Empirical Evaluation of Upstream Throughput in a DOCSIS Access Network

Swapnil Bhatia, Radim Bartoš, and Chaitanya Godsay
Department of Computer Science
University of New Hampshire
Durham, NH 03824, USA
Email: {sbhatia, rbartos, cgodsay}@cs.unh.edu

Abstract— **We present empirical measurements of the upstream throughput of a DOCSIS[1] 1.1 link. In contrast to all previous simulation-based studies, our measurements have been obtained from actual cable-modems (CMs) and head-ends, both from two different vendors each. We have constructed an exhaustive database of measurements of a large subset of the space of parameters affecting upstream throughput. Using a well-known non-parametric hypothesis test, we query this database for obtaining statistically robust answers to key questions about the effect of parameter changes on the throughput. We also present some examples of such questions. Our results indicate that for a single CM scenario, packet concatenation is most effective whereas piggybacking is effective and better than concatenation only in some cases. Using both enhancers decreases throughput for a single CM scenario. Our results are robust across head-end implementations and are of immediate interest to network and protocol architects as well as device developers.**

I. MOTIVATION

The DOCSIS 1.0 and 1.1 standard [1] was a result of research and development into the performance implications of various QoS mechanisms [2], [3], [4], [5], [6], [7]. Since these were preliminary, prototyping studies, they were conducted using analytical and/or simulation models. There has been no extensive study of the DOCSIS protocol using actual implementations, to our knowledge, and for good reason. Researchers wishing to study the performance of a QoS algorithm are well-advised in abstracting away from the details of any particular implementation in order to obtain results applicable to the protocol itself instead of any of its particular implementations. Nonetheless, studies based on actual implementations provide a wealth of complementary information of interest to designers and operators alike. For one, models, simulation or analytical, can rarely capture the full complexity of a system. Models are attractive due to their elegance and simplicity, the price for which is paid by abstraction. If deep complex details and inter-module interactions in a system were included, the model would quickly become intractable. Measurements from an implementation on the other hand, offer a tractable way of capturing and evaluating a system in its entirety. They offer a means of obtaining absolute numerical performance estimates of immediate and direct relevance to developers. Discrepancies between empirical measurements and estimates from modeling can be accurate indicators of subtle implementation bugs. Furthermore, such measurements play a key role in the design and choice of algorithms in an implementation. For network service providers and operators, empirical data obtained from actual implementations is an indispensable input for the design and upgrade process. Providers and operators require some form of field performance report from device vendors, before a device can be deployed in a live traffic environment. Just as analytical models provide a rich framework for designers to ask “what if” questions, similarly, an empirical, numeric model constructed from an actual deployed system can be an invaluable decision tool to operators. Inarguably, no model can match the precision and relevance of the information provided by a complete and accurate characterization of an actual system. Needless to say, an empirical model must be tested for robustness using standard statistical techniques. Without such testing for confidence, no useful and reliable conclusions can be drawn from the data.

In this paper, we report results from our extensive performance study of the upstream portion of a DOCSIS 1.1 link. Using our measurements as a database, we can answer key performance questions that are of immense value to the network designer. We provide examples of such questions answerable with an empirical model. We also hope that the results obtained provide some insight into the strengths and shortcomings of the various performance enhancing features provided by the protocol. This paper is organized as follows: In the next section, we present a brief overview of the DOCSIS 1.1 protocol. We then present the details of our testbed and experiments. Finally, we present and discuss some conclusions that can be drawn from the data and summarize our findings in the last section.

II. AN OVERVIEW OF THE DOCSIS 1.1 MAC LAYER

A DOCSIS network uses the existing cable television infrastructure to deliver data services to subscribers. The network, owing to the structure of the pre-existing cable television plant, forms a tree. The root of the tree is located at the service provider’s office and is connected to a Cable Modem Termination System (CMTS). Subscribers connect to the network through a cable modem (CM) located in their home and connected as a leaf of the tree. The link from the CMTS to CM, termed as downstream, is point-to-multipoint broadcast, whereas the upstream from the CM to CMTS is

---

[1] DOCSIS is a registered trademark of CableLabs.

a multipoint-to-point time-division multiplexed link arbitrated by the CMTS. The downstream link is designed to match cable television channels (90-850 MHz) in its bandwidth and is therefore 6 MHz wide whereas the upstream, located in the lower spectrum (5-42 MHz), can be a maximum of only 4 MHz wide. DOCSIS offers several combinations of upstream data rate and modulation formats. The channel can use either Quadrature Amplitude Modulation (QAM) for low noise environments or Quadrature Phase Shift Keying (QPSK) to counter noise. The channel can transmit at any fixed rate from the set {0.64, 1.28, 2.56, 5.12, 10.24} Mbps. The downstream is also configurable, but is not the focus of this paper. The subscriber connects Customer Premise Equipment (CPE), such as a computer, telephone, etc., to the CM which forwards data from and to the CPE. Typically, there are 1500 to 2000 CMs connected to a single CMTS with any CM being no more than 50 miles from the CMTS.

Each CM on the network is required to perform a registration handshake with the CMTS to obtain network parameters only after which is it allowed to use the upstream channel. Individual CM parameters are described in a configuration file that is downloaded and shared with the CMTS by the CM as a part of the registration handshake. Among other parameters, QoS policies are also specified in this file. The CMTS describes the allocation of the upstream bandwidth for a future interval of time using a map message regularly broadcast downstream. In each mapped interval, the CMTS reserves portions of the upstream bandwidth for new CM registration and bandwidth requests from existing CMs. A CM wishing to transmit data on the upstream first requests bandwidth by transmitting a message during the bandwidth request interval, then waits to receive a bandwidth grant in a map message, and finally transmits data during its map-designated time slot.

The bandwidth request interval is contention-based and is therefore prone to collisions from overlapping request messages from many different CMs. Therefore, as an alternative to the contention-based request interval, a CM can also request bandwidth during a data transmission opportunity, i.e., it can use part of the transmission interval acquired for data transmission through contention-based requesting to make further requests for bandwidth. This is known as *piggybacking* and is one of the enhancements that can be provided to or prohibited for a CM through the configuration file. Piggybacked bandwidth requests use a small part of the data bandwidth to make further requests and thus avoid delays in acquiring bandwidth due to lost request messages. In addition to piggybacking, *packet concatenation* is another available enhancement. In order to minimize overhead and reduce latency, a CM can send a longer burst of concatenated packets in a single transmission opportunity instead of stepping through the request-grant-transmit cycle repeatedly for a sequence of small packets.

In this paper we focus on characterizing the performance of the two enhancers as well as the effect of the channel rate and packet length on the throughput. We describe our experiments in the next section.

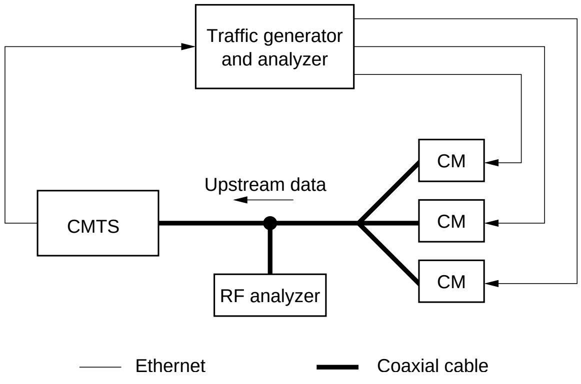


Fig. 1. Experimental setup.

III. EXPERIMENTAL DESIGN

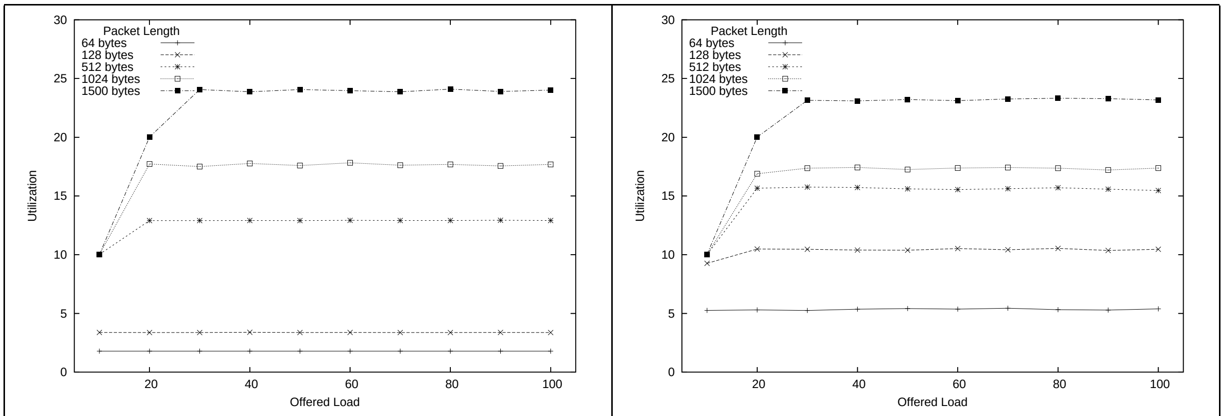
The experiments presented in this paper were conducted on the testbed shown in Figure 1. The test network consisted of a CMTS connected by a coaxial cable plant to one or more CMs. Upstream traffic was generated using a traffic generator, injected into the cable plant through the CM, and captured beyond the CMTS by a traffic analyzer. The distance between the CMTS and the CM was minimal (a few feet).

Our experiments were conducted in two phases dictated primarily by the availability of equipment for the necessary duration. Access was available to several units of two different CM implementations (CM_A and CM_B) and one unit each of two different CMTS implementations ($CMTS_C$ and $CMTS_D$) individually, at different times. In the first phase, we conducted pilot experiments involving CM_A and CM_B against a single $CMTS_C$. In this phase, the experiments spanned a broad range of parameters and were intended to provide the big picture. Upstream traffic was injected into the CM at a maximum rate such that there was negligible packet loss. In the first phase, we conducted pilot runs to explore the effect of various system parameters on the performance. Effect of packet length, channel width, offered load, modulation format, number of CMs and enhancers on latency and throughput was studied. A subset of these results are reported in [8]. The complete set of results can be found in [9]. Based on some of the preliminary findings, we devised new experiments to study interactions between parameters and the throughput at a finer level. This was the objective of the second phase of our work.

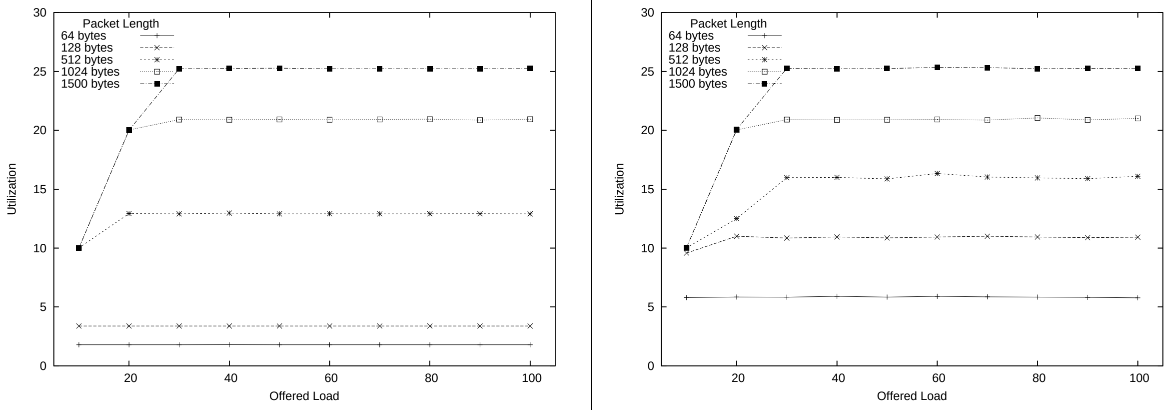
In the second phase, we characterized the effect of every change in the system parameters on throughput. The second phase was designed to be more exhaustive so as to characterize the difference in performance for various system parameters. Data was collected using CM_A against $CMTS_C$ and $CMTS_D$ on a QAM upstream channel. Data was injected into the CM at a constant input rate of 8 Mbps. In each experiment, the system was configured to provide the CM with a different combination of channel rate, data packet length and performance enhancer. Different packet lengths from the set {64, 128, 256, 512, 768, 1024, 1262, 1500} (bytes)

TABLE I

For a 10.24 Mbps channel, throughput on $CMTS_C$ for various loads with (a) No Performance Enhancers, (b) Concatenation, (c) Piggybacking, and (d) Both enabled.



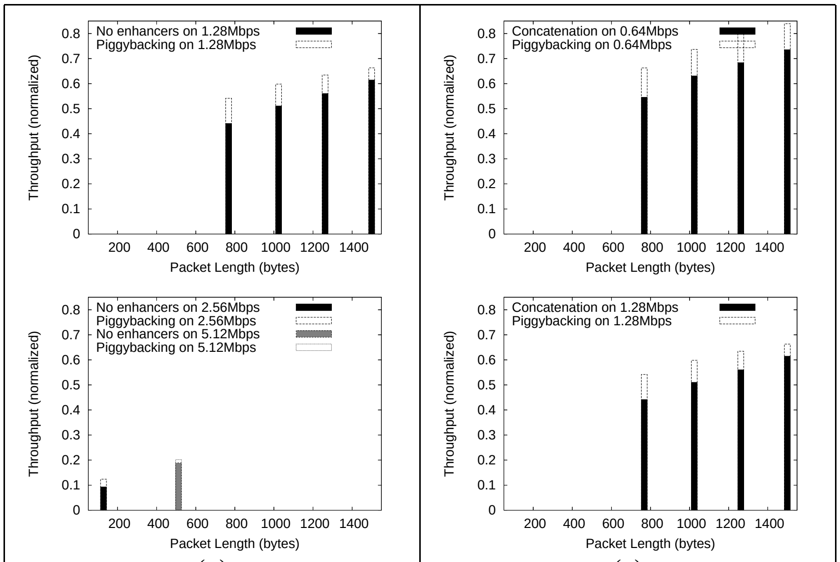
(a) None (b) Concatenation



(c) Piggybacking (d) Both

TABLE II

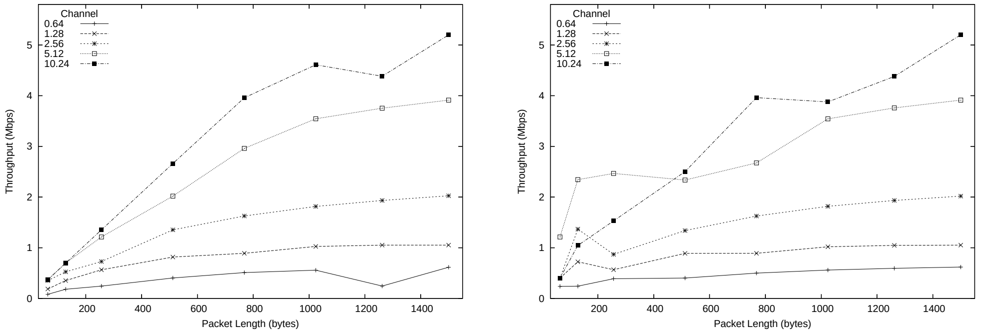
(a) When and by how much is Piggybacking useful? (b) When and by how much is Piggybacking better than Concatenation? (95% significance, points failing this criterion omitted.)



(a) (b)

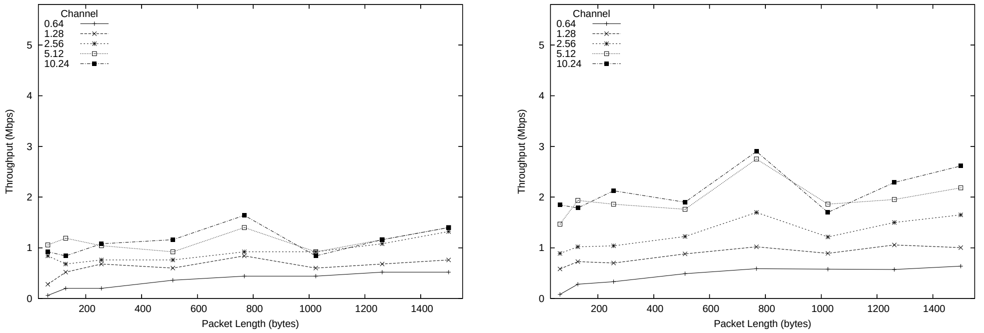
TABLE III

For two CM_A, throughput on $CMTS_C$ with (a) piggybacking (b) piggybacking and packet concatenation. Throughput on $CMTS_C$ for (c) 1 (d) 2 (e) 3 (f) 4 CM_B(s) with concatenation and piggybacking enabled. Throughput vs. (g) Packet length for various enhancers (h) Enhancer for various channels.



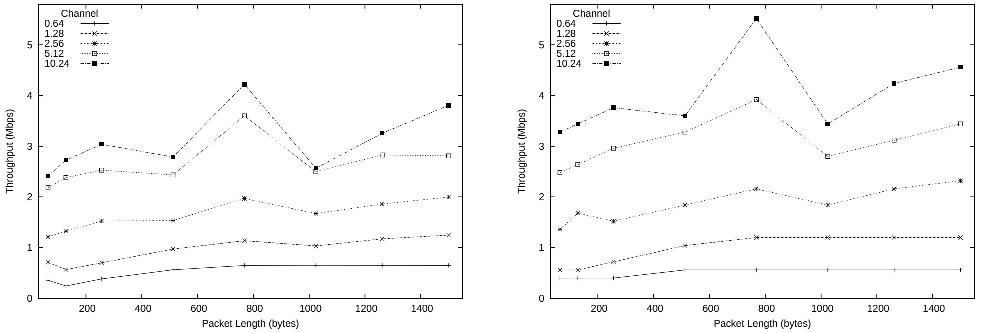
(a) Piggybacking

(b) Both



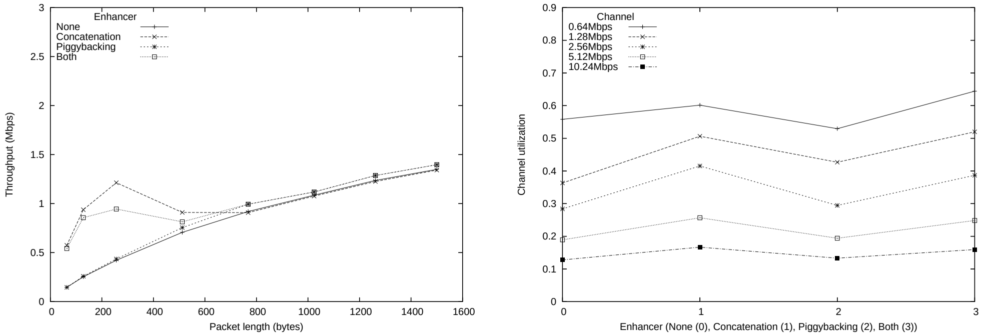
(c) 1 CM_B

(d) 2 CM_B



(e) 3 CM_B

(f) 4 CM_B



(g)

(h)

were used. Finally, all four different performance enhancer combinations (packet concatenation, piggybacking, neither and both) were applied. We also considered the CMTS ($CMTS_C$ vs. $CMTS_D$) as the fourth system parameter since we were interested in obtaining performance measures that were robust across CMTS implementations. The space of all the possible combinations of parameters was explored. Across two CMTS and with 5 channel rates, 8 packet lengths and 4 performance enhancers, a total of 2400 (1120 per CMTS and 160 between CMTS) different transitions from one set of parameters to another leaving the values of all but one parameter unchanged can be performed. For each case, 25 independent observations were recorded to ensure high confidence in the conclusions derived. Thus, for each experiment, two sets of 25 throughput values were obtained—with and without the change in value of one parameter.

In order to draw robust conclusions from our data, we used the Wilcoxon Signed Rank Sum (WSRS) test [10] for testing the effect of each of the 2400 transitions from one parameter set to another. The WSRS test is a commonly used non-parametric hypothesis test when the distribution of the data are unknown and is able to deal with paired data. Given a set of ordered pairs $\{(x_i, y_i) | 1 \leq i \leq N\}$, where x_i is the response prior to or without the modification whereas y_i is the response with the modification (or vice versa), the WSRS test computes the probability that the median of the distribution of $x_i - y_i$ is zero (i.e. the distribution of sorted differences is symmetric about zero). This is the *null hypothesis* of the WSRS test and represents the scenario where the modification or solution is ineffective. If this probability is lower than the preset *significance* level, then the test concludes that the null hypothesis of inefficacy of the solution can be safely rejected since such a rejection is amply supported by the available data. To apply the WSRS test to each experiment, we use the null hypothesis that the change in the value of one of the system parameters produced no effect on the throughput. We fix an acceptable level of significance to 95% ($\alpha = 0.05$). Since we compare many different modifications (the 2400 different parameter transitions) we perform 2400 pairwise tests. In this scenario, although each hypothesis taken separately is significant at level α, all the hypotheses taken together are not significant at level α. In order to achieve a significance level of α for all the tests taken together, we use the Bonferroni method [10] and require each of the 2400 tests to be significant at a level $\frac{\alpha}{2400} = 2.08333 \times 10^{-5}$ (i.e., 99.99% significance individually). The the parameter-set pairs with their significance levels were recorded into a database.

IV. DISCUSSION OF RESULTS

Table I shows some of the results from our preliminary study in the first phase. This experiment was performed to characterize the saturation behavior of the CMs. Although Table I presents the results for a single channel rate (10.24 Mbps), results for all other channel rates are available in [9] and show a similar trend. The decision to use 8 Mbps as the input rate in the second phase was based on these results since this rate was sufficiently high to operate the CM in the saturation region and was the highest rate that could be reliably generated by the traffic generator.

The measurements in Table III (A)–(F) are also from our preliminary experiments. Since we had access to multiple CMs in phase 1, we conducted pilot runs to characterize the throughput in the presence of multiple CMs. Table III (A)–(B) show the aggregate throughput for two CM_A against $CMTS_C$ with piggybacking enabled with or without concatenation respectively. Comparing with Table IV (C),(D) and (G),(H) we see that the measured throughput for two CMs is roughly double that for a single CM. Table III (C)–(F) shows the throughput for 1–4 CM_B. We did not have access to CM_B in phase 2 and hence were unable to conduct experiments with them. Nonetheless, comparing Table III (A) with (D)–(F) shows that again, for multiple CMs, the throughput is roughly a multiple of the throughput for a single CM. These measurements were obtained in phase 1 using the highest input traffic rate that showed negligible packet loss. It is also important to note that the measurements in Table III (A)–(F) are based on fewer data points than in phase 2 and have not been subjected to any statistical analyses. Due to the particularly voluminous nature of these experiments, link throughput characterization with multiple CMs remains a subject for future study.

We now present the results from our experiments in phase 2. Table IV shows the throughput of a single CM_A on $CMTS_C$ ((A)–(D)) and on $CMTS_D$ ((E)–(H)). The results are all presented with 99% confidence intervals[2]. Comparing the two columns in Table IV we see that the throughput of both the CMTS shows a similar trend in each case. Table IV (B) and (F) show a marked increase in throughput for smaller packet lengths when concatenation is enabled (compare with (A) and (E)). The drop in performance for $CMTS_C$ for packet size of 512 bytes cannot be explained. Given that the data point passes the confidence test, we speculate that it may be related to an implementation bug and are working with the vendor to identify its cause. Although not as large, (C) and (G) in Table IV show the improvement in throughput for larger packet sizes when piggybacking is enabled. Finally, (D) and (H) show the throughput with both enhancers enabled. Variations in throughput in each column may be of interest to network designers and operators whereas variations between columns are valuable to the CMTS vendors. It is important to keep in mind that the data points presented are bound by tight confidence intervals. Thus, very small differences in performance may still be statistically significant.

We now turn our attention to the relative improvement in throughput offered by various combinations of enhancers. Table III (G) shows throughput as a function of packet length for various enhancers. Clearly, for smaller packet sizes, concatenation proves to be extremely effective. Moreover, it should be noted that concatenation together with piggyback-

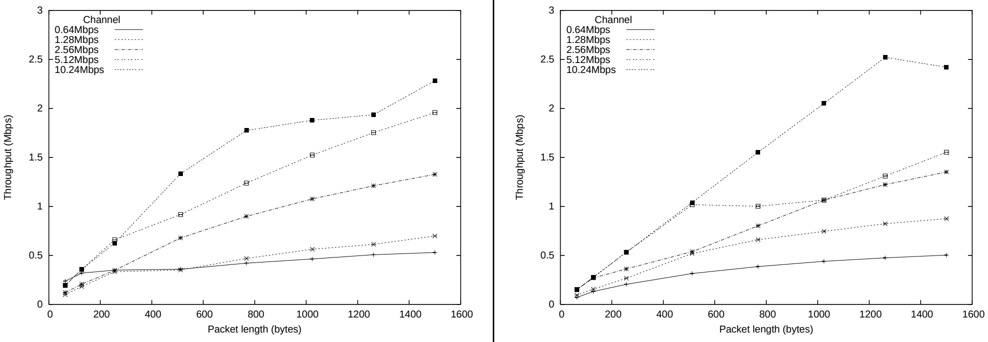
---

[2] The 99% confidence intervals are plotted in the graphs but may be too small to be visible at most of the points.

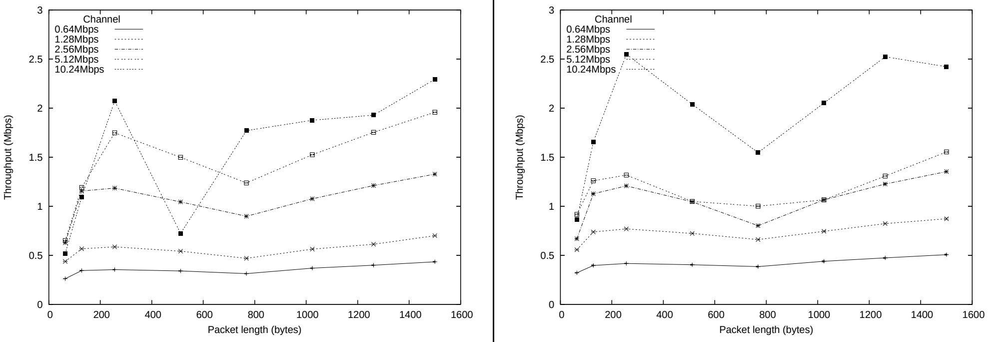
TABLE IV

THROUGHPUT OF CM_A ON $CMTS_C$ ((A)–(D)) AND $CMTS_D$ ((E)–(H)) (WITH NO ENHANCERS, CONCATENATION, PIGGYBACKING AND BOTH ENABLED RESPECTIVELY). (99% CONFIDENCE)

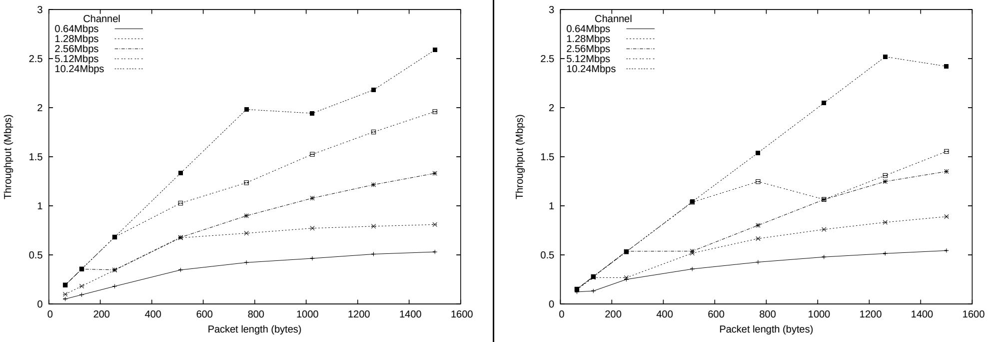
$CMTS_C$ $CMTS_D$



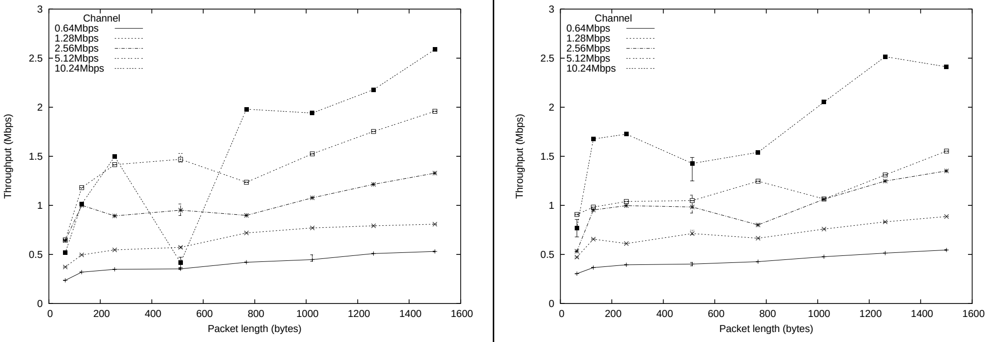
(A) NONE (E) NONE



(B) CONCATENATION (F) CONCATENATION

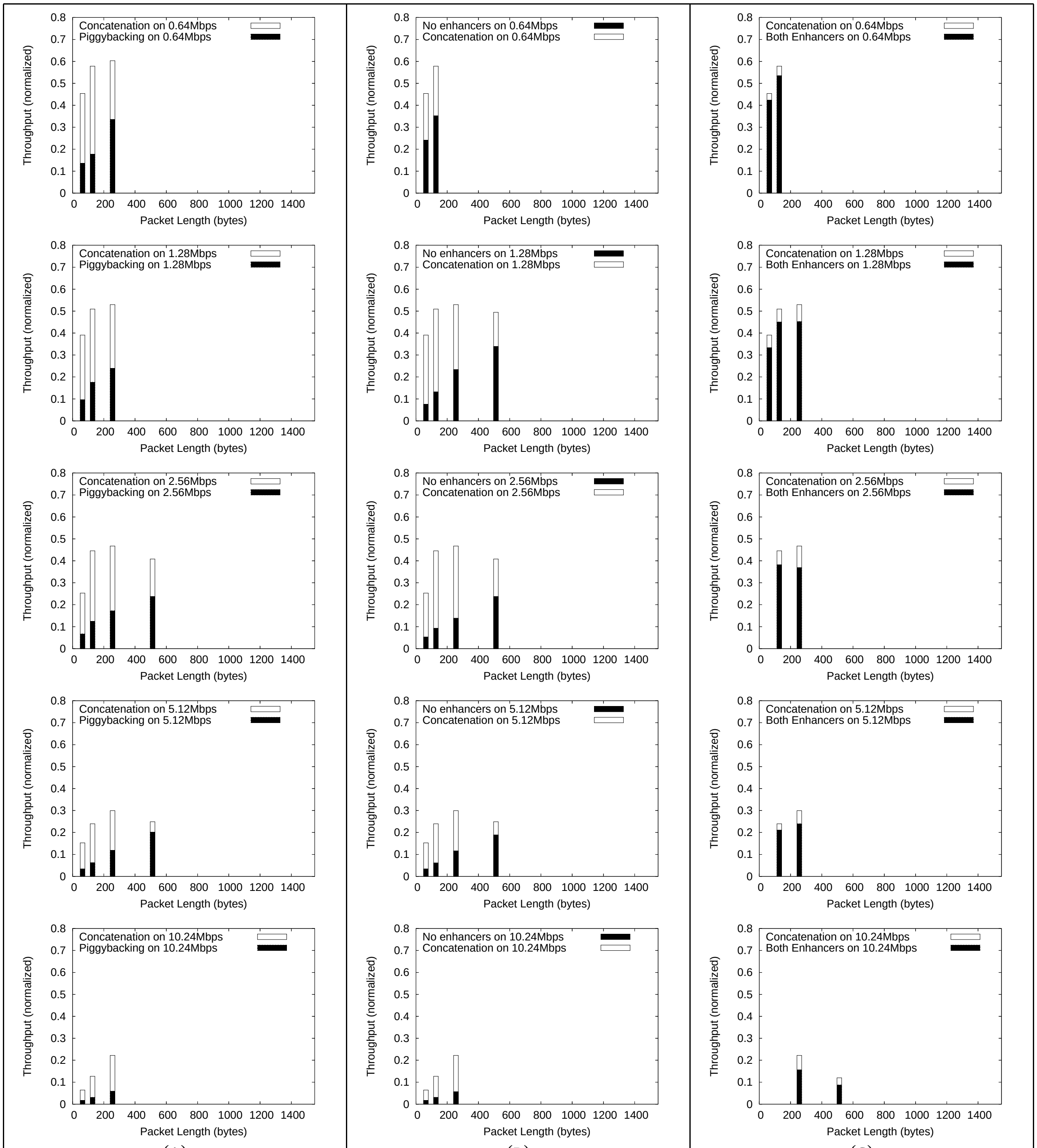


(C) PIGGYBACKING (G) PIGGYBACKING



(D) BOTH (H) BOTH

TABLE V
(A) WHEN AND BY HOW MUCH IS CONCATENATION BETTER THAN PIGGYBACKING? (B) WHEN AND BY HOW MUCH IS CONCATENATION USEFUL? (C) WHEN AND BY HOW MUCH IS PIGGYBACKING AND CONCATENATION WORSE THAN JUST CONCATENATION?
(95% SIGNIFICANCE, POINTS FAILING THIS CRITERION OMITTED.)



(A)

(B)

(C)

ing is less effective than concatenation alone. This can be attributed to the fact that for a single CM, piggybacking provides no benefit for smaller packet sizes, but does consume a small portion of the bandwidth provided. However, piggybacking provides a slight improvement in throughput for larger packet sizes where concatenation does not seem to have a pronounced effect. Table III (H) shows the change in channel utilization with enhancers for different channel rates. Again, concatenation is most effective, more than piggybacking alone or in combination. Based on these initial observations, we can query the measurement database for some more specific questions. For example, it would be of interest to network designers to know, for an actual system to be deployed, when and by how much piggybacking improves throughput. Table II (A) answers this question with 95% significance for a single CM scenario. To answer the question, we pick all those transitions of parameter values from our measurement database in which the initial parameters have no enhancers enabled whereas the final parameters have only piggybacking enabled. Moreover, we only pick those points satisfying this condition which show the same trend—either an increase or decrease in throughput—across both CMTS at the required significance level of 95%. For the current question, we find statistically significant improvements only for the points shown in Table II (A). For the remaining cases, either piggybacking is not useful or there is not enough evidence to conclude so with 95% significance. Table II (B) answers the related question if, when and how much better piggybacking performs compared to concatenation. We find that for larger packets and smaller channels piggybacking is capable of improving throughput even in the single CM scenario by as much as 10-20%[3]. Table V (A) on the other hand answers the complementary question: when and by how much is concatenation better than piggybacking? As alluded to by the results in Table III (G), concatenation proves much better than piggybacking for smaller packet sizes. Table V (B) illustrates the cases where throughput with concatenation is higher than that with no enhancers. Again smaller packet sizes are more amenable to concatenation. Finally, Table V (C) asks the question if, when and by how much throughput is lowered by using both enhancers instead of concatenation by itself. We find that throughput can be lowered by as much as 10-15% for smaller packet sizes if both enhancers are used simultaneously in a single CM scenario. The questions asked above are only a small sample—many other such answers can be mined from the database.

V. CONCLUSION

We have conducted an extensive evaluation of the throughput of an actual DOCSIS upstream link. We report the results of our measurements and based on the data collected answer some questions of interest to network designers and operators as well as device developers. We conclude that concatenation is most effective for small packet lengths but is outdone by piggybacking for larger packet sizes and lower channel rates. In general, even in the single CM scenario, piggybacking is useful for some packet lengths across medium sized channels. We also find that at least in the single CM scenario, concatenation alone works much better than in combination with piggybacking. Our conclusions are robust across two different CMTS implementations and are acceptable with 95% significance. The WSRS test is well-suited for testing significance of differences between alternative schemes.

VI. FUTURE WORK

The second set of our experiments is still incomplete. For example, piggybacking must be studied with multiple CMs. In general, the performance of the system in multi-CM scenarios remains largely unexplored. In addition to these simple performance enhancers, DOCSIS also provides a set of guaranteed flows such as unsolicited grants (guaranteed bandwidth), polled real time and non-real time grants (for voice and video traffic). Two other schemes, payload header suppression and fragmentation also remain open for study.

ACKNOWLEDGMENT

We would like to thank the University of New Hampshire's InterOperability Laboratory (http://www.iol.unh.edu) for their support. We also thank the CMTS and CM vendors for providing us with the necessary configuration and operation information for their equipment.

REFERENCES

[1] *Data-Over-Cable Service Interface Specifications-Radio Frequency Interface Specification*, Cable Television Laboratories, Inc. Standard, Rev. SP-RFI v1.1-I06-001215, Version 1.1, 2000.

[2] V. Sdralia, C. Smythe, P. Tzerefos, and S. Cvetkovic, "Performance Characterization of the MCNS DOCSIS 1.0 CATV Protocol with Prioritized First Come First Serve Scheduling," *IEEE Trans. Broadcast.*, vol. vol. 45 no. 2, pp. 196–205, June 1999.

[3] S.-H. Cho, J.-H. Kim, and S.-H. Park, "Performance Evaluation of the DOCSIS 1.1 MAC Protocol According to the Structure of a MAP Message," in *Proceeding of IEEE ICC 2001*, Helsinki, Finland, June 2001.

[4] N. Golmie, F. Mouveaux, , and D. Su, "A Comparison of MAC Protocols for Hybrid Fiber/Coax Networks: IEEE 802.14 vs. MCNS," in *Proc. IEEE ICC'99*, Nagoya, Japan, June 1999.

[5] G. Chandrasekaran, M. Hawa, and D. W. Petr, "Preliminary Performance Evaluation of QoS in DOCSIS 1.1," Information and Telecommunication Technology Center, University of Kansas, Tech. Rep. ITTC-FY2003-TR-22736-01, Jan. 2003.

[6] R. Domdom, B. Espey, M. Goodman, K. Jones, V. Lim, and S. Patek, "A Simulation Analysis of the Initialization of DOCSIS-Compliant Cable Modem Systems," in *Systems Engineering Capstone Conference*, University of Virginia, 2000.

[7] Y. D. Lin, C. Y. Huang, and W. M. Yin, "An Investigation into HFC MAC Protocols: Mechanisms, Implementation, and Research Issues," *IEEE Comm. Surveys*, vol. Third quarter, 2000.

[8] R. Bartoš, C. K. Godsay, and S. Fulton, "Experimental Evaluation of DOCSIS 1.1 Upstream Performance," in *Proc. of the IASTED International Conference on Parallel and Distributed Computing and Networks (PDCN)*, Innsbruck, Austria, Feb. 2004.

[9] C. K. Godsay and R. Bartoš, "Experimental Evaluation of DOCSIS 1.1 Upstream Performance," Dept. of Computer Science, Univ. of New Hampshire, Tech. Rep. TR 04-01, January 2004.

[10] J. A. Rice, *Mathematical Statistics and Data Analysis.* Duxbury Press, 1994, ch. 11, 12.

---

[3] All percentage changes reported reflect the trend. Real values vary widely from a few to a few hundred percent.