algorithms. For all experiments, we provide access to the experimental data, and the tools to replicate them, online.[1]

We present our analysis in three separate parts: the Good, the Bad and the WiFi. First, the Good: we compare steady state behaviour of the algorithms in a mix of traffic scenarios designed to be representative of a residential Internet setting: measuring latency under load, and real-world application performance of VoIP and HTTP applications, with minimal tuning of the algorithms applied. The tested algorithms perform significantly better than FIFO queueing in these scenarios.

Second, the Bad: we test the impact of the AQMs on fairness between TCP flows of unequal RTT, and analyse the transient behaviour of the algorithms when flows start up. We compare the goodput of four flows with RTTs varying almost two orders of magnitude. We find that the AQM algorithms exacerbate the tendency of unfairness between the TCP flows compared to FIFO queueing. We also look at the development of measured delay over time when competing TCP flows start up and start to claim bandwidth at the bottleneck link. This analysis shows that two of the AQM algorithms (PIE and CoDel) have severe issues in quickly controlling the induced delay, showing convergence times of several seconds with very high delay spikes when the flows start up.

Finally, the WiFi: recognising that wireless networks play an increasing role in modern residential networks, we evaluate the algorithms in a setup where a WiFi link constitutes part of the tested path. We find that the algorithms fail to limit latency in this scenario, and it is quite clear that more work is needed to effectively control queueing in wireless networks.

The analysis of these three aspects of AQM behaviour contributes to a better understanding of residential network behaviour. It points to several areas that are in need of further evaluation and more attention from algorithm developers. One possible solution that has been deployed with promising results [8] is fairness queueing, exemplified by algorithms such as Stochastic Fairness Queueing (SFQ) [9] or the hybrid AQM/fairness queueing of fq_codel [10]. Hence, we have included three such algorithms in our evaluations along with the AQM algorithms. We find that they give vastly superior performance when compared with both FIFO queueing and the tested AQM algorithms, making the case that these types of algorithms can play an important role in the efforts to control queueing delay.

The rest of the paper is structured as follows: Section 2 discusses related work. Section 3 presents the experimental setup and the tested path characteristics, and Section 4 describes the tested algorithms. Section 5 presents the measurements of steady-state behaviour and their results, while Section 6 does the same for the experiments with fairness and transient behaviour. Section 7 covers WiFi and finally, Section 8 concludes the paper and outlines future work.

## 2. Related work

A large number of AQM algorithms have been proposed over the last two decades, employing a variety of approaches to decide when to drop packets; for a comprehensive survey, see [11]. Similarly, several variants of fairness queueing have been proposed, e.g. [12–14]. We have limited our attention to those algorithms proposed as possible remedies to the bufferbloat problem over the last several years. This section provides an overview of previous work on evaluating these algorithms and their effectiveness in combating bufferbloat.

The first evaluations of the AQM algorithms in question were performed by their inventors, who all publish extensive simulation results comparing their respective algorithms to earlier work [5–7]. All simulations performed by the algorithm inventors examine queueing delay and throughput tradeoffs in various straightforward, mainly bulk, traffic scenarios. Due to being published at different times and with different simulation details, the results are not easily comparable, but overall, the authors all find that their proposed algorithms offer tangible improvements over the previously available algorithms.

In an extensive ns2-based simulation study of AQM performance in a cable modem setting [15], White compares CoDel, PIE and two hybrid AQM/fairness queueing algorithms, SFQ-CoDel and SFQ-PIE. Various traffic scenarios were considered, including gaming, web and VoIP traffic as well as bulk file transfers. The simulations focus specifically on the DOCSIS cable modem hardware layer, and several of the algorithms are adjusted to better accommodate this. For instance, the PIE algorithm has more auto-tuning intervals added, and the fairness queueing algorithms have the number of queues decreased. The study finds that all three algorithms offer a marked improvement over FIFO queueing. The study concludes that PIE offers slightly better latency performance than CoDel but has some issues with bulk TCP traffic. Finally, the study finds that SFQ-CoDel and SFQ-PIE offer very good performance in many cases, but note some issues in specific scenarios involving many BitTorrent flows.

Khademi et al. [16] have performed an experimental evaluation of CoDel, PIE and ARED in a Linux testbed. The experiments focus on examining the algorithms at a range of parameter settings and measure bulk TCP transfers and the queueing delay experienced by the packets of the bulk TCP flows themselves. The paper concludes that ARED is comparable to PIE and CoDel in performance.

Rao et al. [17] perform an analysis of the CoDel algorithm combined with a simulation study that compares it to the

[1] http://www.cs.kau.se/tohojo/good-bad-wifi/.