FASTER INTERNET PROTOCOLS

PRELIMINARY STUDY OF CODEL AQM IN A DOCSIS NETWORK

Prepared by:
Greg White
Principal Architect
g.white@cablelabs.com

Joey Padden
Architect
j.padden@cablelabs.com

CableLabs Project Lead:
Jean-François Mulé
jf.mule@cablelabs.com

CableLabs Program Lead:
Dan Rice
d.rice@cablelabs.com

November, 2012

DISCLAIMER

This document is published by Cable Television Laboratories, Inc. ("CableLabs®") to provide information to the cable industry. CableLabs reserves the right to revise this document for any reason including, but not limited to, changes in laws, regulations, or standards promulgated by various agencies; technological advances; or changes in equipment design, manufacturing techniques or operating procedures described or referred to herein. This document is prepared by CableLabs on behalf of its cable operator members to facilitate the rendering, protection, and quality control of communications services provided to subscribers.

CableLabs makes no representation or warranty, express or implied, with respect to the completeness, accuracy or utility of the document or any information or opinion contained in this document. Any use or reliance on the information or opinion is at the risk of the user, and CableLabs shall not be liable for any damage or injury incurred by any person arising out of the completeness, accuracy or utility of any information or opinion contained in this document.

This document is not to be construed to suggest that any manufacturer modify or change any of its products or procedures, nor does this document represent a commitment by CableLabs or any member to purchase any product whether or not it meets the described characteristics. Nothing contained herein shall be construed to confer any license or right to any intellectual property, whether or not the use of any information herein necessarily utilizes such intellectual property.

This document is not to be construed as an endorsement of any product or company or as the adoption or promulgation of any guidelines, standards, or recommendations.

ACKNOWLEDGMENTS

The authors would like to acknowledge and thank the inventor of the CoDel algorithm, Kathleen Nichols, Pollere, Inc., for her contributions to this study. In particular, for her development of the ns-2 system and traffic models and portions of the simulation code, for her contributions to the analysis and interpretation of the simulation results, and, perhaps most importantly, for her guidance in the application of the CoDel algorithm to address buffering latency in the DOCSIS 3.0 cable modem.

Table of Contents

List of Figures

List of Tables

EXECUTIVE SUMMARY

The "Controlled Delay" (CoDel) Active Queue Management technique is implemented in a simulated DOCSIS 3.0 Cable Modem, and its performance in a range of conditions is compared against a model of an existing DOCSIS 3.0 Cable Modem (which utilizes tail-drop queuing), both with and without use of the DOCSIS 3.0 Buffer Control feature.

In comparison to tail-drop queuing without Buffer Control, these preliminary results show that CoDel provides radically better performance than tail-drop for latency sensitive traffic, with VoIP MOS scores increasing by an average of 2.7 MOS points in loaded conditions, and web page load times dropping by nearly a factor of 10 in most of the test conditions. CoDel achieves this result with only a slight (0-2%) drop in TCP throughput.

In comparison to tail-drop queuing with the Buffer Control feature enabled and set to the CableLabs recommended 50 ms value, we see more of a mixed result. The TCP throughput results are unchanged, with tail-drop still providing slightly better results than CoDel. In VoIP MOS performance, tail-drop improved significantly, and provided equivalent performance to CoDel, with a 0.1-0.2 MOS point difference between the two in most cases. However, CoDel still provided markedly better web page load times, though in most cases, the magnitude of the improvement was diminished.

CoDel has advantages over tail-drop in that it does not need to be tuned to the subscriber's data rate, and further it automatically seeks a good operating point even when the subscriber data rate changes, for example in Power Boost situations, or conversely, episodes of upstream channel congestion.

1 INTRODUCTION

Simulation testing is used to study the impact of applying CoDel queue management in the upstream direction of cable modems (CMs). The approach is to create a model of the CM where the queue management can be either tail-drop or CoDel and where the upstream transmission opportunity grant rate can be varied to indicate congestion or lightly-loaded periods. The model is driven by traffic loads chosen to be representative of home usage patterns of particular interest in studying delay, or “bufferbloat”, issues: uploading a large file, over-the-top voice-over-IP (VoIP), and web browsing.

In the experiments, tail-drop (unmanaged) and CoDel managed queues will be compared and some variations in the application of CoDel will be tested. Results will focus on differences in packet delay, overall and per-flow throughput, and loss rates.

Note: the term "tail-drop" is more commonly used in general discourse to describe the queue management practice of discarding the newest arriving packet when the buffer is exhausted, but the ns-2 simulator calls its corresponding buffer model "DropTail". The terms are interchangeable. When referring to the simulation model, this paper uses the term DropTail, whereas when speaking more generally, the term "drop-tail" is used.

1.1 CODEL ACTIVE QUEUE MANAGEMENT

The CoDel AQM algorithm was developed to address the bufferbloat issue. It is intended to be a lightweight "knobless" algorithm that directly addresses the issue caused by bufferbloat: queuing latency. The CoDel algorithm was published into the public domain on May 2012 [CoDel], and the authors make an open source implementation available.

The core of the CoDel algorithm involves measuring the "sojourn-time" of each packet being forwarded by the device, where sojourn-time is defined as the total time that the packet takes to transit the device. When a packet is being de-queued at the head of the queue, CoDel makes a drop decision based upon the sojourn time of the packet, and the recent history of sojourn time.

The drop decision is based upon comparing the sojourn times to a target value of 5 ms. If, over the last 100 ms, sojourn times remain above 5 ms, CoDel enters a "drop state" in which it begins dropping packets. Upon entering the drop state, CoDel starts with a fairly low drop rate (drop 1 packet every 100 ms), but that rate increases with each packet drop as long as the sojourn time remains above the target latency.

The result is that CoDel allows applications such as TCP to fill the buffer in a network element as long as the queue build-up is short-lived and drains down to an empty (or nearly-empty) state. If the buffer doesn't drain on its own, CoDel sends TCP a congestion signal (via packet drops) to force it to drain.

1.2 PERFORMANCE EXPECTATIONS

In comparing DropTail w/Buffer Control to CoDel, expectations are that DropTail with an appropriately sized buffer (e.g., resulting in a maximum queuing latency of ~50 ms at the maximum sustained traffic rate (MSR)) will provide good performance in a variety of scenarios. In particular:

1. TCP should be able to maintain a steady-state transfer at the MSR.
2. When the upstream is uncongested and the CM is transferring TCP data at the MSR, the buffer will result in a consistent 40-50 ms of latency, which in most cases will only slightly degrade VoIP quality, and will result in on the order of N * 50 ms of additional page load time, where N is the number of serialized object fetches required to load a page (on the order of 15-20).

3. When the CM is not loaded with TCP upload traffic, the buffer will remain largely empty and upstream traffic will experience low latency.

However, when either the upstream channel is congested or the CM is bursting at the Peak Traffic Rate, the fixed buffer size of the DropTail queue will result in its being mismatched to the data rate, in one case larger than desired, and in the other case smaller. More specifically:

1. When the upstream is congested, the now oversized buffer will result in excessive latency that will degrade user experience for latency sensitive traffic beyond what is caused by the congestion alone.
2. When the upstream is uncongested, and the Peak Traffic Rate and Max Traffic Burst are set in such a way as to provide a significant data burst (*a la* PowerBoost), the buffer will be significantly undersized, which may result in an inability to achieve the Peak Traffic Rate.
3. Furthermore, if Buffer Control is not configured by the operator, then the DropTail buffer size could be highly mismatched to the data rate, resulting in extremely poor performance during loaded conditions. Historically, queuing latencies on the order of seconds have been reported, which will severely degrade latency sensitive applications (such as VoIP, real-time gaming, and web-browsing).

Also, TCP will seek to keep the DropTail queue completely full (in fact it will constantly ramp up its sending rate until the full buffer causes a packet drop, signaling TCP to temporarily slow down). As a result, when one or more TCP sessions are running, the buffer has very little remaining capacity for other flows to utilize.

CoDel is expected to match DropTail performance in cases where DropTail performs well, and exceed DropTail performance in the scenarios described above, which are troublesome for DropTail.

2 SIMULATION MODEL

2.1 OVERVIEW

The basic structure of an ns-2 topology is composed of *nodes* that are connected by *links*. A *node* represents a network element such as a host, switch or router. A *link* represents the physical and MAC layers of the connection between two nodes. Traffic is generated via source and sink *agents* that are bound to individual nodes. Buffering and any buffer management is performed by the link.

In this study, the CM is modeled in ns-2 as a rate-shaped queue (RsQ) feeding into a model of the transmission queue (TxQ). The RsQ is controlled by token bucket parameters of rate, burst size, peak rate. The TxQ is controlled by a modified token bucket where a variable number of grant bytes can be allocated every 2 ms. Short-term variations in the grant size are permitted through randomization around a mean value that can be varied a few times over the course of the simulation through the TCL script used to run the tests. Packets are sent upstream from the CM to a CMTS model when there are sufficient bytes in the token bucket.

2.2 MODELING DOCSIS MODEM WITH CODEL AQM

The simulation approach involves the development of an ns-2 simulation model for a single CM with a single upstream Service Flow on a DOCSIS 3.0 link, with link congestion modeling.

The DOCSIS upstream (CM model) is implemented as two cascaded queues (ns-2 links): first a rate shaping queue (RsQ) using the DOCSIS token bucket algorithm + CoDel AQM, then a tx queue (TxQ) that models the upstream MAC and channel. In this initial model, the TxQ is a DropTail queue with a fixed buffer size.

The CoDel algorithm utilizes a packet "sojourn time" measure. The sojourn time is defined as the total time a packet spends in a network device, from ingress on the receiving interface to egress on the transmitting interface. In this model we will focus on measuring sojourn time only in the RsQ, but will consider extensions to measure upstream sojourn time for the two queues together.

The DOCSIS downstream (CMTS model) is implemented as a single rate shaping queue (similar or identical to upstream) with a configurable delay and link bandwidth.

2.2.1 RATE SHAPING QUEUE (RSQ)

The RsQ is the primary location that packet buffering occurs when a CM is operating with an uncongested DOCSIS channel. In many CM implementations, the buffer used by the RsQ is oversized and responsible for the bufferbloat phenomenon.

The rate shaping queue implements the token bucket rate shaper in DOCSIS with the CoDel drop decision applied at head.

- CoDel "sojourn time" is measured within the RsQ only.
- Packets dropped by CoDel are not counted against the token bucket.

  DOCSIS token bucket parameters:

    Max Sustained Rate (token rate)

    Max Burst (token bucket depth)

    Peak Traffic Rate (limits burst rate)

For comparison purposes, a second RsQ is implemented using a DropTail queue rather than CoDel, and can be swapped into the system model discussed below.

2.2.2 TRANSMIT QUEUE (TXQ)

The TxQ queuing model is what separates this study from previous CoDel simulation studies. This model enables the exploration of the interaction between CoDel and the channel access behavior unique to DOCSIS. The model includes the request/grant loop, Queue Depth Based Requests (QDB), Continuous Concatenation and Fragmentation (CCF), and request piggybacking to achieve close alignment with DOCSIS. The specifics of these behaviors are discussed below.

DOCSIS US MAC & Channel Model Assumptions:

- Assume fixed MAP rate (configurable, 2 ms default).
- CM can get a grant and/or transmit a request at the beginning of each MAP.
- Model does not include contention request collisions and backoff.
- CM assumed to piggyback requests until TxQ is empty.
- Request/Grant Loop set to 2 MAP intervals (4-6 ms). This effectively delays the first packet transmission by 2 MAP intervals when the TxQ transitions from empty to occupied.
- Will model upstream channel bonding via appropriate configuration of link bandwidth (0.8*ModulationProfile*ChannelCount).
- Grant size varied to model congestion.

Detailed implementation of TxQ:

- At start of each MAP interval, the model calculates the max grant given by CMTS.

  The grant is a uniform random variable with mean grant size (maxgrant_) and +/- range (mgvar_) passed in via TCL script. These parameters can be tuned to model a range of congested/uncongested plant scenarios:

  > Uncongested RF = large mean with low variance
  >
  > Congested RF = lower mean and/or higher variance

- Requests are tracked via a vector REQ() of length two. REQ(0) represents the outstanding request sent to the CMTS. REQ(1) tracks the occupancy of the queue.
- After receiving a grant, the model performs the following steps:

  Determine the actual grant by: grant = min(max grant, REQ(0))

  Update the request vector: Move REQ(1) into REQ(0) and add in any ungranted bytes from the previous request (REQ(0)- actual_grant)

  Calculate number of unrequested bytes in TxQ and record as REQ(1)

  Utilize actual grant to schedule transmit serialization times for packets in queue.

- Packets are transmitted as a whole. CCF is modeled by adjusting the link bandwidth for the duration of the “fragmented” packet, resulting in a packet arrival time at the next hop as if the packet had been fragmented.

2.2.3 NOTES ON INTERACTION BETWEEN RSQ AND TXQ

The TxQ contains all packets for which the CM has either sent a DOCSIS REQ or will send a REQ at the next opportunity. These are packets that have already passed the CoDel decision algorithm. Therefore, they are no longer eligible for being dropped by CoDel, so holding them in a buffer outside of the CoDel algorithm is appropriate. However, when the upstream channel is congested, particularly when the average channel capacity is less than the token bucket rate configured in the RsQ, there will be queue build-up in the TxQ. The current model uses a fixed size (32 packet) DropTail queue for the TxQ, and thus allows packet drops at the interface between the RsQ and the TxQ. While this is a simple implementation, and one that may be attractive to CM developers, we believe that it is suboptimal in handling the congested or near congested case since the queue build-up moves entirely (or in part) to the TxQ, which is outside of the control of CoDel. As a result, the CoDel algorithm does little to impact the performance of the CM; rather, the performance is largely driven by the size of the TxQ.

Alternative implementations, which will be studied in future work, include a) calculating the CoDel sojourn time across both queues (while still performing CoDel drops at the head of the RsQ) or b) introducing a "back-pressure" mechanism, whereby the TxQ doesn't drop packets upon buffer saturation, but instead causes packets to be held in the RsQ. The second of these two approaches seems to be the most likely to provide the best performance in the congested case. Additionally, it would be worthwhile to evaluate whether different target latencies (i.e., other than 5 ms) for CoDel would provide better performance in conjunction with either of these two approaches.

2.3 NS-2 SYSTEM MODEL

Figure 1 below depicts the system model used in this study. The node n0 represents a LAN device behind the CM, and is the initiator of all of the traffic injected into the system. Nodes n1 and n2 represent the CM, where in the case of upstream traffic, the link between n1 & n2 represents the RsQ, and the link egressing node n2 represents the TxQ and DOCSIS link. As stated above, the link between n1 & n2 can be configured to use DropTail or CoDel depending on the test case. The ns-2 environment requires the presence of node n2 to model cascaded queues. The link between n1 and n2 has zero delay and infinite bandwidth. This configuration allows the two cascaded queues to be essentially a single network entity, the cable modem. The DOCSIS link includes 1 ms of fixed latency in each direction (in addition to the upstream MAC latency). Node n3 represents the CMTS and WAN. The server nodes n4-n10 are modeled as having independent GigE links to node n3. Each of the server nodes is configured with a different latency. Node n4 (the FTP server used for TCP background load) is configured to either 18 ms (9 ms each direction) or 98 ms (49 ms each direction) for a total fixed round trip time (including the fixed latency of the DOCSIS link) of 20 ms or 100 ms. Node n5 (the VoIP endpoint) is configured with zero latency; however, additional latency is added in the VoIP quality metric calculation to represent the connection to the other party. Node n6 (the CBR endpoint) is similarly configured with zero additional latency. Nodes n7-n10 (the HTTP servers) are configured with different latency values as indicated in the figure.

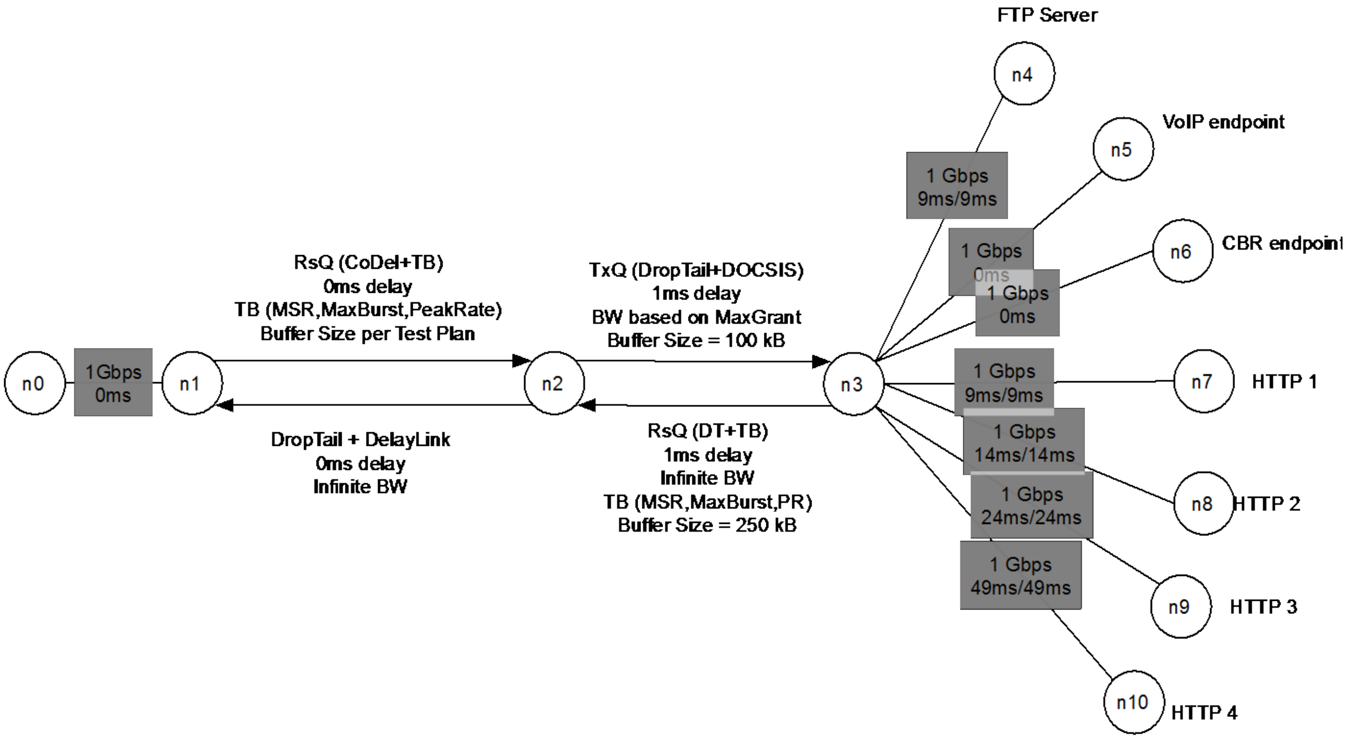


Figure 1. NS-2 System model used in this study

3 TEST METHODOLOGY

3.1 TEST CONFIGURATIONS

3.1.1 DOCSIS SERVICE FLOW CONFIGURATION

A single CM configuration will be utilized:

- Downstream

  Max Sustained Rate: 20 Mbps

  Traffic Burst: 20 MB

  Peak Traffic Rate: 50 Mbps

- Upstream

  Max Sustained Rate: 5 Mbps

  Traffic Burst: 10 MB

  Peak Traffic Rate: 20 Mbps

The Token Bucket link model also includes a "Peak Burst" setting, which is set to 1522 Bytes per the DOCSIS 3.0 specification.

3.1.2 QUEUE MANAGEMENT SCENARIOS

All test cases will be run using each of the two queue management techniques in the RsQ:

- DropTail with a buffer size set equal to 1s * MaxSustainedRate
- DropTail with a buffer size set equal to 50 ms * MaxSustainedRate
- CoDel with a buffer size set equal to 1s * MaxSustainedRate

A limitation with the current ns-2 model is that it limits queues based on a configured number of packets, rather than total bytes, as would be the case for a real CM implementing Buffer Control. As a result, buffer sizes were set based on the maximum packet size of 1500 bytes (i.e., 21 packets for the 50 ms config. and 417 packets for the 1s config.). In test scenarios where the vast majority of upstream packets are 1500 bytes long, this limitation is immaterial. However, in some test cases (particularly the 21 packet buffer cases), this limitation may result in skewed results when a large number of small packets arrive at the queue, triggering packet drops when a real CM would not have dropped them. A future enhancement to the queuing model should consider re-writing the DropTail link module in ns-2 so that it utilizes a byte limit for queues rather than a packet limit.

The TxQ buffer size is set to 32 packets for all test cases.

3.1.3 RF CONGESTION SCENARIOS

The following settings for upstream DOCSIS RF congestion will be explored:

1. No congestion (grants not limited by CMTS).
2. Borderline congestion. Average available capacity of DOCSIS link = CM Max Sustained Rate. Mean Max_grant_size = MSR*MAP_interval (1250 Bytes), grant variance set to 20%. NOTE: this will result in the CM being unable to achieve Peak Traffic Rate, and may cause queue build-up in the TxQ.

Note, the current document only provides results for the No Congestion case. The results of the borderline congestion case were deemed to be too dependent on the particular implementation of our simulator model, and so provide a misleading view as to CoDel performance in the presence of congestion.

3.2 TRAFFIC MODELS

For each test condition we will utilize a simultaneous combination of three traffic types:

- "Bulk" TCP traffic - where the primary metric is throughput
- OTT VoIP traffic - where the primary metric is estimated voice quality
- Web Browsing - where the primary metric is page load time

In addition to the three primary metrics, we will also monitor RsQ size and RsQ latency

3.2.1 BULK TRAFFIC SCENARIOS

Five scenarios to be tested:

1. No background traffic
2. Single long-term upstream FTP

   Single TCP connection in a steady-state bulk transfer (very large file)
3. Multiple simultaneous upstream FTPs

   5 simultaneous FTP sessions, each transferring a moderately sized file. Once each file transfer finishes, a new file transfer is initiated.
4. TCP + UDP: Same as 3, but include 1 Mbps UDP flow alongside TCP to see how CoDel manages traffic that isn't entirely TCP (w/congestion avoidance).
5. Bursty TCP case: single upstream FTP/TCP session transfers 18.75 MB file, waits 16 seconds, then repeats (results in 47% transmit duty cycle: 5.3 s @ 20 Mbps, 8.7 s @ 5 Mbps, 16 s idle).

3.2.1.1 TCP Stacks

The testing is configured to use a variety of TCP stacks. This is primarily due to limitations in ns-2, which make it challenging to use the same TCP stack for multiple traffic scenarios. However, it also is useful to exercise multiple TCP implementations, since many of these are in use in the wild. The TCP stacks utilized are summarized in the following table:

Table 1. TCP Stacks

Bulk FTP	Linux TCP (CUBIC)
Bursty TCP	Reno TCP
Web traffic	BSD TCP w/ SACK

3.2.1.2 Network RTT Scenarios

As described in the simulation topology section, the bulk FTP traffic will be simulated with two different network RTT values: 20 ms & 100 ms, in order to see the impact on performance.

3.2.2 OTT VOIP APPLICATION

This traffic flow will evaluate the impact of CoDel vs DropTail on VoIP MOS.

OTT VoIP will be simulated by an upstream UDP flow that models a G.711 voice call as CBR traffic with 20 ms packetization, 218 byte frames.

Measure mean latency and mean packet loss rate (counting packets with jitter > 60 ms as lost).

Calculate estimated voice quality R-value based on a derivation of the ITU E-Model for G.711 [Cole]].

R= 94.2 - 0.024*Latency - 0.11*max(0,Latency-177.3 ms) - 30*log(1 + 15*Loss).

In addition to the measured one-way latency arising due to the simulation topology, 20 ms of latency is added to the measurements in order to capture additional network latency between the CMTS and the listener.

3.2.3 WEB BROWSING APPLICATION IMPACT

Model single user web page download as follows:

- Web page modeled as single HTML page + 100 objects spread evenly across 4 servers. Web object sizes are currently fixed at 25 kB each, whereas the initial HTML page is 100 kB. Appendix A provides an alternative page model that may be explored in future work.
- Server RTTs set as follows (20 ms, 30 ms, 50 ms, 100 ms).
- Initial HTTP GET to retrieve a moderately sized object (100 kB HTML page) from server 1.
- Once initial HTTP GET completes, initiate 24 simultaneous HTTP GETs (via separate TCP connections), 6 connections each to 4 different server nodes
- Once each individual HTTP GET completes, initiate a subsequent GET to the same server, until 25 objects have been retrieved from each server.

Measure total page load time from initial HTTP GET until 100th object is received. Wait 5 seconds then repeat for the duration of the simulation, and calculate median page load time and median absolute deviation of PLT.

3.3 TEST CASE SUMMARY

All test cases are run with the following common configuration:

- Service Configuration: 20 Mbps downstream x 5 Mbps upstream (details listed above)
- Upstream VoIP traffic (single simulated G.711 voice call)
- Web page load test (single user downloading a page)

The table below summarizes the parameters that change from test case to test case.

Table 2. Test Case Summary

TEST CASE #	QUEUE MGMT	RF CONGESTION	BULK TRAFFIC	BULK TRAFFIC RTT
D01	DropTail	None	None	N/A
D02	DropTail	None	Single TCP	20 ms
D03	DropTail	None	Single TCP	100 ms
D04	DropTail	None	Multiple TCP	20 ms

TEST CASE #	QUEUE MGMT	RF CONGESTION	BULK TRAFFIC	BULK TRAFFIC RTT
D05	DropTail	None	Multiple TCP	100 ms
D06	DropTail	None	UDP + Multiple TCP	20 ms
D07	DropTail	None	UDP + Multiple TCP	100 ms
D08	DropTail	None	Bursty TCP	20 ms
D09	DropTail	None	Bursty TCP	100 ms
D11	DropTail	Yes	None	N/A
D12	DropTail	Yes	Single TCP	20 ms
D13	DropTail	Yes	Single TCP	100 ms
D14	DropTail	Yes	Multiple TCP	20 ms
D15	DropTail	Yes	Multiple TCP	100 ms
D16	DropTail	Yes	UDP + Multiple TCP	20 ms
D17	DropTail	Yes	UDP + Multiple TCP	100 ms
D18	DropTail	Yes	Bursty TCP	20 ms
D19	DropTail	Yes	Bursty TCP	100 ms
C01	CoDel	None	None	N/A
C02	CoDel	None	Single TCP	20 ms
C03	CoDel	None	Single TCP	100 ms
C04	CoDel	None	Multiple TCP	20 ms
C05	CoDel	None	Multiple TCP	100 ms
C06	CoDel	None	UDP + Multiple TCP	20 ms
C07	CoDel	None	UDP + Multiple TCP	100 ms
C08	CoDel	None	Bursty TCP	20 ms
C09	CoDel	None	Bursty TCP	100 ms
C11	CoDel	Yes	None	N/A
C12	CoDel	Yes	Single TCP	20 ms
C13	CoDel	Yes	Single TCP	100 ms
C14	CoDel	Yes	Multiple TCP	20 ms
C15	CoDel	Yes	Multiple TCP	100 ms
C16	CoDel	Yes	UDP + Multiple TCP	20 ms
C17	CoDel	Yes	UDP + Multiple TCP	100 ms
C18	CoDel	Yes	Bursty TCP	20 ms
C19	CoDel	Yes	Bursty TCP	100 ms

4 RESULTS

4.1 TCP PERFORMANCE

4.1.1 DROPTAIL WITH NO BUFFER CONTROL

Figure 2 below shows the throughput of a single TCP session and of five simultaneous TCP sessions running on a simulated CM that has no buffer control (buffer set to 417 packets, equivalent to 1 second at the MSR of 5 Mbps). The initial Peak Traffic Rate burst at 20 Mbps is clearly seen, as is the transition to the Maximum Sustained Traffic Rate of 5 Mbps. These plots show the upside of over buffering. Due to the large buffer, TCP ramps up quickly (0.6 sec) to the Peak Traffic Rate and is able to keep the pipe consistently full for the duration of the simulation. Figure 3 on the other hand, shows the downside. Even a single TCP session will aim to keep the buffer as full as it can, causing unacceptable latency for competing traffic. Note here that the buffer is set to a fixed number of packets, which results in a maximum buffering latency of 250 ms during the ~5.5 seconds of the initial 20 Mbps burst, then jumps to 1 second from that point forward.

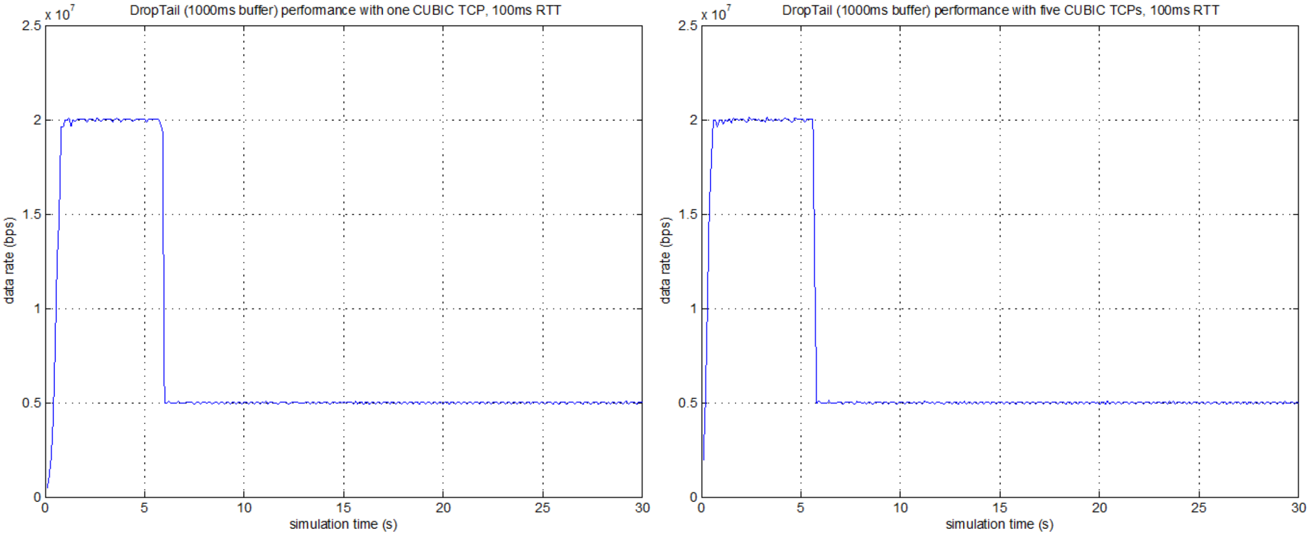


Figure 2. TCP Throughput of Simulated Cable Modem with Bufferbloat

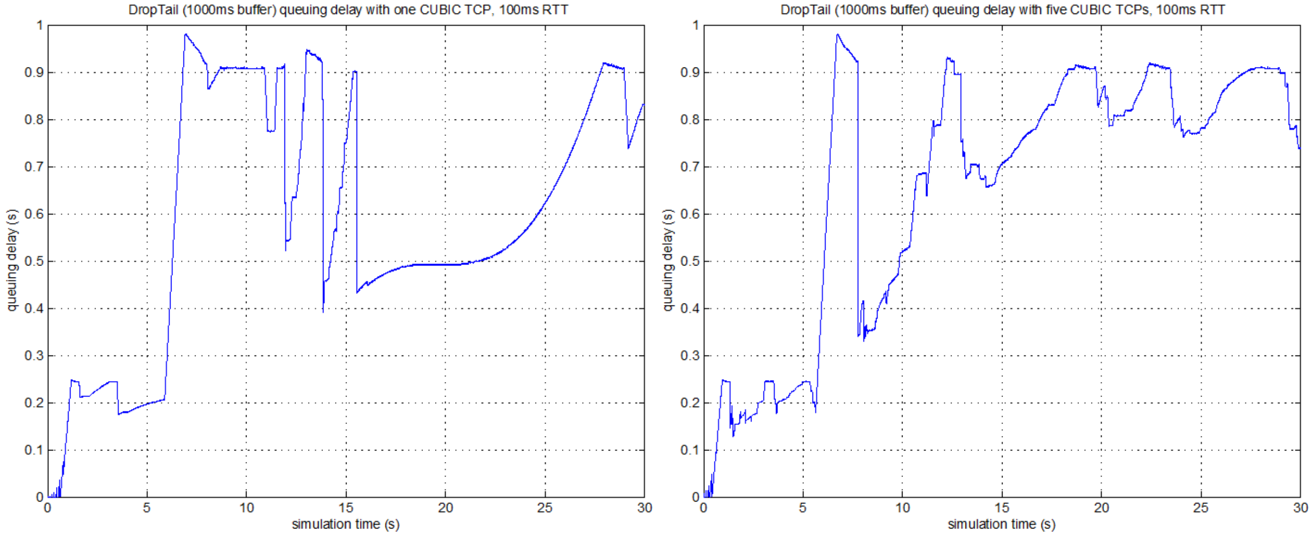
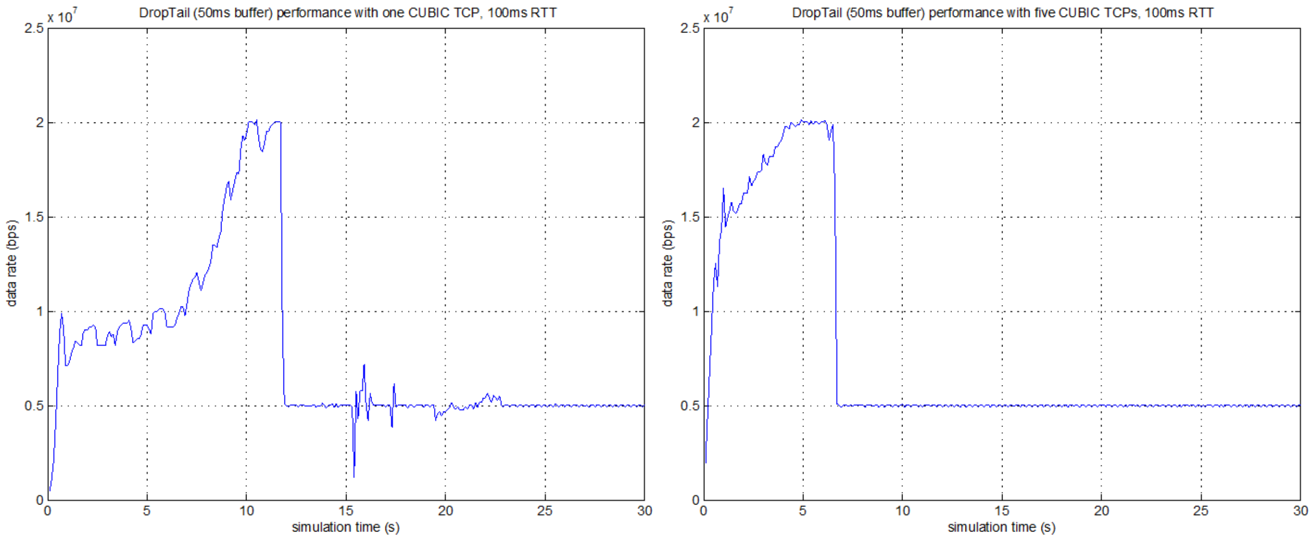


Figure 3. Queuing Delay of Simulated Cable Modem with Bufferbloat

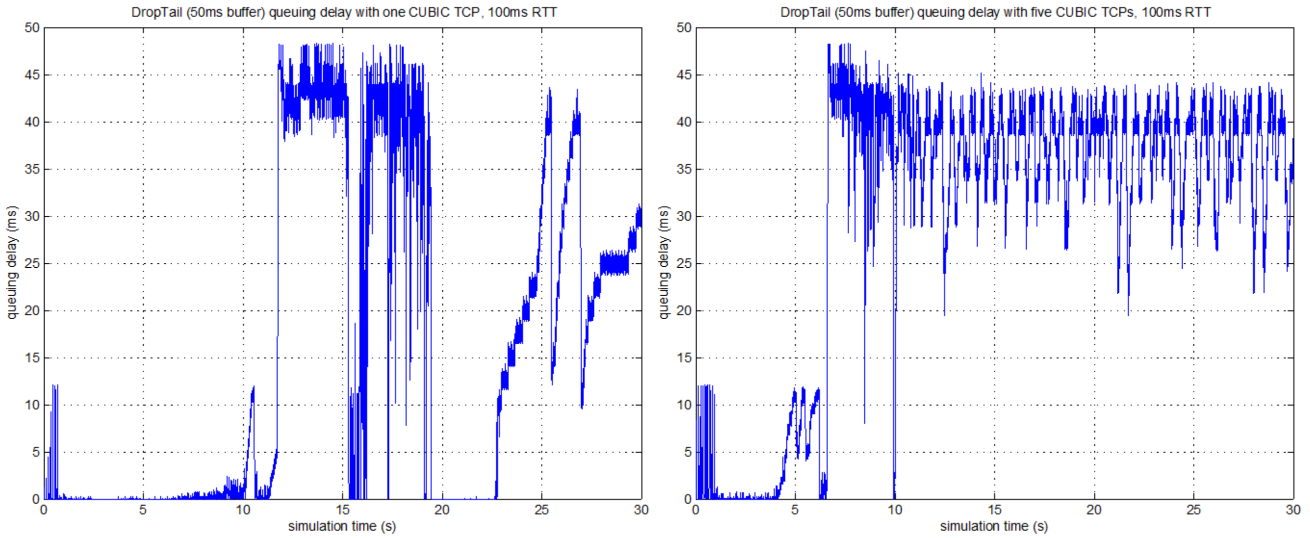
4.1.2 DROPTAIL WITH BUFFER CONTROL

Figure 4 shows the single TCP and five TCP throughput for a DropTail modem in which the modem is configured with Buffer Control enabled and set to 21 packets (equivalent to 50 ms at the 5 Mbps MSR). This selection of buffer size provides a reasonable tradeoff in performance between TCP bulk transfer applications, and latency sensitive applications, and is the value recommended by CableLabs in [GL-BUFFER]. One clear downside to the DropTail with a short fixed-size buffer is that TCP has trouble making use of the Peak Traffic Rate. As can be seen in the figures, the single TCP took 10 seconds to slowly work its way up to the 20 Mbps Peak Traffic Rate, and took 12 seconds to utilize the 10 MB traffic burst (vs. 6 sec with no buffer control). Even five simultaneous TCPs aren't able to quickly ramp up to the peak (taking 4.5 seconds), and took 6.7 seconds to utilize the 10 MB traffic burst (vs. 5.8s for no buffer control). Figure 5 shows the evolution of queuing delay, and illustrates the gain provided by Buffer Control, whereby the worst case queuing delay is bounded by the 50 ms buffer size. Yet it can be seen in the 5 TCP case that the TCPs keep the small buffer very full, which leaves a very small amount of free space in the buffer to accept new streams.


DropTail (50ms buffer) performance with one CUBIC TCP, 100ms RTT

DropTail (50ms buffer) performance with five CUBIC TCPs, 100ms RTT

Figure 4. TCP Throughput of Simulated Cable Modem with Buffer Control Enabled


DropTail (50ms buffer) queuing delay with one CUBIC TCP, 100ms RTT

DropTail (50ms buffer) queuing delay with five CUBIC TCPs, 100ms RTT

Figure 5. Queuing Delay of Simulated Cable Modem with Buffer Control Enabled

4.1.3 CODEL

Figure 6 and Figure 7 show the performance of the CoDel AQM in the same scenarios. The CoDel AQM was configured with the same buffer size as was used in the bufferbloat (no buffer control) version above (417 packets). We see that CoDel allows the single TCP and the five TCPs to quickly ramp up to 20 Mbps (due to the large buffer), but that CoDel reacts to the large buffer and forces the TCPs to back off to the point where they have minimal queue sitting in the buffer, but still keep the channel occupied. There are some interesting aspects to these results. For the single TCP case, while the TCP ramps up to 20 Mbps quickly, after 2.5 seconds the rate drops down to 15 Mbps and does not recover within the 5.5 seconds remaining for the traffic burst. This appears to be due to a specific interaction between CoDel and the CUBIC TCP that results in the TCP getting stuck at 75% link utilization, but the specific mechanism is unclear. This effect is dependent on buffer size. When the buffer is configured for 130 or 170 packets, this issue goes away. Also of note is that, while CoDel generally keeps the queuing latency below the 5 ms target, at the TCP start and at the rate transition from Peak Traffic Rate to Max Sustained Rate, CoDel allows a spike in queuing latency that takes some time (~2.5 seconds) to subside. With the selection of a 417 packet buffer, the spikes in queuing latency can be rather large (up to 370 ms in the 5 TCP case).

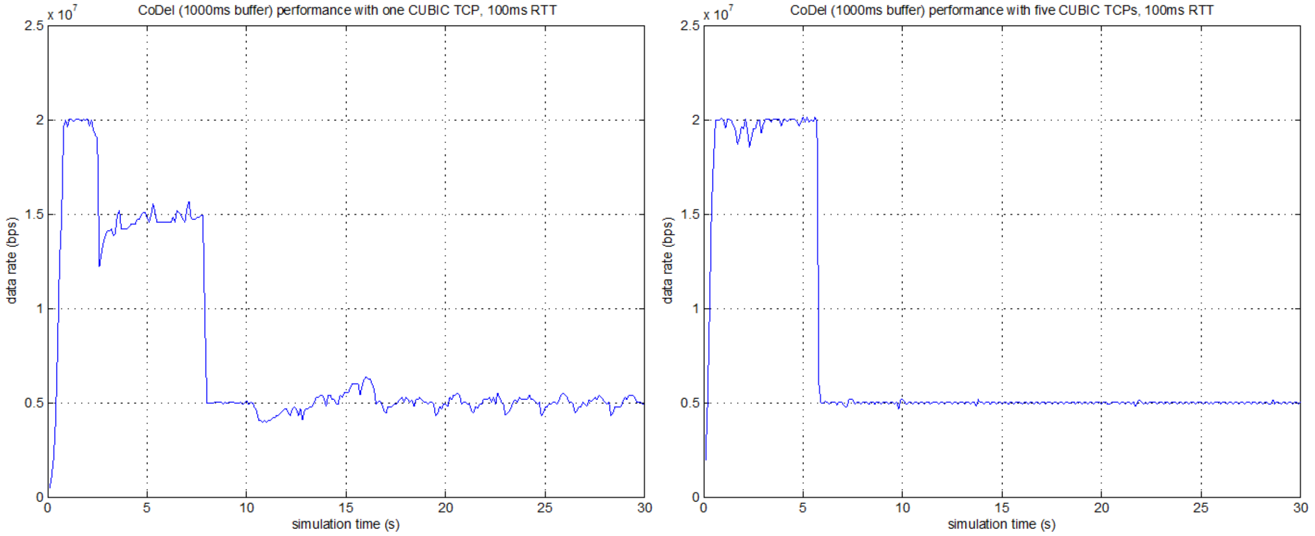


Figure 6. TCP Throughput of Simulated Cable Modem with CoDel AQM

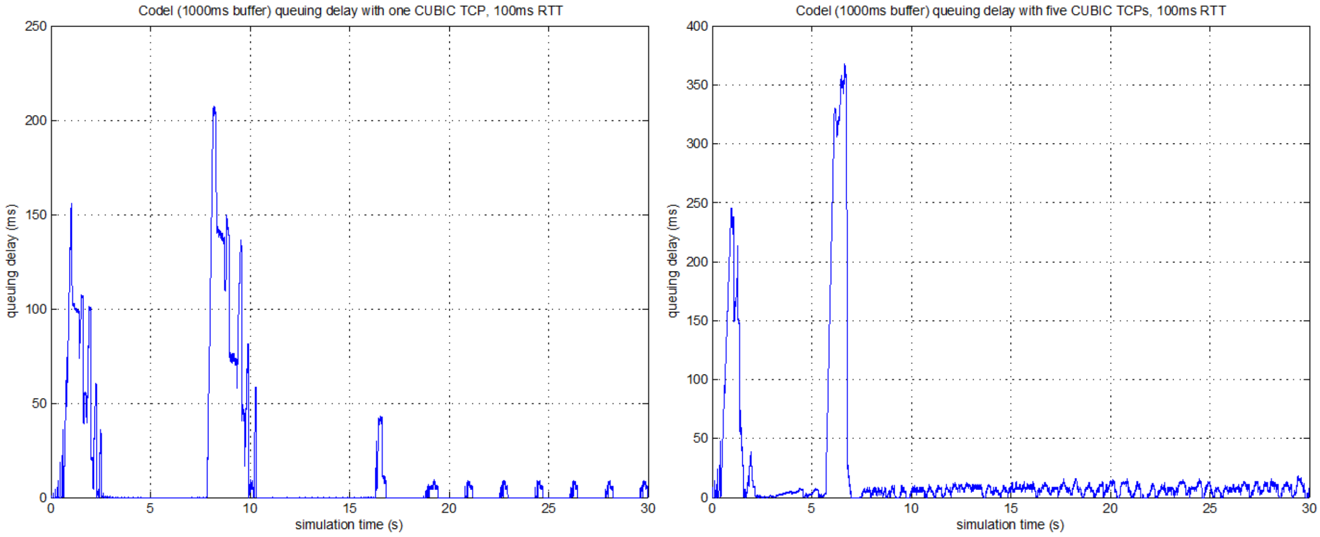


Figure 7. Queuing Delay of Simulated Cable Modem with CoDel AQM

4.2 VOIP PERFORMANCE

4.2.1 DROPTAIL WITH NO BUFFER CONTROL

Table 3 shows the VoIP performance results for DropTail and CoDel, both configured with a 417 packet buffer capacity (equivalent to 1s at the 5 Mbps MSR). We see that the percentage of late packets is dramatically higher for DropTail. Late packets are dropped at the receiver and add to the overall packet loss ratio, which has a negative impact on the call quality. In addition, late packets use network resources then are dropped at the receiver, effectively wasting DOCSIS channel resources and contributing to the queuing latency of subsequent packets. CoDel by comparison controls queuing latency, resulting in very few packets dropped at the receiver. CoDel provides a clear improvement over no buffer control DropTail in the presence of any background traffic. Also of note is that in the case of multiple TCP connections with short RTT, CoDel too begins to experience traffic loss that negatively affects call quality. Alternative implementations (which may be studied in future work) using multiple queues per cable modem, for example in stochastic fair queuing CoDel (sfq_codel) would address the packet loss in this scenario.

Table 3. Comparison of DropTail (without Buffer Control) and CoDel - VoIP

Test Cases	Traffic Load				DropTail (417pkt)			CoDel (417pkt)		
	Scenario	FTP RTT	FTP Load	CBR Traffic	lost %	late %	total plr	lost %	late %	total plr
D01, C01	1	N/A	0	0	0.10%	0.00%	0.10%	0.10%	0.00%	0.10%
D02, C02	2	20 ms	1	0	0.69%	99.18%	99.87%	0.66%	0.12%	0.78%
D03, C03	2	100 ms	1	0	0.23%	97.09%	97.32%	0.51%	1.12%	1.64%
D04, C04	3	20 ms	5	0	2.15%	97.78%	99.93%	8.77%	0.34%	9.11%
D05, C05	3	100 ms	5	0	0.96%	98.96%	99.92%	2.31%	0.37%	2.68%
D06, C06	4	20 ms	5	1 Mbps	2.01%	97.93%	99.93%	10.53%	0.38%	10.90%
D07, C07	4	100 ms	5	1 Mbps	1.70%	98.20%	99.90%	3.67%	0.39%	4.06%
D08, C08	5	20 ms	1, bursty	0	0.48%	86.12%	86.60%	0.92%	0.49%	1.41%
D09, C09	5	100 ms	1, bursty	0	0.27%	20.54%	20.81%	0.18%	0.70%	0.88%

4.2.2 DropTail with Buffer Control

Table 4 shows the VoIP performance results for DropTail configured with a 21 packet buffer capacity (equivalent to 50 ms at the 5 Mbps MSR) compared to CoDel with 417 packet buffer capacity similar to above. DropTail VoIP performance is greatly improved when properly configured using D3.0 Buffer Control and the CableLabs recommended 50 ms. The late arrival percentage went to zero in all cases because the max queue latency, and therefore induced jitter, was 50 ms which is below the late criteria of 60 ms (as stated above in Section 3.2.2). We can see that the performance is slightly better than CoDel overall. The resulting differences in call quality between the two cases are very slight. Similar to above, the two traffic scenarios with short RTT and multiple TCP flows provide the biggest loss ratio for both DropTail and CoDel.

Table 4. Comparison of DropTail (with Buffer Control) and CoDel - VoIP

Test Cases	Traffic Load				DropTail (21pkt)			CoDel (417pkt)		
	Scenario	FTP RTT	FTP Load	CBR Traffic	lost %	late %	total plr	lost %	late %	total plr
D01, C01	1	N/A	0	0	0.00%	0%	0.00%	0.10%	0.00%	0.10%
D02, C02	2	20 ms	1	0	1.15%	0%	1.15%	0.64%	0.14%	0.79%
D03, C03	2	100 ms	1	0	0.60%	0%	0.60%	0.53%	0.60%	1.13%
D04, C04	3	20 ms	5	0	11.41%	0%	11.41%	9.14%	0.34%	9.49%
D05, C05	3	100 ms	5	0	1.00%	0%	1.00%	2.15%	0.33%	2.48%

Test Cases	Traffic Load				DropTail (21pkt)			CoDel (417pkt)		
	Scenario	FTP RTT	FTP Load	CBR Traffic	lost %	late %	total plr	lost %	late %	total plr
D06, C06	4	20 ms	5	1 Mbps	8.51%	0%	8.51%	10.73%	0.32%	11.05%
D07, C07	4	100 ms	5	1 Mbps	2.99%	0%	2.99%	3.61%	0.34%	3.95%
D08, C08	5	20 ms	1, bursty	0	0.91%	0%	0.91%	0.98%	0.50%	1.48%
D09, C09	5	100 ms	1, bursty	0	0.14%	0%	0.14%	0.17%	0.53%	0.70%

4.3 WEB CLIENT PERFORMANCE

4.3.1 DROPTAIL WITH NO BUFFER CONTROL

Table 5 shows the web browser performance for DropTail and CoDel under the same set of conditions as described previously. For both queue management approaches, the web client performance is summarized by two statistics calculated from the page load time, the median and the median absolute deviation. Additionally the count of page loads that completed within the simulation duration of 600 seconds is shown. CoDel shows a clear benefit in all cases where TCP background loading is present, resulting in median page load times that in many cases are about an order of magnitude faster than with the DropTail case, and very consistent performance, with median absolute deviations in the majority of circumstances being < 3% of the median value.

Table 5. Comparison of DropTail (without Buffer Control) and CoDel - WWW

Test Cases	Traffic Load				DropTail (417pkt)			CoDel (417pkt)		
	Scenario	FTP RTT	FTP Load	CBR Traffic	median	MAD	count	median	MAD	count
D01, C01	1	N/A	0	0	2.756	0.002	77	2.756	0.002	77
D02, C02	2	20 ms	1	0	23.934	2.936	21	2.996	0.018	74
D03, C03	2	100 ms	1	0	23.654	3.032	23	2.832	0.035	75
D04, C04	3	20 ms	5	0	30.070	4.154	16	3.352	0.370	69
D05, C05	3	100 ms	5	0	33.030	4.625	15	3.042	0.088	73
D06, C06	4	20 ms	5	1 Mbps	28.076	3.220	17	3.647	0.646	62
D07, C07	4	100 ms	5	1 Mbps	28.414	2.503	16	3.066	0.076	73
D08, C08	5	20 ms	1, bursty	0	6.902	1.158	51	2.978	0.034	75
D09, C09	5	100 ms	1, bursty	0	3.400	0.640	69	2.762	0.008	75

4.3.2 DROPTAIL WITH BUFFER CONTROL

When buffer control is introduced in the DropTail case, a benefit is seen in nearly all cases, but still CoDel provides better performance.

Table 6. Comparison of DropTail (with Buffer Control) to CoDel - WWW

Test Cases	Traffic Load				DropTail (21pkt)			CoDel (417pkt)		
	Scenario	FTP RTT	FTP Load	CBR Traffic	median	MAD	count	median	MAD	count
D01, C01	1	N/A	0	0	4.026	0.002	67	2.756	0.002	77
D02, C02	2	20 ms	1	0	7.367	0.873	34	3.002	0.026	74
D03, C03	2	100 ms	1	0	4.412	0.224	61	2.836	0.035	76
D04, C04	3	20 ms	5	0	>600	-	0	3.310	0.333	69
D05, C05	3	100 ms	5	0	13.696	7.646	21	3.084	0.072	72
D06, C06	4	20 ms	5	1 Mbps	>600	-	0	3.320	0.358	67
D07, C07	4	100 ms	5	1 Mbps	20.570	7.577	18	3.049	0.051	72
D08, C08	5	20 ms	1, bursty	0	4.578	0.238	57	2.960	0.038	73
D09, C09	5	100 ms	1, bursty	0	4.034	0.006	65	2.760	0.006	75

The short buffer used for DropTail created particular problems for the web client when in the presence of multiple TCP sessions (Traffic Scenarios 3 & 4) as is highlighted in the table.

As stated previously, a current limitation of the simulation model is that it implements buffer control via a limit on the number of packets in queue, rather than the number of bytes in queue. This caused a pathological condition to occur in test cases D04 and D06 (5 competing TCP flows w/ 20 ms RTT). In these test cases, the median page load time could not be calculated due to the fact that no page loads completed during the 600s simulation time. In examining the detailed traces from the simulations, it can be seen that the 5 FTP sessions succeed in keeping the buffer nearly full at all times (98.5% of the time there are 5 or fewer empty slots). Figure 8 shows the queue depth as a function of time during the simulation, and Figure 9 provides the CDF of queue depth. This served to block the web client completely, since it attempted to created 24 simultaneous TCP sessions, largely via back-to-back TCP SYN packets. The majority of these SYN packets were dropped due to the perpetually full buffer, which caused TCP to perform a backoff and retry, only to have the retries dropped as well. The result was that some of the TCP sessions were never able to establish a connection, let alone transfer the web objects being requested. Connection attempts were seen at T=1, 4, 10, 22, 46, 94, 154, 214, 274, etc., indicating a binary exponential backoff beginning with a 3-second value, and topping out at 60 seconds.

In the D05 and D07 test cases (5 competing TCP flows 2/ 100 ms RTT) there were similar problems, though with less catastrophic results. Notable is that the median absolute deviation of the page load time is very large (56% and 37% of the median value, respectively) indicating significant dispersion of the results. In looking at the individual page load times for each of these two conditions, we see cases with significantly higher than median value page load times, with the maximum PLT being 84 seconds and 160 seconds in the two cases.

In a real web client, it is possible that the TCP SYNs will be spaced out more, and so the probability of dropping them all would be lower. Additionally, since the TCP SYN packets are 50 bytes, if the DropTail implementation were to limit buffer size in bytes rather than packets, the drop probability for the TCP SYNs would be significantly lower as well. As a result, we feel that these specific results don't provide an accurate portrayal of real-world performance.

Nonetheless, the queue depth CDF paints a somewhat dismal picture for the prospects of other traffic flows that would seek to compete for a very small number of open slots in the buffer.

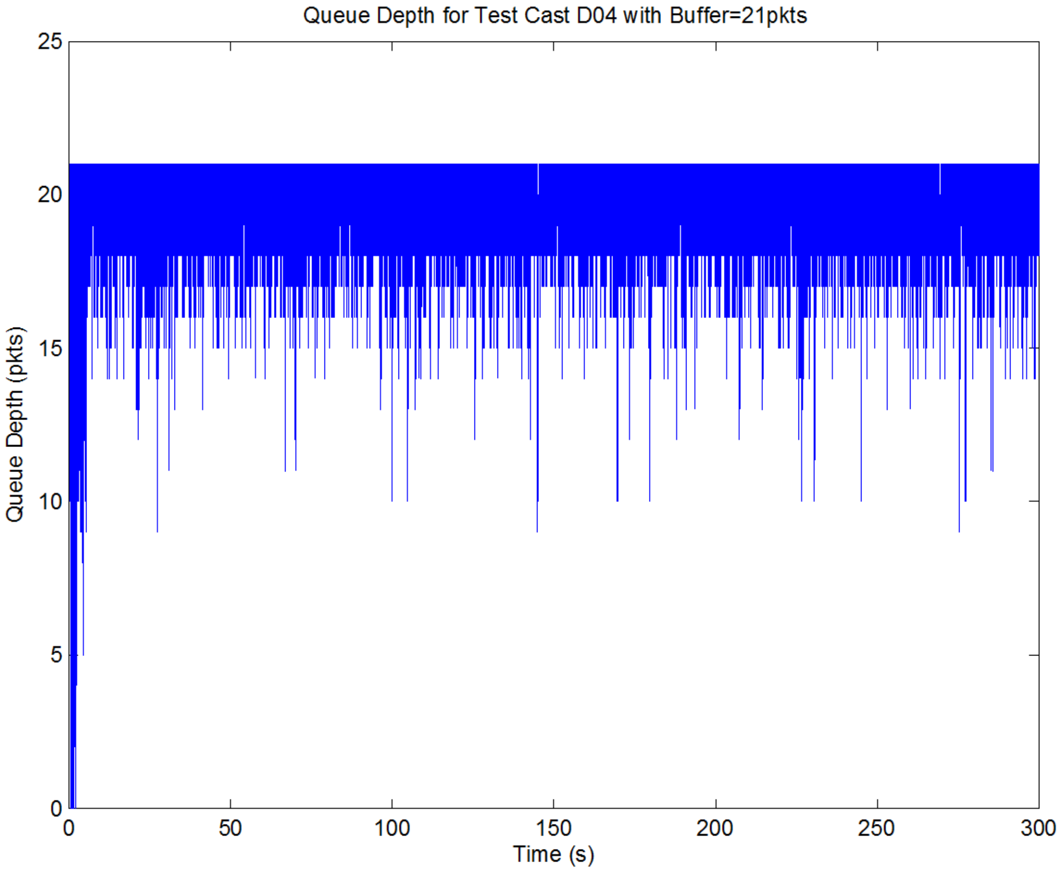


Figure 8. Queue Size vs Time for D04 Test Case

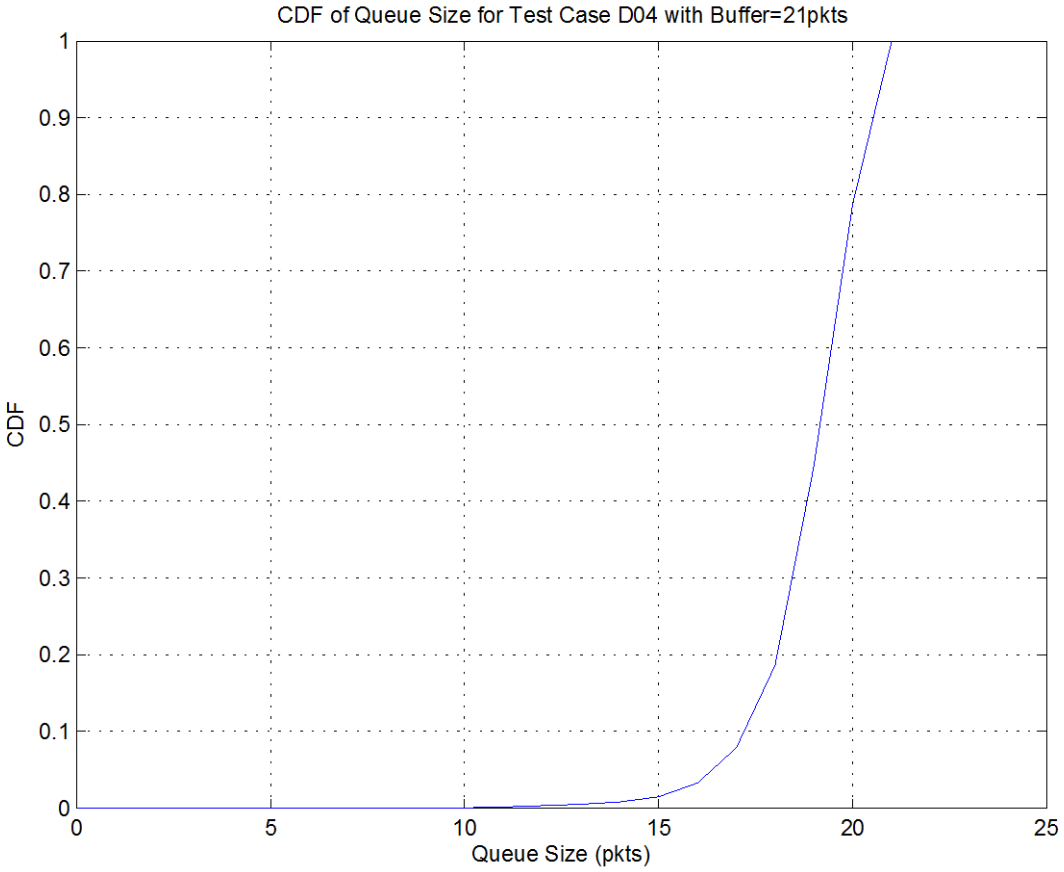


Figure 9. Cumulative Distribution of Queue Size for D04 test case

4.4 PERFORMANCE SUMMARY

Table 7 and Table 8 give a summary of the results presented in the preceding sections. CoDel shows significantly better performance than DropTail with no buffer control for both VoIP and web page load time in all of the tested conditions except for the D01,C01 case with no background traffic, where it provided the same performance as DropTail. When compared to DropTail with Buffer Control, CoDel provided equivalent performance to DropTail for VoIP, but still provided a marked benefit for web page load time.

Table 7. Comparison of DropTail (without Buffer Control) and CoDel - Summary

Test Cases	Traffic Load				VoIP MOS			FTP kBytes in 600 sec.			Median Page Load Time (s)		
	Scenario	FTP RTT (ms)	FTP Load	CBR Traffic (Mbps)	DropTail	CoDel	Delta	DropTail	CoDel	CoDel/DT	DropTail	CoDel	CoDel/DT
D/C01	1	N/A	0	0	4.4	4.4	0.0	0	0	N/A	2.8	2.8	100%
D/C02	2	20	1	0	1.1	4.4	3.3	375614	368198	98%	23.9	3.0	13%
D/C03	2	100	1	0	1.1	4.3	3.2	375247	367641	98%	23.7	2.8	12%
D/C04	3	20	5	0	1.0	3.5	2.5	376472	369714	98%	30.1	3.3	11%
D/C05	3	100	5	0	1.0	4.2	3.2	376393	368505	98%	33.0	3.1	9%
D/C06	4	20	5	1	1.0	3.4	2.4	302337	305263	101%	28.1	3.3	12%
D/C07	4	100	5	1	1.0	4.0	3.0	302263	296446	98%	28.4	3.0	11%
D/C08	5	20	1, bursty	0	1.1	4.3	3.2	371211	368136	99%	6.9	3.0	43%
D/C09	5	100	1, bursty	0	2.7	4.4	1.7	368325	363973	99%	3.4	2.8	81%

Table 8. Comparison of DropTail (with Buffer Control) and CoDel - Summary

Test Cases	Traffic Load				VoIP MOS			FTP kBytes in 600 sec.			Median Page Load Time (s)		
	Scenario	FTP RTT (ms)	FTP Load	CBR Traffic (Mbps)	DropTail	CoDel	Delta	DropTail	CoDel	CoDel/DT	DropTail	CoDel	CoDel/DT
D/C01	1	N/A	0	0	4.4	4.4	0.0	0	0	N/A	4.0	2.8	68%
D/C02	2	20	1	0	4.3	4.4	+0.1	374087	368198	98%	7.4	3.0	41%
D/C03	2	100	1	0	4.4	4.3	-0.1	369865	367641	99%	4.4	2.8	64%
D/C04	3	20	5	0	3.3	3.5	+0.2	379098	369714	98%	>600	3.3	<0.6%
D/C05	3	100	5	0	4.3	4.2	-0.1	375764	368505	98%	13.7	3.1	23%
D/C06	4	20	5	1	3.6	3.4	-0.2	306335	305263	100%	>600	3.3	<0.6%
D/C07	4	100	5	1	4.1	4.0	-0.1	302361	296446	98%	20.6	3.0	15%
D/C08	5	20	1, bursty	0	4.3	4.3	0.0	370465	368136	99%	4.6	3.0	65%
D/C09	5	100	1, bursty	0	4.4	4.4	0.0	366821	363973	99%	4.0	2.8	68%

5 FUTURE WORK

The simulation models built for the purposes of this study provide what we believe to be good models of the DOCSIS 3.0 MAC. That said, a couple of limitations have been noted that point to improvements to the model which would allow a more complete study of certain scenarios. Specifically, the DropTail buffer control model currently implements queue size limits based on the number of packets rather than number of bytes in queue. This caused a more significant degradation in web page load time performance during conditions of heavy TCP load than would occur in reality. Secondly, our implementation of queuing behavior in the presence of upstream RF congestion is known to be problematic, and so the results of testing in those scenarios are not reported here.

Further, other queue management algorithms have been discussed during the preparation of this study: in particular, stochastic fair queuing (sfq) and the hybrids, fair queue CoDel (fq_codel) and stochastic flow CoDel (sfq_codel).

6 CONCLUSIONS

The CoDel active queue management algorithm shows considerable promise in providing an improvement in application performance when compared to the current alternatives of DropTail with and without buffer control.

Further study could provide more guidance with respect to performance in a larger variety of scenarios, and provide guidance on CoDel performance relative to other active queue management approaches.

APPENDIX A FILE SIZES FOR WEB PAGE MODEL (FOR FUTURE WORK)

The file sizes are generated via a log-normal distribution, such that the $\log_{10}$ of file size is drawn from a normal distribution with mean = 3.34 and standard deviation = 0.84. The file sizes (y_i) are calculated from the resulting 100 draws (x_i) using the following formula, in order to produce a set of 100 files whose total size =~ 600 kB (614400 B):

$$y_i = round(k * 10^{x_i})$$

where

$$k = \frac{614400}{\sum 10^{x_i}}$$

The resulting files are distributed to the four servers as follows:

	SERVER1	SERVER2	SERVER3	SERVER4
index.html	102400			
file 1	2997	5009	7105	355
file 2	3225	1643	6683	3815
file 3	7679	1048	1017	9498
file 4	22	510	116	645
file 5	4742	6679	190	544
file 6	1763	769	2424	2931
file 7	224	961	22	12393
file 8	401	132	828	85
file 9	1299	457	91289	12229
file 10	11	3175	91	826
file 11	1859	5744	797	28522
file 12	533	2548	4892	1595
file 13	7432	2527	297	41
file 14	49238	420	1652	4019
file 15	1685	235	547	1062
file 16	23627	256	6169	2457
file 17	860	175	781	1144
file 18	79	1961	276	1127
file 19	2591	235	774	2662
file 20	39	10030	147107	8649
file 21	734	3098	182	51108
file 22	8342	6201	249	2511

	SERVER1	**SERVER2**	**SERVER3**	**SERVER4**
file 23	459	740	449	186
file 24	233	901	51	4380
file 25	1077	5229	2278	3518

APPENDIX B REFERENCES

[CoDel]	Nichols, Kathleen, and Van Jacobsen, Controlling Queue Delay, ACM Queue, May 6, 2012, http://queue.acm.org/detail.cfm?id=2209336.
[Cole]	Cole, Robert G., and Joshua H. Rosenbluth. "Voice over IP performance monitoring." ACM SIGCOMM Computer Communication Review 31, no. 2 (2001): 9-24.
[GL-BUFFER]	DOCSIS Best Practices and Guidelines, Cable Modem Buffer Control, CM-GL-Buffer-V01-110915, September 15, 2011, www.cablemodem.com.