# Managing Fairness and Application Performance with Active Queue Management in DOCSIS-based Cable Networks

James Martin
School of Computing
Clemson University
Clemson, South Carolina
jim.martin@cs.clemson.edu

Gongbing Hong
School of Computing
Clemson University
Clemson, South Carolina
hgb.bus@gmail.com

James Westall
School of Computing
Clemson University
Clemson, South Carolina
westall@cs.clemson.edu

## ABSTRACT

We evaluate modern delay-based AQM algorithms in downstream DOCSIS 3.0 cable environments. Our focus is on fairness and application performance capabilities of two recently proposed delay-based AQM algorithms, CoDel and PIE. The evaluation involves scenarios that include tiered service levels and application workloads that include FTP, HTTP-based adaptive streaming, and VoIP traffic. Our results provide a snapshot of our current effort to evaluate AQM schemes that are likely to be deployed in emerging DOCSIS cable networks.

## Categories and Subject Descriptors

C.2.2 [**Network Protocols**]: Protocol Architecture

## Keywords

Cable Access; Packet Scheduling Algorithms; Network Performance Evaluation;

## 1. INTRODUCTION

The evolution of cable network technology is at an intriguing crossroads. Traditional broadcast video is converging with Internet video broadcast. Network operators must engineer their access networks to competitively support broadband Internet access which might be at the expense of losing traditional broadcast subscribers. Further fueling this rapid evolution is the rate at which cable access technology is advancing. Current generation systems based on Data-Over-Cable Service Interface Specifications (DOCSIS) version 3.0 can support multiple downstream, fixed-size channels bonded together and multiple upstream channels bonded together [1]. A common configuration involves 8 bonded downstream channels and 4 bonded upstream channels which can support data rates up to 320 Mbit/s downstream and 120 Mbit/s upstream. Emerging systems will be based on the recently released DOCSIS 3.1 (D3.1) standard [2] which increases data rates by an order of magnitude.

Scheduling and queue management disciplines are fundamental to computer networking and have been studied from many different perspectives. In spite of this body of knowledge, access networks can still suffer from known problems such as bufferbloat, TCP RTT unfairness, and vulnerability to unresponsive flows [3-6]. Cable access presents additional challenges including: 1) network neutrality issues require transparent bandwidth management; 2) packet scheduling that operates at the cable modem requires low complexity queue management.

Two active queue management (AQM) algorithms have been proposed in recent research: Controlled Delay (CoDel) [4,7] and Proportional Integral Controller Enhanced (PIE) [8]. The D3.1 standard requires that cable modems manage upstream best effort traffic using PIE. The D3.0 standard has been modified to indicate that PIE is recommended [9]. For downstream traffic, a cable 'head-end' device (called a Cable Modem Termination System or CMTS) must support a published AQM algorithm for both D3.0 and D3.1, not necessarily PIE.

We have developed an ns2-based simulation model of DOCSIS 3.0[1]. In prior work, we have verified the basic operation of the DOCSIS model [10]. Our analysis framework utilizes realistic scenarios including FTP, VoIP, and HTTP-based adaptive streaming (HAS). The results presented in this paper address the following questions. 1) How effectively do CoDel and PIE support fairness and application performance in realistic cable network scenarios? 2) Are there issues when the AQM interacts with tiered service levels? 3) How effectively do the schemes isolate responsive traffic from unresponsive flows?


*CSWS'14,* August 18 2014, Chicago, IL, USA

http://dx.doi.org/10.1145/2630088.2630092

[1] An extended version of this paper along with the ns2 CoDel and PIE source code that was used in this study is available at http://www.cs.clemson.edu/~jmarty/AQM/AQMPaper.html.

This paper is organized as follows. Section 2 summarizes relevant research that has been published in the literature. Section 3 introduces the experimental methodology. Section 4 presents the analysis. Section 5 provides conclusions and identifies our next steps.

## 2. RELATED WORK

Bufferbloat has received significant attention recently [3,4]. Bufferbloat is caused by network equipment that is provisioned and configured with large unmanaged buffer queues, and its effects are persistently large queue levels that in turn lead to large and sustained packet latency. AQM has long been considered a solution to bufferbloat. The Random Early Detection (RED) algorithm manages a queue actively by randomly dropping packets in a manner that introduces a feedback control loop that adjusts the random drop rate dynamically based on an average queue size estimate and a configured maximum allowed drop rate (referred to as *maxp*) [6].

While RED is widely deployed, it is not widely used. It has been shown that the average queue delay with RED is sensitive to traffic loads and to parameter settings [11]. Adaptive RED (ARED) is a simple extension to RED that adapts the random drop process such that the average queue level does not exceed a target queue level [12]. This adaptation is performed periodically (we refer to this parameter as the *control_interval*).

CoDel and PIE are two recently proposed AQM algorithms that specifically address RED issues. Both AQMs are delay-based as they proactively drop packets to maintain an average packet queue delay less than a *target_delay* parameter. Both AQMs expose a second configuration parameter that defines the timescale of control (i.e., a *control_interval* parameter). CoDel's delay estimate is based on a per packet latency monitor. PIE's delay estimate is based on an estimate of the recent departure rate at the queue. The two AQMs conceptually deal with bursts in a similar manner by tolerating infrequent bursts of traffic. However, the details of exactly how burst control is performed are different. CoDel assumes early packet drops are performed as queued packets are serviced. PIE assumes early drops are done as packets arrive at the queue. The DOCSIS vendor community has expressed concern in implementing a head-drop AQM in a CM due to complications with hardware buffer control logic. The concern is likely to be true in other network devices (switches or WiFi APs). Our long term research goal is to identify an AQM that offers predictable service over a broad range of networking technologies. Therefore, we limit the scope of the research presented in this paper to low complexity AQM schemes.

### 2.1 Overview of DOCSIS Cable Networks

Modern broadband Internet access over cable involves a hybrid-fiber coaxial (HFC) infrastructure with a DOCSIS MAC protocol operating between the CMTS and the subscriber cable modems (CMs). Downstream operation is based on time division multiplexing. A downstream scheduler operating at the CMTS manages the allocation of bandwidth among competing service flows. Upstream operation is a shared bus relying on a hybrid contention-based request MAC protocol for managing access to the channel by competing cable modems. The centralized upstream scheduler operating at the CMTS allocates periodic grants to modems (i.e., the unsolicited grant service) or the CM requests a bandwidth allocation using either a request piggy-backed to a previously allocated transmission or a contention-based request mechanism.

## 3. METHODOLOGY

Figure 1 illustrates the simulated network that we used in the analysis. We model a single DOCSIS cable domain where one CMTS interacts with a number of cable modems. Each cable modem represents a subscriber that is likely to have multiple TCP/IP devices interacting with a variety of Internet services. The model assumes ideal channels that offer downstream and upstream link layer data rates of 38 Mbps and 30 Mbps respectively. While the simulation model supports downstream bonded channels, the results reported in this paper are limited to single channel scenarios. Our analysis considers scenarios with and without imposition of service rate limits. The regulator component of the simulator, shown in Figure 1, models a DOCSIS token bucket with an unlimited peak capacity. The service rate parameters are the desired maximum sustained rate and the bucket size (specified in bytes).

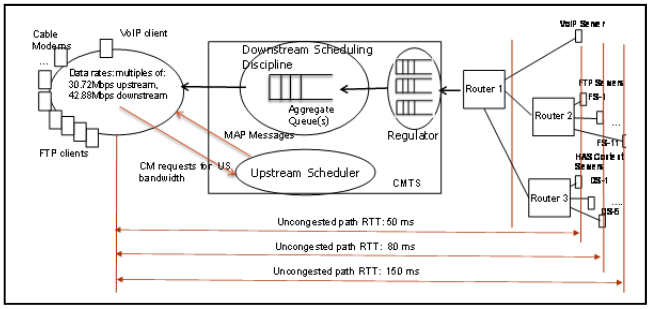


**Figure 1. Simulated Cable Network Diagram**

We analyze several scenarios using varying numbers of downstream application flows including FTP, VoIP, and Internet streaming based on HAS. The client side of the applications run on nodes attached to cable modems. The server side run on nodes located outside the cable network. For the results presented in this paper, we assume each subscriber runs a single application. To model Internet path diversity, we set the propagation delay of the application server access link to different values. As illustrated in Figure 1, the range of uncongested path round trip times is from 50 ms to 150 ms. In the simulations that are presented, all packets are 1500 bytes with the exception of VoIP traffic which is configured to send packets of size

238 bytes (this is explained in more detail in the next section). Assuming an available data rate of 38 Mbps and fixed packet size of 1500 bytes, this corresponds to bandwidth*delay products (BDP) ranging from 158 to 455 packets.

The simulations are designed such that the downstream queue managed by the CMTS is the only bottleneck. We identify a set of scenarios and analyze system performance when different downstream buffer management methods are used. For the results presented in this paper, the maximum buffer capacity of the bottleneck link, referred to as the QUEUE_CAPACITY, is set to 2048 packets. We set the maximum buffer capacity at all other routers to twice the QUEUE_CAPACITY. To provide a fairness reference point, we compare the results of single queue management to results based on an approximation of fair queuing using deficit round robin (DRR) [13]. For the results presented in this paper, each per flow queue was configured with a capacity of QUEUE_CAPACITY/16 where 16 is the largest number of active flows in any simulation. The selection of the QUEUE_CAPACITY along with specific AQM algorithm parameters that impact how much of the buffer is actually used greatly affects the results. For example, it is possible to configure ARED such that it maintains a target delay as long as the link data rate is known and remains constant. Tuning ARED to maintain a *target_delay* and *control_interval* similar to that used by CoDel and PIE will lead to similar results. However, the lesson the community has learned with RED is that an AQM algorithm must not require configuration tuning that might be specific for particular workloads or environments [14]. This requirement initiated interest in delay-based AQM where the only configuration parameter that might need adjustment is the delay target. Our methodology assumes each AQM's recommended default settings based on a specified QUEUE_CAPACITY. The buffer management scheme determines how it will use the buffer allocation.

All TCP flows use TCP/Sack with a receiver's maximum advertised window set to twice the *QUEUE_CAPACITY*. In all simulations, we verified that the upstream channel was never congested. Upstream traffic was limited to TCP acknowledgements and HTTP GET requests for the simulations that involved video streaming.

## 3.1 Traffic Models and Performance Metrics

The results presented in this paper involve workloads consisting of FTP, HAS, and UDP CBR traffic. We have developed a HAS traffic model in ns2. In a HAS system, video content is encoded and available at servers in multiple 'representations' that reflect a range of possible encoded bit rates, each corresponding to a different level of video quality. The client requests portions of the stream in chunks referred to as segments. We configured the segment size to be 12.5 seconds. The client monitors the arriving video stream and determines when to switch to a lower or higher quality stream. The client maintains a playback buffer that serves to smooth variable content arrival rates and requests one or more new segments (tagged with the desired bitrate) once the playback buffer drops below a certain threshold. HAS tends to produce bursty traffic patterns as the client requests segments at the available TCP data rate but then goes idle until the buffer reaches a threshold. For the results presented in this paper, we set the HAS playback buffer to hold up to 60 seconds of the video stream. The rate at which data is dequeued from the playback buffer is the *videoPlaybackRate*. The adaptation algorithm attempts to match the encoded content bit rate with an estimate of the available bandwidth. Our adaptation algorithm is straightforward: it drops the level of video content quality once it determines that the available bandwidth has dropped and then it restores the video quality once available bandwidth will support the higher bitrate stream. The time between adaptations cannot exceed a configured threshold (we used 100 seconds). We configured the server side of the HAS traffic model to include the following encoded bit rate choices (in Kbits per second): 64, 128, 500, 1000, 1500, 2600, 3500, 3750, and 4200. Since video quality is quite subjective, we quantify HAS performance by showing the average player video rate and the frequency of bit rate adaptations (as suggested by recent research [15,16]).

To assess the impact that high bandwidth applications might have on low bandwidth, latency sensitive flows, we establish a VoIP performance monitor between a server node and a cable modem. We estimate the call quality (referred to as the R-value) using the technique described in [17]. The R-value ranges from 0 to 100. Toll quality requires an R-value of 70 or higher. The metric assumes that 2 10 ms 'chunks' of digitized voice data are sent in an IP packet (of size 238 bytes) every 0.020 seconds.

We quantify fairness outcomes using Jain's Fairness Index (JFI) [18]. We assume that throughput allocation based on a max-min criteria is the desired fairness objective. An allocation of resources is max-min fair if it is not possible to increase the rate allocated to any user in the system without decreasing the rate allocated to any other user who is receiving an already lower or equal rate [19]. We define the normalized throughput for the *i'th* user among *n* users competing for channel capacity as $x_i = {}^{T_i}/_{r_i}$ where $T_i$ is the achieved throughput of the *i'th* flow and $r_i$ is the expected outcome based on a max-min criteria. The JFI is defined as: $JFI = \frac{[\sum_{i=1}^{n} x_i]^2}{n \sum_{i=1}^{n} x_i^2}, x_i \geq 0$. We also use the min-max ratio which is defined as: $\frac{Min\{x_i\}}{Max\{x_i\}}$. Both indexes

range from 0 to 1, with ideal fairness represented by a value of 1.

## 3.2 Experimental Definitions

Table 1 identifies four experiments that are presented in this paper. Each experiment involves sets of simulations that explore five different queue management schemes: drop tail, adaptive RED (ARED), CoDel, PIE, and DRR. **Drop tail (DT)** implies a single queue managed by a first-come-first-served (FCFS) scheduling discipline that drops packets that arrive to a full queue. **Adaptive RED (ARED)** is a simple extension to RED that adapts *maxp* such that the average queue level tracks a target queue level (we used the recommended target of *minth + (minth+maxth)/2)*. Configuration parameters are based on recommendations from [12]. The *minth* and *maxth* thresholds are the *QUEUE_CAPACITY* divided by 20 and by 2 respectively. The *control_interval* parameter is 0.50 seconds. The initial setting of *maxp* is 0.10.

**Table 1. Experiment Definition**

| Experiment ID | Summary |
|---|---|
| EXP1 | All flows not limited by service rates, vary the number of competing DS FTP flows |
| EXP2 | Same as EXP1 except add five HAS flows |
| EXP3 | Same as EXP1 except:<br>1)Set all flow services rates to 6 Mbps (refer to these flows as Tier 1 flows);<br>2)Add one additional competing FTP flow with a service rate of 12 Mbps that is active only during the time 500 – 1500 seconds. Refer to this flow as a Tier 2 flow. |
| EXP4 | Same as EXP1 except add a 12 Mbps DS UDP flow (starts at 500 seconds, stops at 1500 seconds) |

The original **CoDel** description recommends a *target_delay* parameter set to 0.005 [4]. We determined that a setting of 0.020 seconds provided more predictable results in our scenarios. The *control_interval* parameter is set to 0.100 seconds. The original CoDel algorithm proactively drops packets at dequeue time. Due to potential implementation issues of head dropping, we implemented a tail drop variant of CoDel where packets are proactively dropped at enqueue time. Our implementation is based on an ns2 tail-drop variant of CoDel[2]. PIE is also delay-based. We set the *target_delay* parameter to 0.015 seconds value and the *control_interval* parameter to 0.016 seconds. The *max_burst* parameter is 0.142 seconds. The internal burst control parameters, $\alpha$ and $\beta$, are set to 2.5 and 0.25 respectively. These settings are based on the recommended settings for CMs described in the D3.1 specification [2].

# 4. Analysis

We organize our analysis to follow the three questions posed in the introduction. First, we evaluate how effectively each AQM maintains fairness and application performance in scenarios that do not involve service rate limits. Second, we explore the impact of service rates. Third, we explore the impact of unresponsive flows.

## 4.1 How Effective are CoDel and PIE?

Figure 2a shows the aggregate throughput achieved by all competing downstream FTP flows for each simulation. For the results with a single flow, DT (identified as FCFS-Droptail in the figures) achieved the highest throughput (37.2 Mbps) and PIE achieved the lowest throughput of 31.2 Mbps. In order to maintain the *target_delay*, PIE and CoDel impose higher loss rates which can lead to reduced throughput compared to DT or ARED. Once multiple flows are involved, the mean throughput of all competing flows for each AQM converge to the same average throughput. However, for a specific AQM, the allocation to each flow is highly variable.

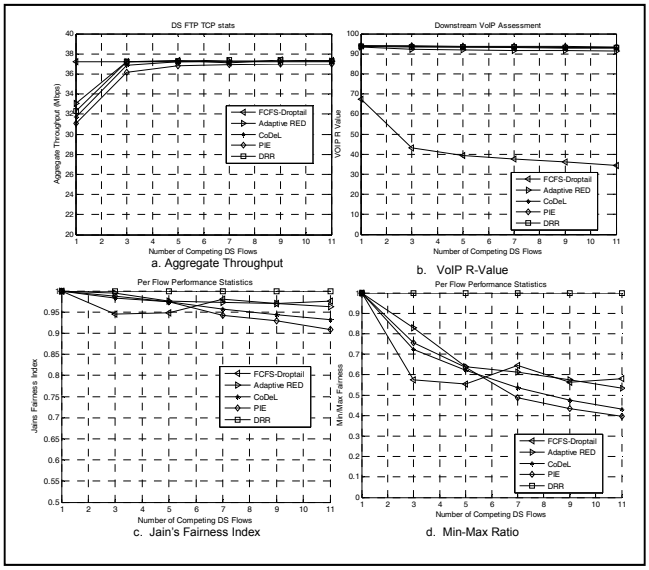


**Figure 2. EXP1 Results**

Figures 2c and 2d illustrate the fairness results. The JFI results confirm that DRR is max-min fair. The results suggest that DT allocations were the most variable for the scenarios involving 3 and 5 competing FTP flows. PIE and CoDel exhibited the highest level of variation as the number of competing flows increased to 11. Figure 2d supports this as the difference between the minimum and maximum TCP throughput achieved by the flows for each AQM (except DT) increases with the number of competing flows. The DT behavior is likely due to synchronization effects. For a given simulation, the achieved throughput for each flow is inversely proportional to the path RTT. In both the CoDel and PIE simulations, the TCP throughput over the shortest RTT path was 6.1 Mbps and the throughput over the longest RTT path was 3.1 and 2.7 Mbps respectively. PIE and CoDel appear to be the most sensitive to TCP/RTT unfairness. TCP/RTT unfairness is in part caused by short RTT path TCP flows starving longer path flows of buffer resource at a bottleneck. Since DT and

[2] The CoDel implementation used in this paper is based on the ns2 module contributed by CableLabs http://sourceforge.net/p/nsnam/ patches/24/

ARED use more of the buffer capacity, these schemes exhibit less RTT unfairness than the schemes that use only a fraction of available buffer capacity. To confirm this conjecture, we ran additional simulations (these results are not shown) setting the *target_delay* parameter for CoDel and PIE to 0.080 seconds. The fairness results were very similar to those of DT and ARED.

Figure 2b illustrates the ability of the AQM scheme to isolate greedy FTP flows from a loss and latency sensitive VoIP flow. The R-value for DT drops from 68 to 32 as the number of competing FTP flows increase from 1 to 11. When 11 flows are competing for bandwidth, ARED, CoDel, and PIE exhibit an R-value of 91.4, 92.75 and 93.5 respectively. In additional simulations, we found that the R-value remains above 85 as the number of competing DS FTP flows is increased to 70 for both CoDel and PIE.

Figure 3 illustrates the results of EXP2. Due to the adaptive nature of HAS, workloads consisting of HAS and FTP managed by DRR allocation might not be max-min fair. For the EXP2 results, we compute the expected max-min allocation for each simulation by assuming each HAS flow demand is 4.2 Mbps (the bandwidth required to support the highest quality video stream) and each FTP flow demand is effectively infinite. For the simulation with 1 competing FTP, the max-min allocation for the FTP flow is 17 Mbps. With 3 competing FTPs, the max-min allocation for each is 5.67 Mbps. In both examples, the max-min allocation for HAS is 4.2 Mbps. For all other simulations, both HAS and FTP flows should be allocated an equal share of bandwidth.

Figure 3a and 3b suggest greater deviation from max-min fair than what was observed in EXP1. As in EXP1, the DT results are different from the other results. Synchronization effects, especially in the simulations involving 1 and 3 competing FTP flows, likely contribute to the highly variable allocation. With the other schemes, the deviation from max-min fair allocation increases as the level of congestion increases. DT exhibits less allocation variability as the number of FTP flows increase. We believe this is an artifact of DT's use of larger buffer capacity.

As the network becomes more congested, we would expect an adaptive application such as HAS to react more aggressively to congestion than FTP. In additional analysis (not shown due to space limitations) we find the average throughput of all the HAS flows and then of all the FTP flows for each buffer management scheme. We quantify how far the average allocation deviate from the expected result by finding the percentage difference. We refer to this as the allocation error. Using the simulation that involved CoDel AQM and 9 FTP flows as an example, the average throughput of all the HAS and the FTP flows was 2.11 Mbps and 2.49 Mbps respectively. The computed max-min fair allocation, which is based on ideal allocation that involves no overhead, is 2.71 Mbps. This corresponds to an allocation error of 22% and 8% for HAS and FTP respectively. We compute the HAS allocation error for all results in EXP2 and find that DT, ARED, CoDel, PIE, and DRR achieved HAS allocations that were lower than the max-min fair allocation by 21.1%, 6.6%, 19.3%, 5.8%, 5.4% respectively.

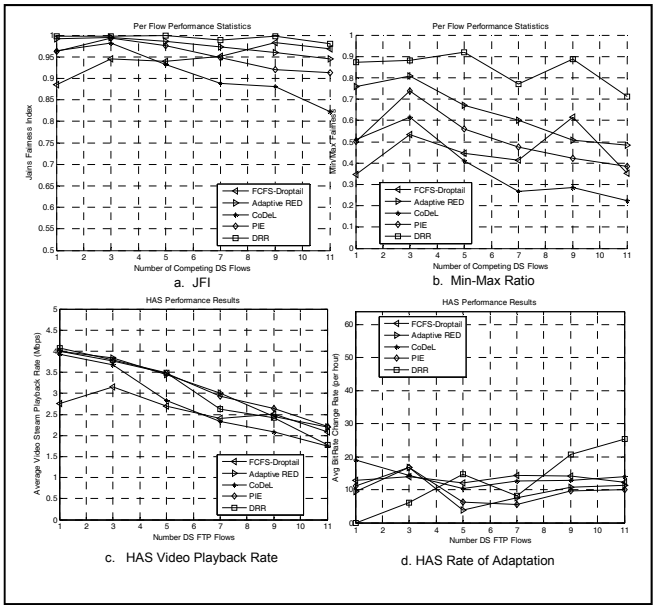


**Figure 3. EXP2 Results**

Figures 3c and 3d illustrate HAS application results. All AQMs lead to similar *videoPlaybackRate* results with the exception of DT at low levels of congestion. Fundamental to HAS is that it might chose to give up available bandwidth in hopes of enhancing the user perceived quality of experience. The HAS allocation error was highest for DT and this is reflected in the *videoPlaybackRate* result. The HAS specific metrics such as the average number of adaptations per hour and the frequency of rebuffering events (the latter results are not shown), do not clearly confirm if the quality perceived by a user is actually improved by adaptation.

## 4.2 Interaction with Service Rate Management

Table 2 summarizes the allocation obtained by Tier 1 and Tier 2 flows for the simulations. Each result shows the average throughput achieved by the Tier 1 and Tier 2 flows while the flows were active (i.e., Tier 2 flows are only active for 1000 seconds). The general result is that the Tier 2 allocation is larger than the max-min fair allocation. However, the Tier 2 allocation for CoDel and PIE does become lower than the max-min fair allocation as the level of congestion increases. This is likely due to the additional overhead caused by higher packet loss rates associated with CoDel and PIE.

**Table 2. EXP3 Results**

| Buffer Mgmt Scheme | 5 Tier 1 Flows Avg Flow Throughput (Mbps) | | 7Tier 1 Flows Avg Flow Throughput (Mbps) | | 9Tier 1 Flows Avg Flow Throughput (Mbps) | | 11 Tier 1 Flows Avg Flow Throughput (Mbps) | |
|---|---|---|---|---|---|---|---|---|
| | Tier 1 | Tier 2 | Tier 1 | Tier 2 | Tier 1 | Tier 2 | Tier 1 | Tier 2 |
| DT | 5.24 | 10.22 | 4.68 | 8.35 | 3.26 | 9.01 | 2.80 | 8.07 |
| ARED | 5.73 | 8.56 | 4.47 | 5.02 | 3.52 | 3.91 | 2.98 | 3.38 |
| CoDel | 5.70 | 8.24 | 4.26 | 5.94 | 3.54 | 4.10 | 2.90 | 2.98 |
| PIE | 5.63 | 8.13 | 4.21 | 4.64 | 3.51 | 3.62 | 2.87 | 2.75 |
| DRR | 5.92 | 7.79 | 4.68 | 4.65 | 3.74 | 3.72 | 3.12 | 3.10 |
| Max/min Fair Allocation | 6.0 | 8.0 | 4.75 | 4.75 | 3.8 | 3.8 | 3.17 | 3.17 |

## 4.3 Management of Unresponsive Flows

EXP4 is similar to EXP3 except that service rates are disabled and the higher speed (Tier 2) TCP flow is replaced with an unresponsive UDP CBR flow. The UDP flow was configured to send 1500 byte packets at a constant rate of 12 Mbps during the time period 500 to 1500 seconds of the 2000 second simulation. The results show that DRR consistently allocates the max-min allocation. For the other buffer management schemes, the unresponsive flow was allocated 12 Mbps and the FTP flows were allocated the remaining bandwidth.

# 5. CONCLUSIONS

In this paper, we explored fairness and application performance capabilities of five packet scheduling disciplines: FCFS drop tail, Adaptive RED, CoDel, PIE, and DRR. We focused on downstream queuing in a DOCSIS cable environment. CoDel and PIE are quite effective at maintaining the target queue delay and consequently better able to address bufferbloat than DT and ARED. We observed that CoDel and PIE are more sensitive to TCP/RTT unfairness than DT and ARED. Addressing bufferbloat at the expense of fairness might prove problematic in emerging converged broadcast networks that must deal with net neutrality issues. The current results suggest that the content providers that physically locate content closest to the subscriber might be allocated a larger share of access network resources compared to a content provider that does not have the resources to locate content at the access network. In future work we will develop low complexity AQM schemes, possibly involving multiple queues, to better address the combined problem of fairness and application performance management.

# 6. REFERENCES


[1] Cable Television Laboratories, Inc., Data-over-cable Service Interface Specifications DOCSIS 3.0, MAC and Upper Layer Protocols Interface Specification, April, 2014.

[2] Cable Television Laboratories, Inc., Data-over-cable Service Interface Specifications DOCSIS 3.1, MAC and Upper Layer Protocols Interface Specification, March, 2014.

[3] J. Gettys, “Bufferbloat: Dark Buffers in the Internet”, IEEE Internet Computing, Vol. 15, No. 3, 2011.

[4] K. Nichols, V. Jacobson, “Controlling Queue Delay”, ACM Queue , Vol 10, No 5, 2012.

[5] S. Floyd, K. Fall, “Promoting the Use of End-to-end Congestion Control in the Internet”, IEEE/ACM Transactions on Networking, Vol. 7, No.4 August 1999.

[6] S. Floyd, V. Jacobson, “Random Early Detection Gateways for Congestion Avoidance”, IEEE/ACM Transactions on Networking, Vol. 1, No.4, August 1993.

[7] K. Nichols, V. Jacobson, “Controlled Delay Active Queue Management”, IETF RFC Draft draft-nichols-tsvwg-codel-01, February 2013.

[8] R. Pan, P. Natarajan, C. Piglione, M. Prabhu, V. Subramanian, F. Baker, B. VerSteeg, “PIE: A Lightweight Control Scheme to Address the Bufferbloat Problem”, Proceedings of IEEE HPSR July 2013.

[9] G. White, D. Rice, “Active Queue Management in DOCSIS 3.x Cable Modems”, CableLabs Technical Report, May 2014. http://www.cablelabs.com/resources/publications

[10] J. Martin, J. Westall, “A Simulation Model of the DOCSIS Protocol”, Simulation: Transactions of the Society for Modeling and Simulation, Vol. 83, No. 2, 2007.

[11] M. May, J. Bolot, C. Diot, B. Lyles, “Reasons not to Deploy RED”, Proceedings of the 7th International Workshop on Quality of Service, June 1999.

[12] S. Floyd, R. Bummadi, S. Shenker, “Adaptive RED: An Algorithm for Increasing the Robustness of RED’s Active Queue Management”, Tech report, ICSI, 2001.

[13] M. Shreedhar, G. Varghese, “Efficient Fair Queuing Using Deficit Round-robin”, IEEE/ACM Transactions on Networking, Vol. 4, No. 3, June 1996.

[14] F. Baker, G. Fairhurst, “IETF Recommendations Regarding Active Queue Management draft-ietf-aqm-recommendation-04”, IETF draft available online at http://tools.ietf.org/html/draft-ietf-aqm-recommendation-04, May 2014.

[15] F. Dobrian, A. Awan, D. Joseph, A. Ganjamm J. Zhan, V. Sekar, I. Stoica, H. Zhang, "Understanding the Impact of Video Quality on User Engagement", Proceedings of SIGCOMM, August 2011.

[16] A. Balachandran, V. Sekar, A. Akella, S. Seshan, I. Stoica, H. Zhang, "A Quest for an Internet Video Quality-of-Experience Metric", Proc of ACM HotNets'12, Oct 2012.

[17] R. Cole, J. Rosenbluth, “Voice Over IP Performance Monitoring”, Proceedings of ACM SIGCOMM, April, 2001.

[18] R. Jain, D. Chiu, W. Hawe, “A Quantitative Measure of Fairness and Discrimination for Resource Allocation in Shared Computer Systems”, DEC Technical Report, 1984.

[19] Jean-Yves Le Boudec, "Rate Adaptation, Congestion Control and Fairness: A Tutorial," Ecole Polytechnique Federale de Lausanne(EPFL), Chapter 1, November 22, 2005