DOCSIS Performance Evaluation:
Piggybacking versus Concatenation

Stefen Howard
Clemson University
100 McAdams Hall
Clemson, S.C. 29634-0974
+1 828 206 2453
showard@cs.clemson.edu

Jim Martin
Clemson University
100 McAdams Hall
Clemson, S.C. 29634-0974
+1 864 656 4529
jim.martin@cs.clemson.edu

ABSTRACT
Demand for high-speed internet access is growing rapidly and new technologies have been developed to supply broadband access to homes and small businesses. One of these technologies, DOCSIS, delivers high-speed internet connections through existing cable television service. While many factors affect performance, an evaluation of the process by which cable modems request upstream bandwidth has not been studied under realistic web traffic conditions. Our study uses the network simulator *ns* along with our own DOCSIS extension to model these conditions. We studied the DOCSIS protocol, looking specifically at the performance effects of two enhancements to the bandwidth request mechanisms: piggybacking and concatenation. Our results show that piggybacking alone is the most effective method for enhancing the overall performance of DOCSIS.

Categories and Subject Descriptors
C.2.5 [Computer-Communication Networks]: Local and Wide-Area Networks - *access schemes, high-speed, internet*.

General Terms
Measurement, performance.

Keywords
DOCSIS, broadband access, Data over Cable, cable networks, TCP performance.

1. INTRODUCTION
As the demand for high-speed internet access has grown, dial-up connections have become unable to provide the type of service and performance users demand. One technology that has been developed to meet this demand uses cable television connections and can now achieve bandwidths from 0.32 Mbps to 30 Mbps upstream and from 27 Mbps to 56 Mbps downstream. This technology, the Data-Over-Cable Service Interface Specification (DOCSIS) protocol was developed by the Multimedia Cable Network System (MCNS) group and connects end users' cable modems (CMs) to a headend or cable modem termination system (CMTS). Although other protocols such as DVB/DAVIC and IEEE 802.14 [11] for delivering broadband access over cable exist, DOCSIS is emerging as the industry standard.

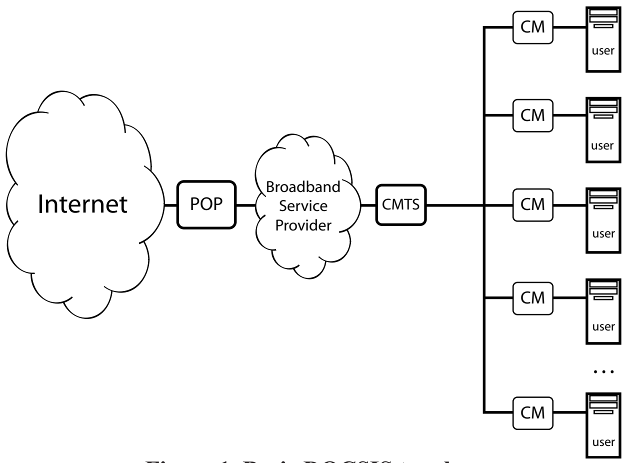

Figure 1. Basic DOCSIS topology

A DOCSIS network consists of a CMTS connected to hundreds of CMs sharing an RF (radio frequency) channel via an Ethernet-style bus topology [12]. Figure 1 illustrates a simple DOCSIS network. The CMTS manages requests for bandwidth which is partitioned into discrete time slices called minislots. Each minislot is typically long enough to transmit 8 to 32 bytes of data depending on the configuration of the system. The CMTS is responsible for scheduling opportunities for individual CMs to access these minislots. The CMs are responsible for accepting packets in the downstream direction and passing them on to the TCP stack for delivery to applications. They are also responsible for transmitting packets from applications on the local computer to the CMTS for transmission across the internet. When a packet arrives at a CM from the TCP stack, the CM must request enough bandwidth from the CMTS to transmit this packet. Requests for bandwidth are placed using one of the available request mechanisms. The CMTS periodically transmits a MAP message which specifies the allocation of minislots over a specific time interval. The size of a MAP message is a system setting, however, 2 ms has been shown to result in good performance [12][4]. The MAP message specifies which slots in a particular MAP are reserved for management, maintenance, contention requests, and data transmission. It also specifies which slots have been allocated for this time interval and to which CMs they were awarded. Figure 2 illustrates the structure of a MAP message. Without any other mechanisms in place for requesting minislots, all CMs must contend for request slots in a

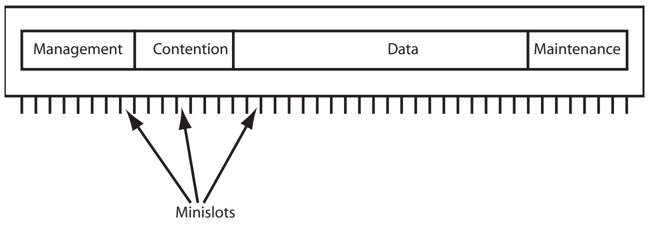

Figure 2. MAP message

given MAP by transmitting a request frame at the designated time. Since there are typically only a few slots reserved for contention requests, there is the possibility that two or more CMs will choose to request bandwidth during the same contention slot. This results in a collision and the slot is wasted. Intuitively it makes sense that avoiding collisions will lead to fewer wasted slots and better performance. For a DOSCIS best-effort service, two approaches to lowering the overhead associated with requests for bandwidth are *piggybacking* and *concatenation*. Piggybacking involves transmitting a request for more bandwidth along with an outgoing frame and concatenation is the technique of combining two or more small packets into one DOCSIS frame and making a single request for enough bandwidth to transmit this aggregate of packets. Our study focused on evaluating the effectiveness of piggybacking and concatenation when applied to a realistic web traffic scenario found in a typical DOCSIS environment. This paper is structured as follows: section two presents an overview of related work in the field and how this study differs from previous work, section three provides details of the study, section four presents and analyzes the results, and section five presents our conclusions.

2. RELATED WORK

While other work has been done to evaluate the performance of DOCSIS and the similar IEEE 802.14 MAC protocol, the focus has been primarily on such areas as contention resolution, minislot allocation, MAP size, and bandwidth allocation. In [4] the authors study the effects of MAP size and the number of contention slots on throughput and access delay in DOCSIS, concluding that a MAP size of 2 milliseconds and 6 contention slots yields the best performance. In [10] the authors compare and contrast IEEE 802.14 and DOCSIS scheduling algorithms with an emphasis on request minislot allocation schemes and collision resolution strategies. The authors include a study of the effects of piggybacking and conclude that when piggybacking is enabled, more request minislots are needed during medium loads than during low or high loads. The work done in [7] compares IEEE 802.14 and DOCSIS performance with IP versus ATM traffic. This includes an evaluation of DOCSIS performance increase with concatenation, showing that concatenation results in an improvement. This was not a significant part of the study which was focused on comparing IEEE 802.14 and DOCSIS performance, not DOCSIS with concatenation versus DOCSIS without concatenation. Their conclusion was that DOCSIS is more suited for IP traffic while IEEE 802.14 is better for ATM traffic. Other works such as [6] and [14] both look specifically at the performance impact of piggybacking and concatenation in DOCSIS. In [6] the author evaluates the two techniques over a network with only one CM and a constant bit rate (CBR) traffic generator. Two types of CBR generators are used, one described as distributed packet length (DPL) and one described as constant packet length (CPL). Only the results for the experiments involving a CPL-CBR traffic generator are shown. In [14] the authors are studying the effects of concatenation on low bit rate (LBR) streams intended to simulate audio, voice, and low quality video streams. It does consider a more realistic number of CMs than [6], and shows some interesting results. However, this study considers only a subset of the types of traffic carried over DOCSIS networks. Another work [8] focuses primarily on contention resolution algorithms for IEEE 802.14, but it offered a section on other factors that impact performance, and it included a small study of the effects of piggybacking and concatenation. Although these previous works have explored the benefits of these two techniques, none have included a study of a realistic web-traffic scenario. Since DOCSIS primarily serves home users, and home users typically generate web traffic such as http requests and ACKS, we felt it was useful to study how piggybacking and concatenation perform in this situation.

3. METHODOLOGY

Our study used the Network Simulator (ns) and the ns DOCSIS model developed previously [15]. Our DOCSIS simulator strives to be as close as possible to an actual implementation. It allows for a wide range of configuration parameters to be set by the user such as queue sizes, number of contention slots, number of service management slots, MAP time, and MAP frequency. There are two features of our implementation that are of particular interest to this study. First, as the protocol specifies, an extended MAC header may not be used with a concatenated frame [3] and our implementation complies with this requirement. The extended header is where a piggybacked request would be carried, so this rules out the possibility that a piggybacked request can be carried by a concatenated frame. However, when piggybacking and concatenation are both enabled, and a non-concatenated frame is transmitted, a piggybacked request is permissible. Second, although the specifications do not explicity state how to determine the number of contention request slots per MAP, our implementation assigns any unallocated minislots in a MAP to be extra contention opportunities. The following subsections describe our simulated environment in greater detail.

3.1 Simulated Environment

Our DOCSIS network consists of *n* CMs attached to one CMTS, where *n* is a parameter of the TCL script used to build the network.

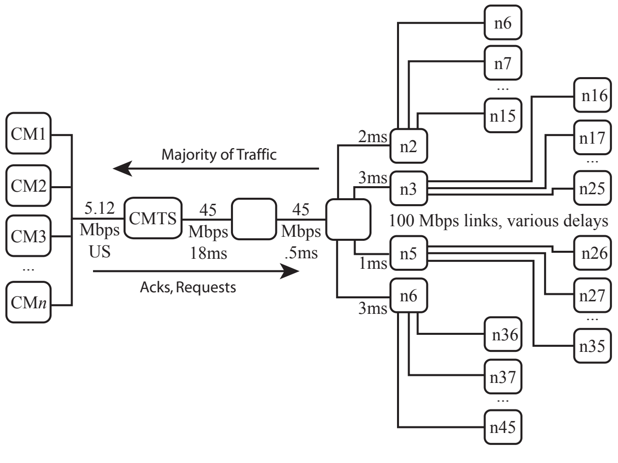

Figure 3. Simulated DOCSIS environment

Table1. DOCSIS configuration

Parameter	Value
Upstream channel capacity	5.12 Mbps adjusted for FEC to 4.71Mbps
Downstream channel Capacity	30.34 Mbps adjusted for FEC to 28.91 Mbps
Ticks per minislot	5 (18 bytes/per slot)
Maptime	2 ms (64 minislots per map)
Contention slots	12
Concatenation threshold	1

The CMTS is attached to a node serving as a point-of-presence (POP) for the network to which the CMTS is attached. This POP is in turn attached to another node which serves as an entry point to the network where our servers are located. Four groups of ten servers each connect to one of four nodes which are attacheded to this entry node. The links connecting this entry node to these 40 nodes are all 100 Mbps links with varying propagation delays. Figure 3 illustrates our simulated environment and table 1 describes the relevant DOCSIS configuration parameters we used. Based on previous work, we chose to set the concatenation threshold to one. That is, when concatenation is enabled, the CM may combine at most one other packet from the queue with the current packet into one DOCSIS frame. It may seem that specifying a low threshold like this is counter-intuitive, but when the threshold is high, occurrences of ACK compression increase which leads to poor TCP performance overall [1][13].

Table 2. Traffic generators

Parameter	Distribution	Average	shape
Session interval	Exponential	1	NA
Interpage arrival	Pareto	10	2.0
Interobject arrival	Pareto	0.5	1.5
Object size	Pareto	12	1.2

To create realistic web traffic, our simulated environment provides a mixture of traffic representative of home users. This includes flows to simulate downstream audio and video as well as a variety of web connections. Each CM on the DOCSIS LAN generates web sessions with random intersession times. Each session contains 250 pages, and each page is requested at random intervals. Additionally, these pages are further broken down into different size pages, with varying arrival times and object sizes. Random variable generators are used to control the generation of these sessions. Table 2 describes the parameter, the type of distribution used, and the settings for each of these random number generators. Since these are all downstream sessions, the only upstream traffic generated is requests for pages, requests for bandwidth, and acknowledgements for data received. We have designed the traffic models to reflect actual usage patterns as measured by [2][12].

The model described above does not include any upstream UDP traffic to simulate streaming audio or video since this is not presently a significant portion of real world web traffic. However, with the growing market for VoIP (Voice over IP) services provided by companies such as Vonage, more significant amounts of telephony traffic may be carried by DOCSIS in the near future. For one set of experiments, we included traffic to approximate upstream VoIP traffic. The experiments are described in the next subsection.

3.2 Experiments

For the basic web model, The first step in gathering our data was to perform runs with 100, 200, … , 600 CMs using neither piggybacking nor concatenation. We called these our baseline runs and they were simulated for 60 seconds. Next, we performed a set of runs with identical parameters except for the piggybacking and concatenation parameters. We performed a set of runs with only piggybacking, one with only concatenation, and one where both piggybacking and concatenation were enabled. We also configured the simulator to output information about network performance to various files which we then extracted and analyzed. We refer to the above set of experiments as experiment 1.

To understand how these techniques perform when VoIP traffic is introduced, we configured our scripts to set a fixed number of 200 CMs on the LAN. Our first run included none of them running VoIP and we progressively assigned 10%, 20%, …, 50% of these CMs to each be running a single 56k CBR (Constant Bit Rate) UDP connection in addition to the other web connections. This is a rough approximation of a VoIP connection, and future studies will include more accurate models for VoIP traffic. This set of experiments is referred to as experiment 2.

In both sets of experiments we ran a ping client on a CM on the DOCSIS network which was set to ping a server on the external network approximately every half second. This CM generated web and VoIP traffic (for experiment 2) as well as the ping packets. We collected RTT (round trip time) and loss rate data for this ping application.

3.3 Metrics

To evaluate performance, the DOCSIS metrics we looked at were upstream and downstream link utilization as seen by the CMTS, CM contention request collision rate, and mean access delay. Collision rate measured the percentage of contention requests that resulted in collisions, and thus a wasted slot. Mean access delay is defined by [8] as “The average transmission time for a packet from the moment it is generated until it is received at the headend.” This time includes all time spent by the CM in contention for request slots, resolving collisions, transmitting the bits, and propagation delay. It also includes time spent in queues by the request packet as well as data packets [8]. The CMTS captured bandwidth data which it reported as link utilization. Each CM calculated the access delay for each packet, kept track of contention requests generated, and tallied up contention collisions. The individual CM mean access delays for each run were averaged to arrive at an overall average for that run. We also looked at average ping RTT and ping loss rate. We ensured that we ran the simulations long enough to allow the utilization and collision rate statistics to converge to a 99% confidence interval. The next section presents our findings.

4. RESULTS

Our expectations were that both piggybacking and concatenation would show a measurable improvement over the baseline. While there was no evidence that either would ever perform much worse

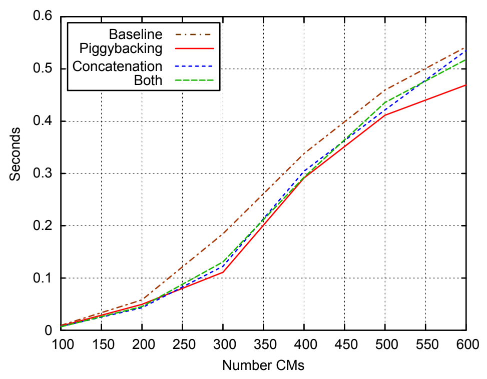

Figure 4. Mean access delay (experiment 1)

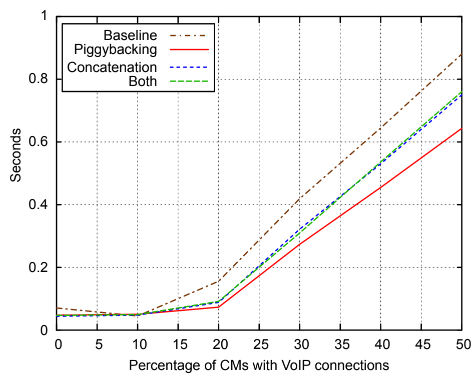

Figure 8. Mean access delay (experiment 2)

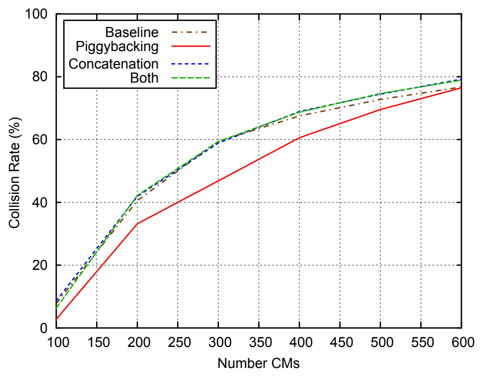

Figure 5. Collision Rate (experiment 1)

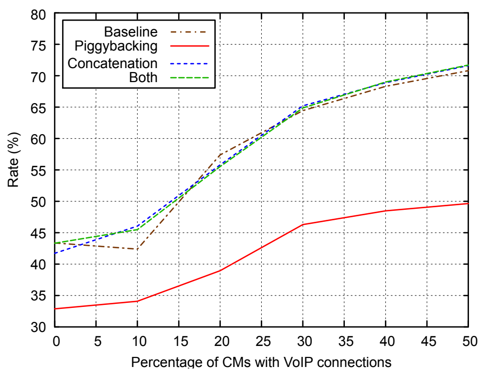

Figure 9. Collision Rate (experiment 2)

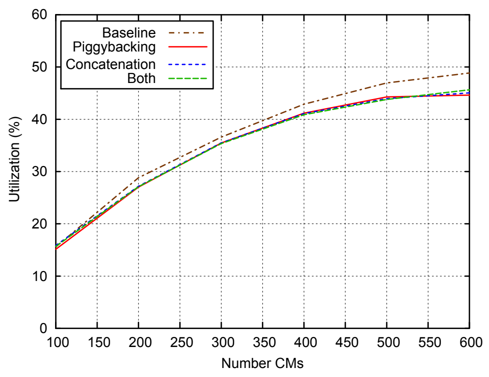

Figure 6. Upstream link utilization (experiment 1)

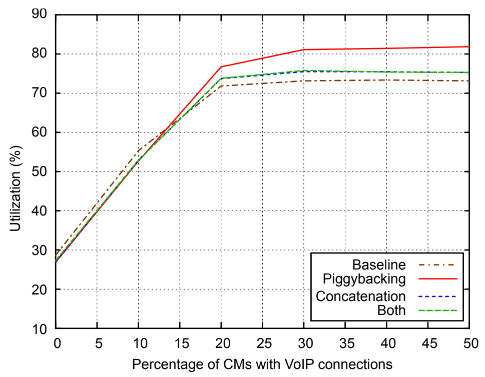

Figure 10. Upstream link utilization (experiment 2)

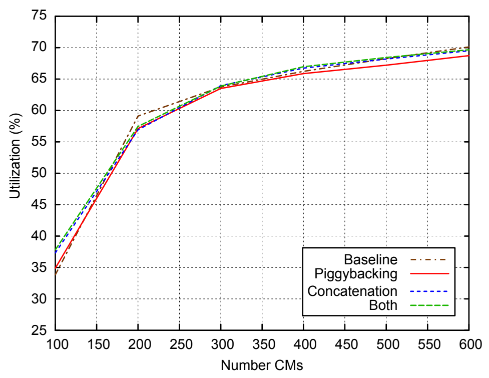

Figure 7. Downstream link utilization (experiment 1)

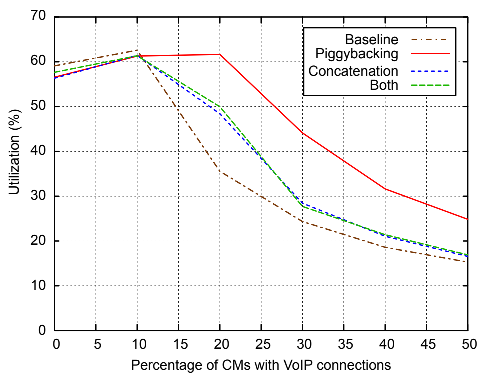

Figure 11. Downstream link utilization (experiment 2)

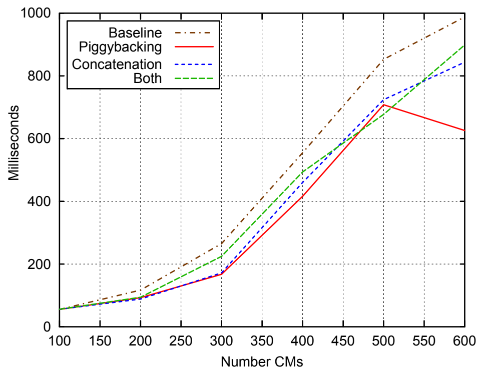

Figure 12. Ping RTTs (experiment 1)

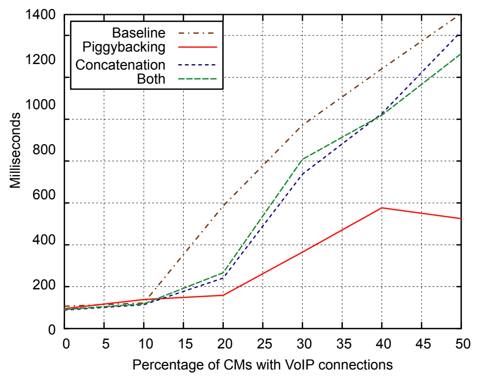

Figure 14. Ping RTTs (experiment 2)

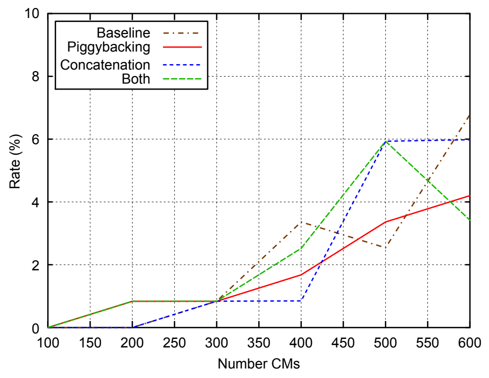

Figure 13. Ping loss rate (experiment 1)

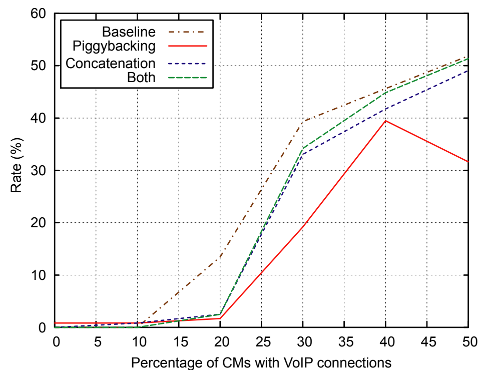

Figure 15. Ping loss rate (experiment 2)

than baseline, in most cases piggybacking was better. Piggybacking was able to improve performance over the baseline in most of the tested areas and a significant improvement with concatenation was only seen in mean access delay. In most of the experiments, concatenation performance was similar to the baseline, or was closer to the baseline than to piggybacking. However, in mean access delay, concatenation's performance was closer to that of piggybacking. Given the requirement that requests could not be piggybacked on concatenated frames, we were unsure how combining the techniques would effect the performance. It turned out in both sets of experiments that combining them resulted in almost identical performance as concatenation alone. Clearly concatenation was dominant in our trials. The web traffic scenario generates bursts of activity followed by periods of inactivity. This simulates a user loading a web page and then pausing to read it. During on periods, acks tend to arrive in clumps at the CM resulting in sustained queue levels in the upstream direction at high load levels. Adding VoIP traffic further increases the average queue level. Both situations are favorable for concatenation to dominate. Overall, the results were as expected. Reducing the dependence on contention requests reduces collisions, which in turn, speeds up access to the network and can improve utilization.

4.1 Experiment 1 Results

Piggybacking gave the best results for reducing both the mean access delay and the collision rate in experiment 1. That is, on average, packets took less time to reach the CMTS with piggybacking than without and the collision rate of contention requests was lower. At the 100 and 200 CM level the differences are small, but above 200, the gap widens. Figure 4 plots the mean access delay of all our simulated runs for experiment 1. It is shown that concatenation also provides an improvement over the baseline for access delay, though not as significant as piggybacking. Piggybacking also emerged as the better of the two techniques for reducing contention request collision rates (between 5 and 20 percentage points). The curve (figure 5) makes sense since, at light loads, all unused minislots will be designated as extra contention slots and the impact of enhancements will be small, but as loads increase and extra contention slots diminish, performance enhancement techniques begin to have an effect. At the upper end of the graph, when loads are heavy, all curves converge. Concatenation made almost no difference, and in many runs was the same as baseline. In the area of utilization (figures 6 and 7) there was no significant difference between any of the techniques, but the curves for ping RTT and loss rate (figures 12 and 13) show an improvement for piggybacking over the rest.

4.2 Experiment 2 Results

For experiment 2 (figures 8, 9, 10, and 11), with VoIP traffic included in the mix, piggybacking came out on top in all areas. When half of the 200 CMs were each running a VoIP connection, mean access delay was cut by over two tenths of a second with respect to the baseline, and the collision rate was cut from just over 70% at baseline to just under 50% with piggybacking. When running web traffic without VoIP (experiment 1), Link utilization showed virtually no difference between any of the techniques, but when VoIP traffic is added, piggybacking was considerably better than both baseline and concatenation. Concatenation and the combination of the two techniques again performed virtually identically to one another with only slight increases over baseline except in mean access delay. Downstream link utilization suffered in all areas as the percentage of VoIP connections increased. This can be explained by the introduction of VoIP traffic. As more upstream traffic is present, acks take longer, which slows down the TCP sender. In the case of ping/CM loss rates (figures 14 and 15) and ping RTTs, experiment 2 produced results similar to those of experiment 1. The performance increase was especially evident with ping RTTs where piggybacking was able to produce times of less than half of those for baseline and concatenation.

5. CONCLUSIONS AND FUTURE WORK

From these results, which are based on the the algorithms used in our implementation of the protocol, we show that piggybacking alone is the best choice for enhancing the performance of the DOCSIS protocol. It was best at reducing the wait time for a packet to reach the CMTS, reducing contention request collision rates, reducing ping RTTs, reducing ping loss, and, in some cases, improving link utilization. We also demonstrated that concatenation can, in a more limited capacity, improve performance, and we explained why combining the two techniques is no more effective than concatenation alone. Although we did not study concatenation with any threshold settings other than 1, It is likely that a higher threshold could improve the performance of concatenation, especially for experiment 2, when we have queues full of small VoIP packets. However, we don't know how this would effect overall TCP performance.

We are curious if we can enhance the algorithms to build on the potential that piggybacking offers and thus design a better, more efficient system. We will further develop our ns DOCSIS model and our web traffic model and continue to explore the effects of other system parameters on these bandwidth request mechanisms. We want to understand why concatenation could not achieve similar results to those of piggybacking and intend to combine new algorithms with a system configuration that delivers excellent all around performance.

6. REFERENCES

[1] H. Balakrishnan, et. Al., *TCP Behavior of a Busy Internet Server: Analysis and Improvements*, IEEE INFOCOM98, 1998.

[2] P. Barford, M. Crovella, *Generating Representative Web Workloads for Network and Server Performance Evaluation*, Proceedings of Performance, ACM SIGMETRICS98, 1998.

[3] Cable Television Laboratories, Inc., *Data-Over-Cable Service Interface Specifications: DOCSIS 2.0, Radio Frequency Interface Specifications*, 1999-2003.

[4] S. Cho et. Al., *Performance Evaluation of the DOCSIS 1.1 MAC Protocol According to the Structure of a MAP Message*, ICC 2001, Helsinki, Finland. June, 2001.

[5] O. Elloumi et. Al., *A Study of TCP Dynamics over HFC Networks*, 1997.

[6] C. Godsay, *Upstream Performance Whitepaper*, University Of New Hampshire, InterOperability Laboratory DOCSIS Consortium.

[7] N. Golmie et. Al., *A Comparison of MAC Protocols for Hybrid Fiber/Coax Networks: IEEE 802.14 vs. MCNS*, Proceedings of the 16th International Conference on Communications, Vancouver, Canada, June, 1999.

[8] N. Golmie et. Al., *A Review of Contention Resolution Algorithms for IEEE 802.14 Networks*, IEEE Communications Surveys, first quarter 1999.

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

[10] Y. Lin et. Al., *Allocation and Scheduling Algorithms for IEEE 802.14 and MCNS in Hybrid Fiber Coaxial Networks*, IEEE Transactions On Broadcasting, Vol. 44, No. 4, December 1998.

11] Y. Lin et. Al., *An Investigation into HFC MAC Protocols: Mechanisms, Implementation, and Research Issues*, IEEE Communications Surveys, third quarter 2000.

[12] J. Martin and N. Shrivastav, *Modeling the DOCSIS 1.1/2.0 MAC Protocol*, Proceedings of the 2003 International Conference on Computer Communications and Networks", Dallas TX, October 2003.

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

[14] Tzerefos et. Al., *Delivery of Low Bit Rate Isochronous Streams Over DOCSIS 1.0 Cable Television Protocol*, IEEE Transactions on Broadcasting, Vol. 45, No. 2, June 1999.

[15] N. Shrivastav, A Network Simulator Model of the DOCSIS Protocol and a Solution to the Bandwidth-hog Problem in the Cable Networks, Masters Thesis, North Carolina State University, 2003.