# A New Error Control Scheme for Packetized Voice over High-Speed Local Area Networks

*Bert J. Dempsey* *Jörg Liebeherr* *Alfred C. Weaver*

Computer Science Department,
University of Virginia,
Charlottesville, VA 22903,
U. S. A.

May 24, 1993

**Abstract**

Distribution of packetized digital speech across high-speed LANs has become increasingly feasible, and packet voice protocols supporting audio conferencing are available. These protocols must provide mechanisms to address errors and delay introduced by the network that threaten the quality of the voice playback at the receiving site. A common mechanism for ensuring continuous voice playback in the presence of delay jitter is the use of a control time for the first packet in each talkspurt. Packet loss is generally not recovered in packet voice protocols, though techniques such as forward-error correction (FEC) and priority channels can be used to reduce losses. In this paper we propose a new error control mechanism for packet voice, referred to as Slack ARQ (S-ARQ). S-ARQ is based on extending the control time for the first packet in each talkspurt to allow for the timely retransmission of lost packets. S-ARQ is orthogonal to the use of preventive error control schemes such as FEC or priority channels. It does not require hardware support, imposes little overhead on network resources, and could be easily integrated into extant voice protocols. Experiments using a simulation model developed in this paper indicate that the extended control time required for S-ARQ can be modestly larger than the control time for jitter control and within the limits imposed by end-to-end delay requirements. The feasibility of the S-ARQ scheme is thus established.

*Key Words: Error Control, ARQ, Retransmission, Packet Voice, Continuous Media, Local Area Networks.*

# 1 Introduction

Continuous-media services such as voice and video were traditionally carried over circuit-switched networks. Using packet-switched networks for these services has become increasingly attractive due to technology trends that have enabled high-speed multiservice networks (e.g., FDDI, DQDB, and ATM) and that have increased the viability of distributed multimedia applications on desktop platforms. Since voice and video are inherently variable rate sources [4, 17], the statistical multiplexing of packets in a packet-switched network represents a more efficient use of network resources than circuit switching. Statistical multiplexing, however, causes delay variation, referred to as *delay jitter*, and losses due to buffer contention. In a local area network (LAN) delay jitter is introduced by variable network access delay, operating system scheduling, and protocol processing. Packet losses due to transmission errors are rare in a local area network, but hardware buffers and switches can lose packets during periods of high load and transient periods of overload in the network.

Experiments with the transmission of packetized digital speech across computer networks date to the pre-LAN era, with a network voice protocol (NVP) for the Internet having been specified in 1976 [9]. Packet voice over satellite channels and mobile radio was experimented with in the early 1980's [10, 18, 23]. Standards have recently been developed for wideband packet technologies [7, 8]. With the advent of high-speed LANs and powerful desktop computers with audio hardware support, protocols and conferencing tools have become available for multiple-party connections in a local area environment. Examples include NVP, the vat protocol [12], and the NEVOT audio conferencing tool [21].

The distribution of packetized interactive voice across a network requires consideration of appropriate encoding schemes, end-to-end network delay, delay jitter, and packet loss, all of which significantly affect the speech quality at the receiving site. The selection of a voice encoding scheme represents a trade-off between consumption of bandwidth on the network and the ability to reconstruct natural sounding speech at the receiving side. End-to-end delays have a significant impact on the quality of interactive voice transmissions, as shown in Table 1 [24]. Variation in the network delay experienced by individual packets, i.e., delay jitter, can lead to interruptions of the continuous playback of the voice stream at the receiver (*voice clipping*). Unlike data transmission, packet voice does not need reliable delivery, but its tolerance for packet loss is low. It is estimated that playback loses intelligibility if losses exceed 2 percent [15]. Recently developed schemes for ATM networks

| One-Way Delay | Effect on Speech Quality |
|---|---|
| $> 600$ *ms* | Speech becomes incoherent and unintelligible. |
| 600 *ms* | Speech is barely coherent. |
| 250 *ms* | Conversation style affected by long delays. |
| 100 *ms* | Imperceptible if listener hears only from network, not off the air. |
| 50 *ms* | Imperceptible even if listener is in the same room with speaker and hears off the air and from the network. |

Table 1: Effect of End-to-End Delays on Speech Quality.

raise this figure to around 5 percent [25]. Note however, that even the loss of a single packet may noticeably degrade service quality.

In this paper we address the problem of error recovery from lost packets in packet voice protocols. In contrast to extant approaches to error recovery in packetized voice transmissions, our solution is based on a retransmission scheme. We show that the mechanisms of packet voice protocols that account for delay jitter can be extended to provide a high probability of successful error recovery. Our scheme, referred to as *Slack ARQ* (S-ARQ), attempts to retransmit lost packets that can be recovered in a timely fashion, i.e., before the lost packet is due for playback. We develop a simulation model and show that in a local area environment a significant amount of transparent error recovery through retransmissions is indeed feasible.

The remainder of this paper is structured as follows. In section 2 we review the component mechanisms in extant voice protocols. In section 3 we present our *Slack ARQ* scheme. In section 4 we develop a simulation model of our error recovery scheme for a local area network environment. Our simulation takes into account the delays incurred at the endsystems due to operating systems scheduling and protocol processing. We present several simulation experiments to evaluate the effectiveness of S-ARQ. In section 5 we give conclusions and directions for future work.

# 2 Packet Voice Protocols

Packet voice protocols need to address all issues which may lead to a reduction of speech quality at the receiver, i.e., speech encoding, end-to-end delays, delay jitter and error control. In this section we discuss how existing voice protocols attempt to maintain a high level of speech quality and examine approaches from the literature.

## 2.1 Voice Coding and Packetization

An important protocol parameter of packet voice protocols is the number of digitized voice samples placed in each network packet at the voice source. The sampling period represented by each packet is the *packetization interval*. When a packet contains the specified number of samples, it is submitted to the network. Typical packetization intervals range from $10 - 50$ $ms$ [21]. Since the frame size in broadcast LANs can be large, a network voice packet is typically carried in a single physical frame.

Given a fixed packetization interval, the encoding/decoding scheme determines the actual number of bits per packet. The ubiquitous pulse code modulation (PCM) encoding scheme for voice [6] samples every 125 $\mu s$ with 8 bits per sample to yield a 64 $kbit/s$ channel. Bandwidth reduction can be achieved through the use of one or more of the following: fewer bits per sample, less frequent sampling, suppression of transmissions during silence periods (*digital speech interpolation*), and compression of the digitized data. Adaptive differential pulse code modulation (ADPCM) [7], for example, is a widely accepted coding technique in which only the differences between consecutive samples are encoded, reducing the number of bits per sample to $2 - 5$. Coding techniques with very low bit rates such as Linear Predictive Coding (LPC) exist, though speech fidelity is frequently poor [14]. Bandwidth reduction through compression of encoded data offers the potential for significant bandwidth savings, but is computation intensive at the endsystems unless supported by specialized hardware.

## 2.2 Delay Jitter Reduction Techniques

If the network delay of voice packets is not constant, i.e., packets are subject to delay jitter, the receiver may observe *gaps*, which result in interruptions in the continuous playback of the voice

stream. Delay jitter in packetized voice transmission is commonly addressed through a *control time* at the receiving system. The first packet in a voice stream is artificially delayed at the receiver for the period of the *control time* in order to buffer sufficient packets to provide for continuous playback in the presence of jitter. Note however, that the the control time cannot be arbitrarily large due to constraints on the end-to-end delay (see Table 1). Since voice data consists of an alternating series of *talkspurts* and *silence periods* and since talkspurts are generally isolated from each other by relatively long silence periods [4, 5], voice protocols typically impose the control time on the first packet of each talkspurt.

We refer to the *playback time* of a packet as the point in time at which playback of the packet must begin at the receiver in order to achieve a zero-gap playback schedule for the talkspurt. We refer to the *slack time* of a packet to denote the time difference between its arrival time at the receiver and its playback time. Note that, independent of whether a control time is used or not, the arrival of the first packet in a talkspurt at the receiving side determines the start of the zero-gap schedule and thus the playback time of all packets in the talkspurt. Due to delay jitter, a packet may arrive before or after its playback time. In the former case, the packet is placed in a queue, the so-called *packet voice receiver* (PVR) queue, until it is due for playback. In the latter case, a gap has occurred and the packet is played back immediately. [1]

In Figure 1 we illustrate the occurrence of gaps due to delay jitter and the elimination of gaps through the introduction of a control time. The time sequence shown in Figure 1 illustrates the transmission of a talkspurt consisting of 5 packets. The four timelines pictured represent, from top to bottom, packet generation at the voice source, packet transmissions at the sender, packet arrivals at the receiver, and the playback of voice samples at the receiver. Packet generation is taken to occur periodically, at the end of a fixed-size packetization interval. Packets are indicated by a vertical bar. Note that in a high-speed network the transmission time of a packet is negligible compared to the length of a packetization interval. In Figure 1 the delay incurred by a packet due to protocol processing, scheduling delay, and media access delay is indicated by the arrows from the second timeline (Packetization) to the third timeline (Arrival at the Receiver). We assume that propagation delay in a LAN is negligible. At the bottom of Figure 1 we present two scenarios for

[1] An alternative policy is to play back only that part of a late packet that has arrived within its playback time, thus ensuring that the talkspurt has the same duration at the receiver as it did at the source.

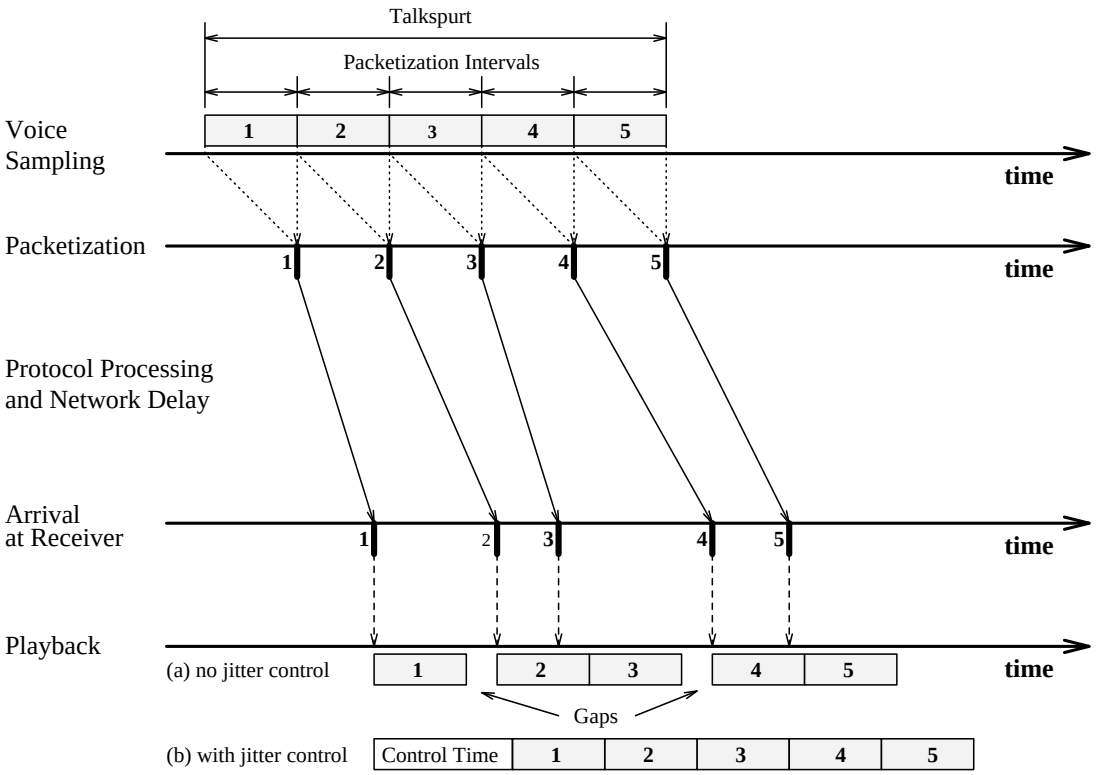


Figure 1: Transmission and Playback of a Talkspurt.

the playback of the voice. Scenario (a) depicts playback without a control time, scenario (b) with a control time. In scenario (a) two gaps are observed, between the first and second packet, and the third and fourth packet. In scenario (b) the presence of a control time delays the first packet of the talkspurt and eliminates gaps. Thus, in this scenario, the playback of the voice samples is continuous.

A packet voice protocol with control times must be able to identify the beginning of talkspurts. This can be done explicitly with a dedicated bit or with a combination of timestamps and sequence numbers. Determination of the control time is more difficult since it requires knowledge of the network delay distribution. Note that to eliminate gaps completely the control time must be set equal to the maximum variation of the network delay. Numerous methods have been proposed for estimating the control time of a talkspurt, based on network delay measurements [19], on stochastic assumptions of the network delay [1, 2], or both [20]. One result of note for our work is that suitable control times were found to be in the range of $2 - 3$ times the mean network delay.

## 2.3 Error Control

Error control for continuous media streams such as voice cannot be equated with that for traditional end-to-end protocols performing nonreal-time tasks such as file transfer. Note that a network service can not provide a service that simultaneously bounds latency and ensures reliable transmission since retransmissions may cause arbitrarily long delays in the progress of data delivery at the receiver. Thus, in contrast to nonreal-time error control which provides 100 percent data completeness, any error control mechanisms for packet voice must be viewed as merely improving service quality by ensuring that, statistically, fewer voice samples will be lost or arrive late at the receiver than would be the case without these mechanisms.

Error control for dropped or corrupted packets is, in all packet voice protocols known to the authors, absent. Recall that data retransmissions found in conventional reliable end-to-end protocols are unacceptable since they are delay-insensitive. Retransmission of time-constrained data such as voice packets risks being counterproductive since the retransmitted data may arrive at the receiver too late to be useful.

Since the real-time constraints of packet voice transmissions do not allow recovery from lost packets in all cases, approaches have been suggested to reduce the likelihood of losing packets. Forward-error correction (FEC) allows limited error recovery without retransmission. By sending redundant information with the original information, lost data can be reconstructed using the redundant information. However, FEC results in an increase of required network bandwidth. In [3] it was shown that the negative effects of added congestion on a network due to FEC overhead can more than offset the benefits of FEC recovery. Also, FEC requires hardware support at end-systems in order to encode and decode the redundant information with sufficient speed. Finally, FEC does not guarantee that corrupted or lost packets can always be recovered.

Channel coding refers to a class of approaches that separate the voice signal into multiple data streams with different priorities [11, 26, 27]. These priorities are then used to tag voice packets so that during periods of congestion the network is more likely to discard low priority packets which carry information that is less crucial in reconstructing the original signal. By controlling the cells that the network discards, these schemes enable multiple-priority channels to maintain a higher quality of service over larger loss ranges than channels using a single priority for all voice packets. Channel coding requires that the network be able to control packet loss during congestion through a

priority mechanism, and the use of different streams for different priorities requires synchronization at a per-packet granularity in order to reconstruct the voice signal.

Forward error correction and channel coding schemes have limitations both in errors that can be corrected and in the network architectures for which they are applicable. The investigation of different and/or orthogonal error control mechanisms for voice transmissions is therefore an ongoing research topic. In the next section we propose an error control scheme which is based on retransmission but takes into the account the necessity of timely delivery of retransmitted voice packets.

# 3 A Delay-Sensitive Retransmission Scheme for Packet Voice

In this section we introduce a novel approach to error recovery for packetized voice protocols, referred to as *Slack Automatic Repeat Request* (S-ARQ). S-ARQ does not preclude the use of preventive error control schemes such as FEC or channel coding and may be used in conjunction with preventive error control. The new scheme does not require hardware support, imposes little overhead on network resources, and could be easily integrated into extant voice protocols.

S-ARQ is an error control scheme based on the retransmission of lost packets. Recall that due to the need for continuous playback of voice packets, retransmitted packets must arrive before they are due for playback. The principle behind S-ARQ is to extend the control time at the beginning of a talkspurt and use the extended control time at the PVR such that the slack time (as defined in section 2.2) of arriving packets is lengthened. With this simple mechanism timely retransmissions of lost packets can be achieved with a high probability.

In S-ARQ, whenever a lost packet is detected, the receiver requests a retransmission of the missing packet. The packet voice receiver assumes that a packet is lost if it receives a packet out of sequence. Note that the packet voice receiver will wrongly assume that a packet is lost if packets are misordered in the network. However, the misordering of packets very rarely occurs in broadcast LANs. If the retransmission is attempted but it is lost or late, the packet voice receiver does not hold back subsequent correctly received packets nor does it attempt any additional retransmissions of the lost data. Therefore, S-ARQ does not guarantee that lost packets are successfully recovered. The percentage of retransmissions that are completed successfully is largely dependent on the appropriate choice of the control time. Note that the likelihood of successful retransmissions

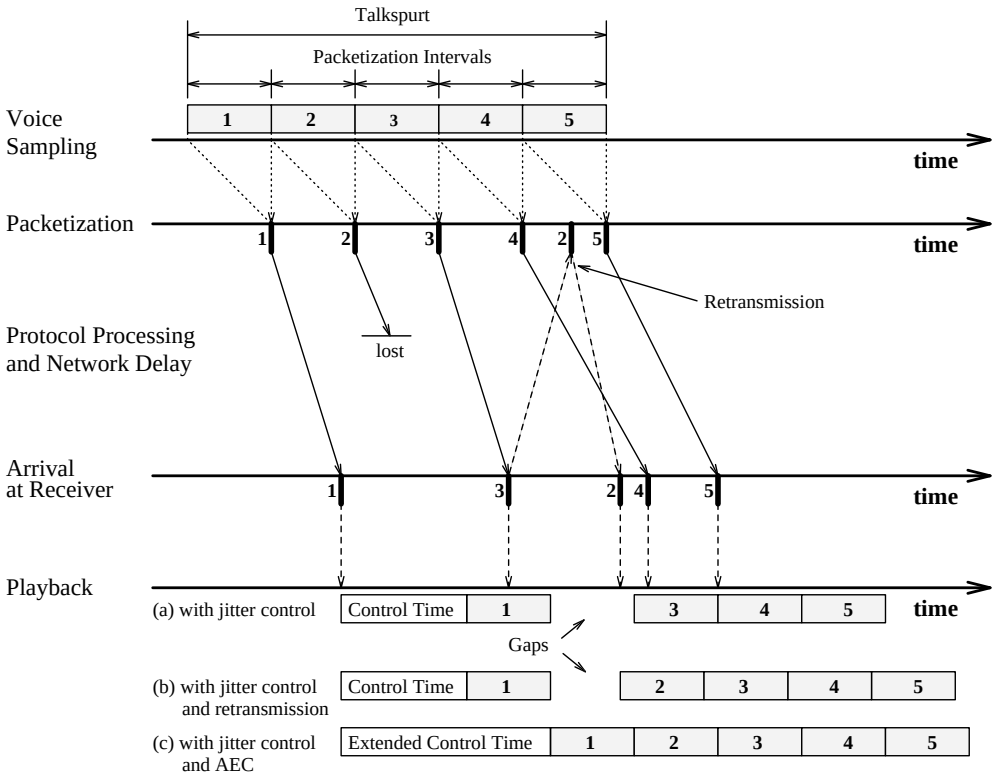


Figure 2: Transmission of a Talkspurt in the Presence of Errors.

decreases if consecutive packets are lost.

We illustrate the advantages of S-ARQ in Figure 2. In Figure 2 we show the same transmission scenario as in Figure 1, i.e., the transmission of a talkspurt consisting of five packets. Here however, we assume that the second packet of the talkspurt is lost. At the bottom of Figure 2 we show three scenarios. For all scenarios we assume the existence of jitter control with an appropriately selected control time. Scenario (a) shows error handling typically found in extant voice protocols, i.e., no retransmissions are attempted. If a packet is lost, playback of the lost packet is skipped. As shown in Figure 2 skipping the playback of a lost packet results in a gap equal to the time of packetization interval. In scenarios (b) and (c), the receiver requests a retransmission of the lost packet. In both scenarios the lost packet is detected upon arrival of the third packet to the receiver. Scenario (b) shows the drawback of a retransmission scheme without S-ARQ. If a packet is lost, playback of all packets is discontinued until the retransmission has been completed. Since the retransmission of the second packet is not completed before its playback time, a gap is observed. In scenario (c)

we assume an S-ARQ scheme. Due to the extended control time, the retransmission of the second packet is completed before its playback time.

S-ARQ depends on appropriately extending the control time of the PVR queue in order to increase the slack time of arriving packets. Packet voice protocols provide control times to account for delay jitter, but these control times do not consider the roundtrip time of the network, which is central to packet retransmissions. In the calculation of control times for jitter purposes, packet voice protocols must estimate the delay variation of packets traversing the network, either through performing measurement of this the variation or through the use of network-specific constants. The additional amount of control time required for S-ARQ can be calculated using these same mechanisms to derive estimates for the network roundtrip delay. Hence, the implementation of slack ARQ in an extant or future packet voice protocol will add little overhead. Note that in the common case where no packets in the talkspurt are lost, the extended control time for S-ARQ provides an additional measure of protection from gaps in the playback due to delay jitter.

Note that S-ARQ can be improved in the sense that retransmissions are suppressed when the probability of timely retransmission is low. In this case, before S-ARQ requests retransmission of a lost packet it would examine the length of the PVR queue and compare it against the measured or estimated retransmission time.

A possible drawback of S-ARQ is that it increases the end-to-end delay of all voice packets. Recall that the end-to-end delay in interactive voice transmissions significantly affects the speech quality. Therefore, before any refinement to the S-ARQ schemes are given, e.g., for calculating the extended control time and thresholds for retransmission suppression, one needs to show that the increase in end-to-end delay due to S-ARQ does not interfere with requirements for speech quality. In the next section we will show that in a LAN environment, S-ARQ can successfully recover from almost all single packet losses in a talkspurt with only modest extensions to the control time used for jitter control.

# 4 Evaluation of Slack ARQ

In order to evaluate the effectiveness of our S-ARQ scheme we present a simulation model of digital speech distribution across a local area network. We conduct experiments with the simulation model and provide answers to the following questions:

- *How much control time is needed for S-ARQ to ensure a high probability of successful retransmissions?*

  Note that the control time at the PVR results in increased end-to-end delays for all packets. However, since voice transmission is sensitive to end-to-end delay (see Table 1) the control time cannot be increased arbitrarily.

- *How does the control time required for jitter control compare to the control time required for S-ARQ ?*

  We expect that our S-ARQ scheme will not be acceptable for extant packet voice protocols if the control time needed for S-ARQ is not in the same order of magnitude as the control time used for jitter control.

- *How does S-ARQ perform if consecutive packets are lost?*

  Due to the simplicity of the error detection scheme in S-ARQ we expect the effectiveness of S-ARQ to degrade if consecutive packets are lost. However, S-ARQ should be able to recover at least partially from consecutive packet losses.

- *How sensitive is S-ARQ to particular network delay distributions ?*

  Although we expect the effectiveness of S-ARQ to vary for different network delay distributions, S-ARQ should be applicable for a wide range of network delay distributions.

In the following subsection we discuss the parameters of our simulation model. Then we present a set of experiments to evaluate the S-ARQ scheme for different sets of system parameters.

### 4.1 Simulation Model

For our simulation model we chose a local network environment with multi-user workstations as endsystems and an FDDI network as the communication network. Since we are primarily interested in studying the behavior of the queue at the packet voice receiver queue, we model a single unidirectional voice channel. The voice traffic stream is modeled as alternating talkspurts and silence periods whose lengths are exponentially distributed with means 350 $ms$ and 650 $ms$, respectively. These figures represent a talk activity of 35 percent as suggested in [5]. The packetization interval, $T_p$, determines the duration of speech (in milliseconds) captured by each network packet. The number of packets in a talkspurt will be the length of the talkspurt in milliseconds divided by $T_p$.

The one-way delay of the network, $T_{Net}$, consists of three components, protocol processing at the sender, $T^S_{Prot}$, network access delay, $T_{Acc}$, and protocol processing at the receiver, $T^R_{Prot}$ :

$$T_{Net} = T^S_{Prot} + T_{Acc} + T^R_{Prot} \tag{1}$$

For simplicity, processing characteristics of sender and receiver are assumed to be equivalent. The processing delays $T^S_{Prot}$ and $T^R_{Prot}$ are assumed to consist of a fixed component $C_{const}$ and a variable component $C_{var}$ as given here:

$$T^S_{Prot} = T^R_{Prot} = C_{const} + C_{var} \tag{2}$$

The fixed component $C_{const}$ represents the minimum time required for processing a packet at the endsystems. For representing the variable component of the processing delay we choose a truncated Cox distribution $\Phi(x_1, x_2, x_3)$ with mean value $x_1$, coefficient of variation $x_2$, and a maximum value of $x_3$. Selecting values from a Cox distribution for the variable delay allows to consider different degrees of variances of the distribution. For example, selecting $x_2 = 1$ as the coefficient of variation will yield an exponential distribution, values $x_2 < 1$ will result in a so-called Erlang distribution with low variations, and $x_2 > 1$ will result in a highly variable hyperexponential distributions. Since system measurements show that values for processing delays do not grow arbitrarily large we select a truncated distribution. The minimum processing delays of packets at endsystems is assumed to be low, i.e., we set $C_{const} = 1$. However, due to multitasking at the endsystem the packet processing delay is highly variable. Therefore, we select the variable component of processing delays $T^S_{Prot}$ and $T^R_{Prot}$ from a truncated Cox distribution, specifically $\Phi(1, 6, 10)$.

We assume that the FDDI network is heavily loaded. Network measurements show that non-negligible packet error rates in FDDI networks are observed only under conditions of high network load. We assume that all voice data is transmitted as synchronous traffic. With the recommended default value for the Target Token Rotation Time (TTRT) of 8 *ms* [13] an upper bound of the access delay is given by $2 \cdot TTRT$ [16]. Assuming that $T_{Acc}$ is exponentially distributed, we obtain a truncated Cox distribution $\Phi(8, 1, 16)$ for the network access delay.

The key parameters of a packet voice protocol are the control time at the PVR queue, $T_V$, and the packetization interval, $T_p$. Both $T_V$ and $T_p$ are constrained by the end-to-end delay requirements of the channel. To achieve a voice channel with high speech quality, we do not consider values of $T_V > 150$ *ms* and $T_p > 150$ *ms*.

| Parameter | Description | Min. | Max. | Avg. |
|---|---|---|---|---|
| $T_{Net}$ | One-way network delay | 2 | 36 | 12 |
| $T^S_{Prot}$ | Processing delay at sender | 1 | 10 | 2 |
| $T^R_{Prot}$ | Processing delay at receiver | 1 | 10 | 2 |
| $T_{Acc}$ | Network access delay | 0 | 16 | 8 |
| $T_V$ | Control time | - | 150 | - |
| $T_p$ | Packetization interval | - | 150 | - |

Table 2: Default Parameters of the Simulation Model (in milliseconds).

The error model of our simulation model arbitrarily generates a period of time in the talkspurt where packets are dropped, the so-called *error period*. The start of the error period is uniformly distributed among the duration of the talkspurt. There is at most one error period in a talkspurt. However, in one error period multiple consecutive packets may be lost. An error period in which one packet in a talkspurt is lost is called a *1-error*, an error period in which two consecutive packet are lost is called a *2-error*, and so on. In the case of a loss of multiple consecutive packets, the receiver sends a single request for the retransmission of lost packets. This is a realistic assumption since FDDI frames are large relative to the amount of data in a voice packet. In Table 4.1 we summarize the default parameters of the simulation model.

## 4.2 Experiments

The simulation model was developed using the SES/Workbench simulation package [22]. All simulation experiments were run on a Sun Sparc2 workstation. Each data point represents the observation of at least 6000 talkspurts.

We present three sets of experiments. In the first two experiments, we investigate the degree to which packetization interval and control time affect the ability of S-ARQ to recover lost packets. In the third experiment we examine the sensitivity of our S-ARQ scheme towards variations in the network delay behavior. Our performance measure is the probability $Prob[no\ gap]$, the probability that no gaps are observed during the playback of a talkspurt. If the talkspurt contains packet losses then $Prob[no\ gap]$ is equivalent to the probability of successful retransmissions. We show the

values of $Prob[no\ gap]$ for single and multiple packet losses. For reference, we always include the simulation results of $Prob[no\ gap]$ for error-free voice transmissions.

### 4.2.1 Experiment 1

Figures 3, 4 and 5 show the effect of increasing the control time both on jitter control and on the probability of successful retransmissions using the S-ARQ scheme. The packetization interval is assumed to be constant and is given by $T_p = 10\ ms$ in Figure 3, $T_p = 25\ ms$ in Figure 4, and $T_p = 50\ ms$ in Figure 5.

In each Figure, the curve labeled *no errors* depicts the probability that in an error-free transmission a packet arrives at the receiver before its playback time. The other curves show the probability that S-ARQ will successfully recover from the loss of $n$ consecutive packets, where $1 \leq n \leq 5$. Thus, the curve labeled *1-error* in Figure 3 represents the probability of recovery from single packet losses.

Note that without a control time, i.e., $T_V = 0$, the probability that no gaps occur ranges between $75 - 91$ percent for all simulation runs in this experiment. A control time of $T_V = 15\ ms$ is needed to reduce the probability of a gap in the playback schedule to less than 1 percent. Short control times of $T_V \leq 15\ ms$ result in a low probability of error recovery, only 20 to 30 percent for the *1-error* case.

The extended control time necessary for S-ARQ to be effective can be directly obtained from Figures 3,4, and 5. In Figure 4, for example, complete coverage of single error losses is achieved with a control time of $T_V \geq 55\ ms$. The following argument explains the observed behavior of S-ARQ. Define the *virtual slack* of a lost packet as the slack time of that packet if it had not been lost, but had instead arrived at the receiver. When a single packet is lost, the slack time at the receiver of the packet following the lost one is approximately the virtual slack of the lost packet minus one packetization interval, $T_p$. In order for a retransmission to occur before the playback time of the lost packet, the slack time of the out-of-sequence packet must be greater than a roundtrip time in the network, i.e., $2T_{Net}$. Thus, the virtual slack of the lost packet should have a value of $T_p + 2T_{Net}$. For example, in the simulations shown in Figure 4, $T_p = 25\ ms$ while the maximum value of the network roundtrip time is given by $2T_{Net} = 72\ ms$.

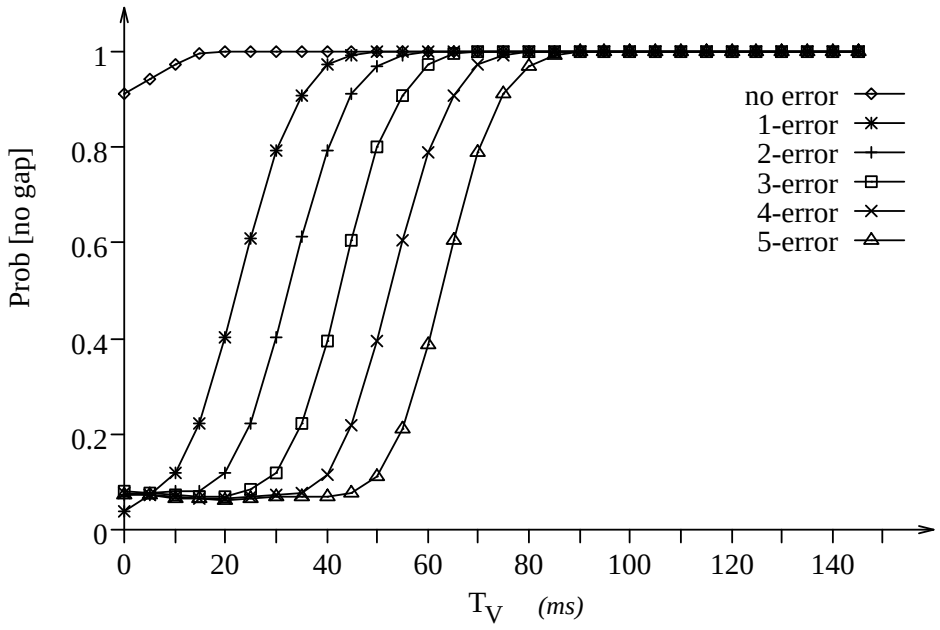


Figure 3: Probability of Continuous Playback ($T_p = 10\ ms$).

Therefore, we are guaranteed that a control time of $T_V = 97\ ms$ allows to successfully retransmit all single packet losses. Yet, in Figure 4 we observe that a control time of $T_V = 60\ ms$ allows us to almost certainly recover from single packet losses, i.e., $Prob[no\ gap] \approx 1$ if $T_V \geq 60\ ms$ in the *1-error* case. An extension of the argument above to the case of consecutive packet losses shows that each additional consecutive packet lost in an error burst adds approximately the duration of one packetization interval $T_p$ to the control time $T_V$ required for successful recovery of the entire lost burst. This explains why the retransmission curves in Figures 3, 4, and 5 are parallel with a distance of approximately $T_p\ ms$.

Summarizing, we note that the additional control time needed by S-ARQ to recover from single packet losses is low for small values of the packetization delay $T_p$ (Figures 3 and 4), but may be unacceptable for long packetization delays (Figure 5).

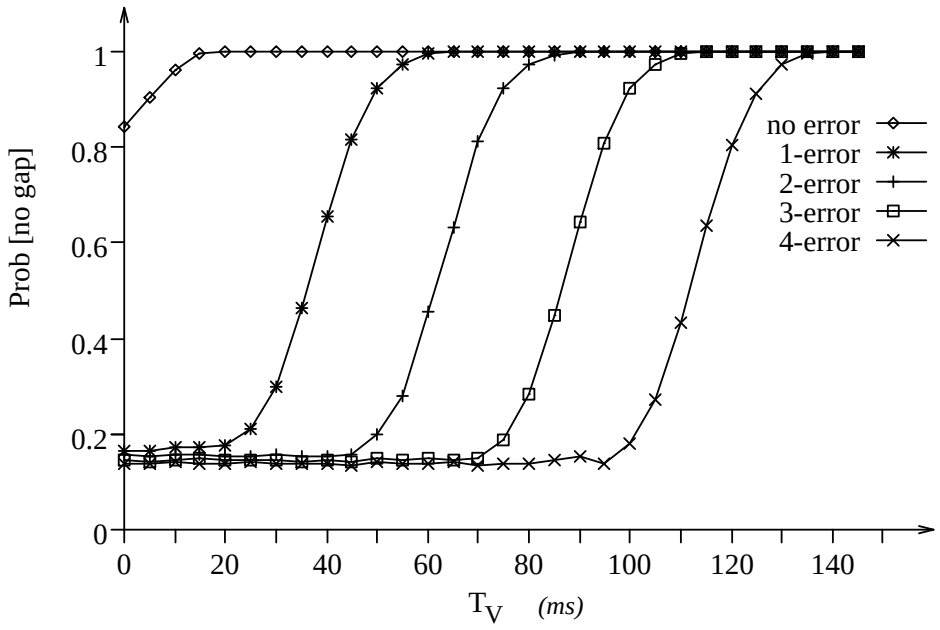


Figure 4: Probability of Continuous Playback ($T_p = 25\ ms$).

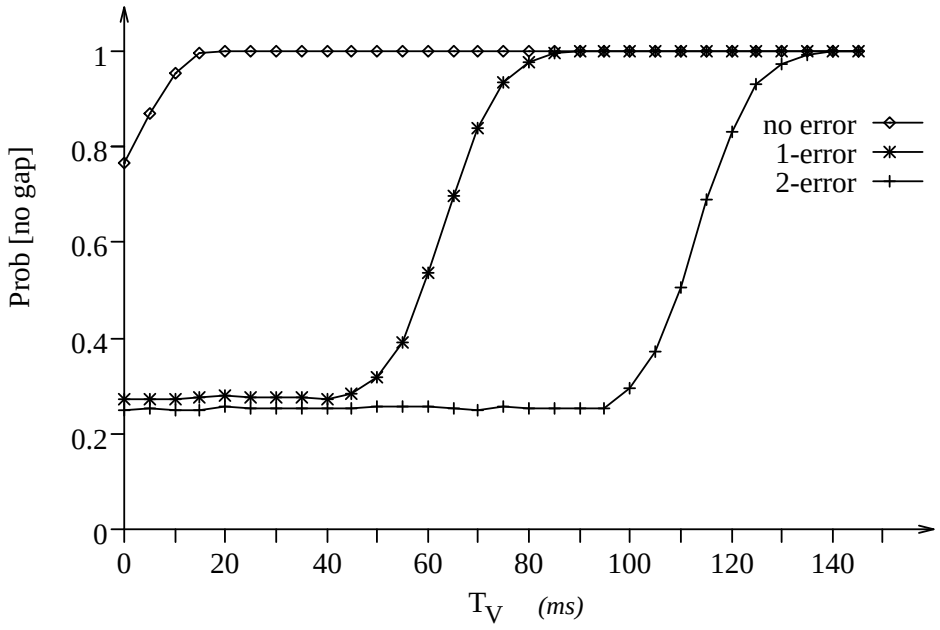


Figure 5: Probability of Continuous Playback ($T_p = 50\ ms$).

### 4.2.2 Experiment 2

In this experiment we investigate the effects of different packetization intervals $T_p$ on S-ARQ. We assume a fixed control time at the PVR queue, $T_V = 50\ ms$. Other simulation parameters are as given in Table 4.1. The results of this experiment are summarized in Figure 6. Recall from the previous experiment that since a control time of $T_V = 50\ ms$ is higher than required for jitter control in an error-free environment, the *no error* scenario always yields $Prob[no\ gap] = 1$.

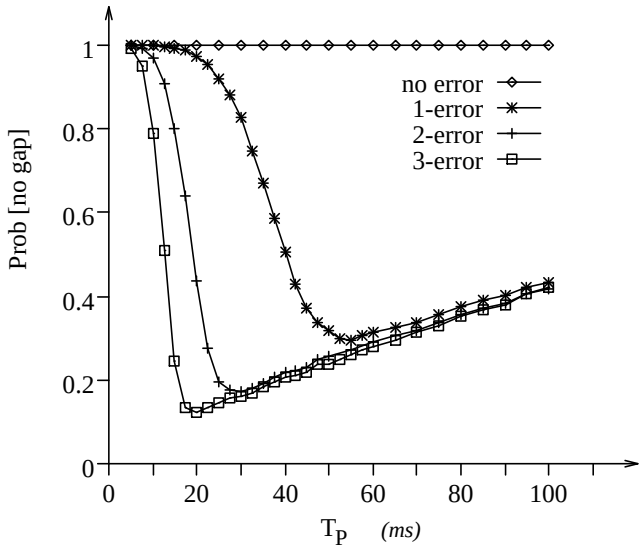


Figure 6: Probability of Continuous Playback ($T_V = 50\ ms$).

In Figure 6 we note the non-monotonic behavior of curves for successful error recoveries using S-ARQ. Two opposing phenomena are influencing the behavior of the curves. On the one hand, as $T_p$ grows, the probability of successful retransmissions is lessened. For example, in the *1-error* curve in Figure 6 this effect can be observed for $T_p \leq 20$. However, as $T_p$ is increased, the number of packets in a talkspurt decreases. But fewer packets per talkspurt increase the likelihood that the lost packet is the first packet in the talkspurt. Note that the first packet in a talkspurt can always be recovered since the playback schedule can still be changed until the first packet is played back. Since network roundtrip times in a local area network are small relative to end-to-end delay constraints for speech quality, retransmission of the first packet is always preferred and always successful.

| Distribution | $T_{Acc}$ | $T^S_{Prot}, T^R_{Prot}$ | |
|---|---|---|---|
| | | $C_{const}$ | $C_{var}$ |
| Exponential | $\Phi(8, 1.00, 16)$ | 1.0 | $\Phi(8, 1.00, 24)$ |
| Erlangian | $\Phi(8, 0.25, 16)$ | 1.0 | $\Phi(8, 0.25, 24)$ |
| Hyperexponential | $\Phi(8, 1.50, 16)$ | 1.0 | $\Phi(8, 3.00, 32)$ |

Table 3: Delay Parameters in Experiment 3 (in milliseconds).

#### 4.2.3 Experiment 3

In this experiment we investigate the effects of different distributions for the network delay $T_{Net}$ on the control times required for jitter and error control. The packetization time is fixed at $T_p = 25$ *ms*. Three different network delay scenarios are investigated, and the parameters for $T_{Acc}$ and $T_{Prot}$ in these three scenarios are given in Table 3. Recall that $\Phi(x_1, x_2, x_3)$ denotes the truncated Cox distribution with mean $x_1$, coefficient of variation $x_2$, and maximum $x_3$.

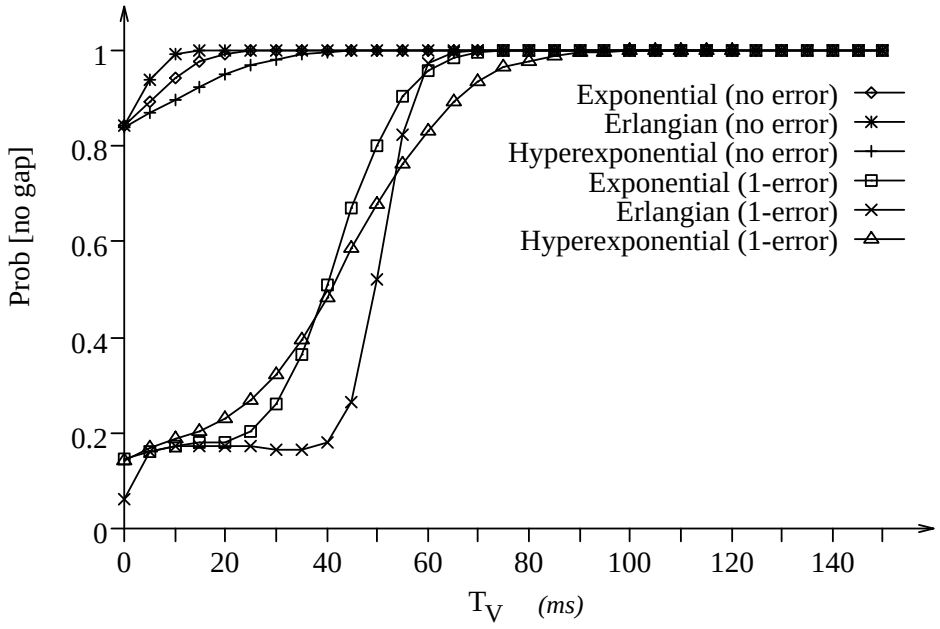


Figure 7: Probability of Continuous Playback for Different Distributions for $T_{Net}$.

The curves in Figure 7 represent the *no error* and *1-error* curves for the three $T_{Net}$ delay

distributions. The *no error* curves in Figure 7 show that higher variation in $T_{Net}$ creates the need for a longer control time to handle jitter. The hyperexponential scenario, for example, requires 35 *ms* for jitter control while the erlangian scenario requires only 15 *ms*. The *1-error* curves show differences as well in the control times required for full coverage of single packet losses.

As the control time increases, the convergence toward full coverage is quite dependent on the delay distribution of $T_{Net}$. The Erlangian scenario exhibits a convergence behavior approximating a discrete jump as it approaches the control time that allows for successful recovery of a single packet. Thus, for a control time of 40 *ms*, error recovery would be very poor while with 60 *ms*, almost all lost packets are recovered. The high variation represented by the hyperexponential scenario allows for better recovery at lower control times than the other scenarios, but its convergence to full coverage takes longer.

While the distributions here represent a wide range of variation in the behavior of $T_{Net}$, note that the overall effects are small. Single packet error recovery can be covered for all the distributions with a conservative control time $T_V = 85$ *ms* and 80 percent coverage is provided with a control time of $T_V = 55$ *ms*.

# 5 Conclusion and Future Work

A new unified approach towards the delay and error constraints that influence the quality of a digitized voice over asynchronous local computer networks has been proposed. This approach, called *Slack ARQ* (S-ARQ), proposes an extension of the control time mechanism for jitter control in order to provide error recovery through retransmissions in the case of packet losses in the network. While existing packet voice protocols do not employ retransmission schemes, our simulation studies indicate that the extended control time can provide significant error coverage while remaining within the same order of magnitude as the control time require for jitter control. Simulation experiments revealed that our concept is clearly feasible within the end-to-end delay constraints in packet voice transmissions.

To the authors knowledge, the Slack ARQ scheme represents the first appearance in the literature of a delay-sensitive retransmission mechanism for packet voice error recovery. The scheme is orthogonal to preventive error control mechanisms such as FEC and channel coding.

Having established the feasibility of the Slack ARQ scheme, we intend to examine implementa-

tion mechanisms for Slack ARQ and their trade-offs in future work. Two mechanisms are crucial for exploiting the full potential of the Slack ARQ scheme. First, we will provide methods for calculating appropriate control times for S-ARQ. Secondly, we will develop a methodology for estimating the probability of a successful retransmission. This probability can be used to suppress retransmission requests. In order to understand the behavior of the packet voice receiver queue more fully, we plan to develop an analytical model using an embedded Markov chain approach. Finally, we intend to implement a Slack ARQ scheme in an existing voice protocol or audio conferencing tool.

## References


[1] G. Barberis. Buffer Sizing of a Packet-Voice Receiver. *IEEE Transactions on Communications*, COM-29(2):152–156, February 1981.

[2] G. Barberis and D. Pazzaglia. Analysis and Design of a Packet-Voice Receiver. *IEEE Transactions on Communications*, COM-28(2):217–227, February 1980.

[3] E. Biersack. Performance Evaluation of Forward Error Correction in ATM Networks. *Sigcomm '92*, pages 248–258, August 1992.

[4] P. T. Brady. A Technique for Investigating On-Off Patterns of Speech. *Bell System Technical Journal*, 44:1–22, 1964.

[5] P. T. Brady. Effects of Transmission Delay on Conversational Behavior on Echo-Free Telephone Circuits. *Bell System Technical Journal*, pages 115–134, January 1971.

[6] CCITT. Recommendation G.711 - Pulse Code Modulation of Frequencies, October 1984.

[7] CCITT. Recommendation G.727 - 5-, 4-, 3-, 2-Bits/Sample Embedded ADPCM, July 1990.

[8] CCITT. Recommendation G.764 - Voice Packetization: Packetized Voice Protocol, July 1990.

[9] D. Cohen. Specification for the Network Voice Protocol (NVP). Technical Report RFC 741, Information Sciences Institute, Los Angeles, CA, January 1976.

[10] G. Falk et.al. Integration of Voice and Data in the Wideband Packet Satellite Network. *IEEE Journal on Selected Areas in Communications*, SAC-1(6):1076–1083, December 1983.

[11] M.W. Garrett and M. Vetterli. Joint Source/Channel Coding of Statistically Multiplexed Real-Time Services on Packet Networks. *IEEE/ACM Transactions on Networking*, 1(1):71–81, February 1993.

[12] V. Jacobson. VAT Protocol Specification. Lawrence Berkeley Laboratory, University of California, Berkeley.

[13] R. Jain. Performance Analysis of FDDI Token Ring Networks: Effect of Parameters and Guidelines for Setting TTRT. *IEEE Lightwave Telecommunications Systems*, pages 16–22, May 1991.

[14] N.S. Jayant. High-Quality Coding of Telephone Speech and Wideband Audio. *IEEE Communications Magazine*, 28(1):10–20, January 1990.

[15] N.S. Jayant and S.W. Christensen. Effects of Packet Losses in Waveform Coded Speech and Improvements Due to Odd-Even Sample-Interpolation Procedure. *IEEE Transactions on Communications*, COM-29:101–109, February 1981.

[16] M.J. Johnson. Proof that Timing Requirements of the FDDI Token Ring Protocol Are Satisfied. *IEEE Transactions on Communications*, COM-35:620–625, June 1987.

[17] G. Karlsson and M. Vetterli. Packet Video and Its Integration into the Network Architecture. *IEEE Journal on Selected Areas in Communications*, 7(3):739–751, June 1989.

[18] S.A. Mahmoud, W.-Y. Chan, J.S. Riordon, and S.E. Aidarous. An Integrated Voice/Data System for VHF/UHF Mobile Radio. *IEEE Journal on Selected Areas in Communications*, SAC-1(6):1098–1111, December 1983.

[19] W.A. Montgomery. Techniques for Packet Voice Synchronization. *IEEE Journal on Selected Areas in Communications*, SAC-1(6), December 1983.

[20] W.E. Naylor and L. Kleinrock. Stream Traffic Communication in Packet-Switched Networks: Destination Buffering Considerations. *IEEE Transactions on Communications*, COM-30(12):2527–2534, December 1982.

[21] H. Schulzrinne. Voice Communication Across the Internet: A Network Voice Terminal. Technical report, University of Massachusetts, July 1992.

[22] SES. SES Workbench Release 2.1, February 1992. Scientific and Engineering Software.

[23] N. Shacham, E.J. Craighill, and A.A. Poggio. Speech Transport in Packet-Radio Networks with Mobile Nodes. *IEEE Journal on Selected Areas in Communications*, SAC-1(6):1084–1097, December 1983.

[24] A. Shah, D. Staddon, I. Rubin, and A. Ratkovic. Multimedia over FDDI. In *17th IEEE Conference on Local Computer Networks*, pages 110–124, September 1992.

[25] K. Sriram, R. McKinney, and M. Sherif. Voice Packetization and Compression in Broadband ATM Networks. *IEEE Journal on Selected Areas in Communications*, 9(3):703–712, April 1991.

[26] K. Sriram and W. Whitt. Traffic Smoothing Effects of Bit Dropping in a Packet Multiplexer. *IEEE Transactions on Communications*, 37(7):703–712, July 1989.

[27] N. Yin. Congestion Control for Packet Voice by Selective Packet Discarding. *IEEE Transactions on Communications*, 38(5):674–683, May 1990.