Modeling The DOCSIS 1.1/2.0 MAC Protocol

Jim Martin
Department of Computer Science
Clemson University
Clemson, SC
jim.martin@cs.clemson.edu

Nitin Shrivastav
Department of Computer Science
North Carolina State University
Raleigh, NC
nshriva@unity.ncsu.edu

***Abstract*—We have developed a model of the Data over Cable (DOCSIS) version 1.1/2.0 MAC and physical layers using the '*ns*' simulation package. In this paper we present the results of a performance analysis that we have conducted using the model. The main objective of our study is to examine the performance impact of several key MAC layer system parameters as traffic loads are varied. We focus on the DOCSIS best effort service. We measure the level of ACK compression experienced by a downstream TCP connection and show that even under moderate load levels DOCSIS can cause TCP acknowledgement packets to compress in the upstream direction leading to bursty (and lossy) downstream dynamics. We explore the use of downstream rate control on network performance and find that it does reduce the level of ACK compression for moderate load levels compared to an equivalent network scenario without rate control. However, as the number of active subscribers on a given channel increase, the level of ACK compression grows implying that other mechanisms should be looked at to improve performance during periods of high usage.**

Keywords- Broadband Access; DOCSIS; TCP Performance;

I. INTRODUCTION

It is predicted that the number of high speed Internet (i.e., broadband) cable users in North America will increase from about 12 million (in 2002) to about 26 million by the end of 2006 [1]. Although the market is largely residential, the small office environment is also able to take advantage of the relatively low cost Internet access. Both residential and corporate traffic patterns and requirements are evolving rapidly. New applications such as peer-to-peer, audio/video streaming and VoIP are in various stages of deployment. While several approaches to broadband access over cable have been proposed (e.g., DVB/DAVIC and IEEE 802.14 [2]), the industry is converging on the architecture developed by the Multimedia Cable Network System (MCNS) group referred to as the Data-Over-Cable Service Interface Specification or DOCSIS standard[3]. The DOCSIS Radio Frequency Interface specification defines the Media Access Control (MAC) layer as well as the physical communications layer [4]. The original DOCSIS MAC interface (v1.0) provides a best effort service with simple prioritization. DOCSIS 1.1, which is currently being deployed, provides a set of ATM-like services (i.e., services equivalent to ATM's constant bit rate, non-real-time variable bit rate, unspecified bit rate). The physical layer supports a maximum downstream data rate of roughly 30.34Mbps. Upstream channels of 3.2Mhz offer maximum data rates up to 10.3.Mbps that is shared by all CMs using a TDMA based system. DOCSIS 2.0 increases upstream capacity to 30 Mbps through more advanced modulation techniques and by increasing the RF channel allocation to 6.4 Mhz.

Figure 1 illustrates a simplified DOCSIS environment. A Cable Modem Termination System (CMTS) interfaces with hundreds or possibly thousands of Cable Modem's (CMs). The Cable Operator allocates a portion of the RF spectrum for data usage and assigns a channel to a set of CMs. A downstream RF channel of 6 Mhz (8Mhz in Europe) is shared by all CMs in a one-to-many bus configuration (i.e., the CMTS is the only sender). The CMTS makes upstream CM bandwidth allocations based on CM requests and QoS policy requirements. The upstream channel is divided into 'mini-slots' which, depending on system configuration, normally contain between 8 to 32 bytes of data. The CMTS periodically sends a 'MAP' message to all CMs on a downstream channel that indicates upstream bandwidth allocation over the next 'MAP time'. The MAP provides slot assignments for particular CMs (i.e., data grants), provides opportunities for a CM to request bandwidth using a contention-based request process and identifies which slots are to be used for system overhead Figure 2 illustrates the MAP layout used in our model. DOCSIS allows a CM to describe one or more flows with a Service Flow ID (referred to as a SID). DOCSIS defines a set of upstream scheduling services that operate on a per SID basis: Unsolicited Grant Service (UGS), Real-Time Polling Service (rtPS), Unsolicited Grant Service with Activity Detection (UGS-AD), Non-Real-Time Polling Service (nrtPS) and Best Effort Service (BE).

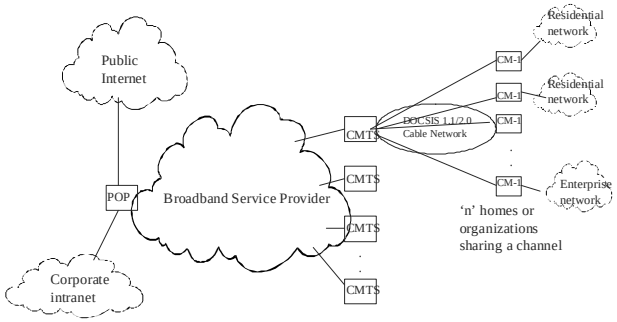


Figure 1. DOCSIS cable access environment

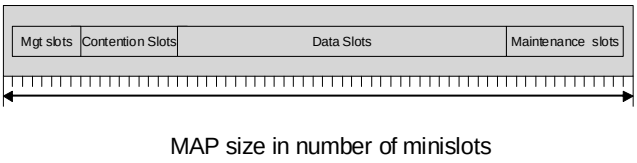


Figure 2. DOCSIS upstream TDMA channel

A critical component of the DOCSIS MAC layer is the upstream bandwidth allocation algorithm. The DOCSIS specification purposely does not specify these algorithms so that vendors are able to develop their own solutions. However any upstream bandwidth management algorithm will share a certain set of basic system parameters. These include the amount of time in the future that the scheduler considers when making allocation decisions (we refer to this parameter as the MAP_TIME), the frequency at which MAPs are issued, the frequency of contention slot offerings and the range of collision backoff times. While the settings of these parameters have a major impact on system performance, the DOCSIS MAC layer has been largely ignored by the research community.

We have developed a model of the Data over Cable (DOCSIS) 1.1/2.0 MAC and physical layers using the '*ns*' simulation package [5]. Although other simulation packages exist, *ns* provides widely accepted models of TCP and application traffic. It is the simulation tool of choice by the TCP research community. In addition to presenting the model to the community, the objective of our study is to examine the performance impact of several key MAC layer system parameters as traffic loads are varied. We focus on the DOCSIS best effort service. In addition to traditional performance metrics such as loss, delay and collision rates, it is important to assess the level of ACK compression created by the network and its subsequent impact on TCP performance. ACK compression distorts the ACK stream which can lead to bursty sender behavior and higher loss rates. It has been shown that ACK compression occurs in the Internet [6,7]. Further, it has been shown that even a relatively low level of compression leads to a significant increase in packet loss [8].

We measure the level of ACK compression experienced by a TCP connection and show that even under moderate load levels DOCSIS causes TCP acknowledgement packets to compress in the upstream direction leading to bursty (and lossy) downstream dynamics. We explore the use of downstream rate control on network performance and find that it does reduce the level of ACK compression for moderate load levels compared to the same network but without rate control and subject to equivalent application loads. However, as the number of active subscribers on a given channel increase, the level of ACK compression grows implying that other mechanisms should be looked at to improve performance during periods of high usage.

The rest of this paper is organized as follows. The next section presents the operation and features of our DOCSIS model. We identify and demonstrate the tuning parameters using simple FTP upstream and downstream scenarios. We end the paper with a discussion of related work, present conclusions and identify future work.

II. SUMMARY OF THE MODEL

The model implements the basic DOCSIS architecture defined in [3]. At initialization time, each CM registers itself with the CMTS. At least two service flows are created for each CM, one in the upstream and one in the downstream direction. Currently a CM supports a single default best effort service flow and any number of UGS services.

Packets sent over the downstream channel are broken into 188 byte MPEG frames each with 4 bytes of header and trailer. The model accounts for physical overhead including forward error correcting data. For our simulations we assume a FEC overhead of 4.7% (8% in the upstream direction) [9]. We model this overhead by reducing downstream channel capacity by 4.7% (8% in the upstream direction). The downstream channel supports an optional token bucket based service rate. The rate and maximum token bucket size are simulation parameters. Each SID service queue is treated in a first come first serve manner. Depending on traffic dynamics, queueing can occur at either the SID queue or the downstream transmission queue. The maximum size of either queue is a simulation parameter.

All CMs receive periodic MAP messages from the CMTS over the downstream channel that identify future scheduling opportunities over the next MAP time. When the CM has data to send, and if it has been provisioned with a periodic grant, the CM can send at its next data grant opportunity. For best effort traffic, it is more likely that bandwidth will be requested during contention transmission opportunities specified by the MAP. DOCSIS allows the CM to combine multiple IP packets into a single DOCSIS frame by issuing a concatenated request. If a CM receives a grant for a smaller number of mini-slots than were requested (even for a concatenated request), the CM must fragment the data to fit into the assigned slots. To minimize the frequency of contention-based bandwidth requests, a CM can piggyback a request for bandwidth on an upstream data frame. To help us evaluate the benefits of these features, we have configuration parameters that enable fragmentation, piggybacked requests and the maximum number of packets that can be included in a concatenated MAC frame.

Each time the CMTS generates a MAP, the upstream bandwidth scheduling algorithm examines all existing requests for bandwidth and implements an earliest deadline first scheduling policy. All UGS service requests are scheduled and whatever bandwidth remains is shared by best effort requests based on a first-come-first-served discipline. A UGS request fails to meet a deadline is considered a provisioning problem. During periods of high load, the CMTS maintains a queue of pending requests. Once a new request for bandwidth arrives from a CM, the CMTS responds with a data grant pending indicator to the CM. This informs the CM that the contention-based request succeeded and to expect a data grant at some point in the future.

The scheduler has a configured MAP time (i.e., a MAP_TIME parameter) which is the amount of time covered in a MAP message. The MAP_FREQUENCY parameter specifies how often the CMTS sends a MAP message. Usually these two parameters will be the same (in the 1 – 10 millisecond range). The scheduling algorithm supports

dynamic MAP times through the use of a MAP_LOOKAHEAD parameter which specifies the maximum MAP time the scheduler can 'lookahead'. If this parameter is 0, MAP messages are limited to MAP_TIME amount of time in the future. If set to 255, the scheduler may allocate up to 255 slots in the future. The MAP_LOOKAHEAD can be as large as 4096 slots, however we will see in the next section that a large MAP_TIME can severely degrade performance.

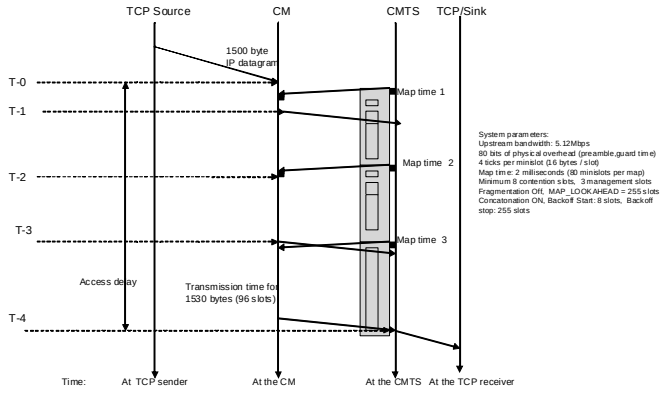


Figure 3. Upstream transmission

Figure 3 illustrates the upstream transmission of a 1500 byte IP datagram from a TCP source directly connected to a CM to a sink connected to the CMTS. We show the upstream slot assignment with a time line where time progresses in the downwards direction. Based on the system parameters listed in Figure 3), the nominal MAP size is 80 slots however 96 slots are required to carry the entire packet. Each MAP provides a configured number of contention slots defined by the CONTENTION_SLOTS parameter[1]. When the datagram arrives, the CM requests a transmission opportunity by issuing a contention based request. The DOCSIS contention algorithm requires the CM to randomly select a number of contention slots to skip before sending. This number is drawn from a range between 0 and a value that is sent by the CMTS in the MAP (the BKOFF_START). For this example the CM skips 2 slots and is therefore able to transmit a 16 byte bandwidth request message during MAP time number 1. After transmission, if the CM does not receive an indication that the request was received, the CM must randomly select another number of contention slots to skip before retrying the contention-based transmission request. The CM is required to exponentially backoff the range with each collision with the maximum backoff specified by a maximum backoff range parameter contained in each MAP. The CM will drop the packet after it has attempted to send the request 16 times.

In our example, we assume collisions do not occur. The small dark square box positioned before each MAP time in the figure represents the transmission of the MAP message in the downstream direction. Our model sends the MAP at the beginning of each MAP time. Each MAP describes the next MAP time (i.e., rather than the current MAP time which has already been described by the previous MAP). The IP packet arrives at the CM at time T-0. The CM sends the bandwidth request message at time T-1 and receives the data grant at time T-2 (i.e., in the MAP describing MAP time 3). The grant occurs in MAP time 3 The CM sends the frame at Time T-3 and is received by the CMTS at time T-4. The time between T-4 and T-0 is the access delay which represents the total time a packet is delayed over the DOCSIS access network.

III. ANALYSIS

We define the maximum upstream throughput as $T_{\max-us} = \frac{BytesSent*8}{D_{total-access}}$, where *BytesSent* is the total data sent in the request and $D_{total-access}$ is the total access delay from the time the packet arrives at the CM until when the data has been received by the CMTS. The best case scenario illustrated in Figure 3 assumes no system delay. The minimum access delay (i.e., $D_{total-access}$) is roughly 2 MAP times plus the transmission time of the frame. Assuming a MAP time of .002 seconds, no concatenation, framing and physical layer overhead of 30 bytes and an upstream bandwidth of 4.7Mbps (adjusted for FEC overhead), the highest throughput to be expected is roughly 1.8Mbps. We have verified this with the simulation model with a single TCP connection over an unloaded network configuration.

When a packet arrives at a CM to be sent in the upstream direction, it is unlikely to experience the ideal scenario depicted in Figure 3. Delay caused by the contention-based bandwidth request process along with the delay incurred waiting for the CMTS to issue a grant can increase the total access delay significantly. If multiple packets have arrived at the CM, it can piggyback a request for additional bandwidth on a data frame. This has significant benefits in a busy system as it avoids the contention process leading to lower upstream channel collision rates.

The MAP_LOOKAHEAD parameter is intended to enhance performance when either a large packet might not fit in a single MAP time or when two packets travel back-to-back. In the first case, the intent is to provide a knob that can reduce the amount of fragmentation. In the second case, the intent is to improve efficiency by increasing the average frame size (i.e., and thereby reducing the overhead associated with multiple smaller frames). If the CM issues a concatenated request (i.e., for 3056 bytes), the CMTS will grant the request if its MAP_LOOKAHEAD is large enough (at least 219 slots for the configuration described in Figure 3) and if it can meet other scheduling requirements. Assuming best case conditions, the minimum access delay will be 2 MAP times plus the transmission time of the concatenated frame. For our example, this results in a maximum throughput of roughly 2.6Mbps which we have validated in the model. If we decrease the MAP_TIME to .001 second, the maximum TCP upstream throughput increases to 3.4 Mbps.

We provide a similar discussion for downstream. Figure 4 illustrates a downstream scenario involving the arrival of 4 back-to-back segments at the TCP sink. We assume that the

[1] An actual DOCSIS implementation would adjust the number of contention slots per MAP dynamically. We use a static setting to help us explore the impact of different settings on performance.

sink generates two acknowledgement packets (i.e., one ACK for every other segment that arrives) which arrive at the CM separated only by the downstream transmission time of two back-to-back segments. We assume concatenation is enabled (with a concatenation limit of 2 packets) allowing the CM to request bandwidth for the transmission of the two ACK packets in a single frame. In the best case, the ACK's will be received at the CMTS roughly within 2 MAP times from the time the ACKs arrive at the CM plus the transmission time of the data frame. Therefore, assuming a MAP_TIME of .002 seconds and assuming rate control is disabled, we expect a maximum downstream throughput by any single CM to be limited to about 12Mbps (4 segments metered out every 4 milliseconds). After setting the TCP window size very large and also making sure the buffers at the CM were large we did observe a maximum downstream TCP throughput of 12Mbps.

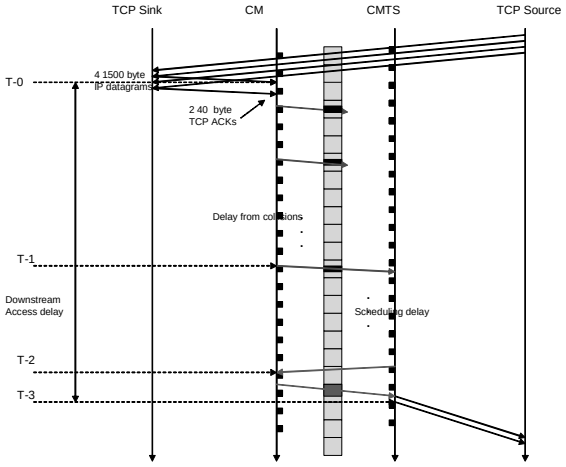


Figure 4. Downstream scenario

The maximum downstream throughput will be roughly 26Mbps after taking into account physical and link level headers. The TCP throughput however is limited by the ACK rate which is the amount of data that is acknowledged per second (as observed by the TCP sender). The maximum downstream TCP throughput is given by $T_{\max-ds} = \frac{BytesACKed*8}{D_{total-access}}$ where *BytesACKed* is the amount of data that is acknowledged in the bandwidth request. Increasing upstream or downstream channel capacities will not increase the single connection TCP throughput. Throughput will increase only if the amount of data acknowledged in a data frame increases or if the MAP_TIME is reduced.

We base further analysis on two simple scenarios involving the cable network illustrated in Figure 5. For simplicity, we assume that a CM is a TCP /IP host with one or more applications. The first scenario involves upstream FTP traffic and the second scenario examines the downstream case. For each scenario we ran three experiments each with a different system parameter under study: the MAP_TIME, the number of contention slots per MAP (i.e., CONTENTION_SLOTS) and the contention backoff start range (BKOFF_START) respectively. The MAP_TIME varies between .001 and .01 seconds. The CONTENTIONS_SLOTS varies between 3 and 20 slots per map. The BKOFF_START ranges from 8 slots to 128 slots. For each experiment, we conducted specific simulation runs designed to show system performance for these values of the system parameters as the traffic load increased (i.e., as the number of CMs increased). For brevity, we focus only on the results for experiment 1 which pertains to the MAP_TIME parameter.

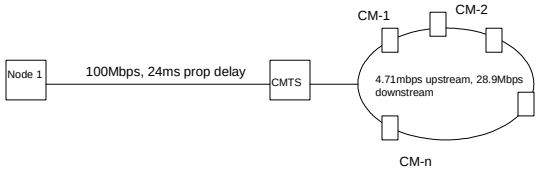


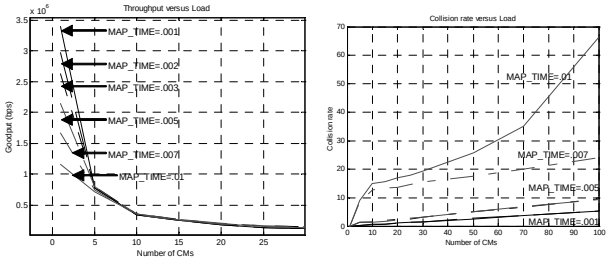
Figure 5. Simple cable scenario

Model Parameters
Upstream bandwidth 4.71Mbps, 80 bits of physical overhead (preamble,guard time)
Downstream bandwidth 28.9Mbps, 4 bytes overhead per 188 bytes of data.
4 ticks per minislot (16 bytes / slot)
Map time: 2 milliseconds (80 minislots per map)
Minimum 8 contention slots, 3 management slots
Fragmentation Off, MAP_LOOKAHEAD = 255 slots
Concatonation ON
Backoff Start: 8 slots, Backoff stop: 128 slots

Figure 6. Model parameters and experiment description

A. Upstream FTP Traffic Scenario

Each CM has a single TCP FTP traffic source which sends continuously to a sink located at node 1. Looking first at the experiment involving the MAP_TIME, we run a set of 10 simulations with the MAP_TIME initially set to .001 seconds and vary the number of CMs from 1 to 50. Next, we set the MAP_TIME to .002 and do another 10 iterations and so on. Figures 7a illustrates the throughput experienced by one of the FTP connections as the MAP_TIME varies. As expected we see that in an unloaded system (i.e., less than 5 CMs), throughput is highest for small MAP_TIME values. However, for higher loads, the MAP_TIME setting does not make a difference. Figure 7b illustrates the upstream collision rates. The results shows that that the collision rate increases with MAP_TIME. This is due to our contention slot allocation policy which allocates a fixed number of slots per MAP. The number of offered contention slots decreases as the MAP_TIME gets large.

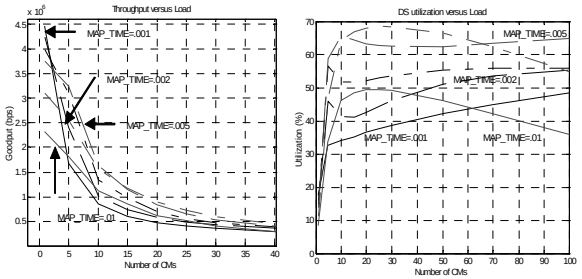


a. Upstream throughput b. Upstream collision rate

Figure 7. Upstream experiment 1 results

B. *Downstream FTP Traffic Scenario*

The downstream FTP scenario utilizes the model configured per the settings shown in Figure 6. FTP traffic generators are located at node 1 and the TCP sinks are located at the CMs (one TCP sink per CM). We have disabled downstream rate control and so each FTP connection attempts to obtain as much bandwidth as possible. Figure 8a suggests that throughput is optimized with a MAP_TIME setting of .001 but only for the single CM case. Once there are 5 or more CMs competing, larger values of MAP_TIME tend to yield slightly better performance. As the number of CMs increase, loss occurs in the downstream direction but only for MAP_TIMES of .001, .002 and .003 (increasing from 0 to about 7% as the load increases). Smaller settings of MAP_TIME lead to bursty TCP dynamics which increases the TCP loss rate, the TCP timeout rate and reduces the link utilization (confirmed in Figure 8b).



a. Downstream throughput

b. Downstream utilization

Figure 8. Downstream experiment 1 results

The TCP loss rate is significantly higher for lower MAP_TIME settings in Experiment 1 because of the interaction between the MAP_TIME and the timing of the ACK stream in the upstream direction. The MAP_TIME determines how long a CM waits before it is able to send an ACK. During this delay, additional packets might arrive from the downstream direction causing ACK's to be queued at the CMs upstream queue. The impact is that downstream TCP connections will experience ACK compression.

We are interested in observing the impact that downstream rate control has on network dynamics and in particular on the level of ACK compression. We ran experiment 1 again but applied a downstream service rate of 512000 bps. Because of the smoothing effect of rate control, the network operates at a higher utilization level (80% versus about 55%) and with lower loss rates (roughly .5% for the largest CM load level). At a load of less than 30 CMs, the level of ACK compression is low compared with the same load level. At higher loads the level of ACK compression and the TCP burstiness is about the same as that observed in the runs without rate control. However while in high load situations, the network is able to transfer more application data, the utilization remains high and the loss rates are 0 (for low MAP_TIMES). Even though the traffic arrival process becomes bursty, the per CM downstream rate control queues absorb the connection burstiness and effectively smooths the arrival process at the downstream transmission queue.

C. *Summary of Results*

While all three of the system parameters under investigation directly impact system performance, the MAP_TIME is perhaps the most critical. For the upstream FTP scenario, TCP throughput decreases as the MAP_TIME increases. For the downstream FTP scenario, performance drops as the MAP_TIME setting decreases in size because the level of ACK compression increases. We have also shown that both upstream and downstream TCP throughput can be limited not by channel rates but by low packet transfer rates in the upstream direction. The packet rate available to a CM will decrease as the MAP_TIME setting increases or as upstream access delay increases. The drawback to small MAP_TIMES is overhead. Each MAP message consumes CMTS processing cycles and more importantly downstream bandwidth. Although we found that the MAP_TIME should be less or equal to .002 seconds, more analysis needs to be done to further validate this observation.

Although we did not present the results from the other experiments, the CONTENTION_SLOTS and BKOFF_START parameters also impact performance as they determine the level of collisions. If the CMTS offers too few contention opportunities when the network is subject to even moderate downstream traffic, performance degrades as the upstream channel will not be able to support the bandwidth required for upstream ACKs. However too many contention slots can lead to inefficiency. We found that a value of at least 12 contention slots is needed although this parameter could be dynamically adjusted. The contention backoff start range also has a tradeoff. A low starting range helps keep the upstream access delay low, especially during periods of low usage. A higher starting range can reduce the collision rate at higher loads. We found that the backoff starting range should be at least 8 contention slots.

The concatenation limit and the MAP_TIME have the greatest impact on the level of ACK compression. The purpose of concatenation is to improve efficiency in the upstream direction. For downstream traffic, we saw the reverse might happen as extreme levels of ACK compression leads to higher loss rates and poor network utilization. Other scenarios, such as those involving variable size packets from multiple upstream flows, would benefit from higher levels of concatenation.

IV. RELATED WORK

The cable industry has been in a continual state of evolution since its inception. Delays in the development of a data over cable standard by the 802.14 IEEE Working Group lead the industry to adopt the DOCSIS architecture. While the intent of the 802.14 effort was to provide ATM services over a hybrid fiber coaxial (HFC) medium, the operation of the MAC layer is similar to that supported by DOCSIS. Previous research studied the performance of TCP over an 802.14 MAC/PHY interface [10,11]. In [10], the authors show that an HFC network presents difficulties for TCP because it provides an asymmetric path with potentially high loss rates (possibly as high as 10-50%). As long as the cable plant is well engineered, packets are seldom dropped. The authors in [11] found that TCP throughput is low primarily due to ACK compression.

The authors propose two solutions: one involving piggybacking and a second involving TCP rate smoothing by controlling the ACK spacing. The authors found that piggybacking can help reduce the burstiness associated with the ACK stream in certain situations. However it is limited in its abilities to effectively match offered load over a range of operating conditions. The author's second solution is to control the TCP sending rate by measuring the available bandwidth and calculating an appropriate ACK rate and allowing the CM to request a periodic grant that provides sufficient upstream bandwidth to meet the required ACK rate. Our work confirms and further explores these findings in a DOCSIS environment.

The performance of TCP over asymmetric paths has been thoroughly studied [12,13,14]. A network exhibits asymmetry with respect to TCP performance if achieved throughput is not solely a function of the link and traffic characteristics of the forward direction but in fact depends on the impact of the reverse direction. Prior work was focused on highly asymmetric paths with respect to bandwidth where the normalized asymmetry level (i.e., the ratio of raw bandwidths to the ratio of packet sizes in both directions) is on the order of 2-4 [12]. In DOCSIS, depending on the service rate configuration, the level of bandwidth asymmetry is small (or nonexistent). Instead, DOCSIS exhibits a packet rate asymmetry caused by the upstream TDMA MAC layer. The impact on downstream TCP connections that we have observed (i.e., reduced bandwidth and possibly poor network behavior caused by ACK compression) is similar to the observed behavior of TCP over asymmetric paths[12]. Various methods have been proposed to alleviate the problems [12,13,14,15] including header compression and modified upstream queue policies (drop-from-front, Ack prioritization, Ack filtering). A CM that supports ACK filtering could drop 'redundant' ACKs that are queued. While this would increase the acknowledgement rate, it would also increase the level of ACK compression. ACK reconstruction could be implemented in the CMTS to prevent the increased level of ACK compression from affecting performance.

V. CONCLUSIONS AND FUTURE WORK

We have presented a simulation model of the DOCSIS1.1/2.0 MAC level protocol. We are in the process of validating the model using both analytic and measurement based techniques. Based on initial results, we believe the model is accurate. The focus of this paper has been to analyze system performance when subject to varying loads of best effort traffic as the MAP_TIME, the CONTENTION_SLOTS and the BKOFF_START parameters are varied. We have seen that DOCSIS impacts the level of ACK compression experienced by TCP connections and more importantly that it leads to high packet loss rates and poor downstream channel utilization. The use of concatenation, the number of packets sent in a concatenated frame along with the access delay time determine the level of compression. The access delay time increases with the number of competing CMs, the level of upstream traffic and with the system parameters that we have examined. We have shown that downstream rate control can stabilize network performance for moderate loads. However, as the load increases, the access delay increases resulting in equivalent levels of ACK compression as when rate control is disabled. At high loads, downstream rate control will smooth the downstream arrival process such that loss rates will be low. However, downstream rate control will not protect the outside network from bursty TCP behavior. We plan on exploring upstream rate control to minimize ACK compression. A further approach is to restore the ACK stream after the ACKs have been received by the CMTS. This would serve two purposes. First it is an alternative method for downstream rate control and second it repairs the damage caused by ACK compression.

REFERENCES

[1] Cablelabs project overview presention available at http://www.cablemodem.com/downloads/slideshow.ppt

[2] V. Licea, "A Comparison of the DVB/DAVIC, DOCSIS and IEEE 802.14 Cable Modem Specifications", IEEE 2000 ICC, May 2000.

[3] Cable Television Labs Inc. , CableLabs, http://www.cablemodem.com-/specifications/specifications20.html.

[4] Cable Television Labs Inc. , CableLabs, "Data-Over Cable Service Interface Specifications- Radio Frequency Interface Specification", SP-RFIv2.0, available at http://www.cablemodem.com-/specifications/specifications20.html.

[5] The Network Simulator. Available at : http://www-mash.cs.Berkeley.EDU/ns/.

[6] J. Mogul, "Observing TCP Dynamics in Real Networks", Technical Report, Digital Western Lab, April 1992.

[7] V. Paxson, "Measurements and Analysis of end-to-end Internet Dynamics", Ph.D. dissertation, Univ. California, Berkeley, CA, 1997.

[8] H. Balakrishnan, V. Padmanabhan, S. Seshan, M. Stemm, R. Katz, "TCP Behavior of a Busy Internet Server: Analysis and Improvements", IEEE Infocom98, March 1998.

[9] "Understanding Data Throughput in a DOCSIS World", Cisco Technical Report available at: http://www.cisco.com/warp/public-/109/data_thruput_docsis_world_19220.pdf

[10] O. Elloumi, et. Al., "A Simulation-based Study of TCP Dynamics over HFC Networks", Computer Networks, Vol. 32, No. 3, pp 301-317, 2000.

[11] R. Cohen, S. Ramanathan, "TCP for High Performance in Hybrid Fiber Coaxial Broad-band Access Networks", IEEE/ACM Transactions on Networking, Vol. 6, No. 1, pp. 15-29, February 1998.

[12] H. Balakrishnan, V. Padmanabhan, R. Katz., "The Effects of Asymmetry on TCP Performance", ACM/IEEE International Conference on Mobile Computing and Networking, Sept. 1997.

[13] T. Lakshman, U. Madhow, B. Suter, "Window-based error recovery and flow control with a slow acknowledgement channel: a study of TCP/IP performance", INFOCOM97, April 1997.

[14] L. Kalampoukas, A Varma, K. Ramakrishnan, "Improving TCP Throughput over Two-Way Asymmetric Links: Analysis and Solutions", SIGMETRICS 98, June 1998..

[15] V Jacobson, "Compressing TCP/IP Headers for Low-Speed Serial Links", Feb 1990, RFC 1144.

.