Cable Modem Buffer Management in DOCSIS Networks

Jim Martin, James Westall
School of Computing
Clemson University
Clemson, SC
jim.martin,westall
@cs.clemson.edu

Terry Shaw, Greg White
CableLabs
Louisville, CO
t.shaw,g.white
@cablelabs.com

Rich Woundy
Software and Operations
Comcast Cable
Philadelphia, PA
richard_woundy
@cable.comcast.com

Jeff Finkelstein
Network Architecture
Cox Communications
Atlanta, GA
jeff.finkelstein
@cox.com

George Hart
Network Strategy
Rogers Communications
Toronto, Ontario
george.hart
@rci.rogers.com

***Abstract*—A critical component of subscriber management in a DOCSIS-based cable access network is the buffer management strategy that is in operation at the upstream service flow queue located in the cable modem. The strategy must contend with conflicting goals: large buffers might be required to ensure TCP flows can utilize available bandwidth, however large buffers can impact application flows that are latency sensitive. In this study, we have explored the relationship between application performance and upstream queue management. We define the optimal queue capacity as the buffer size that maximizes throughput and minimizes packet delay. Our results agree with previous work in wired Internet router contexts that suggest that the queue capacity should not exceed a bandwidth*delay product (BDP) amount of data. However, the upstream data rate available to a cable modem varies with the number of competing cable modems. Upstream service flow queues that are provisioned to hold a BDP of data might suffer queue delays exceeding several seconds during periods of congestion. Active queue management such as the Random Early Discard algorithm is able to reduce average queue levels but is not able to provide a consistent balance between the needs of both high throughput and latency sensitive applications. The conclusion is that an adaptive queue management algorithm is required to maintain a consistent balance between throughput and delay.**

Keywords-Cable Network; DOCSIS; Broadband Access;

I. INTRODUCTION

A recent Pew Internet Project study estimates that 63% of adults in the United States use broadband Internet access and that cable is the most common access technology (41% used cable while 33% use DSL) [1]. While all access technologies are evolving rapidly, perhaps the most significant change is occurring in cable access networks. The current Data-over-Cable Systems Interface Specification Version 3 (DOCSIS 3.0) offers significantly higher service rates than previous standards by allowing multiple downstream and upstream channels to be bonded together to provide a logical pipe to subscribers that can provide an effective data rate of hundreds of megabits per second [2]. It is expected that future cable modems (CMs) will be equipped with anywhere from 4 to 32 downstream receivers and 4 or more upstream transmitters.

In spite of these advancements, there are rudimentary challenges that broadband cable service providers face. One such issue is subscriber bandwidth management. A crucial component to subscriber management in a DOCSIS-based cable access network is the buffer management strategy that is in operation at the upstream service flow queue located in the cable modem. While the upstream scheduling algorithm operating at the Cable Modem Termination System (CMTS) ensures quality-of-service commitments and determines the order that cable modems access the channel, the upstream buffer management strategy significantly impacts the performance of best effort applications during periods of congestion.

When a packet arrives at the cable modem from the subscriber's network destined for the Internet, the packet will be queued in an appropriate service flow queue. Assuming the packet is associated with a best-effort service, if a request for upstream bandwidth to the CMTS is not already in progress, a new request will be sent using the DOCSIS contention-based request mechanism. The default buffer management strategy used by a cable modem to support upstream best effort traffic is a drop-tail packet drop policy with a queue capacity set by the manufacturer of the cable modem.

Rules-of-thumb based on sizing a router's buffer capacity with an amount equal to the effective bandwidth*delay product (BDP) are widely used in wired networks. Given the shared nature of a DOCSIS upstream channel and due to complex second-order effects of the channel access protocol, finding the optimal upstream queue capacity at a cable modem is more difficult than at a traditional router. Several recent studies have characterized broadband access link properties with metrics that include round trip time (RTT), jitter, loss, queue length, and drop policies [3]. The results suggest that 30% of cable modems that were deployed had sufficient buffering to cause upstream packet delays in excess of 3 seconds. A smaller scale study (limited to a single service provider) found the capacity of the upstream queue in a set of cable modems ranged from 8 Kbytes to larger than 60 Kbytes [4].

Active Queue Management (AQM) has been introduced in the Internet community as a method to better manage queues. The Random Early Detection (RED) algorithm manages the queue in a more adaptive manner than a simple tail-drop policy by randomly dropping packets using a drop probability that grows as the average queue length grows [5]. It has been shown that RED can reduce the level of packet loss by better tolerating bursty traffic, reduce queue delay by maintaining a lower average queue level, and improve fairness of competing TCP flows. However, it has also been shown that the average queue delay with RED is sensitive to traffic loads and to the RED parameter settings. To the best of our knowledge, the research community has not studied upstream buffer management in cable networks.

In this paper, we present the results of a simulation study that addresses the issue. We limit the study primarily to DOCSIS 2.0 cable networks (we briefly consider a DOCSIS 3.0 situation involving three bonded upstream channels) and to scenarios that involve best effort traffic. For brevity, we have limited the results to those involving 'uncapped' scenarios where subscribers are not limited to a maximum service rate. The motivating questions for the research include the following:

- Do traditional router queue provisioning 'rules-of–thumb' apply to cable modems?
- What is the optimal upstream queue capacity for a cable modem?
- Are there advantages if an AQM algorithm such as RED manages the upstream queue?

The rest of this paper is organized as follows: section 2 provides an overview of DOCSIS focusing on the upstream operation. Section 3 describes the experimental methodology and Section 4 presents the analysis. Section 5 provides conclusions and identifies future work.

II. BACKGROUND AND RELATED WORK

A. DOCSIS

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 modems. The most widely deployed version of the standard, DOCSIS 2.0, uses 6 MHz of bandwidth for the shared downstream channel (8 MHz in certain regions) and up to 6.4 MHz of bandwidth for the shared upstream channel. The downstream channel supports raw data rates up to 42.88 Mbps (55.62 Mbps in 8 MHz regions) and the upstream channel supports raw data rates up to 30.72 Mbps. A single channel (downstream or upstream) may be shared by dozens or as many as a few hundred cable modems.

Downstream operation is based on time division multiplexing. 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 [6]. 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. All upstream data transmissions are contention-free using time division multiple access. The latest standard, DOCSIS 3.0, introduces multiple channels for both downstream and upstream operation. Future services based on bonded channels that offer service rates of 100 Mbps or higher will increase the sensitivity of system performance to upstream buffer management techniques.

B. Active Queue Management

Resource allocation mechanisms used in TCP/IP are host centric, based on feedback, and utilize window control. Routers generally apply a FCFS scheduling discipline at queues that build at output interfaces. When a packet is forwarded to an output interface in a router that has a full queue, the arriving packet is dropped. Dropped packets provide implicit signals of network congestion back to the end points. The TCP congestion control algorithms provide the basic elements of control for managing congestion and for sharing network resources among competing flows.

For routers with queue management based on FCFS scheduling and tail-drop packet discard policy, the capacity of the queue is a readily accessible 'control knob' that can impact performance. Large buffers (i.e., queues configured with large capacity) are useful to absorb the inherent burstiness associated with data and are generally required to support high speed TCP flows. However, large buffers can lead to large queue delay. The performance experienced by subscribers using latency sensitive applications such as VoIP or network games is highly sensitive to queue delay. Finding the optimal buffer capacity is difficult as the best setting depends on the set of active flows and their underlying application requirements.

Advanced queue disciplines can provide more granular control of router resources. For example, if traffic can be grouped into classes, a priority queue algorithm can be used. The challenge with strict priority queueing is to ensure that low priority traffic does not get starved. Algorithms such as weighted round robin, deficit round robin, and weighted fair queueing have been developed to provide a framework to implement a desired policy. AQM algorithms such as RED manage the queue in a more adaptive manner than a simple tail-drop policy by randomly dropping packets using a drop probability that grows as the average queue length increases. The drop rate increases linearly as the average queue length grows from a minimum threshold (*minth*) to a maximum threshold (*maxth*). The drop rate will not exceed a maximum level set by the *maxp* parameter. It has been shown that RED provides the following benefits: reduced level of packet loss by better tolerating bursty traffic, reduced queue delay by maintaining a lower average queue level, and improved fairness of competing TCP flows. However, it has been shown that the average queue delay with RED is sensitive to traffic loads and to the RED parameter settings [7, 8, 9]. Many modifications to RED have been proposed to address this issue, including Adaptive RED and BLUE [10, 11]. Adaptive RED adapts the *maxp* parameter to keep the average queue length between *minth* and *maxth*. BLUE, on the other hand, asserts that local queue level does not provide sufficient information about the level or nature of congestion forcing RED to require a range of parameter settings to operate optimally. BLUE replaces the RED congestion decision with an algorithm that is based on packet loss and link utilization. BLUE claims to reduce packet loss rates (compared to RED) even when operating with small capacity queues. In ongoing work we found that aspects of BLUE do not map well to the DOCSIS MAC protocol although the premise of BLUE, that queue level alone does not provide sufficient detail of the level of congestion, is relevant. For the results presented in this paper, we limit the scope to drop tail and RED managed queues.

C. Related Work

The problem of sizing router buffers has been thoroughly studied in the context of wired networks. The work in [12] finds that due to the dynamics of TCP, a router needs to buffer an amount equal to the effective BDP that is based on the average RTT of active flows. The work in [15] suggests that the BDP rule of thumb is incorrect for backbone routers subject to many flows and should be scaled by *1/SQRT(n)* where *n* is the number of active flows. Rules-of-thumb involving maximum link delay contributions, such as suggestions that routers have at least 250 ms of buffering, have also been discussed [13, 14]. In spite of the volume of academic research in the area of buffer management, there has been little attention paid to the issue in cable access networks. Aside from several measurement studies designed to provide insight in how service providers are actually managing the upstream buffer [3, 4, 15], to the best of our knowledge, the issue of how to optimally manage upstream DOCSIS queues has not been explored.

III. METHODOLOGY

We have developed a simulation model of DOCSIS 1.1/2.0 for the *ns2* simulator[1]. In this section we describe the simulation model and then summarize the experimental methodology used for the research. Refer to [19] for further details of the simulation model.

A. Simulation Model

Figure 1 illustrates the network that is modeled by the simulator. In the figure, the box on the left represents a cable modem. Packets from the subscriber's network arrive at the CM where they are stored in per service flow queues as they await an upstream transmission opportunity. For simplicity, we add a traffic generator and a TCP/IP stack to the CM node (these components of the model are not illustrated in Figure 1). Requests for upstream bandwidth (for best effort service flows) arrive at the CMTS and are queued and processed in FCFS order. The CM can request a transmission opportunity by issuing a contention-based request. The CMTS might chose to grant the CM a partial request in which case the DOCSIS fragmentation process is utilized. It is also possible for the CM to avoid contention by piggy-backing a request on a scheduled upstream data transmission. Our simulation model supports any of these possibilities. While the simulation model supports other types of DOCSIS services such as Unsolicited Grant Service (UGS), and real-time Polling Service (rtPS), the results presented in this paper focus exclusively on best effort flows.

We have implemented the RED algorithm as defined in [5]. In our implementation, the algorithm is executed each time a packet arrives from the subscriber network that has to be queued in the upstream service flow buffer. The parameters are the buffer capacity (*Qmax*), the maximum drop probability for an early drop (*maxp*), and the average queue weight factor (*wp*). The buffer capacity is an experimental parameter that ranges from 4 to 1024 packets. The RED *maxth* parameter is set to ½ buffer capacity. The *minth* parameter is 1/10 the buffer capacity. The units for the buffer capacity and the *minth* and *maxth* parameters are in packets. The *maxp* parameter specifies the maximum drop probability that would ever be applied. The original RED algorithm used a maximum drop probability of 0.02, however it has been pointed out that the value should be much higher [16]. We experimented with different values and selected a default value of 0.5. The weight factor is the time constant used in the low pass filter that maintains the average queue level. We kept *wp* at the recommended value of 0.002.

B. Experimental Design

Figure 1 illustrates the simulated network. CMs 1 through *N* attach to a 42.88 Mbps downstream channel and a 30.72 Mbps upstream channel. The model assumes ideal channels. Two types of traffic generators are used in the simulations: FTP and VoIP. Depending on the experiment, the FTP send side is attached to either the wired server or to one of the CMs. We model VoIP sessions as an 'always-on' stream of G.711 voice data using a CBR traffic generator configured to send two voice frames concatenated into a single IP packet every 20 milliseconds. At the end of a simulation, statistics summarizing the loss and latency experienced by each VoIP flow is collected. The E-model is a useful tool for predicting call quality over a packet network based on measurable network statistics such as loss rate and latency. We have implemented an E-model VoIP performance metric in *ns2* based on the ideas presented in [17, 18]. The metric estimates the R-value which describes the call quality. The R-values are mapped to a human-oriented, perceived quality assessment based on the following scale: 100-90: best quality, 90-80: high call quality, 80-70: medium quality, 70-60: low quality, 60-0: poor quality.

As illustrated in Figure 1, one CM is under observation (referred to as the monitor CM) and one or more additional CMs provide competing network traffic (referred to as background CMs). An experiment is defined by the number of background CMs that are active, by the workload generated by the monitor CM and by the additional CMs, and the queue management policy that is in effect at the upstream CM queue.

The objective of the experiments is to explore the impact of system design and configuration settings on application

[1] The open source Networking Simulator is available at http://www.isi.edu/nsnam/ns/

performance. Elemental to the analysis is our definition of the "optimal queue capacity". We assume that the optimal queue capacity is the minimum queue capacity provisioned within a cable modem that supports maximum throughput and minimum packet delay. At the simplest level, the optimal setting will allow the network to remain at the "knee" of the throughput-load curve [20]. The parameter space of the experiments involves three aspects:

Path distance: The propagation delay of the link connecting the wired server is set to one of two values: 2 ms and 80 milliseconds. These two settings correspond to an end-to-end RTT (when the path is not congested) of 20 ms and 175 milliseconds respectively. The default settings of the simulation model restrict the maximum throughput that can be achieved by a single service flow to less than 9.0 Mbps[2]. Based on these assumptions, the BDP for each setting is 21.97 Kbytes and 192.26 Kbytes respectively which corresponds to 15 and 131 full size (i.e., 1500 byte) IP packets. The BDP for the experiment that provides an approximation of a DOCSIS 3.0 scenario is 1.05 Mbytes or roughly 700 full size IP packets.

Traffic intensity and workload: The number of CMs that provide background traffic will be varied from 0 to 50. This allows the analysis to consider network congestion levels ranging from unloaded to heavily congested.

Upstream queue capacity and management strategy: The maximum capacity of the upstream buffer is varied from 4 packets to 1024 packets. The results presented in this paper are limited to drop tail and RED queue management policies.

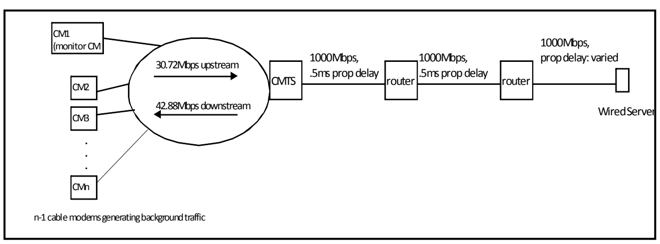

Figure 1. Simulated Network

We study two scenarios. The first, which provides baseline results, involves a single TCP/Sack session configured to send data upstream using an FTP traffic generator. In addition, the single CM sources an upstream best-effort VoIP flow. Using this scenario, seven experiments are defined that involve different combinations of maximum TCP receiver advertised window settings (32 Kbytes through 2.25 Mbytes), path RTT settings (uncongested RTTs ranging from 20 ms to 175 ms), and DOCSIS properties. Experiments 1.1 through 1.6 involve a typical DOCSIS 2.0 configuration. In Experiment 1.7, we modify the simulation parameters to support a higher link rate. This experiment loosely models a DOCSIS 3.0 scenario involving 3 upstream channels[3].

While Scenario 1 involves local congestion generated by a single CM, Scenario 2 involves network-wide congestion caused by many CMs. A downstream TCP flow configured with an FTP traffic source is configured to operate between the observed CM and the wired server. An upstream VoIP flow is also established over this path. In addition, a variable number of CMs are configured (1 through 50) each adding an upstream TCP/Sack connection configured with an FTP traffic generator. All the experiments in this scenario involved the network settings defined in Experiment 1.6 from the first scenario. Table 1 summarizes the parameter combinations that are used in both scenarios.

Experiment ID	BDP (Packets)	Max Rx Advertised Window (Packets)	Min Path RTT (ms)
EXP 1.1	14.25	22	20
EXP 1.2	14.25	44	20
EXP 1.3	14.25	685	20
EXP 1.4	131.25	22	175
EXP 1.5	131.25	44	175
EXP 1.6	131.25	685	175
EXP 1.7	700.0	1500	175
EXP2.1-EXP2.6	131.25	685	175

Table 1. Experimental Parameters

IV. ANALYSIS

A. Scenario 1

Figure 2 illustrates the results from the seven experiments from Scenario 1. For each experiment, the upstream service flow queue capacity ranged from 4 packets to 1024 packets. At the end of a simulation run, the average throughput obtained by the upstream TCP connection is computed. The results are plotted with the queue capacity on a logarithmic scale on the X-axis. Figure 2a shows that the experiments that were limited to a maximum TCP window less than or equal to 64 Kbytes experienced the lowest achieved throughput over the high RTT path (i.e., they were window-bound). The figure shows that Experiments 1.1 through 1.6 converge to a throughput of 8.3 Mbps which corresponds to

[2] While the upstream channel rate can be as high as 30.72 Mbps in a DOCSIS 2.0 network, the effective data available to CMs will be limited by many factors including provisioned service rates, CM channel access behaviors, and CMTS scheduling related behaviors that include the frequency of upstream scheduling assignments (referred to as the MAP time), allocation 'packing' strategies, and allowed maximum grant sizes. As an example, in a DOCSIS system that allows 'uncapped' service rates (i.e., a CM is not restricted to a bandwidth limits that are less than the channel capacity) and a MAP time setting of 0.002 seconds, the maximum upstream throughput a single flow can obtain is 9.0 Mbps. This example corresponds to three 1500 byte IP packets sent upstream every other MAP time. Refer to [19] for further details.

[3] We model bonded upstream channels by increasing the channel capacity to 90 Mbps, decreasing the MAP time from 0.002 to 0.001 seconds, and increasing the maximum number of packets that can be included in a concatenated frame from 3 to 8. In ongoing work we are extending the simulation model to accurately model the DOCSIS 3.0 protocol.

the maximum throughput that can be achieved by a single flow for the given configuration. Experiment 7 converges to 48 Mbps which is the highest throughput expected for that configuration. For the experiments that were not constrained by the TCP receiver advertised window, the optimal queue capacity was equivalent to a BDP amount of data.

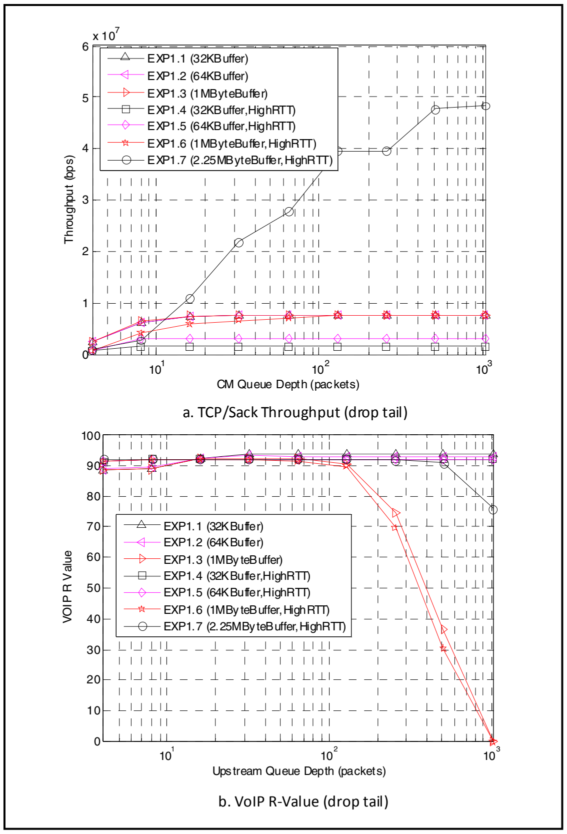

Figure 2. Scenario 1 Results

Figure 2b suggests that flows configured with large TCP receiver advertised window settings (i.e., Experiments 1.3 and 1.6) might experience an extreme drop in VoIP call quality once the upstream buffer capacity exceeds 128 packets. An upstream TCP flow that has a large maximum TCP window can consume up to a window's worth of buffer at the queue. As the CM upstream queue capacity increases to the maximum of 1024, the queue delay experienced by upstream packets (including the VoIP packets) approaches 1 second. It is well known that once the mouth-to-ear latency exceeds 0.3 seconds, the perceived call quality (R-Value) drops significantly [17]. We observe in Figure 2b that toll quality is achieved for all experiments as long as the queue capacity does not exceed a BDP amount of data. If we repeat the scenario using RED, the average queue delay is reduced such that the R-value for all experiments (including Experiments 1.3 and 1.6) does not drop when the queue capacity is large. However when using RED, the throughput achieved in EXP1.7 is less than the drop tail case (it is in fact dependent on the RED parameter settings).

B. *Scenario 2*

Figure 3a plots the average throughput achieved by each upstream TCP flow in the Scenario 2 experiments. Figure 3b plots the queue delay experienced by upstream TCP packets. The dashed vertical line that goes through each plot represents the queue capacity that matches the BDP for the path. In Figure 3a, the arrowed line segments labeled 'Optimal Points' connect the points representing the optimal queue capacity for each experiment. The optimal queue capacity is a function of the network load. For the system configuration defined by Scenario 2, the results suggest that the optimal queue capacity is less than or equal to the BDP and is in fact a function of the number of active CMs. Each optimal queue capacity point observed in Figure 3a can be mapped to Figure 3b to find the expected queue delay for the experiment. By expanding the figure in Figure 3b (we do not show this) we find that the queue delay at the optimal queue capacity points range from 60 ms to 110 ms.

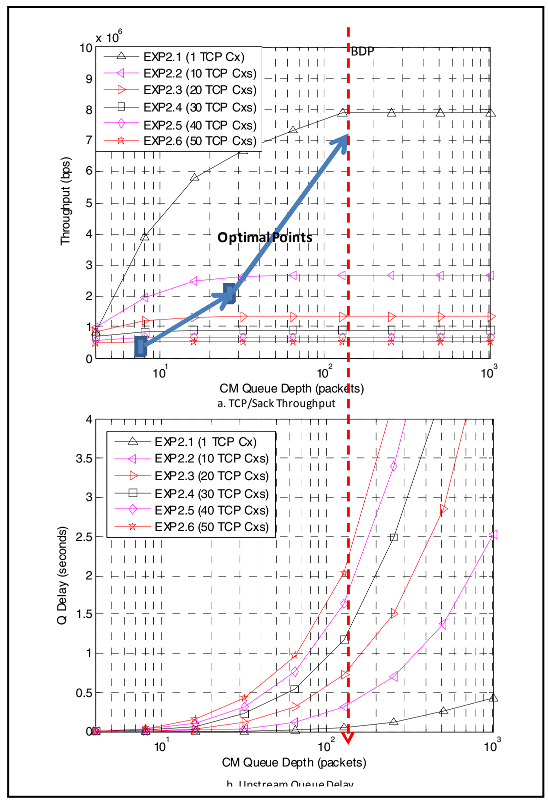

Figure 3. Average Upstream TCP Throughput (Scenario 2)

To better map the results of Scenario 2 to application performance, we plot the VoIP metric R-Value associated with the experiments. Figure 4a plots the drop tail results (i.e., which correspond to the results described by Figure 3). Figure 4b plots the results for the identical experiments except that the upstream CM queue is managed by RED. For both drop tail and RED, the R-value drops significantly as the number of competing upstream TCP flows increase beyond 20. While Figure 3b suggests packet delay at the optimal queue capacity setting in each experiment is less than 110 ms, Figure 4a shows that toll quality VoIP (i.e., an R-value of 70 or larger) is not possible when the number of flows exceed 20. A breakdown of the components of VoIP impairment indicates that the major impairment when the queue capacity is less than 32 packets is packet loss and then switches to latency for larger queue capacities.

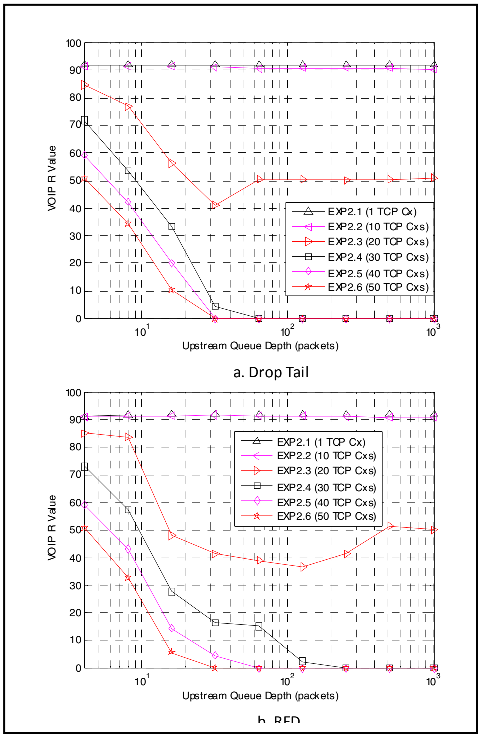

Figure 4. VoIP Results (Scenario 2)

V. CONCLUSIONS

In the research reported in this paper, we have explored the relationship between cable modem upstream buffer management and best effort application performance. We defined the optimal queue capacity as the minimum queue capacity that maximizes throughput and minimizes packet delay. Our results agree with previous work in wired Internet router contexts that suggest that the queue capacity should not exceed a bandwidth*delay product (BDP) amount of data. For upstream service flow queues that are provisioned to hold a BDP of data, the queue capacity will be too large during periods of congestion. We conclude that widely used rules-of-thumb for sizing router buffers based on a BDP are applicable and can serve as an upper limit. However the optimal queue capacity will likely fluctuate over time depending on the available data rate.

We found that RED offers minimal benefits compared to drop tail when managing an upstream CM queue. This is not surprising as RED is primarily intended for high speed routers that are subject to hundreds of flows at any given time. We did observe that RED can maintain a lower average queue level when the queue capacity is large. However, the exact behavior of RED depends on network load as well as on the RED parameter settings.

We focused on a small number of configurations and scenarios. Our study is also limited because we did not consider service rates, scenarios involving non-TCP flows, or scenarios involving multiple TCP flows at cable modems. We conjecture that our results are applicable in these broader scenarios however further study is required. Based on our results, we conclude that an adaptive queue management algorithm is required to maintain a consistent balance between throughput and delay. We plan to address this in future work.

REFERENCES

[1] J Horrigan, "Home Broadband Adoption 2009", Pew Internet & American Life Project, June 2009, available at: http://www.pewinternet.org/~/media//Files/Reports/2009/Home-Broadband-Adoption-2009.pdf.

[2] Data-Over-Cable Service Interface Specifications DOCSIS 3.0, CableLabs, May, 2009.

[3] M. Dischinger, A. Haeberlen, K. Gummadi, S. Saroiu, "Characterizing Residential Broadband Networks", Proceedings of the Internet Measurement Conference (IMC'07), October 2007.

[4] M. Claypool, R. Kinicki, M. Li, J. Nichols, "Inferring Queue Sizes in Access Networks by Active Measurement", Proceedings of the 5th Passive and Active Network Measurement Workshop, April, 2004.

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

[6] H. Peyravi, "Medium Access Control Protocols for Space and Satellite Communications: A Survey and Assessment", IEEE Communications Magazine, March 1999.

[7] M. Christiansen, K. Jeffay, D. Smith, "Tuning RED for Web Traffic", IEEE/ACM Transactions on Networking, Vol. 9, No.4, June 2001.

[8] M. May, J. Bolot, C. Diot, B. Lyles, "Reasons not to Deploy RED", Proceedings of the 7th International Workshop on Quality of Service (IWQoS'99), June 1999.

[9] T. Ott, T. Lakshman, L. Wong, "SRED: Stabilized RED", Proceedings of the IEEE Infocom, March 1999.

[10] S. Floyd, R. Bummadi, S. Shenker, "Adaptive RED: An Algorithm for Increasing the Robustness of RED's Active Queue Management", Technical report, ICSI, 2001. Available at http://www.icir.org/floyd/papers/adaptiveRed.pdf

[11] W. Feng, K.Shin, D. Kandlur, D. Saha, "The Blue Active Queue Management Algorithms", IEEE/ACM Transactions on Networking, Vol. 10, No.4, Aug 2002.

[12] C. Villamizar, C. Song, "High Performance tcp in ansnet", ACM Computer Communications Review, October 1994.

[13] C. J. Fraleigh, "Provisioning Internet Backbone Networks to Support Latency Sensitive Applications", PhD Thesis, Stanford University, Department of Electrical Engineering, June 2002.

[14] C. Fraleigh, F. Tobagi, C. Diot, "Provisioning IP Backbone Networks to Support Latency Sensitive Traffic", Proceedings of the IEEE Infocom, April 2003.

[15] G. Appenzeller, I. Keslassy, N. McKeown, "Sizing Router Buffers", Proceedings of ACM SIGCOMM, September 2004.

[16] S. Floyd, "RED: Discussions of Setting Parameters", available at http://www.icir.org/floyd/REDparameters.txt , Nov. 1997.

[17] ITU-T Recommendation G.107, "The E-Model, a Computational Model for Use in Transmission Planning", December, 1998.

[18] R. Cole, J. Rosenbluth, "Voice over IP Performance Monitoring", Proceedings of ACM SIGCOMM, April 2001.

[19] J. Martin, M. Westall, "Validating an 'ns' Simulation Model of the DOCSIS Protocol", Proceedings of the 2006 International Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS'06), July 2006.

[20] L. Kleinrock, "Power and Deterministic Rules of Thumb for Probabilistic Problems in Computer Communications", Proceedings of the International Conference on Communications (ICC'79), June 1979.