Voice and Data Session Capacity over Local Area Networks

Theresa A. Fry[1], Ikhlaq Sidhu[2], Guido Schuster[2], and Jerry Mahler[2]
[1]Department of Electrical and Computer Engineering
Northwestern University, Evanston, IL 60208-3118
[2]3COM Advanced Technologies Research Center

Abstract

We analyze the effects of adding voice traffic to a packet switched network. We develop and validate separate simple and general analytical models for data and voice traffic and then combine them. Our analytical models assume no particular MAC layer and no priority of service. We analyze the effects on the average voice packet delay of the number of voice and data sessions, packaging of multiple voice frames in a single packet, contention at the MAC layer, and LAN utilization.

Our first goal was to develop a model which can predict the number of real time voice sessions which can coexist with (TCP) data sessions. By appropriately parameterizing the model for narrowband voice codecs like G.723.1, we find some very interesting results. First, we find that hundreds of such voice sessions can share a 10 Mbps Ethernet link while maintaining appropriately low delays (below 30 ms). (See Figure 5) Next we see the surprising result that with as few as 5-10 simultaneous data sessions, the number of supportable voice sessions drops to zero. (See Figure 6)

As a second important result, we show that combining multiple voice frames into a single larger packet before transmission increases the number of both data and voice sessions that can be supported. (See Figure 8)

1. Introduction

In the past, networks have typically been designed to accommodate one specific type of traffic. There is the Public Switched Telephone Network (PSTN) for voice traffic; and Wide Area Networks (WANs), Local Area Networks (LANs), and the Internet for data traffic. Recent efforts have been towards combining all traffic types in a single network. One important reason for working toward a single network which accommodates all traffic types is that it is less costly and simpler to install and maintain one network instead of two. For this reason and many others it is important to study the performance of multiple traffic type networks.

Existing packet-switched networks were designed to accommodate data traffic, according to its specific set of characteristics and traffic requirements. Voice and video traffic have their own stipulations which are very different from data traffic requirements. For example, voice traffic requires limited packet delays (and delay variance) for acceptable quality of service, packets are small, and arrivals are deterministic. Voice traffic can also tolerate a certain percentage of errors and permits no retransmission because of added delays. Local area network protocols have been optimized for data traffic, which consists of much larger packets, is generally bursty, tolerates long delays, requires low error-rate transmission, and allows packet retransmissions.

Until recently, Local Area Networks such as Ethernet have not been considered as viable media for traditionally circuit-switched traffic. Other networks and protocols have been proposed which are more suited for voice traffic. In particular, Asynchronous Transfer Mode (ATM) switching networks have been championed as the solution to the single network problem. ATM networks provide different Quality of Service (QoS) levels and divide the channel into fixed length cells, which allow real-time traffic to acquire the channel at constant intervals, limiting the delay variance.[1] Another solution synchronizes packetized voice and data, utilizing the master frame format of Time Division Multiplexing (TDM) to ensure real-time service delays.[2] Also, the IEEE 802.1 Internetworking Task Group is currently working on the 802.1p standard which will enhance the Ethernet MAC layer to allow different traffic classes and implement priorities.[3] However, all of these solutions require changes to existing Internet and Intranet networks. Our goal is to analyze the ability of the existing LAN segments to carry voice traffic.

Ethernet is one of the most common LAN technologies in the market today and many capacity measurement and performance studies have been done to determine its capabilities.[4,5,6,7] Some of the original assumptions and conditions which eliminated Ethernet from consideration for voice traffic have changed, and have made it worth another look. Ethernet's CSMA/CD utilizes the Binary Exponential Backoff algorithm for contention resolution. Studies have shown this scheme causes a capture effect, in which the more times a packet contends the longer its backoff time becomes. Thus newly arriving packets will most likely be transmitted before packets which have been waiting. Because of this capture effect, Ethernet has been thought to be unsuitable for real-time traffic, which must be ensured certain packet delay times. Some have proposed a new contention resolution algorithm, the Binary Logarithmic Arbitration Method, which requires changes to the MAC level of the existing Ethernet.[8] However, the collapsed backbone Ethernet which fosters the problematic contention is commonly being replaced with a switched network, which may increase its capacity for real-time traffic.

Previous studies have focused on the Ethernet MAC layer without considering higher network layers. For example, Slothouber uses a queueing model to represent a Web server but ignores TCP/IP protocol effects.[9] Paxson measures network delays of all layers and then removes the upper layer effects of TCP to distinguish the network dynamics.[10] In practice, TCP sends bursts of data and then waits for acknowledgment. We argue that gaps between TCP traffic bursts can be used to transmit the relatively small voice packets. Because of the widespread existence of Ethernet in today's market, it is likely to end up supporting mixed voice and data services. Thus, it is

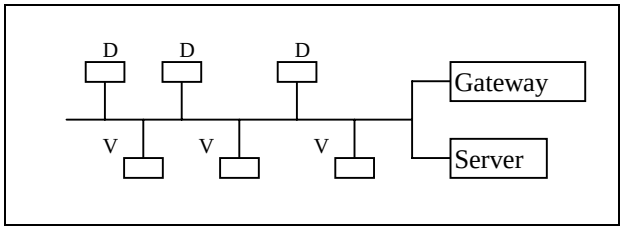
useful to analyze how well the existing Ethernet can support these traffic types.

We first develop an analytical model to determine the average data packet delay based on the number of data sessions on a single Ethernet segment. The model is based on a TCP windowing scheme and assumes no specific MAC layer protocol. We then devise a model for a network of only voice traffic to determine the average voice packet delay based on the number of voice users on the subnet. Although voice packet arrivals are deterministic, we use an M/M/1 model to find an upper bound on the voice packet delay. Our final model incorporates both data and voice traffic. We analyze the effects on the average voice packet delay of the number of data and voice sessions, and show the trade-offs of adding voice vs. data sessions. Our results show that the average voice packet delay is highly sensitive to the number of data sessions on the subnet. Also, when the number of voice frames per packet is increased, we show that the average voice packet delay can be substantially reduced. Thus more data and voice sessions can be supported on the subnet while still providing an acceptable voice QoS on the average. This packetization scheme permits Intranets to service a considerable amount of voice traffic while still providing an acceptable average voice packet delay.

2. Network Model

In order to model a Local Area Network (LAN) we begin with a 10 Mbps collapsed backbone model, in which multiple workstations are connected through a single network hub. Figure 1 shows this configuration, where 'D' denotes a data session and 'V' denotes a voice session. As in existing LANs, our model assigns no priorities to different traffic types or users. Our model accounts for delays from the transport, network, data link, and physical layers of the network. The transport layer is assumed to be TCP. No particular MAC layer protocol is assumed; instead we presuppose that each user has equal probability to be the next one served.

Figure 1: LAN Segment Model

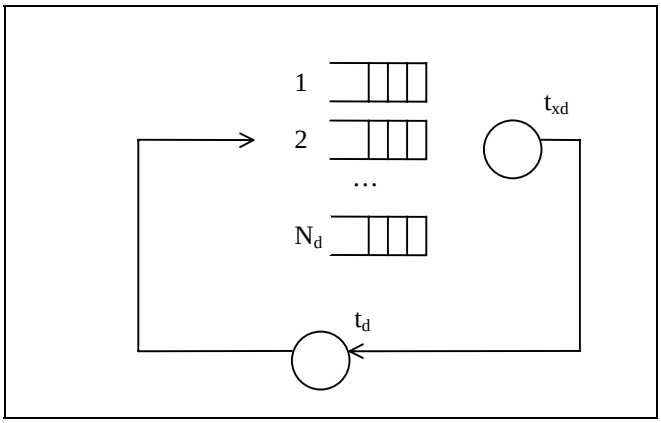


Our users consist of both data and voice sessions. A data session is an end-user terminal connected to the LAN. As a worst-case estimate, we assume that each data session always has data to transmit (as if, for example, each user is constantly sending data via *ftp*), and thus is always contending. As in TCP, data packets are windowed and acknowledged, with one acknowledgment (ACK) per window. A data packet has size L_d, and there are w packets per window. Typical TCP values are shown in Table 1. We simplify the TCP windowing scheme by assuming a constant window size, although TCP changes its window size according to the traffic on the LAN. A voice session is a telephone connected to the network, which transmits one voice packet every 30 ms. Voice packets are not windowed and have size L_v, which is much smaller than the data packet size. Thus, the transmission time of a voice packet is substantially smaller than that of a data packet. For our purposes, transmission time is the amount of time it takes to transmit the packet plus an interpacket gap time (it is assumed that there is some small delay between packets). Typical voice packet values are shown in Table 1.

Description	Data Param	Typical Data Value	Voice Param	Typical Voice Value
Window size	w	1-6 pkts	---	---
Packet size	L_d	12 kbits	L_v	512 bits
Transmit time	t_{xd}	1.44 ms	t_{xv}	.06144 ms
ACK Delay	t_d	25 ms	---	---

Table 1: Model Parameters and Typical Values

Figure 2: Data Queueing Model



2.1 Data Model

We first develop a model of a network in which all users are data sessions. We model data sessions to determine an average data packet delay to validate our results with TCP statistics. Each of the N_d sessions is represented by a queue, as shown in Figure 2. It is assumed that data is sent one window at a time and one ACK is returned per window. Each user constantly transmits data, which means it can be in one of three states: with a window in the queue, transmitting a window, or waiting for acknowledgment. Thus, unless a user is transmitting or waiting for an ACK, it always has a single window waiting in its queue. In this model a user contends only once to transmit w packets, rather than contending w times. (We also modeled the case where each packet in a window contends separately, with one ACK per window. This model resulted in the same average delay to send all w packets (although the delay variance differs), so for analytical simplicity we utilize the model which assumes the window contends only once.) Each queue has an equal probability, on the average, of being the next one served (i.e. that session's window is transmitted), unless it is awaiting an ACK (which means its window was recently transmitted). This assumption follows the theory that the Ethernet is essentially fair over long periods of time, despite the capture effect.[11]

We wish to determine the wait time D_d of a given packet at the MAC layer. Thus, our delay (D_d) represents the delay of the packet in the lower layers of the network, rather than the end-to-end delay which would incorporate bottlenecks in the upper network layers (i.e. application and presentation). In this model, the average number of queues which have data to transmit at any

given time is N_dP_d, where N_d is the number of data sessions on the network and P_d is the probability that a session has data waiting in the queue (i.e. it is not awaiting an ACK). Thus, the average wait time for a packet at the MAC layer for any given data session is simply the transmit time for all packets that arrived before it plus its own transmit time, and is given by

$$D_d = (N_d - 1)P_d t_{xd} + t_{xd} \quad (1)$$

P_d is the ratio of the time a packet is waiting in the queue or being transmitted to the total time it is in the system, and is given by

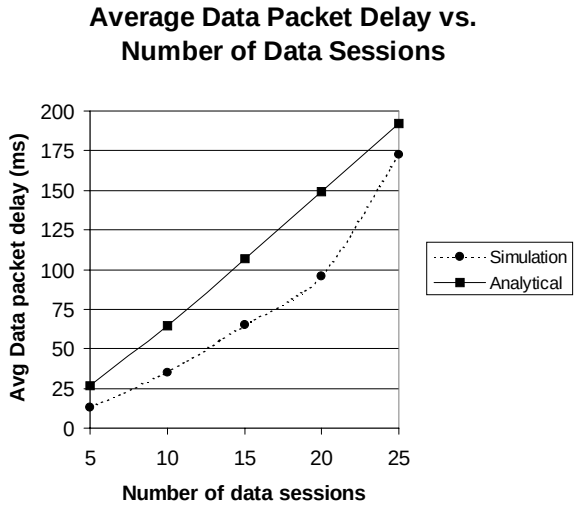
$$P_d = \frac{D_d}{D_d + t_d} \quad (2)$$

Plugging (2) into (1) yields

$$D_d = \tfrac{1}{2}\left(N_d t_{xd} - t_d \pm \sqrt{(N_d t_{xd} - t_d)^2 + 4 t_{xd} t_d} \right) \quad (3)$$

which is plotted in Figure 3 for up to 25 data sessions. The results show that the average packet delay increases linearly with an increasing number of data sessions. This agrees with Boggs *et al* [4], and contradicts the notion that Ethernet transmission delay increases substantially with network load.

Figure 3: Average Data Packet Delay for Data Model



We also simulated the lower network layers using existing TCP and Ethernet simulation models which include CSMA/CD's binary exponential backoff scheme. The simulation model assumes the same number of packets per window and a single acknowledgment for the entire window. However, unlike the analytical model it times out and retransmits the window if no ACK is received within 200 ms of a transmission. As shown in Figure 3, the analytical model is consistent with the simulation model, and is actually an upper bound for the delay. The simulation delays are smaller because when a packet is dropped in the simulation (due to contention backoff) the delay of the retransmission does not include the original packet's delay. In the analytical model, however, packets are not dropped - they simply remain in the queue until transmission. Thus, the delay for a packet will include the entire time it contends (even if it would have been dropped and retransmitted in the simulation), and will be higher on average than the delay in the simulation. Thus, we conclude that our analysis provides a reasonable model of the lower layers of the LAN to determine an upper bound on the data packet delay.

2.2 Voice Model

In this section we model a network in which all users are voice sessions. Each voice session is modeled as a queue which has a packet arrival every 30 ms. We can assume there are always N_vP_v packets waiting to contend, where N_v is the number of voice sessions and P_v is the probability that a voice session has a packet waiting in the queue. The delay of a packet once it starts contending depends on its contention and transmission times. Because each packet contends with $P_v(N_v-1)$ other packets, a single user's service time T_S is

$$T_S = P_v(N_v - 1)t_{xv} + t_{xv} \quad (4)$$

where P_v equals the arrival rate times the service times of all users served before it:

$$P_v = \lambda(N_v - 1)t_{xv} \quad (5)$$

This model is similar to Bolot's [12], in that it assumes an essentially FIFO service order of packets and models delay as a combination of a constant propagation delay plus variable processing and queueing delays. We also modeled a system where each packet's service time is

$$T_S = P_v N_v t_{xv} \quad (6)$$

and

$$P_v = \lambda N_v t_{xv} \quad (7)$$

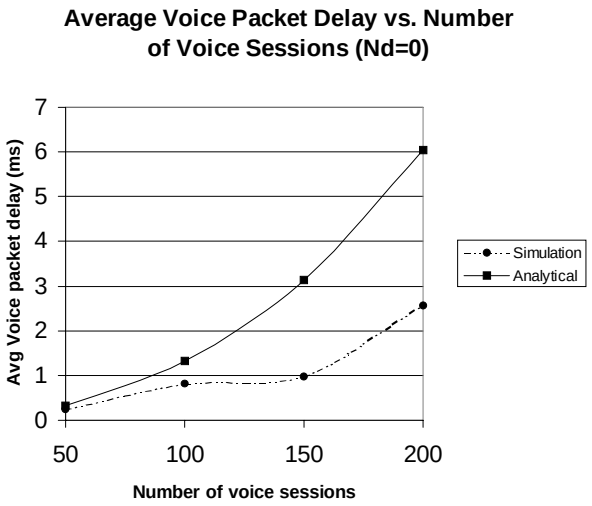
The resulting average delay is virtually identical to the delay when service time is given by (4) and P_v is given by (5), so we use the analytically simpler model given by (6) and (7). In this model, assuming there is a packet at each of the P_vN_v queues, the expected value of the number of times a packet contends before it is serviced (i.e. the number of users which are serviced before it) is a geometric distribution [13]. We approximate the geometric distribution as an exponential distribution in continuous time, and thus assume an exponential service time which includes both contention and transmission time. Thus, although packet arrivals are deterministic, the Markovian M/M/1 model can be used to find an upper bound for the mean delay of voice packets. With arrival rate $\lambda = \frac{1}{30}$ and service rate $\mu = \frac{1}{T_S} = \frac{1}{\lambda(N_v t_{xv})^2}$, the average voice packet delay is given by

$$D_v = \frac{\lambda(N_v t_{xv})^2}{1 - (\lambda N_v t_{xv})^2} \quad (8)$$

The results are shown in Figure 4 for up to 200 voice sessions.

The simulation model sends the voice packets over Ethernet through UDP rather than TCP so that there are no retransmissions or acknowledgments. The simulation results are also shown in Figure 4 and correspond with the analytic results (within 5 ms difference). Using a D/G/1 analytical model would provide more exact results, but is analytically intractable. The results with our M/M/1 model are close enough and provide an upper bound for voice packet delay. Thus, we determine that our analytical model is a reasonable representation of voice over Ethernet.

Figure 4: Average Voice Packet Delay for Voice Model



2.3 Data and Voice Model

After validating each model separately we combine them to represent a LAN with mixed voice and data traffic. Each traffic type has the same characteristics as in the first two cases, except that we simplify the data model by ignoring TCP acknowledgments. Instead we initially assume there is always at least one packet waiting in each of the data queues (i.e. $P_d = 1$). This assumption is reasonable because we also assume that a window consists of multiple packets and each packet contends separately, so that each data session would tend to have at least one packet in its queue. Including acknowledgments would change the model by causing a user to not always be contending, so that $P_d < 1$. This is because the user will sometimes be waiting for an ACK, and thus will not have a packet waiting in the queue for transmission. Although transmission of the ACK would add traffic to the network (which contends with data packets), because ACK packets are small relative to data packets the traffic it adds is less than the traffic from a data packet. We note that because we assume one ACK per window, any user waiting for an ACK does not have a packet contending. Instead its ACK is contending. Thus, because of our assumption that each user always has a packet contending, we can ignore TCP acknowledgments without invalidating our analysis. Actually our analysis results in a higher delay because the data packets are much larger than ACK packets.

We may also obtain a higher delay for another reason. In practice data traffic is bursty, which means it has gaps between bursts of packet transmissions. Our model assumes each user always has packets to transmit, which is equivalent to every user always having a burst of data with no gaps. Thus, in practice the performance of a data network is better than our results, and we obtain an upper bound on average voice packet delay.

2.3.1 Average Voice Packet Delay

We again assume that each voice or data session with a packet waiting in its queue has equal probability of winning contention. There are $N_dP_d+N_vP_v$ sessions contending at any given time, and each has $\frac{1}{N_dP_d+N_vP_v}$ probability of winning contention (i.e. being served). We obtain the delay T for a packet once it starts contending (which includes contention time plus its service time), as the number of contending sessions times the average service time:

$$T = \frac{N_d P_d t_{xd}}{1 - N_v \lambda\, t_{xv}} \tag{9}$$

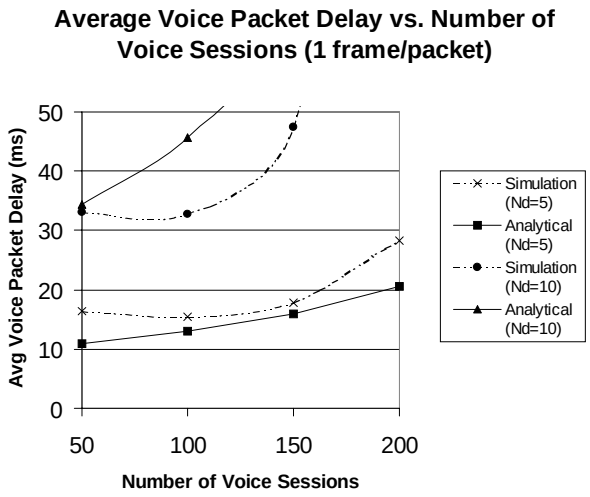
We can represent a single voice user with a Markov D/G/1 model, since the voice arrivals are deterministic. However, by the same argument as in Section 2.2 for the voice model, we can use the M/M/1 model to determine an upper bound on the average voice packet delay for a given voice session. The resulting average voice packet delay is given by

$$D_v = \frac{N_d P_d t_{xd}}{1 - \lambda(N_v t_{xv} + N_d P_d t_{xd})} \tag{10}$$

and is shown in Figure 5, along with simulation results.

As shown in Figure 5, with five data sessions (N_d = 5) the average voice packet delay is less than 30 ms for up to 200 voice sessions. However, with as few as ten data sessions the average voice packet delay increases to higher than 30 ms, which results in an unacceptable delay for voice traffic.

Figure 5: Average Voice Packet Delay for Data & Voice Model



2.3.2 Sensitivity to Number of Sessions

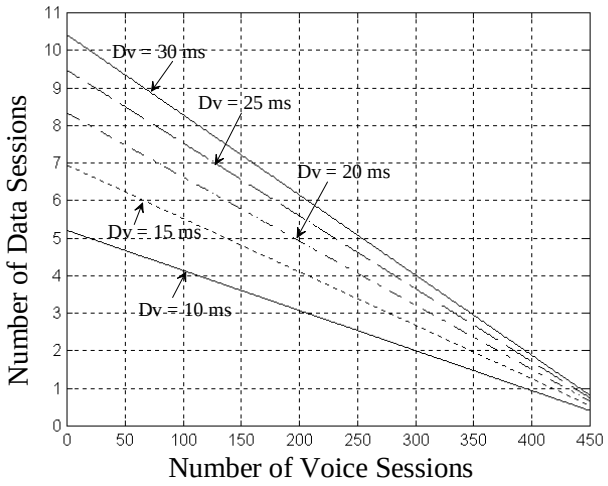
We show the sensitivity of the average voice packet delay to the number of voice and data sessions in Figure 6. From (10) we find the number of data sessions for a given average voice packet delay:

$$N_d = \frac{D_v - D_v N_v \lambda t_{xv}}{P_d t_{xd} + D_v \lambda P_d t_{xd}} \tag{11}$$

Values for average voice packet delay from 10 ms to 30 ms are shown in Figure 6. Average voice packet delays greater than 30 ms are not considered because such delays would cause excessive distortion in the received voice and may cause the sender buffer to overflow.

As shown in Figure 6, the average voice packet delay is much more sensitive to increasing the number of data sessions than increasing voice sessions. This is intuitive because data packet delay is much larger than voice packet delay, and thus adding one data session increases the contention time much more than adding several voice sessions.

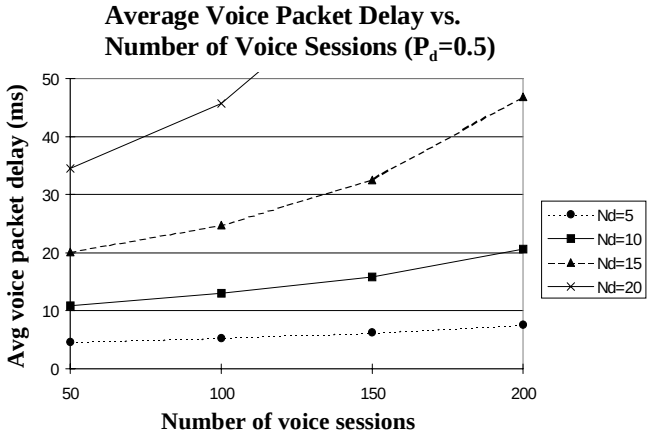
Figure 6: Sensitivity to Number of Voice and Data Sessions



2.3.3 LAN Utilization

The model assumes that every data session always has a packet waiting for transmission (i.e. P_d=1). This corresponds to 100% LAN utilization for the data sessions. As in the data model, the percentage of time in which there is a packet waiting at the MAC (P_d) depends on many factors, including window size, network traffic, and ACK frequency. We can alter this value in the model according to utilization data for a given LAN and thus account for these factors. Figure 7 shows average voice packet delay for 50% LAN utilization (P_d = 0.5). Results are shown for several values of N_d (i.e. number of data sessions). As shown in the graph, decreasing P_d by one half increases the number of data sessions by two for a given number of voice sessions, on the average, without increasing the average delay. (Compare with Figure 5).

Figure 7: Average Voice Packet Delay for 50% LAN Utilization



2.3.4 Multiple Voice Frames per Packet

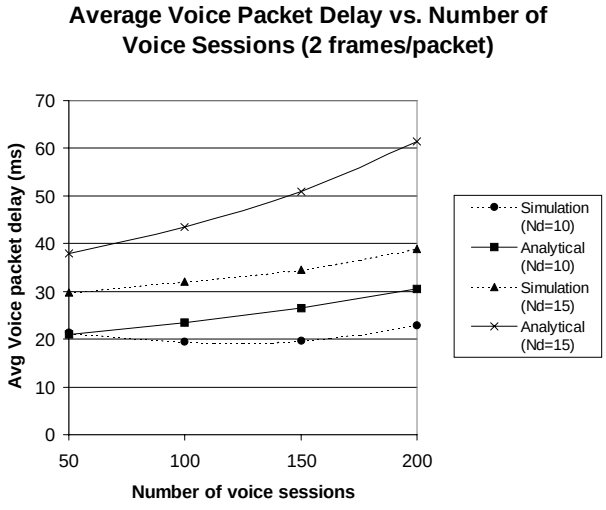
We now study the effects of changing the number of voice frames per packet. The original model assumes each voice packet is a single voice frame, and a voice frame must be received at least once every 30 ms. Because a single voice frame is small, we can combine multiple frames in one packet before transmission over the LAN. If we combine *m* frames in one packet, the voice packet delay can be *m* times longer, since the receiver buffers *m-1* frames and plays one every 30 ms. This also decreases the amount of contention in the network, since the voice packet rate is *m* times smaller than the original rate (i.e. $\lambda = \frac{1}{m \times 30}$). The transmission time is not increased by *m*, since the combined packet size is less than *m* times the original packet size. This is true because the overhead for two frames in two packets is greater than the overhead for two frames in one packet, which has only one header instead of two. Because λt_{xv} decreases, the average voice packet delay also decreases as we see from (10), although the end-to-end delay for a given packet is larger (since the arrival time is *m* times larger).

Figure 8 shows the average voice packet delay vs. number of voice sessions for given values of N_d, when two voice frames are combined in a single packet. The results show that up to 15 data sessions can be combined with 200 voice sessions in this case, while with only one voice frame per packet (see Figure 5) only about 5 data sessions are allowed.

Combining three voice frames per packet further increases the number of data sessions to between 20 and 25 for 200 voice sessions. Thus, we can see that combining multiple voice frames per packet substantially increases the capacity of the LAN, where capacity is measured in number of sessions and is limited by the average voice packet delay.

Simulation results for the combined voice and data model show even lower voice packet delays than the analytical model, as shown in Figure 8. Our model is an upper bound because of the M/M/1 assumption for the voice model and because the analytical data model does not incorporate bursty traffic, TCP ACKs, or retransmissions. (see Sections 2.1, 2.2, and 2.3).

Figure 8: Average Voice Packet Delay for 2 voice frames per packet



3. Conclusions

We have developed a general model for a LAN segment which incorporates both voice and data traffic. By setting appropriate parameters we have determined the number of voice and data sessions which a LAN will support, using average voice packet delay as a limitation. As we have shown, average voice packet delay is extremely sensitive to the number of data sessions. Slightly increasing the number of data sessions substantially decreases the number of supportable voice sessions. This is true because the data packet service time is much larger than the voice packet service time, and thus adding a data session increases the voice packet delay much more than an extra voice session would.

We have also shown that increasing the number of voice frames per packet decreases the average voice packet delay, and thus increases the number of allowable sessions on the LAN. This result is not intuitive because combining several frames in a single packet increases the end-to-end delay for each frame. For example, when we combine two frames per packet the arrival rate λ is halved, but the transfer time t_{xv} does not double because less overhead is needed. Thus, λt_{xv} is much smaller than before and the overall average voice packet delay D_v is smaller. Although the end-to-end delay of a given packet is larger, the maximum allowable delay also increases. This is because the receiver always receives two frames, which means it can wait for 60 ms rather than only 30 ms to receive the next packet, since it buffers the second frame. Thus, the acceptance region for the amount of voice packet delay increases to 60 ms rather than 30 ms.

The model presented in this paper provides a baseline for the study of voice and data over a Local Area Network. Future work will incorporate video traffic, and determine whether proposed network changes such as priority over Ethernet and call admission control schemes would benefit the addition of new traffic types to the LAN . Future work will also evaluate the effects of voice packet delay jitter on session capacity. Because voice traffic is constrained by delay variance, this parameter is important in a discussion of the capacity of multimedia networks.

References

1 J. Lane, "ATM Knits Voice, Data on Any Net," *IEEE Spectrum*, February 1994, pp. 42-45.

2 J. K. Choi, J. U. Seo, and C. K. Un, "Performance Analysis of a Packet-Switched Synchronous Voice/Data Transmission System," *IEEE Transactions on Communications*, vol. 38, no. 9, September 1990, pp. 1419-1429.

3 "MAC Bridges - Traffic Classes and Dynamic Multicast Filtering Services in Bridged Local Area Networks," October 1996 IEEE P802.1p/D4.

4 D. R. Boggs, J. C. Mogul, and C. A. Kent, "Measured Capacity of an Ethernet: Myths and Reality," *Proceedings of the SIGCOMM '88 Symposium on Communications, Architectures, and Protocols*, Stanford, CA, August 1988.

5 R. Gusella, "A Measurement Study of Diskless Workstation Traffic on an Ethernet," *IEEE Transactions on Communications*, vol. 38, no. 9, September 1990, pp. 1557-1568.

6 J. Shoch and J. A. Hupp, "Measured Performance of an Ethernet Local Network," *Communications of the ACM*, vol. 23, no. 12, December 1980, pp. 711-721.

7 W. Leland, M. S. Taqqu, W. Willinger, and D. V. Wilson, "On the Self-Similar Nature of Ethernet Traffic (Extended Version)," *IEEE/ACM Transactions on Networking*, vol. 2, no. 1, February 1994, pp. 1-15.

8 M. L. Molle, "A New Binary Logarithmic Arbitration Method for Ethernet," Technical Report CSRI-298, Computer Systems Research Institute, University of Toronto, 1994.

9 L. P. Slothouber, "A Model of Web Server Performance," StarNine Technologies, Inc., http://louvx.biap.com/webperformance/modelpaper.html.

10 V. Paxson, "End-to-End Internet Packet Dynamics," *Proceedings of the ACM SIGCOMM '97 Conference*, Cannes, France, September 1997, pp. 139-152.

11 R. Seifert, "The Effect of Ethernet Behavior on Networks Using High-Performance Workstations and Servers," Technical Report for Networks and Communications Consulting, March 1995.

12 J. C. Bolot, "Characterizing End-to-End Packet Delay and Loss in the Internet," *Journal of High Speed Networks*, vol. 2, 1993, pp. 305-323.

13 A. Leon-Garcia, *Probability and Random Processes for Electrical Engineering*, Addison-Wesley Publishing Company, Inc., Massachusetts, 1994.