A MEASUREMENT STUDY OF TWO LOCAL AREA NETWORK DATA LINK LEVEL PROTOCOLS

B. Lewis Barnett, III and Michael K. Molloy

Department of Computer Sciences
The University of Texas at Austin
Austin, Texas 78712-1188

TR-88-18 May 1988

A Measurement Study of Two Local Area Network Data Link Level Protocols

B. Lewis Barnett, III

Department of Computer Sciences
Taylor Hall 2.124
University of Texas at Austin
Austin, TX 78712
barnett@im4u.utexas.edu

Michael K. Molloy

Department of Computer Science
4212 Wean Hall
Carnegie-Mellon University
Pittsburgh, PA, 15213
mkm@k.gp.cs.cmu.edu

ABSTRACT

In this paper, two protocols for the data link level of Local Area Networks, the original Ethernet protocol and the EnetII protocol, are compared. The two protocols were implemented on a testbed network. A variety of performance measurement experiments were run for each protocol, and information on throughput and packet delay was collected. The measurement results for Ethernet are compared to simulation results.

1. Introduction

Local Area Networks have become increasingly important in many computing milieux, from the automated office to the university campus. In some installations, the design limits of the current generation of LAN equipment are being stressed. This is an opportune moment to examine alternatives to the predominant LAN system, Ethernet [DEC82], in anticipation of the design of the next generation of local networks. The area of protocol design has been active since the introduction of Ethernet, and many alternatives have been proposed to replace the Ethernet Data Link level component in Carrier Sense Multiple Access/Collision Detection (CSMA/CD) bus networks [Arth84, Brog82, Cape79, Gall78, Kies81, Moll81, Moll83, Toko77, Tows82]. The performance of these protocols has been calculated analytically where possible or simulated, but there have been very few actual implementations, and therefore very few measurement studies.

Ethernet, on the other hand, is a popular networking system and has been the subject of several measurement studies. The first was performed by Shoch and Hupp at the Xerox Palo Alto Research Center [Shoc80] on the installation's experimental three megabit per second Ethernet. The emphasis of this work was to confirm the throughput capacity and stability of Ethernet under high loads and to characterize network traffic under normal conditions. No attempt was made to measure the delay suffered by packets as a result of queueing on the transmitting hosts. A more recent Xerox study by Gonsalves [Gons85] compared the performance of 3 and 10 megabit Ethernets and added packet delay measurements. Gonsalves produced traffic consisting of fixed length packets with uniformly distributed generation times. These results were more complete than those produced by Shoch and Hupp, but still did not investigate heavy loads composed of a realistic mixture of packets.

The significant missing factor is still the absence of direct comparisons of performance measurements between Ethernet and the other proposed protocols. Analytic models and simulations are limited by the accuracy of their assumptions about the system being modeled. In the case of local area networks, it has been shown that many of the assumptions made to keep the analyses tractable do not accurately reflect the actual conditions on the networks being investigated. Thus, measurements of actual systems are needed to augment the modeling and simulation results currently available. This paper presents the results of a performance measurement comparison of the Ethernet protocol and the EnetII protocol [Moll85b]. EnetII belongs to a family of protocols called Collision Resolution Protocols, which includes the Capetanakis tree protocol [Cape79] and the Gallager-Tsybakov protocol [Gall78]. These protocols attempt to partition the group of stations involved in a collision by various means, allowing one group to retransmit while the other defers. This differs from the Ethernet scheme, where colliding stations compute a random backoff and retransmit. The Ethernet mechanism is simple to implement and effective when traffic is light. However, when traffic is heavy the variance of packet delay increases rapidly. It is this problem that the Collision Resolution Protocols address. The performance of the Enet II protocol has previously been simulated by the author [Moll85b] and studied analytically by Liu and Wise [Liu87].

Many of the proposed protocols require a slotted, or synchronous medium. The CSMA/CD bus is an unslotted medium, and thus presents implementation difficulties for slotted protocols. The Enet II protocol performs collision resolution in the unslotted CSMA/CD environment and incorporates techniques which facilitate the implementation of slotted protocols in this environment. The difficulty with implementing most of the slotted collision resolution protocols on a CSMA/CD bus is that the algorithms must be able to determine when an idle 'slot' has occurred. In an unslotted medium, this determination is difficult. To understand how Enet II accomplishes this, it is necessary to note two facts. First, the Ethernet specification [DEC82] and the IEEE 802.3 standard [IEEE82] set a maximum length for cables in CSMA bus networks. Thus, a protocol can assume a fixed propagation delay, a, for the network. A station listening to the ether for $r = 2a$ after a message completes is guaranteed to hear any station which began transmitting after the message completion. Second, collision detection and the subsequent 'jam' (collision consensus enforcement) do not waste very much bandwidth. It is therefore practical to use intentional collisions to gain information about the state of other transmitting stations during the collision resolution period.

In Enet II, a station can be in one of three states; inactive, active, or deferred. Inactive stations either have no packets to send or have just successfully transmitted a packet. Active stations are trying to transmit a packet that may have been involved in a collision. Deferred stations have attempted a transmission but are waiting for active stations to complete their transmissions. In the algorithm statement that follows, r represents twice the propagation delay of the network. The protocol we describe here is referred to as "Naive Enet II." The protocol can be improved by adding counters as described in [Moll85b] to implement an unslotted Capetanakis protocol. The algorithm is described in Figure 1.

The transmit by active stations after the channel is observed idle for r produces a guaranteed collision which signals that no stations flipped heads, so stations should flip their coins again. Collisions observed within r of the original collision are viewed as actual collisions, while collisions observed between r and $2r$ after the original collision are viewed as signals that an idle step has just ended. The initial $3r$ wait serves to allow active and deferred stations to finish their transmissions before any new packets are transmitted.

2. Measurement Environment

This study was conducted using the Local Area Network Testbed (LANT) described in [Barn85]. The aim of LANT is to provide a tool for the investigation of the performance of data link level protocols under a wide variety of traffic conditions, from conditions matching those used in analytic studies to conditions emulating normal network usage patterns. The testbed consists of five Digital Equipment Corporation VAX 11/750 computers connected by an Ethernet Local Area Network. One of the computers acts as a monitor and experiment manager, and all five are used as load generators. The choice of five computers was motivated by a combination of economic considerations and the results presented by Shoch and Hupp. The results for utilization under continuously queued packet sources presented in [Shoc80] indicated that the effect of the number of transmitting hosts on utilization was minimal for more than 5 hosts. The

Algorithm Statement

Inactive $\rightarrow$ **active** (new packet available for transmission)
Sense channel
If idle, wait $3r$, then transmit
If not idle, wait until idle for $3r$, then transmit

Active
Successful transmission $\rightarrow$ inactive
If a collision occurs, flip a coin
If heads is flipped, try to transmit again; active
If tails is flipped, monitor the channel
If channel idle for r, transmit; active
If successful transmission occurs, transmit; active
If collision within r $\rightarrow$ deferred

Deferred
Monitor the channel
If idle for $2r$, transmit; active
Otherwise, repeat deferred action

Figure 1. Statement of the Enet II algorithm.

measurements presented here indicate that this is not the case. The load generation machines are equipped with 3Com 3C300 Unibus Ethernet controllers [Minn83]. These network interfaces were an early offering by 3Com, and performed the Ethernet backoff algorithm in the device driver rather than in hardware or microcode on the interface board. The design is uniquely suited for use in a performance measurement system such as LANT. Many of the proposed protocols can be implemented by modifying the packet transmission and collision handling routines in the UNIX† device driver for the interface. Of the many interesting challenges presented by building a testbed from off the shelf components, two of the most critical are generating adequate throughput from each machine (particularly in a testbed with a small number of machines) and providing accurate timing information. Minnich reported [Minn83] maximum process to process throughput for TCP/IP on a 4.1BSD Unix VAX 11/780 using the 3Com controller to be 355 kilobits per second. This figure is far too low to generate interesting loads with the equipment available for LANT, so high level software was not the answer. It was therefore necessary to modify the network device driver for the network controller to provide adequate generated throughput as well as delay and throughput measurements for the artificially generated traffic loads.

The artificial load generation code is included in the device driver. Throughput for each generator was boosted by bypassing the normal transmission process and using the large onboard packet buffers of the 3Com controller as a packet library. The 3Com has 32K of memory which is used as 16 2K packet buffers. The normal process for transmitting a packet is to copy the packet from main memory into the high end of one of these buffers, write an offset to the beginning of the packet in the first word of the buffer, and then initiate transmission by writing the buffer index into the transmit control register. Since the contents of artificially generated packets is irrelevant, it was possible to preload the buffers with one Ethernet header after the other. By filling fifteen of the sixteen buffers with headers in this way, it is possible to have a header in the proper location for transmitting a packet of any legal size with no copying of data from main memory onto the board. Consequently, to generate a packet of a given size, it is only necessary to look up the buffer number and offset of the proper header in a table, write the offset to the first word of the buffer, and write the buffer number to the control register. This technique allows a single host to generate almost 60% of network capacity for maximum sized packets.

† UNIX is a trademark of Bell Laboratories.

Modifications were also necessary for timing purposes. The clock handling routine and clock interrupt frequency were adjusted to provide interval timers and timestamps of adequate resolution for the traffic measurements. During experiment execution, histograms are collected of the per-packet delay and packet length. Delay is measured from the time at which the packet is submitted for transmission to the time at which the transmission is successfully completed. This is accomplished by taking a timestamp immediately before the transmit control register of the controller is written with the transmit command and another immediately after the "transmit done " bit in the transmit control register is set. A histogram of the number of collisions suffered by packets is also collected.

This testbed does not have the advantage of a large equipment base such as the one used in the Xerox studies, and this fact affects the nature of the results generated. However, LANT does enjoy several advantages over the software and hardware used for the other studies. LANT experiment configuration software allows fully flexible specification of traffic patterns including the ability to define arbitrary distributions for packet lengths and interpacket delays. This is very important since network traffic stubbornly refuses to conform to any of the well know statistical distributions. LANT is a dedicated research network with no other computational or network activity. This allows the researcher to repeat experiments without interference from a user population. When performing comparisons between protocols it is vital to be able to reliably repeat traffic patterns in this way.

3. Enet II Implementation Notes

In two instances the testbed hardware and software made it impractical to implement the Enet II protocol exactly. The Enet II specification calls for timing of events in multiples of r, the round trip propagation delay of the network. For Ethernet, r is approximately 50 microseconds. The operating system clock routines were modified to provide interval timers with greater resolution, but increasing the clock interrupt rate to meet the 50 microsecond requirement affected the performance of the computers to an unacceptable degree. A clock resolution of 200 microseconds was finally implemented. As a result, the specified idle periods of r, $2r$, and $3r$ for the various protocol states are approximately four times as long as they theoretically could be. Throughput and delay characteristics of the resulting protocol are degraded, but not necessarily by a factor of four.

The second problem is also related to the idle monitoring periods, and is an effect of the packet reception process of the network controller used by the generators. In order to monitor network events, a promiscuous receive request is issued to the controller. Any packet which subsequently appears on the network is reported to the device driver, regardless of the packet's intended destination. If a packet was received during the monitored interval, it is indicated by a bit in the receive control register of the controller. A collision is indicated by the reception of a packet smaller than the minimum required packet size. An idle interval is indicated by the absence of the "receive done" bit in the receive control register of the controller. The problem occurs when the controller receives a packet whose transmission time is greater than the monitored interval. Since the "receive done" bit is not set until the transmission is complete, it appears that the ether was idle when the interval expires. This means that the protocol will occasionally mistake a successful transmission for an idle period. The result of this situation is that some packets will join the transmission process sooner than they should, and that some deferred packets will retransmit earlier than they should. Very few packets reach the deferred state, so it is believed that the performance degradation thus caused is negligible.

4. Experiment Design

Experiments were designed with two goals in mind: first, to provide a more complete understanding of the behavior of the protocols being investigated, and second, to determine how well the analytic and simulation models of protocol performance match the actual performance of the protocols. LANT allows the experimenter to control the distribution of packet sizes and the distribution of the interpacket interval for an experiment. Packet lengths can range from 46 bytes to 1500 bytes. The interpacket interval is specified in 200 microsecond units. (This interval is determined by modifications to the UNIX clock handling routine as discussed in Section 3.) For each of these quantities, the experimenter may choose from a menu of standard distributions, such as exponential and binomial, or may specify his own distribution by entering a series of x and y coordinates for the distribution.

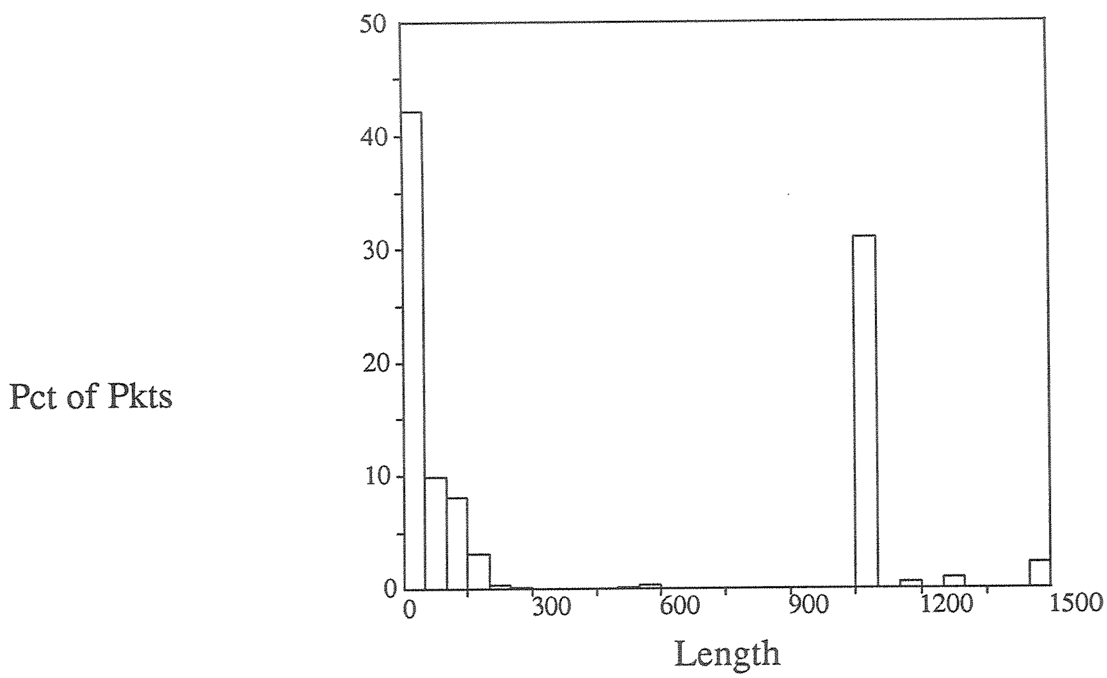

Figure 2. Packet length versus percentage of total packets.

In order to meet the first goal, experiments which closely match the traffic patterns of a large production network were specified. The network in question is the University of Texas campus ethernet. A traffic study of the network was conducted using the ILMON network monitoring package [Barn87]. The network connects some 80 nodes, including mainframes, minicomputers, workstations, fileservers, laser printers and terminal concentrators. Many other machines reside on subnets and contribute to the network's traffic. Several higher level protocols, such as the DOD/ARPA protocol suite, DECnet, and CHAOSNET, are used. The network was monitored for a 24 hour period, with samples summed over six minute intervals. During this period, 8.7 million packets were transmitted, containing 3.8 billion bytes of information, not including the Ethernet headers, preambles and frame check sequences. Figure 2 is a histogram of observed packet lengths. This figure shows the classic bimodal distribution, with a large percentage of small packets carrying terminal traffic, and a smaller percentage of very large packet associated with file transfers. For the sake of efficiency, this distribution was abstracted to a two point distribution which produced 70% minimum sized packets and 30% packets 1024 bytes long. A series of experiments using this packet length distribution with decreasing interpacket intervals was run. However, using this distribution with the available load generation hardware did not produce network saturation, so a second set of experiments was run with scaled up packet sizes.

To meet the second goal, it was necessary to collect data which corresponded well with the models used for analytic and simulation studies. Most of these studies make a number of simplifying assumptions, such as fixed packet lengths and fixed or exponentially distributed packet arrival times. A set of experiments using fixed packet size for various fixed packet intervals was defined to meet this requirement.

The possible load levels for the generators are determined by the modified interval timer. Interpacket intervals are implemented by setting an interval timer when a packet transmission completes. Thus, possible interpacket intervals are multiples of the timer resolution plus the overhead for generating the packet length and interval value. For the results presented here, experiments were run at each available level for the "interesting" portion of the curve (i.e. approaching and exceeding the saturation point) and were spread out at appropriate intervals for the lower loads. At least three experiments of ten minutes duration were run at each level. This duration results in at least 50,000 packets from each generator. For the packet delay figures, this gives us a confidence level of 95% with a confidence interval of 200 microseconds, the resolution of our timer. Offered load for each level was determined by running an experiment using the same *load pattern* (packet length and interpacket interval distributions) on a single load generator, then multiplying by the number of hosts participating in the experiment.

5. Results

The results of the study will be presented in two parts. The first will be a comparison on the performance of the two protocols. Graphical results for throughput versus offered load, queueing delay versus throughput, and queueing delay variance versus offered load will be presented. The second part will be a comparison of the observed throughput and delay of the protocols with the predicted values for these quantities. Throughout this discussion, the following abbreviations will be used. The offered load will be referred to as G. The offered load for a given load pattern is calculated by multiplying the average throughput of a single generator running the pattern in the absence of contention by the number of generators participating in the experiment. S is the observed throughput for the network. It is calculated by summing the average throughput from each generator. D refers to the average per-packet queueing delay. This delay is measured from the time the packet is first submitted for transmission until it is successfully transmitted. This figure includes the transmission time of the packet. V refers to the variance of D.

5.1. Protocol Performance Comparison

The goal of the Enet II protocol is the improve on the delay characteristics of the Ethernet protocol without resorting to a token based system. It was expected that the packet queueing delay would be higher than that of Ethernet at low traffic levels, but would improve on the delay of Ethernet as the offered load increased. It was further expected that the variance of delay would be greatly improved over that of Ethernet. Experiments were performed using both fixed length packets and a mixture of packet lengths.

5.1.1. Fixed Length Packets

Figures 3 through 5 show the behavior of the protocols for 1500 byte packets with fixed interpacket intervals. Five generators were used to load the network for these measurements. Figure 3 shows the throughput versus offered load of the two protocols. The units for both axes are megabits/second (Mbps). As expected, at low loads, the throughput characteristics of the two protocols are very similar. However, as contention begins at approximately 80% utilization, Enet II begins to demonstrate a slight advantage. Both protocols exhibit a peak throughput of 9.8 Megabits/second. However, Enet II achieves this peak at an offered load of 10.26 Mbps, while for Ethernet the peak appears at 11.08 Mbps offered load. This difference is attributable to the lower delay variance for Enet II. Both protocols experience a drastic drop in throughput as queueing delays from contention begin to dominate the transmission time. For Ethernet, this occurs between 11.51 Mbps and 11.95 Mbps offered load. Due to constraints mentioned in Section 4, it was not possible to take measurements at intermediate load levels. Enet II experienced this decrease at lower offered load level, between 11.11 Mbps and 11.54 Mbps. Throughput stabilizes again at approximately 7.5 Mbps for Ethernet and 7.7 Mbps for Enet II.

Figure 4 shows the packet queueing delay versus the throughput. The x axis units are Mbps and the y axis units are microseconds. As expected, Ethernet outperforms Enet II at low packet loads, and both protocols experience little increase in delay until the offered load exceeds 100% of capacity. However, rather than improving upon the delay of Ethernet, Enet II merely decreases the distance between the two curves. As throughput begins to decline after nearing the capacity of the channel, the distance between the curves is approximately 300 microseconds. This difference can be explained by the interval timer features discussed in Section 3. It should be noted that using the larger value of r increases the initial gating delay

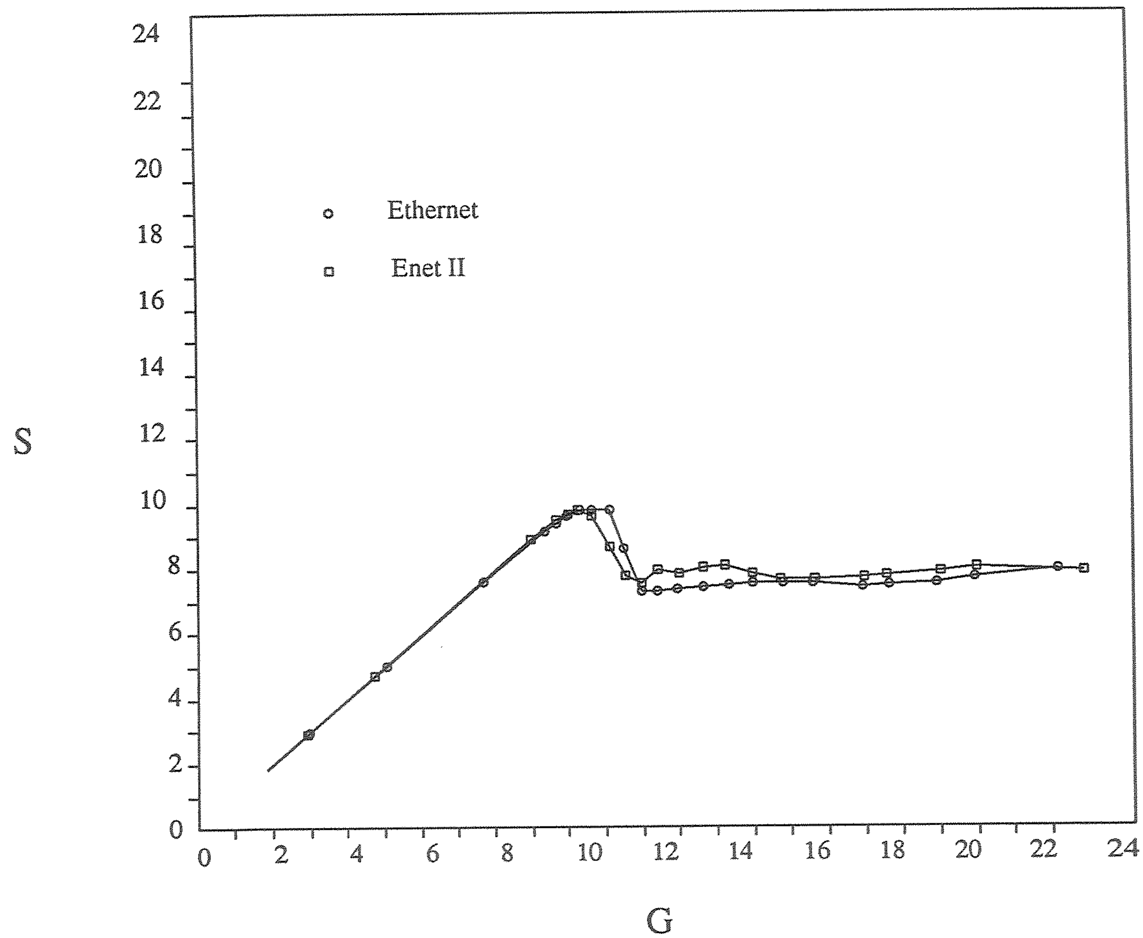

Figure 3. Throughput (S) vs. offered channel load (G) for five generators sending 1500 byte packets. X and Y axis units = megabits per seconds.

by 450 microseconds, so it is reasonable to expect that an exact implementation of the protocol would demonstrate the expected improvement under overload conditions. The sharp increase in delay corresponds to the sharp decrease in throughput shown in Figure 3.

Figures 5a and 5b show the delay variance behavior of the two protocols. The x axis is the offered load in Mbps and the y axis is the delay variance in microseconds. The curve has been split into two graphs due the the large time range covered. Figure 5a shows the curve for the region of normal operation. As predicted, Enet II shows a smaller delay variance than Ethernet over most of the range of operation. Figure 5b shows the curve for overload situations. Here Enet II is shown to achieve much lower variance than Ethernet. This indicates the greater stability of Enet II under extreme overload conditions.

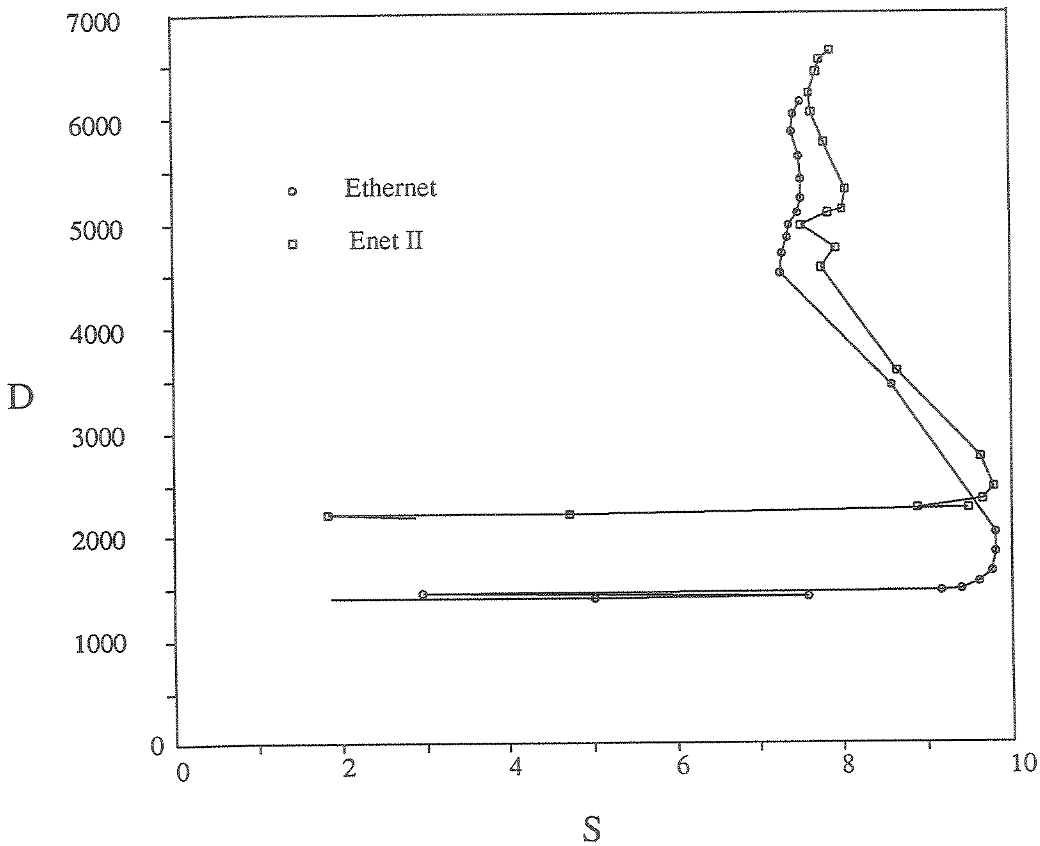

Figure 4. Packet queueing delay (D) versus Throughput (S) for five generators sending 1500 byte packets. X units = megabits per second, Y units = microseconds.

5.1.2. Mixed Packet Lengths

Figures 6 through 8 show the results of experiments with a mixed packet size distribution and fixed interpacket interval. The generators produced 55% packets of 200 bytes and 45% packets of 1500 bytes. Five generators were used for these measurements. Figure 6 shows throughput versus offered load. The introduction of shorter length packets has increased contention and decreased the maximum throughput of both protocols. The performance of the two protocols is nearly identical as offered load increases to 7 Mbps. Ethernet reaches its peak throughput of 7.0 Mbps at an offered load of 12.0 Mbps, while the throughput of EnetII is still increasing at the maximum achievable offered load of 14.59 Mbps. The curves show the same basic structure as the 1500 byte curves, but the changes in slope are less pronounced.

Figure 7 shows the per-packet queueing delay versus the throughput. The y axis is in units of microseconds while the x axis is in Mbps. The features of the curve are similar to those in Figure 4. Enet II closes the gap as the channel becomes congested, and finally crosses the Ethernet curve under extreme overload. The distance between the curves is increased due to smaller packets increasing the contention on the wire. Again, it is suspected that the implementation difficulties have profoundly affected the delay performance of the Enet II protocol.

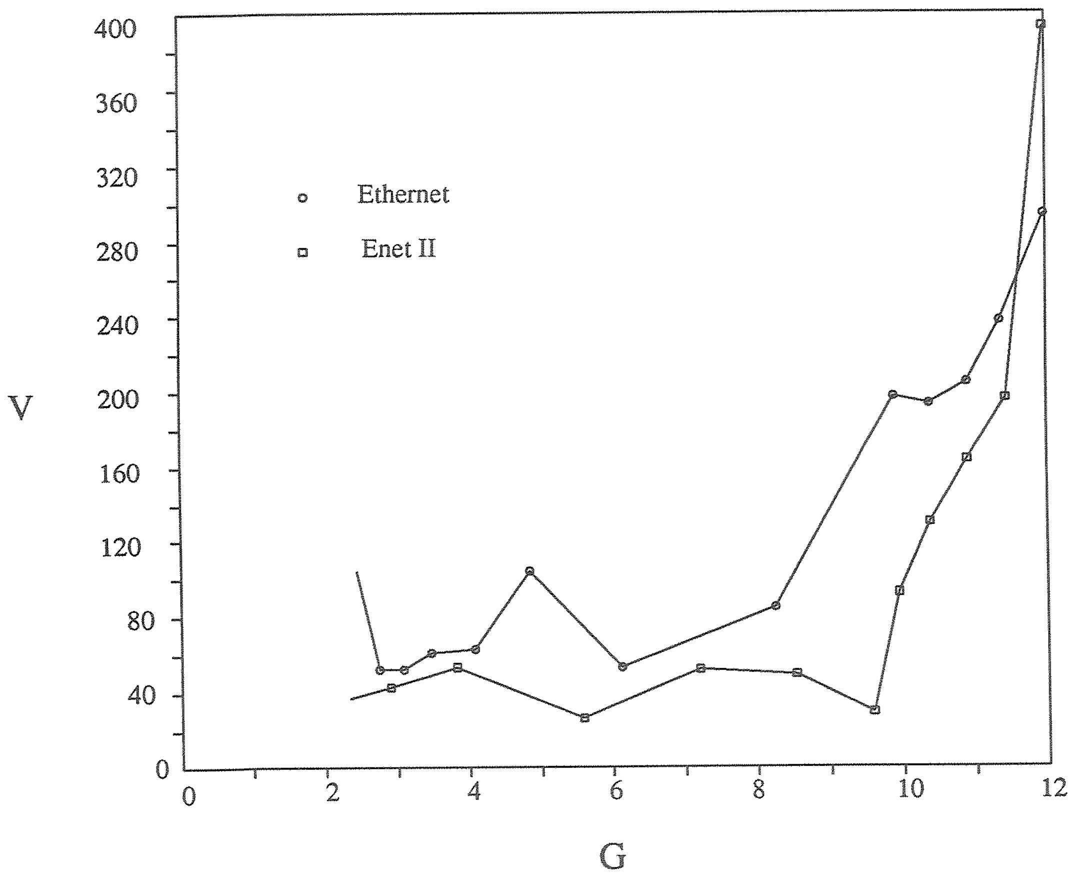

Figure 5a. Delay variance (V) versus offered channel load (G) for four generators sending 1500 byte packets. Y units = microseconds, X units = megabits per second. Normal operating range.

Figure 8 shows the delay variance performance for the two protocols with mixed packet length loads. The x axis shows the throughput in Mbps and the y axis shows the delay variance in microseconds. The delay per packet is greater across the operating range due to the increase in contention caused by the presence of small packets in the traffic. Enet II once again performs slightly better in the normal operating range as throughput approaches capacity and improves vastly on the performance of Ethernet under overload conditions. This once again indicates the superior stability characteristics of the Enet II protocol.

5.2. Comparison to Simulation Results

A simulation of Ethernet was run for comparison purposes. The program simulated a network with an infinite number of transmitting hosts. Packet length was fixed at 1500 bytes. The results are shown in Figures 9 and 10. Figure 9 shows throughput versus offered load for the simulator and the testbed network. The testbed measurements show a peak throughput of 9.82 Mbps, while the simulator reaches a peak of 9.46 Mbps. Since the simulator used an infinite population model, contention was greater and began earlier than in the testbed, which was limited to a population of five generators in the measurements shown here.

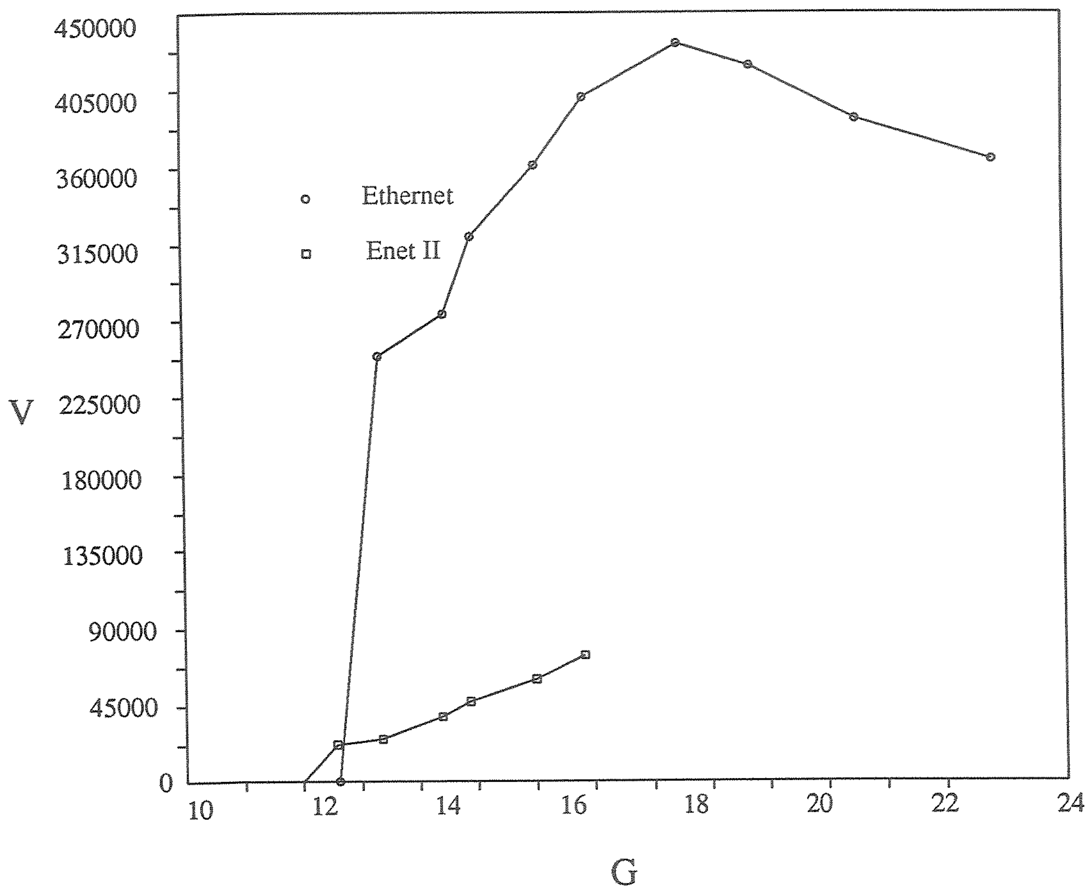

Figure 5b. Delay variance (V) versus offered channel load (G) for four generators sending 1500 byte packets. Y units = microseconds, X units = megabits per second. Heavy overload conditions.

The testbed data also show a dramatic drop in throughput at approximately 11.5 Mbps offered load which is not repeated in the simulation data. This drop occurs at the point where most of the packets in the testbed experiments suffered collisions that involved most or all of the five available generators. In the simulation, with an infinite population of generators, other stations were always waiting to transmit while those involved in a collision backed off. However, when all stations in the testbed collided, an idle period resulted while the colliding stations backed off. Here we see that the relatively small generator population of the testbed had a profound effect upon the generated traffic in spite of the results reported in [Shoc80]. Figure 10 shows the packet queueing delay versus offered load behavior of the testbed and simulator. Again, the effects of the small number of hosts in the testbed are evident. Contention begins in the simulation data at much lower offered load causing increased delay. The simulation data was truncated at just under 7 Mbps offered load to preserve the important features of the curves; the slope of the curve remains nearly constant until capacity is reached, at which time it moderates slightly. The average delay at that point is approximately 110 milliseconds. In the experiments with maximum sized packets, the testbed generators did not begin experiencing significant contention until the combined offered traffic exceeded the

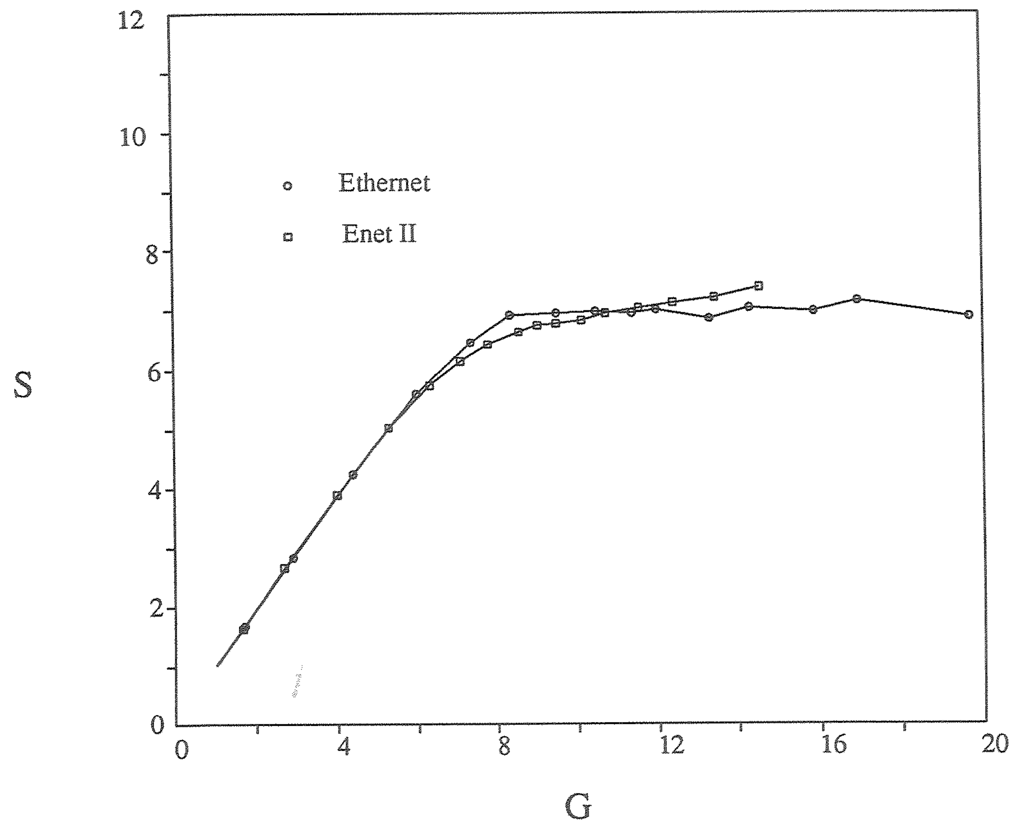

Figure 6. Throughput (S) versus offered channel load (G) for five generators sending mixed length packets. X and Y axis units = megabits per second.

capacity of the network. Thus, the knee in the curve occurs at about 11.5 Mbps. The testbed data also clearly indicates the effect of the truncation of the backoff scheme after 16 retries by Ethernet, as indicated by the leveling off of the delay curve at about 13 Mbps. Of course, the packets which were dropped as a result of this feature would eventually be resent in a real system by a higher level protocol, adding further to the offered traffic.

Gonsalves also reports measurements for experiments with 1500 byte packets. Throughput for 1500 byte packets peaked at approximately 85% of capacity and was stable at that level up to 1000% percent offered load. This once again points up the effect of a small testbed on throughput measurements; the maximum throughput is lower due to greater contention for the network bandwidth, and the drop in throughput due to all stations executing the backoff algorithm after collisions does not appear. Correspondingly, the delay figures show the effects of contention much sooner than the LANT experiments and match the simulation curve well for 1500 byte packets.

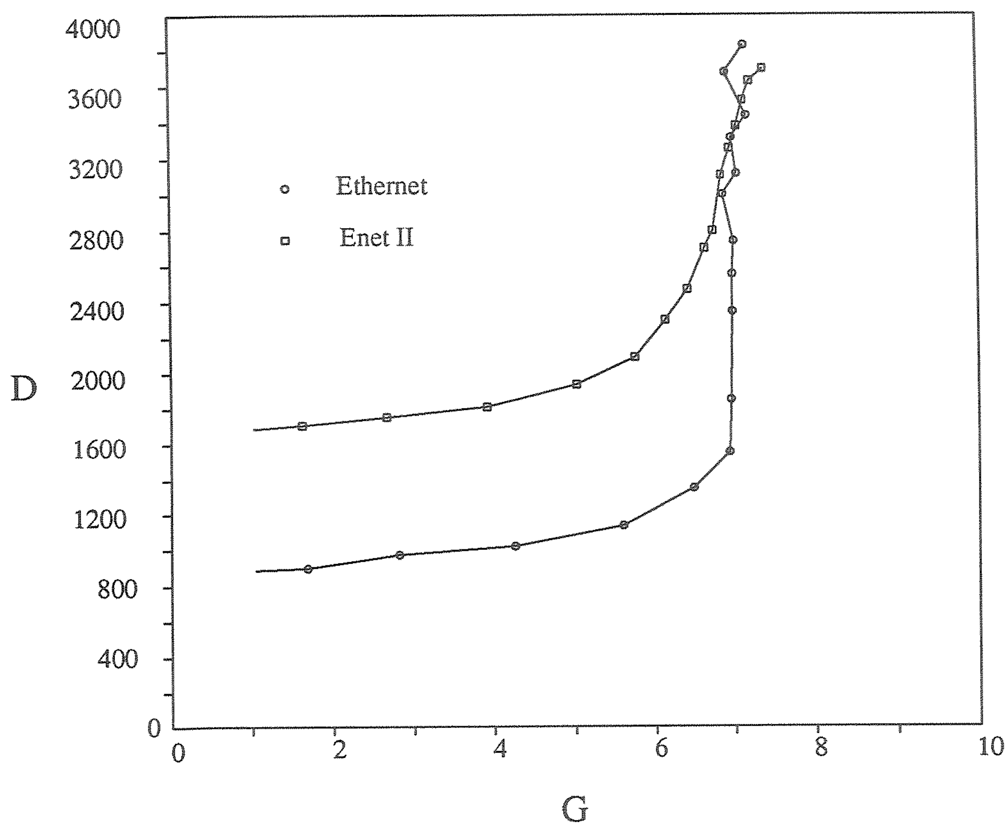

Figure 7. Queueing delay (D) versus Throughput (S) for four generators sending mixed length packets. X axis units = megabits per second. Y axis units = microseconds

6. Conclusions

We have investigated the performance of the Ethernet and Enet II protocols under varying load conditions. The protocols were found to have similar throughput and delay characteristics under normal network conditions, while Enet II achieved higher throughput and greater stability under extreme overload conditions. The predicted delay advantage for Enet II was not observed in these measurements due to implementation compromises, but Enet II was observed to achieve its goal of greatly reducing the variance of the packet queueing delay. It was also noted that, contrary to previously reported results, network utilization is sensitive to the number of traffic producing hosts for five or more hosts. A more appropriate testbed would consist of thirty or more inexpensive workstations with network interfaces similar to those used in LANT and the same experiment configuration capabilities.

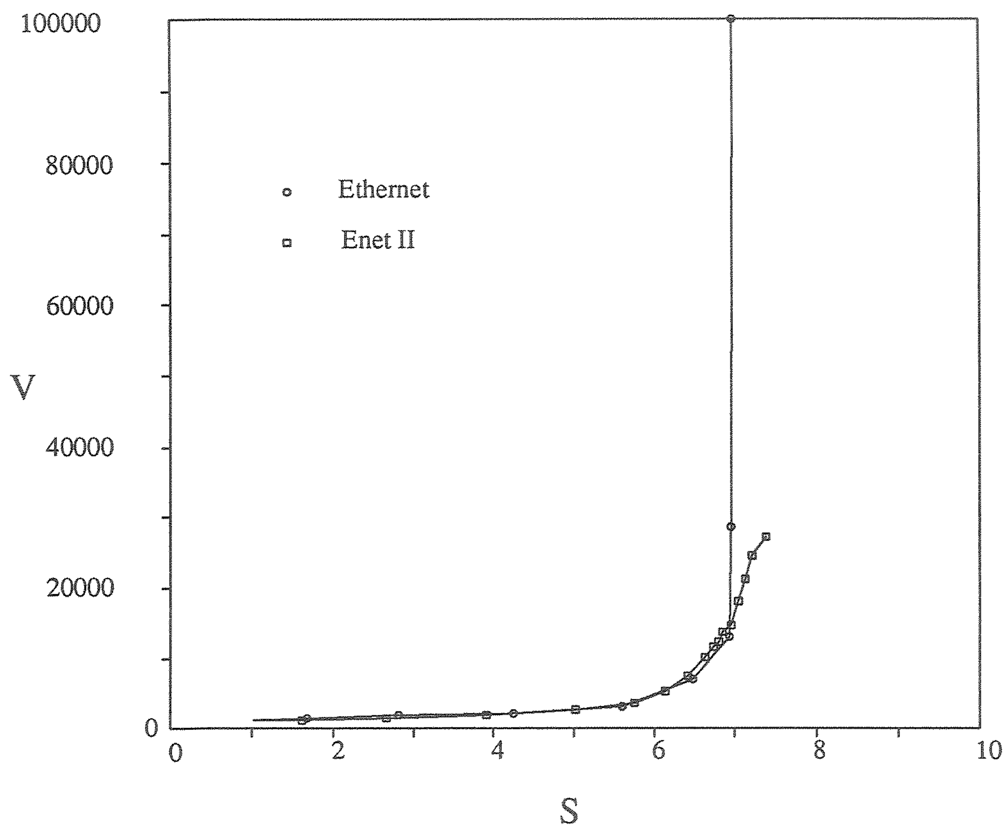

Figure 8. Queueing delay variance (V) versus Throughput (S) for five generators sending mixed length packets. X axis units = megabits per second. Y axis units = microseconds

References

Arth84 E. Arthurs and B.W. Stuck, "A Modified Retry Policy for ETHERNET Version 1.0 Data Link Layer," *IEEE Transactions on Communications,* Vol. COM-32, No. 8, August 1984, pp. 977-979.

Barn85 B. Lewis Barnett and Michael K. Molloy, "Local Area Network Testbed Design," Technical Report TR85-25, Department of Computer Sciences, University of Texas at Austin, May 1985.

Barn87 B. Lewis Barnett and Michael K. Molloy, "ILMON: A UNIX Network Monitoring Facility," *Winter 1987 USENIX Association Conference Proceedings,* Washington, D.C., January 21-23, 1987, pp. 133-144.

Brog82 F. Brogonovo and L. Fratta, "A Collision Resolution Algorithm for Random Access Channels with Echo," *Proceedings of INFOCOM 82,* April 1982, pp. 68-75.

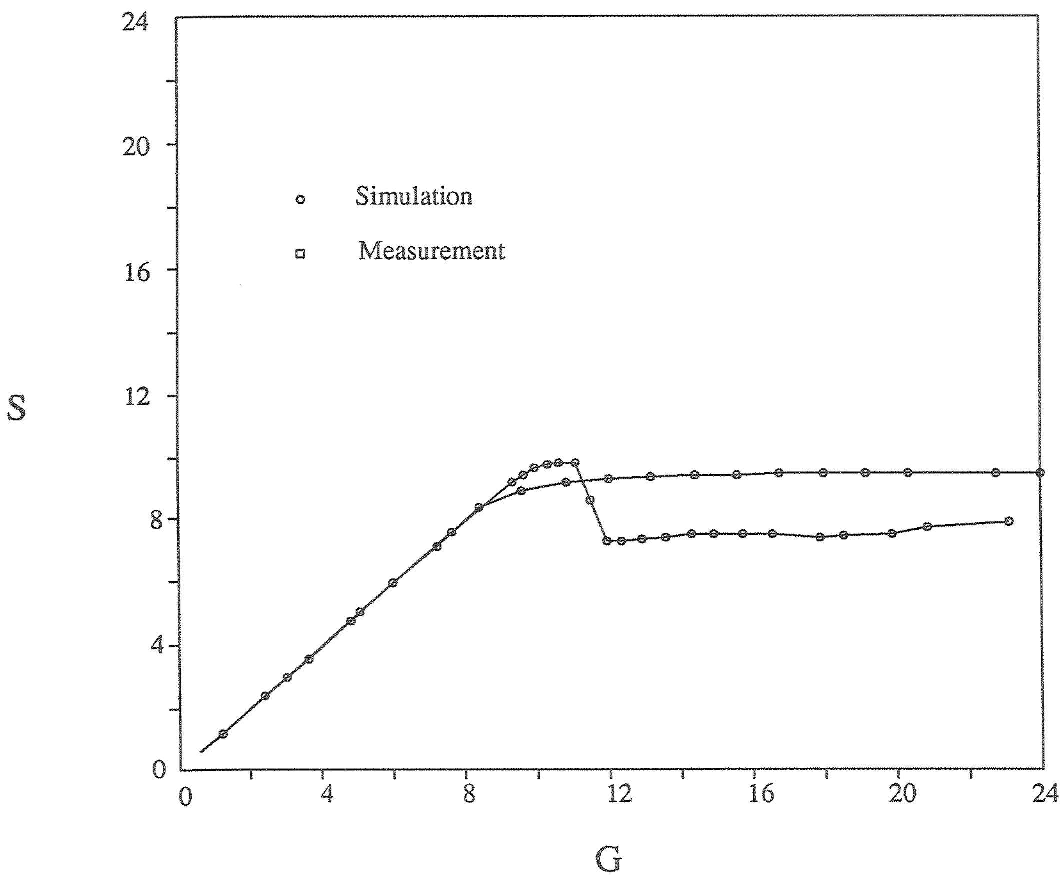

Figure 9. Throughput (S) versus offered load (G) for simulated and measured Ethernet. 1500 byte packets, x and y axes in Mbps.

Cape79 J. Capetanakis, "Tree Algorithms for Packet Broadcast Channels," *IEEE Transactions on Information Theory*, Vol. IT-25, September 1979, pp. 505-515.

Gall78 R. Gallager, "Conflict Resolution in Random Access Broadcast Networks," *Proceedings of the AFOSR Workshop in Communication Theory and Applications*, Sept 1978, pp. 74-76.

Gons85 Timothy A. Gonsalves, "Performance Characteristics of 2 Ethernets: an Experimental Study," *Proceedings of the 1985 ACM SIGMETRICS Conference on Measuring and Modeling of Computer Systems*, August 1985, pp. 78 - 86.

IEEE82 "IEEE Project 802, Local Area Network Standards," Draft IEEE Standard 802.3 CSMA/CD Access Method, Draft D, 1982

Kies81 W.M. Kiesel and P.J. Kuehn, "CSMA-CD-DR: A New Multi-access Protocol for Distributed Systems," *Proceedings of NTC 81*, December 1981, pp. A2.4.1-A2.4.6.

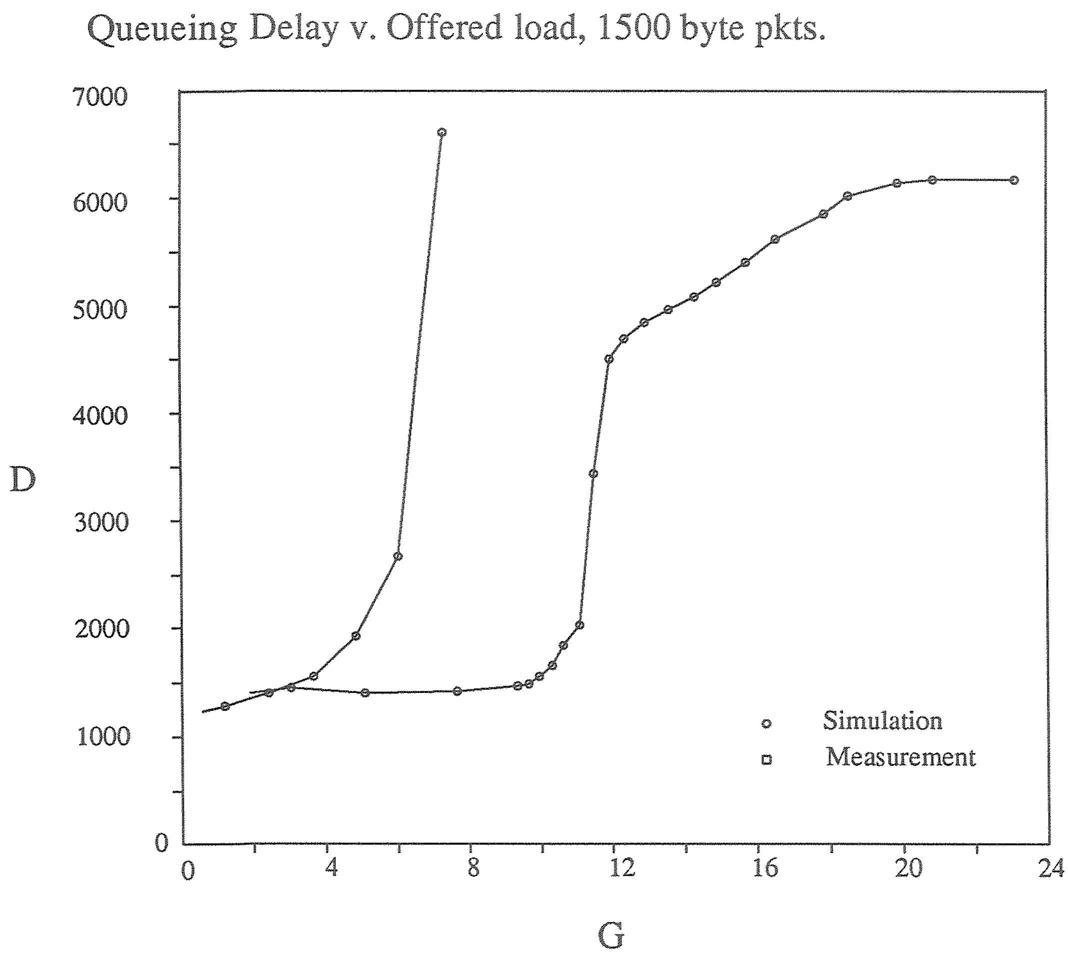

Figure 10. Packet queueing delay (D) versus offered load (G) for simulated and measured Ethernet. 1500 Byte packets, x axis in Mbps, y axis in microseconds.

Liu87 Yih-Chiao Liu and Gary L. Wise, "Performance of a CSMA/CD Protocol for Local Area Networks," *IEEE Journal on Selected Areas in Communications, Vol. SAC-5, No. 6,* July 1987, pp. 948-955.

Mass80 J. Massey, "Collision Resolution Algorithms and Random-Access Communication," UCLA Technical Report UCLA-ENG-8016, 1980.

Minn83 N.M. Minnich and C.J. Cotton, " An Evaluation of Two Unibus Ethernet Controllers," *Proceedings of the 8th Conference on Local Computer Networks,* Minneapolis, Minnesota, October 17-19, 1983, pp. 29 - 36.

Moll81 M. Molle, " Unification and Extensions of the Multiple Access Communications Problem," PhD Thesis, UCLA Technical Report CSD-81073, June 1981.

Moll83 M. Molle, "Asynchronous Multiple Access Tree Algorithms," *Proceedings of SIGCOMM 83 Symposium on Communications Architectures and Protocols,* ACM #533830, March 1983, pp. 214-218.

Moll85a M. Molle and L. Kleinrock, "Virtual Time CSMA: Why Two Clocks Are Better than One," *IEEE Transactions on Communications*, Vol. COM-33, No. 9, September 1985, pp. 919-933.

Moll85b Michael K. Molloy, "Collision Resolution in an Unslotted Environment," *Computer Networks*, Vol. 9, No. 3, March 1985, pp. 209-214.

Shoc80 John F. Shoch and Jon A. Hupp, "Measured Performance of an Ethernet Local Network," *CACM* Vol. 23, No. 12, December 1980, pp. 711-721.

3Com82 *UNIBUS Ethernet Controller Reference Manual*, 3Com Corporation, Mountain View, California, 1982.

Toko77 Mario Tokoro and Kiichiro Tamara, "Acknowledging Ethernet," *Proceedings of COMPCON 77*, September 1977, pp. 320-325.

Tows82 D. Towsley and G. Venkatesh, "Window Random Access Protocols for Local Computer Networks," *IEEE Transactions on Computers*, Vol C-31, No. 8, August 1982, pp. 715-722.