Adaptive joint playout buffer and FEC adjustement for Internet Telephony

Catherine Boutremans, Jean-Yves Le Boudec
LCA - I&C
Swiss Federal Institute of Technology at Lausanne (EPFL)
CH-1015 Lausanne, Switzerland
{Catherine.Boutremans,Jean-Yves.LeBoudec}@epfl.ch

Abstract

We develop a joint playout buffer and Forward Error Correction (FEC) adjustment scheme for Internet Telephony that incorporates the impact of end-to-end delay on the perceived audio quality. We show that it provides a better quality than the adjustment schemes for playout buffer and FEC that were previously published. This is important because of a threshold effect when the end-to-end delay of interactive audio is around 150 ms. We represent the perceived audio quality as a function of both the end-to-end delay and the distortion of the voice signal. We develop a joint rate/error/playout delay control algorithm which optimizes this measure of quality and is TCP-Friendly. It uses a channel model for both loss and delay. We validate our approach by simulation and show that (1) our scheme allows a source to increase its utility by avoiding increasing the playout delay when it is not really necessary, (2) it performs better than direct combinations of existing algorithms in the cases where end-to-end delay is important and (3) adaptive delay aware FEC adjustment brings significant improvements only if it is coupled with an adaptive playout adjustment.

1 Introduction

Transport of real-time, interactive audio over IP networks often suffers from packet loss and delay variation, which called for two types of corrective actions. First, Forward Error Correction (FEC) is used [1] to mitigate the impact of packet losses; it also increases the end-to-end delay, since the destination has to wait for the redundant packet(s) to be received in order to repair packet losses. Moreover, FEC increases the bit rate requirement of an audio source. Second, adaptive playout buffer algorithms at the receiver compensate for jitter, also at the expense of maybe some additional end-to-end delay.

Bolot et al [2] proposed an adaptive rate/error control which optimizes a subjective measure of quality and incorporates a rate control. Their algorithm supposes that the destination plays the best received copy of a given packet. They neither consider losses due to playout buffer overflow nor try to optimize the overall end-to-end delay. In particular, they do not manage the additional delay due to FEC. However, it is recognized that the end-to-end delay has an impact on the perceived quality of interactive conversations, with a threshold effect around 150 ms [3] for strongly interactive conversations, whereas many voice coders can tolerate some small loss without severe penalty. As a result, the FEC scheme in [2] may in some cases increase the delay when it would be more important to keep delay low while accepting some small loss. An adaptive *delay aware* error control was proposed in [4] to overcome this problem. The delay aware algorithm in [4] is based on the assumptions that (1) the destination plays the best copy received and that (2) the playout delay is equal to the delay that would be used if FEC were absent plus an additional delay to be able to use FEC (as in *rat* and *freephone* [5, 6]). Thus [4] uses the philosophy that if the source went to the trouble of adding some redundancy (FEC) then the destination should wait for the redundant information to arrive.

This philosophy is challenged by the work of Rosenberg et al [7], who tackle the problem of the delay introduced by FEC, in the case of non delay aware FEC. They point out that waiting for all the redundant information is inappropriate when network loss rate is low and he proposed a number of playout adaptation algorithms that provide a coupling between FEC and playout buffer adaptation. These algorithms suppose that a *play first* strategy is used at the destination, namely, that the destination plays the first copy correctly received of a given packet. These coupled playout algorithms improve the delay performance for non delay aware FEC schemes, since they allow a destination not to wait for the FEC packets that are not needed. However, this is less clear in the case of delay aware FEC, since the source sends FEC packets only when necessary. This is one of the questions addressed in this paper.

More generally, we consider the joint problem of FEC and playout adjustment at source and destination of an interactive audio source, while accounting for the impact of delay in the perceived utility. We propose and analyze two solution methods of increasing complexity that solve the joint problem. The first one ("partial" method, called *N1* in this paper) adjusts the playout buffer at destination using the results in [7]; the FEC scheme (at the source) is aware of the playout delay computed at the destination and adjusts redundancy as a function of network characteristics and of playout delay, so as to maximize the perceived audio quality. Method N1 supposes that the destination uses a *play first* strategy (to be consistent with the use of a coupled playout algorithm). A second, more elaborate method ("complete" method, called *N2*), jointly chooses both the playout delay and the FEC scheme so as to maximize the perceived audio quality. In the latter method, the playout delay and the FEC scheme are parameters of the optimization problem, while in the former, the playout delay is adapted separately and the best FEC scheme is chosen given this playout delay. In method N2, we consider the use of both play first and play best strategies at the destination.

As a basis for comparison, we also use two straightforward combinations of existing FEC and playout adjustment schemes. The first one (method *O1*) uses the delay aware FEC adjustment method proposed in [4] together with the classical playout adaptation, plus enough delay to wait for all FEC to be received at destination. The second one (method *O2*) combines the non delay aware FEC scheme proposed in [2] with the playout delay adjustment in [7].

In addition to the development of methods N1 and N2, we address the following questions:

1. Are there significant quality improvements by using the complete joint method N2 ? More generally, how do the different methods compare ?

2. Is it worth using a joint FEC/playout adaptation scheme when delay aware FEC is used or, as mentioned earlier, can we rely on the source to send only those FEC packets that are necessary and wait for all FEC to be received ?

3. With the complete method N2, is it preferable to adopt a *play first* or a *play best* strategy ?

Our technical approach for designing N1 and N2 is as follows. We start by considering the perceived audio quality as a function of the audio reconstructed rate at the destination, the packet loss rate *and* the end-to-end delay. We model the channel by a (1) packet loss process that we assume to be a Gilbert loss process and (2) a stationary delay process (not necessarily i.i.d.). We assume that the network delivers packets in sequence, a reasonable assumption since audio applications have a relatively small rate. We show that, given this assumption, our method needs only to know the marginal distribution of delays (i.e. the latter hypothesis encompasses all the needed information on the joint delay distribution). We further suppose that (3) delay and losses are stochastically independent, an assumption which makes sense if packet losses are due to active queue management schemes such as RED. Then we write our FEC and playout control problem as an optimization problem, and solve it numerically. When used over the standard IP best effort service, an audio source should also control its rate in order to react to network congestion and share bandwidth fairly, in some sense [8, 9]. This is why we also incorporate a TCP-friendly module into our design.

We designed and implemented methods N1, N2, O1 and O2 and found by simulation and analytical results that:

1. Our complete scheme N2 performs better than the partial scheme N1 and the combinations of existing schemes O1 and O2, in the cases where end-to-end delay is important.

2. Contrary to some intuition, it is worth using a joint playout adjustment algorithm when a delay aware FEC scheme is used. Classical playout delay adjustments as used in *rat* and *freephone* lead to excessive playout delay.

3. When used with the joint adaptation scheme N2, the *play first* and *play best* strategies have similar performance.

The paper is organized as follows. Section 2 gives the necessary background material on error control for audio applications, playout adjustment algorithms, TCP friendly rate control and audio quality measure. Section 3 describes our utility functions, which are adapted from the E-model. Section 4 presents our main results, namely, the derivation of our joint rate/error/playout delay control methods N1 and N2. Results of simulations implemented in ns2 [10] are given in Section 5.

2 Background Material

2.1 Error recovery

An audio transport protocol may cope with packet losses by [1]: (1) retransmitting dropped packets, (2) using error-concealment algorithms to correct the losses, or(3) applying Forward Error Correction (FEC) to reconstruct the missing packet.

Retransmission algorithms based on Automatic Repeat reQuest (ARQ) have been successful in protocols like TCP, but they are typically not acceptable for real-time audio applications since they dramatically increase the end-to-end delay. FEC, on the other hand, is an attractive alternative to ARQ since it provides relatively low-delay performance. The principle of FEC is to send redundant information, along with the original information, so that lost data can be recovered, at least partially, from this redundant information. When FEC fails to recover from a loss, applications can resort to *error concealment* algorithms at the receiver to correct the effect of missing packets [11]. Error concealment algorithms[11, 12, 13, 14] use repair mechanisms like insertion, interpolation or regeneration. These techniques work well for relatively small loss rates ($\leq 10\%$) and for small packets (4-40 ms of audio) but break down when the loss length approaches the length of a phoneme (5-100 ms). Hence, error concealment schemes should be regarded as complementary to, and not substitute for, FEC.

FEC techniques can be classified as *media independent* and *media specific*. Media independent FEC uses block codes (*e.g.* based on Reed-Solomon [15] or on parity codes [16, 17]) to provide redundant information. Each code takes a codeword of k data packets and generate $n-k$ additional check packets for the transmission of n packets over the network. Such a code, denoted as an (n, k) code, is able recover all losses in the same block if an only if at least k out of n packets are received correctly. These techniques have the advantage of recovering the loss packet in a bit-exact way. On the other hand, they have

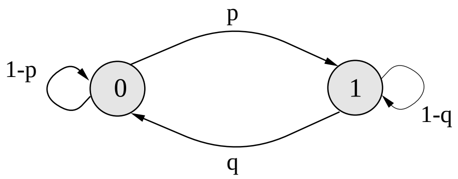


Figure 1: *The Gilbert model*

the disadvantage of introducing additional delays that can be significantly large (up to $n-1$ packet intervals).

Media specific (also called *signal processing*) FEC is used by most audio conferencing tools. The principle of the *signal processing* FEC is to transmit each segment of audio, encoded with different quality coders, in multiple packets. When a packet is lost, another packet containing the same segment (maybe encoded differently) can be able to cover the loss. This approach has been advocated by Hardman *et al* [11] and Bolot *et al* [18] for use on the Mbone, and extensively simulated by Podolsky *et al.* [8].

The first transmitted copy of the audio segment is called primary encoding and subsequent transmissions secondary encodings. With *signal processing* FEC, redundant audio segments are piggy-backed onto a later packet, which is preferable to the transmission of additional packets, as this decreases the amount of packet overhead and routing decisions. For example, in the case of a single redundant segment, packet n contains, in addition to its encoded samples, a redundant version of packet $n-1$. This redundant information is usually obtained using a lower-bit-rate, lower-quality encoding than the primary information. This simple scheme only recovers from isolated losses but can be modified (as proposed in [9]) to recover from consecutive losses as well by carrying in packet n redundant versions of packets $n-1$ and $n-2$, or of packets $n-1$, $n-2$ and $n-3$ or of packets $n-1$ and $n-3$ etc.

2.2 Loss Process of Audio Packets

The efficiency of the FEC depends on the characteristics of the loss process of audio packets. Typically, FEC is more efficient when the consecutive number of lost packets is small.

There has been many research efforts in the measurement and modeling of end-to-end Internet characteristics [19],[20],[21]. The main result is that the correlation structure of the loss process of audio packets can be modeled with low order Markov chains. In particular, a two-state Gilbert model was found to be an accurate model in many studies. Moreover, it was found that the distribution of the number of lost packets in a loss period is approximately geometric [18],[21]. These results confirmed that FEC schemes are well suited for interactive audio applications in the Internet. The Gilbert model (depicted in Figure 1) is a two-state model in which state 1 represents a packet loss and state 0 represents a packet reaching the destination. The parameters p and q denote respectively the probabilities of passing from state 0 (no loss) to state 1 (loss) and from state 1 to state 0. The stationary probabilities to be in state 1 (π_1) and in state 0 (π_0) are given by:

$$\pi_1 = \tfrac{p}{p+q} \quad \text{and} \quad \pi_0 = \tfrac{q}{p+q}$$

In absence of redundant information, the packet loss rate is given by the unconditional probability to be in state 1: $PLR = \pi_1$. The Gilbert model also allows to compute the packet loss rates after reconstruction when FEC is used. The n-stage transition matrix $P^n = [p_{ij}^{(n)}]$, $i, j \in \{0, 1\}$ is given by:

$$P^n = \frac{1}{p+q}\begin{bmatrix} q & p \\ q & p \end{bmatrix} + \frac{(1-p-q)^n}{p+q}\begin{bmatrix} p & -p \\ -q & q \end{bmatrix} \tag{1}$$

2.3 Playout Adjustment Algorithms

Jitter is compensated for at the receiver by means of adaptive playout buffer algorithms at the receiver. Most existing playout adaptation algorithms work by taking some measurements on the delays experienced by packets and updating the playout delay on a talkspurt to talkspurt basis. The main purpose of playout adaptation algorithms is to trade delay for loss. Let D be the playout delay which is defined as the difference between playout time and generation time for all the packets in a talkspurt; clearly, the larger D, the smaller the fraction of late packets.

Classical methods for playout adaptation (in absence of FEC) were proposed in [22, 23]. [22] proposed four adaptation algorithms. Each algorithm computes in some way an estimate of the mean $\hat{d}$ and the standard $\hat{v}$ deviation of the delays experienced by previous packets and adjusts the playout delay at the beginning of each talkspurt to be $D = \hat{d}+4\hat{v}$. All four algorithms compute $\hat{v}$ in the same fashion, using an exponentially weighted moving average. The four algorithms differ only in their computation of $\hat{d}$: Algorithms 1 and 2 in [22] compute an exponential moving average of the delays, but with different weighting factors. Algorithm 3 sets $\hat{d}$ to be the minimum delay experienced during the prior talkspurt. Algorithm 4 performs delay spike detection. During a spike, the delay estimate tracks the delays closely, and after a spike, an exponential weighted moving average is used. For a detailed description of this algorithm, see [22]. The adaptive playout adjustment algorithm proposed in [23] tracks the network delay of recently received packets and maintains delay percentile information. This algorithm also performs spike detection. During spike mode, the delay of the first packet in a talkspurt is used as playout delay. During normal mode, the playout delay is the qth percentile among the last w packets received by the receiver.

When FEC is used, existing audio tools compute the playout delay as if FEC were absent (using one of the methods described above), compute some delay needed to make use of FEC and combine the two. If FEC is not adaptive, it is clear that this method introduces too much delay when network losses are rare. To cope with this problem, new playout adjustment algorithms were proposed in [7] to integrate more smoothly the FEC packets into the playout buffer. The algorithms proposed in [7] include:
- *Virtual delay* algorithms that are all modifications of algorithms proposed in [22, 23]. The virtual algorithms use any of the existing playout adaptation algorithms which compute the playout delay D as a function of the packet delays but substitute the packet delays by the packet *virtual delays* that are defined as the difference in time between the earlier of the arrival and the recovery times, and the generation time of the packets.
- *A previous optimal* algorithm that uses the optimal delay for the previous talkspurt as playout delay for the next talkpurt. The optimal playout delay can be chosen to meet an arbitrarily chosen criteria. In [7], the optimal playout delay for a talkspurt is defined as the minimun delay that achieves a specified loss rate after reconstruction.
- *An analytical* algorithm that works by attempting to model the impact of network loss and delays on the application playout probability and the end-to-end delay. It then uses this model to find the playout delay D that meets a particular loss rate after reconstruction. For a detailed description of these coupled algorithms, see [7].

2.4 Rate Control

It has been suggested that audio applications share resources fairly with each other and with current TCP-based applications. One way to ensure this is to implement some form of congestion control that adapts the transmission rate in a way that *fairly* shares congested bandwidth with TCP applications, thus falling into the category of *TCP-friendly* congestion control [24].

There has been significant previous research on TCP-Friendly control mechanisms and many control schemes were proposed in the literature. Prominent examples of these schemes are: (1) window-based control mechanisms such as TEAR [25], (2) mechanisms based on additive increase, multiplicative decrease (AIMD) such as RAP [26] and LDA [27] and (3) equation-based mechanisms such TFRC [28], and mechanisms designed specifically for video transport [29, 30] and for audio transport [4].

The rate control scheme that we use in this work is the one described in [4]. This TCP-Friendly rate control relies on the Real-Time Transport Protocol (RTP) [31] and its control part, RTCP. With this scheme, the source performs equation-based [32] congestion control based on feedback information contained in RTCP reports and adjusts its sending rate by changing the packet size, the time interval between packets remaining constant.

Table 1: Speech transmission quality classes and corresponding rating value ranges.

R-value range	MOS	Speech transmission quality
$100-90$	4.50-4.34	best
$90-80$	4.34-4.03	high
$80-70$	4.03-3.60	medium
$70-60$	3.60-3.10	low
$60-0$	3.10-1.00	very poor

2.5 An Audio Quality Measure: the E-Model

In this section, we give a brief overview of the E-model. A detailed description can be found in [33, 34, 35, 36].

The E-model predicts the subjective quality that is experienced by an average listener combining the impairment caused by transmission parameters (such as loss and delay) into a single rating. The rating can then be used to predict subjective user reactions, such as the Mean Opinion Score (MOS). According to ITU-T Recommendation G.107, every rating value corresponds to a speech transmission category, as shown in Table 1. A rating below 60 indicates unacceptable quality, while values above 70 correspond to PSTN quality (values above 90 corresponding to very good quality).

The E-model rating R is given by:

$$R = R_0 - I_s - I_d - I_e + A \tag{2}$$

where R_0 groups the effects of noise, I_s represents impairment that occur simultaneously with the voice signal (quantization), I_d is the impairment associated with the mouth-to-ear delay, and I_e is the impairment associated with signal distortion (caused by low bit rate codecs and packet losses). The advantage factor A is the deterioration that callers are willing to tolerate because of the 'access advantage' that certain systems have over traditional wire-bound telephony, e.g. the advantage factor for mobile telephony is assumed to be 10. Since no agreement has been reached for the case of VoIP services, we drop the advantage factor in this study.

2.5.1 Delay Impairment I_d

The delay impairment I_d models the quality degradation due to the one way end-to-end delay d. The delay impairment I_d is the sum of three contributing impairments:

$$I_d = I_{dte}(d, TEL) + I_{dle}(d, LEL) + I_{dd}(d) \tag{3}$$

The terms I_{dte} and I_{dle} capture the impairments due to talker and listener echo respectively. TEL and LEL are the echo losses (in dB) at the points of reflexion and depend on the echo cancellation applied. In this work, we assume that the echo losses at both ends are equal. In the sequel, we denote the echo loss by EL, with $EL = TEL = LEL$. For packetized voice calls[33], $EL = \infty$ corresponds to a perfect echo control and $EL = 51$ corresponds to a simple yet efficient echo controller. The third delay-relative impairment, I_{dd} is the loss of interactivity. In practice, the mouth-to-ear delay d is composed of the Encoding/Decoding delay (algorithmic and packetization delay), the network delay (transmission, propagation and queuing delay) and the de-jitter delay (delay spent in the playout buffer in order to cope with delay variations). Although the voice encoding and decoding process can take a significant amount of time when performed by software based implementations, the algorithmic delay is reducible by using fast hardware implementations. Without access to specific implementation information and processing delay measurement, we consider in this work that this delay is very small and can be neglected. As a consequence, the Encoding/Decoding delay is reduced to the packetization delay, that we assumed to be fixed. The playout delay as defined in Section 2.3 actually represents the sum of network and de-jitter delays. The mouth-to-ear delay is therefore the sum of the packetization delay and the playout delay. [34, 35] gives analytical formaulae to compute I_d as a function of the echo loss and the mouth-to-ear delay.

2.5.2 Equipment Impairment I_e

The impairments introduced by distortion are brought together in I_e. Currently, no analytical expression allows to compute I_e as a function of parameters such as the encoding rate or the packet loss rate. Estimates for I_e must be obtained through subjective measurements. A few values for I_e are given in Appendix A of [36] for several codecs and several packet loss conditions and a recent study [37] provides additional results about the impact of losses on the perceived audio quality. We build our utility function based on the values given in [36], as explained in Section 3.

3 Choice of Utility Functions Which Account for Delay

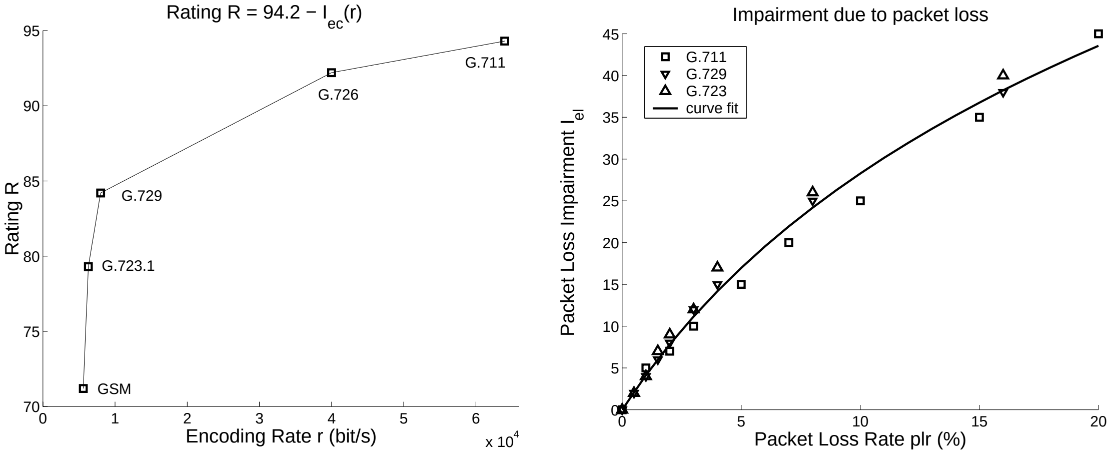
We use the E-model as a basis for defining our utility functions. However, we need to further elaborate on it, for two reasons. First, our optimization framework requires an analytic expression for the rating R as a function of the encoding rate r and the packet loss rate plr, which the E-model does not readily provide. Second, we wish to define different utility functions corresponding to a different modes of interactivity [38] and thus need to consider additional expressions for the delay impairment.

3.1 Influence of Distortion

The equipment impairment I_e captures the distortion of the original voice signal due to 1) the use of low bitrate codec and 2) packet losses (that occur in the network and in the playback buffer). In this paper, we consider these two causes of distortion separately, our goal being to approximate I_e with an expression of the form:

$$I_e(r, plr) = I_{ec}(r) + I_{el}(plr) \tag{4}$$

where I_{ec} would represent the impairment due the audio encoding and I_{el} would be the impairment due packet losses. Since we know that the effect of packet loss is to increase the measured distortion, if the distortion increases in the same way for all the codecs as a function of the packet loss rate, then this approximation would be acceptable. Let us first consider the distortion introduced by the encoder (in absence of packet loss). Various values for the intrinsic impairment of a variety of codecs can be found in [36] and [33]. Figure 2(a) shows the rating R for voice calls using different bit rate codecs, for zero end-to-end delay and no packet loss. Since these values are obtained though subjective tests, we have at our disposal only a few samples of $I_{ec}(r)$ measured for discrete values of r (let us call them r_i,$i = 1, \ldots, n$) corresponding to the encoding rate of n existing codecs. Knowing the value of our utility function for discrete values of r is sufficient in the case where we wish to choose the best codec among existing ones in order to maximize the utility function of our audio source (see Section 4). However, it would be interesting to solve the problem



(a) (b)

Figure 2: *Influence of distortion on quality. (a) Rating as a function of the encoding rate (for delay=0 and no loss), (b) Impairment due to loss as a function of the packet loss rate.*

also in the general case where the audio source would be able to encode audio samples with an infinite (large) number of rates ranging from 0 to 64Kbits/s. This could be made possible with some sophisticated coding techniques such as multiple description coding [39]. To obtain a continuous utility function with respect to the encoding rate, we use the values given for existing codecs and simply interpolate between these values to obtain a piece-wize linear utility function (as shown in Figure 2(a)).

Now let us consider the impact of packet loss on the measured distortion. In Appendix I of [36], values of I_e are given for several codec types as a function of packet loss rate and packet loss burstiness. In this paper, we focus on results corresponding to random losses because 1) they are equivalent to the results obtained with bursty losses if the packet loss rate is small ($\leq$ 5%), which will typically be the case of the packet loss rate after reconstruction, and 2) because a precise description of the algorithm used for generating bursty packet losses is missing from [36]. Furthermore, we assume that all codec implement some form of error concealment, which is a common practice today i.e. it is built-in in G.729A and G.723.1 and can be added on top of G.711.

In Figure 2(b), we plotted the values of $I_e - I_{ec}$ as a function of the packet loss rate plr for a variety of codecs. We can see that I_{el} increases with the packet loss rate. Moreover, we can see that the all codec follow more or less the same trajectory. Therefore, we can use a curve fitting technique to represent with a good approximation the impairment due to packet loss as a continuous function of the loss probability, independently of the codec in use. We use a logarithmic curve (see Figure 2(b)), as advised in [40] to approximate $I_{el}(plr)$ and obtain:

$$I_{el}(plr) = 34.3\ \ln(1 + 12.8\,plr) \tag{5}$$

3.2 Influence of end-to-end Delay

As explained in [38], there is a dimension that is not captured by the E-model, that of different modes of conversation or interactivity requirements . Different types of conversation require different switching speed and thus have a different sensibilities to delay. For example, a business call might require a higher level of interactivity, with shorter messages and higher switching speed than a social call. In [38], they conjecture that the fact that the E-model does not account for different modes of conversation may imply that the curves provided capture some kind of averaging of these different modes.

Besides, various studies [3, 41] concluded that, for natural hearing the end-to-end delay should be approximately 150 ms. While delays lower than 100 ms can not really be appreciated, delays above 150 ms are noticed by the users and become a hindrance to interactivity. As a result, the impairment caused by too long mouth-to-ear delays should be negligible for delays smaller than 150 ms and than increase rapidly as the delay gets larger than this threshold. Moreover, it is also recognized that telephony users find delays of greater than about 300 ms more like half duplex connection than a conversation. In order to account for the great impact of the interactivity threshold (of 150ms) on a quality of conversation, a utility function with a steep decrease around 150 ms was proposed in [4].

In this paper, we consider three different delay impairment functions, representing different sensibilities to the end-to-end delay. The first delay impairment function $I_{d1}(d)$ considered (see Figure 3) is based on the utility function proposed in [4] and characterizes a user with strong interactivity requirements. Based on [4], the delay impairment (that was scaled to fit in the E-model) is expressed I_{d1} as follows:

$$I_{d1}(d) = \begin{cases} \gamma_1 \, d & \text{if } d \leq 150 \\ b_1 \tanh(\beta(d - b_2)) + b_3 & \text{if } 150 < d < 300 \\ \delta + \gamma_2 \, d & \text{if } d \geq 300 \end{cases}$$

where d is expressed in ms, $\gamma_1 = \gamma_2 = 0.01$, $\beta = 0.02$, $\gamma = 50$ and b_1, b_2 and b_3 are constants selected to ensure the continuity of I_{d1}. Figure 3 shows the utility function as a function of end-to-end delay (in absence of distortion impairment). The utility function obtained with I_{d1} presents the following behaviour: for delays below the critical threshold of 150 ms, the quality decreases very slowly as the users do not benefit from getting a lower end-to-end delay. Then, above this threshold, the quality drops steeply as any increase of the end-to-end delay hurts the interactivity. Then, above 300 ms, since the connection is considered as a half duplex conversation anyway, any further increase of the delay only slightly affects the quality (which is already low).

The second and the third delay impairment functions considered are based on the E-model: $I_{d2}(d) = I_d(d, 51)$ represents a user that is annoyed by delay because of echo, but without a clear threshold effect and $I_{d3}(d) = I_d(d, \infty)$ represents a user that attaches a small importance to delay. Figure 3 shows the influence of the mouth-to-ear delay on the utility function for these two delay impariment functions.

3.3 Our Utility Functions

In summary, we consider three different utility functions $f_i : \Re^+ \times \Re^+ \times \Re^+ \rightarrow [0, 100]$, $i = 1 \ldots 3$ that correspond to three different interactivity requirements:

$$f_i(r, d, plr) = 94.2 - I_{di}(d) - I_{ec}(r) - I_{el}(plr) \tag{6}$$

where d is the one-way mouth-to-ear delay, r is the codec bit rate and plr is the residual packet loss rate (after FEC is used), $I_{di}(d)$ $(i = 1 \ldots 3)$, $I_{ec}(r)$ and $I_{el}(plr)$ are defined above.

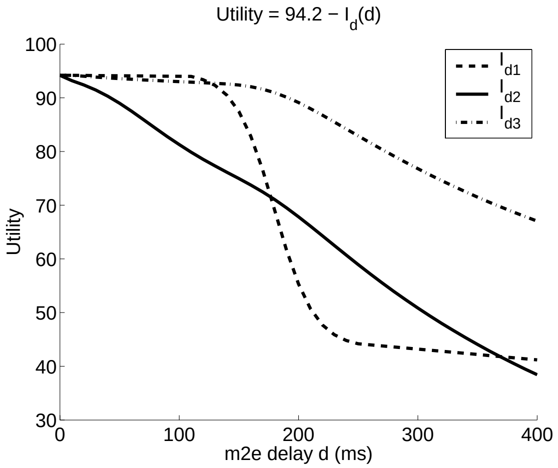


Figure 3: *Utility as a function of the mouth-to-ear delay for different interactivity levels for $r = 64Kbits/s$ and no loss.*

4 Adaptive Joint Playout Delay and FEC Adjustment

In this section, we present two methods that solve the joint problem of FEC and playout adjustment at source and destination, while accounting for the impact of delay in the perceived utility. The "partial" method N1 adjusts the playout buffer at the receiver using the results in [7]; the FEC scheme at the source is aware of the playout delay computed at destination and adjust redundancy so as to maximize the perceived utility. If the source uses Signal Processing FEC (SP FEC) with method N1, a play first strategy has to be used (since it is the strategy used with coupled playout algorithms). If the source uses media independant FEC like Reed-Solomon FEC (RS FEC), since all the copies are identical, there is only one possible strategy (since the first copy is also the best). The "complete" method, on the other hand, jointly chooses both the playout delay and the redundancy at the source so as to maximize the perceived utility. With method N2, no constraint is imposed on the decoding strategy. We thus consider both the play first and play best strategy when signal processing FEC is used. In summary, we study five possible combinations of FEC and playout adjustment method, FEC scheme and decoding strategy: (N1,SP FEC,play first), (N2,SP FEC, play first), (N2,SP FEC,play best), (N1, RS FEC) and (N2, RS FEC).

4.1 General Method

Consider a voice source with the flexibility to encode its samples at a rate x such that $x \in [0, X_{max}]$ (in the general case) or $x \in R$ with $R = \{r_i, i = 1, \ldots, n\}$ (if the source has a limited set of coders at her disposal).

The quality of the voice call is characterized by a utility function $f : \Re^{+} \times \Re^{+} \times \Re^{+} \rightarrow [0, 100]$ as described in equation (6). Our utility function has now three parameters: (1) the reconstructed rate at the destination and (2) the end-to-end delay and (3) the packet loss rate. It should be pointed out that the packet loss rate considered here is the residual packet loss rate after reconstruction of the lost packets using FEC.

The source transmits voice packets to a destination over an unreliable network about which we made some assumptions. For the control scheme, we modeled the network by:

- **a packet loss process:** Y_i which we suppose to be a Gilbert process where $Y_i \in \{0, 1\}$ (see Section 2.2). If the *ith* packet reaches the destination, then $Y_i = 0$, otherwise, $Y_i = 1$. The parameters p and q of the Gilbert model are estimated on-line at the receiver using the maximum likelihood estimator.

- **a stationary delay process:** D_i**.** We suppose that the network delays of voice packets are identically distributed and follow a given distribution $F_D(d) = P(D_i \leq d)$ [1]. Moreover, we assume that the network delivers packets *in sequence*. The latter assumption can be expressed as follows:

  $$Pr(D_{n+1} \leq D_n - T) = 0 \qquad (7)$$

  which, by induction, is equivalent to:

  $$Pr(D_{n+i} \leq D_n - i\,T) = 0 \qquad (8)$$

  where D_n is the network delay of the nth packet and T is the time interval between two consecutive voice packets. We show that this hypothesis allows us to consider only the marginal distribution of delays and not the joint distributions.

- **independence loss-delay**: we assume that packet losses and network delays are mutually independent.

It is clear that this model is quite simple but we have to find a tradeoff between complexity and tractability since our control scheme has to be implemented in audio sources.

The no-reordering assumption allows to write:

Property 1 For a given packet, if the nth copy arrives after playout time, then all the following copies also arrive after playout time

$$Pr(D_{n+i} \leq D - iT | D_n > D) = 0 \quad , i \geq 0 \qquad (9)$$

[1] The parameters p, q and and the parameters characterizing the delay distribution (which are all estimated on-line at the receiver) are sent back to the source via the application specific part (APP) of the RTCP receiver reports.

Property 2 For a given packet, if the nth copy arrives before playout time, then all the preceding copies also arrive before playout time (provided they are not lost in the network)

$$Pr(D_{n-i} > D + iT | D_n \leq D) = 0 \quad , i \geq 0 \tag{10}$$

Property 3 The joint probability $Pr(\{D_n \leq D\} \cap \{D_{n+1} > D - T\})$ is computed as follows:

$$\begin{aligned}
&Pr(\{D_n \leq D\} \cap \{D_{n+1} > D - T\}) \\
&\quad = Pr(D_n \leq D) \times Pr(D_{n+1} > D - T | D_n \leq D) \\
&\quad = Pr(D_n \leq D) \times (1 - Pr(D_{n+1} \leq D - T | D_n \leq D)) \\
&\quad = Pr(D_n \leq D) \times \\
&\qquad (1 - \frac{Pr(D_n \leq D | D_{n+1} \leq D - T) Pr(D_{n+1} \leq D - T)}{Pr(D_n \leq D)}) \\
&\quad = Pr(D_n \leq D) \times (1 - \frac{Pr(D_{n+1} \leq D - T)}{Pr(D_n \leq D)}) \\
&\quad = Pr(D_n \leq D) - Pr(D_{n+1} \leq D - T)
\end{aligned} \tag{11}$$

The "no reordering" assumption allows to obtain all needed probabilities from the sole marginal distribution F_D. All information about the joint distribution is encompassed in the no-reordering assumption.

Let R_{max} be the rate available for the audio flow. R_{max} is the result of our TCP-Friendly rate control scheme (which is introduced in Section 2.4 and described in details in [4]) and is updated upon reception of an RTCP receiver report.

Let D be the playout delay for each talkspurt.

Then, our general problem can be stated as follows: Given that we can send at most K_{max} copies of each voice packet, find the optimal combination of coding scheme (optimal number of copies to send and optimal encoding rate for each copy) and playout delay D so as to maximize the quality of the voice call subject to the rate constraint.

This problem can be can be formulated as an optimization problem and the formulation depends on the combination of (1) the FEC/playout adjustment method, (2) the error recovery technique and (3) the decoding strategy. In the following, we study the different scenarios separately.

4.2 Detailed Formulation of the Sub-Problems

4.2.1 Method N1 - SP FEC - play first

In this case, the optimization problem can be expressed as follows (P1):

$$\begin{array}{ll} \text{Given:} & p,\, q,\, R_{max},\, F_D(d) \text{ and } D \\ \text{maximize} & \sum\limits_{i\in\Gamma} P_{play}(i)\, f(x_i, D+T, PLR_{FEC}) \\ \text{over all} & \Gamma,\ (x_i\,,\, i\in\Gamma) \\ \text{subject to} & \left\{ \begin{array}{l} \sum\limits_{i\in\Gamma} x_i + R_{overhead} \le R_{max} \\ x_i \ge r_0\,,\, i\in\Gamma \end{array} \right. \end{array}$$

where x_i is the encoding rate of the copy placed in ith position in the stream; T is the packetization delay[2]; $P_{play}(i)$ is the probability that the ith copy is sent to the audio driver; $R_{overhead}$ is the bandwidth overhead of the IP/UDP/RTP headers and PLR_{FEC} is the packet loss rate after reconstruction. Γ is the set of copies of a given packet that are sent by the source i.e., $\Gamma = \{i|$ copy placed in ith position in the stream was sent$\}$. A value $i = 1$ corresponds to the primary information, thus the minimal $\Gamma = \{1\}$ corresponds to a source that sends no redundant information. Without loss of generality, we assume that the set Γ is an ordered set, i.e. if $\Gamma(k)$ denotes the kth element in Γ, we can write $\Gamma(k) < \Gamma(k+1)$. $P_{play}(i)$ is the probability that the ith copy is the first copy correctly received and that it is on time. Let nc denote the number of elements in Γ, namely nc is the total number of copies of a given packet that are sent by the source.

Define K to be the position of the last copy of a given packet sent by the source: $K = \Gamma(nc)$. [3] The optimal value of K is a priori unknown and is supposed to be in $[1, K_{max}]$. In practice, the larger K, the longer the destination has to wait to receive all the redundant information.

$P_{play}(i)$ is a function of the playout delay D, the parameters of the Gilbert model p and q and of the redundancy scheme Γ and can be computed as follows:

$$\begin{aligned} P_{play}(i) &= Pr(\{\{Y_{n+i-1} = 0\} \cap \{D_{n+i-1} \le D - (i-1)T\}\} \bigcap \\ &\quad \{\{\{Y_{n+k-1} = 0\} \cap \{D_{n+k-1} > D_{n+i-1} + (i-k)T\}\} \\ &\quad \cup \{Y_{n+k-1} = 1\} \forall k \in \Gamma, k < i\}) \\ &= Pr(\{\{Y_i = 0\} \cap \{D_i \le D - (i-1)T\}\} \bigcap \\ &\quad \{\{Y_k = 1\} \cup \{\{Y_k = 0\} \cap \{D_k > D_i + (i-k)T\}\} \\ &\quad \forall k \in \Gamma, k < i\}) \end{aligned}$$

[2] Since the playout delay is defined as the difference between playout time and generation time of for all the packets in a talkspurt, the mouth-to-ear delay is the sum of playout delay and packetization delay.

[3] As example, if a scheme $(n, n-2)$ is used, $\Gamma = \{1, 3\}$, $nc = 2$ and $K = 3$.

$$
\begin{aligned}
& \text{(since the delay and packet loss processes are stationary)} \\
= \;& Pr(\{\{Y_i = 0\} \cap \{D_i \leq D - (i-1)T\}\} \bigcap \\
& \{\{Y_k = 1\} \forall k \in \Gamma, k < i\}) \\
& \text{(using the no-reordering hypothesis (eq.(8)))} \\
= \;& Pr(\{\{Y_i = 0\} \cap \{D_i \leq D - (i-1)T\}\}) \times \\
& Pr(\{\{Y_k = 1\} \forall k \in \Gamma, k < i\} | \{\{Y_i = 0\} \cap \\
& \{D_i \leq D - (i-1)T\}\}) \\
= \;& Pr(\{Y_i = 0\}) \times Pr(D_i \leq D - (i-1)T) \times \\
& Pr(\{\{Y_k = 1\} \forall k \in \Gamma, k < i\} | \{Y_i = 0\}) \\
& \text{(by the loss-delay independence)} \\
= \;& Pr(D_i \leq D - (i-1)T) \times \underbrace{Pr(\{\{Y_k = 1\} \forall k \in \Gamma, k < i\} \cap \{Y_i = 0\})}_{a_i} \\
= \;& F_D(D - (i-1)T) \qquad \times \qquad\qquad a_i
\end{aligned}
\tag{12}
$$

with

$$
a_i = \\
\begin{cases}
\pi_0 & \text{if } i = 1 \\
\pi_1 \, p_{10}^{(i-1)} & \text{if } i > 1 \text{ and } I_d(i) = 2 \\
\pi_1 \prod_{j=2}^{I_d(i)-1} p_{11}^{(\Gamma(j)-\Gamma(j-1))} \, p_{10}^{(i-\Gamma(I_d(i)-1))} & \text{if } i > 1 \text{ and } I_d(i) > 2
\end{cases}
$$

where $I_d(i)$ is the index of the ith copy in the ordered set Γ (e.g. if $\Gamma = \{1, 3\}$, $I_d(3) = 2$), and the probabilities $\{p_{ij}^{(n)}\}$, $i, j \in \{0, 1\}$ are computed using Equation 1. As expected, we can see that $P_{play}(i)$ is an increasing function of D.

The residual packet loss rate after reconstruction PLR_{FEC} is defined as the probability that none of the different copies can be played at the receiver and is given by:

$$
PLR_{FEC} = 1 - \sum_{i \in \Gamma} P_{play}(i) \tag{13}
$$

The objective function above represents the average quality measured at the destination. In the discrete case, the formulation remains the same but the second constraint is replaced by $x_i \in R, i \in \Gamma$.

4.2.2 Method N2 - SP FEC - play first

In this case, the objective function is the same as in (P1) but the input parameters have changed. The problem (P2) is thus expressed as follows:

$$\begin{array}{ll} \text{Given:} & p,\, q,\, R_{max},\, F_D(d) \\ \text{maximize} & \sum\limits_{i\in\Gamma} P_{play}(i)\, f(x_i, D+T, PLR_{FEC}) \\ \text{over all} & \Gamma,\, (x_i\,,\, i\in\Gamma),\, D \\ \text{subject to} & \left\{ \begin{array}{l} \sum\limits_{i\in\Gamma} x_i + R_{overhead} \le R_{max} \\ x_i \ge r_0\,,\, i\in\Gamma \end{array} \right. \end{array}$$

4.2.3 Method N2 - SP FEC - play best

The problem (P3) is expressed as follows:

$$\begin{array}{ll} \text{Given:} & p,\, q,\, R_{max},\, F_D(d) \\ \text{maximize} & \sum\limits_{\Delta\subseteq\Gamma\,,\,\Delta\neq\emptyset} P(\Delta)\, \max\limits_{i\in\Delta} f(x_i, D+T, PLR_{FEC}) \\ \text{over all} & \Gamma,\, (x_i\,,\, i\in\Gamma),\, D \\ \text{subject to} & \left\{ \begin{array}{l} \sum\limits_{i\in\Gamma} x_i + R_{overhead} \le R_{max} \\ x_i \ge r_0, i\in\Gamma \end{array} \right. \end{array}$$

where x_i is the encoding rate of the copy placed in ith position in the stream, $R_{overhead}$ is the bandwidth overhead of the IP/UDP/RTP headers and PLR_{FEC} is the packet loss rate after reconstruction.

The random variable $\Delta = \{i|\{Y_i = 0\} \cap \{D_i \le D - (i-1)T\}\,,\, i \in \Gamma\}$ represents the set of copies of a given packet that are received at the destination before the playout time of this packet. Without loss of generality, we assume that the set Δ is an ordered set, i.e. if $\Delta(k)$ denotes the kth element in Δ, we can write $\Delta(k) < \Delta(k+1)$.

$P(\Delta)$ is the probability to receive *exactly* the set Δ before playout time. Formally, $P(\Delta)$ is expressed as follows:

$$\begin{aligned} P(\Delta) \;=\; & Pr(\{\{Y_i = 0\} \cap \{D_i \le D - (i-1)T\}, \forall i \in \Delta\} \bigcap \\ & \{\{Y_k = 1\} \cup \{\{Y_k = 0\} \cap \{D_k > D - (k-1)T\}\}, \\ & \forall k \in \Gamma\backslash\Delta\}) \qquad (14) \end{aligned}$$

Now, let us define the events $\{A_i,\, i = 0 \ldots K\}$ as follows:

$$\left\{\begin{array}{lcl} A_0 & = & \{d_1 > D\} \\ A_i & = & \{\{d_i \le D-(i-1)T\} \cap \{d_{i+1} > D-iT\}\} \quad \text{for} \;\; i = 1\dots K-1 \\ A_K & = & \{d_K \le D-(K-1)T\} \end{array}\right.$$

From Properties 1 and 2, we can deduce that $P(\bigcup_{i=0}^{K} A_i) = 1$ and $A_i \cap A_j = \emptyset$ for $i \neq j$. Hence, from the Total Probability Theorem, $P(\Delta)$ can be computed as follows:

$$\begin{array}{rcl} P(\Delta) & = & \displaystyle\sum_{j=0}^{K} P(\Delta|A_j)P(A_j) \\ & = & \displaystyle\sum_{j=K_\Delta}^{K} P(\Delta|A_j)P(A_j) \\ & & (\text{since } P(\Delta|A_j) = 0, \, \forall j < K_\Delta) \\ & = & \displaystyle\sum_{j=K_\Delta}^{K-1} P(\Delta|A_j)(F_D(D-(j-1)T) - F_D(D-jT)) \\ & & + P(\Delta|A_K)F_D(D-(K-1)T) \\ & & (\text{from Property 3:} \\ & & \qquad \left\{\begin{array}{l} P(A_i) = F_D(D-(i-1)T) - F_D(D-iT) \quad \text{for} \;\; i = 1\dots K-1 \\ P(A_K) = F_D(D-(K-1)T) \end{array}\right. \\ & &) \end{array}$$

where K_Δ is the position of the last copy sent by the source and received on time, i.e. $K_\Delta = \Delta(n_\Delta)$ where n_Δ is the number of elements in Δ and $P(\Delta|A_j)$ is given by:

$$\begin{array}{rcl} P(\Delta|A_j) & = & Pr(\{\{Y_i = 0\}, \forall i \in \Delta\} \bigcap \{\{Y_k = 1\}, \forall k \in \Gamma \backslash \Delta, \, k \le j\}) \\ & = & \left\{\begin{array}{ll} \pi_{b_1} & \text{if } j = 1 \\ \pi_{b_1} \displaystyle\prod_{k=1}^{nc_j - 1} p_{b_k b_{k+1}}^{(\Gamma(k+1) - \Gamma(k))} & \text{if } j > 1 \end{array}\right. \end{array}$$

where nc_j is the number of elements in Γ that are inferior or equal to j and b_k is a binary value defined as $b_k = 1 - I(\Gamma(k) \in \Delta)$; , and the probabilities $\{p_{ij}^{(n)}\}$, $i, j \in \{0, 1\}$ are computed using Equation 1.

PLR_{FEC} is the probability that none of the copies arrives on time, and is given by:

$$PLR_{FEC} = 1 \; - \sum_{\substack{\Delta \, \in \, \Gamma \\ \Delta \neq \emptyset}} P(\Delta) \qquad (15)$$

4.2.4 Method N1 - RS FEC

The problem (P4) is expressed as follows:

$$\begin{array}{ll} \text{Given:} & p,\ q,\ R_{max},\ F_D(d),\ D \\ \text{maximize} & f(x, D+T, PLR_{FEC}) \\ \text{over all} & (n,k),\ x \\ \text{subject to} & \begin{cases} x\frac{n}{k} + R_{overhead} \leq R_{max} \\ x \geq r_0 \\ k \leq n \end{cases} \end{array}$$

where x is the encoding rate of each copy, $R_{overhead}$ is the bandwidth overhead of the IP/UDP/RTP headers and PLR_{FEC} is the packet loss rate after reconstruction. PLR_{FEC} is a function of the parameters of the loss process p and q, of the parameters of the FEC scheme (n, k) and is an increasing function of the playout delay D and . The detailed formulation of PLR_{FEC} is a decreasing function of the playout delay D. The detailed expression of PLR_{FEC} as a function of these parameters is given in Appendix A.

4.2.5 Method N2 - RS FEC

The problem (P5) is the same as (P4) except that D is no longer an input but a parameter. (P5) is thus expressed as follows:

$$\begin{array}{ll} \text{Given:} & p,\ q,\ Rmax,\ F_D(d) \\ \text{maximize} & f(x, D+T, PLR_{FEC}) \\ \text{over all} & (n,k),\ x,\ D \\ \text{subject to} & \begin{cases} x\frac{n}{k} + R_{overhead} \leq R_{max} \\ x \geq r_0 \\ k \leq n \end{cases} \end{array}$$

4.3 Resolution

We implemented numerical methods to solve the different optimization problems formulated above. We used a combination of the algorithm proposed in [42] (which is well suited to maximize a sum of weighted utility functions like ours and gives exact solutions) and of an exhaustive search on discrete parameters (like Γ,n,k etc.). We have a running implementation of our audio source/destination in ns2. The method is optimal if the channel complies with the model described above.

5 Simulation Results

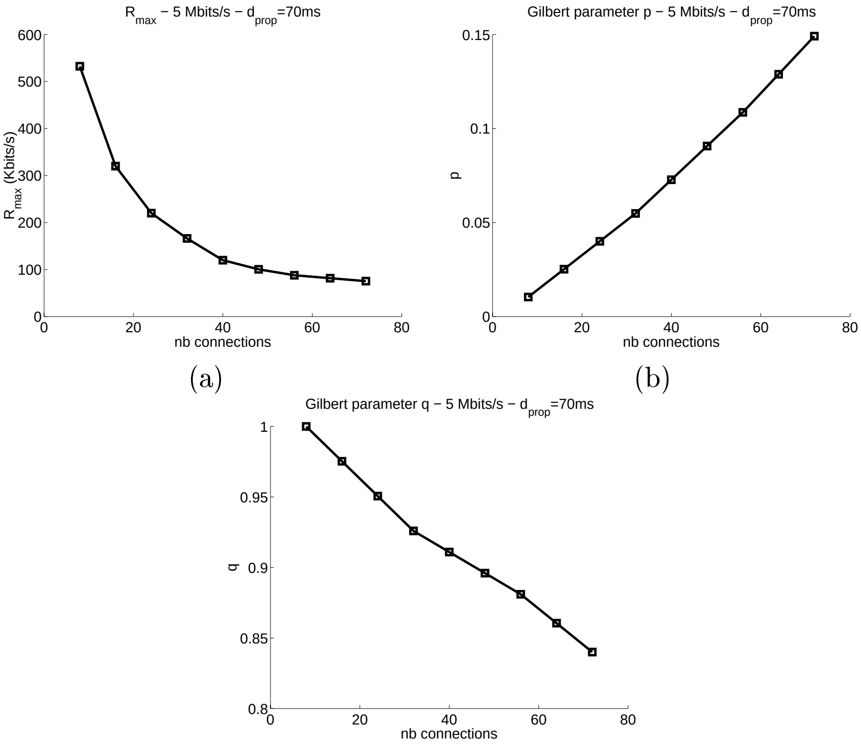
We investigated the behaviour of the different FEC/playout adjustments schemes under a wide range of loss and delay conditions. The results presented here constitute a small sample of our simulation results but are representative of the behaviours we could observe in our simulations.

We consider a simple scenario where n audio sources share a bottleneck link with $3n$ Sack TCP and an ON/OFF source (CBR 500 Kbits/s when ON) with ON and OFF periods exponentially distributed, with average ON and OFF times of 3s. Packet loss rate is varied artificially by changing the number of TCP and audio connections sharing the link. Figure 4 (a) shows the TCP-Friendly rate constraint R_{max} as a function of the number of connections sharing the link. Figures 4 (b) and (c) represent respectively the parameters p and q characterizing the losses experienced by audio flows as a function of the number of connections sharing the link. The bottleneck bandwidth is 5Mbits/s and the one-way propagation delay (without queuing) is 70 ms. The graphs show the mean values averaged over the last 300 s of simulation and over all audio connections.

Figure 5 shows the performances (in terms -from left to right- of mouth-to-ear delay (playout delay + 20 ms of packetization delay), utility measured at the receiver, residual packet loss rate and rate of the reconstructed audio stream at the receiver) for the different methods N1, N2 (Play First), O1 and O2 as a function of the number of connections sharing the link. Figures 5 (a) to (d), (e) to (h) and (i) to (l) correspond respectively to the results obtained with utility functions f_1 (high interactivity requirements), f_2 (delay is a concern but without threshold effect) and f_3 (delay is marginally important) with a Signal Processing FEC coding scheme. Figures 5 (m) to (p) were obtained with utility function f_1 and a Reed-Solomon error coding. Algorithm 1 in [22] was used to adjust the playout delay with method O1 and its virtual version was used with N1 and O2. We tested other playout adjustment algorithms proposed [7] and show the results obtained in Appendix B.1. The conclusions presented in the sequel remain the same for other existing playout adjustment algorithms.

Figures 5 (a) to (p) show that:

- method N2 always gets the higher utility compared to other methods. The gain in utility is significant when utility f_1 is used (Fig. 5(b)) with a mouth-to-ear delay 20 to 30 ms smaller than with other methods (Fig. 5 (a)); the gain in utility is smaller with utilities f_2 (Fig. 5 (f)) and f_3 (Fig. 5 (j)) but still visible, with a mouth-to-ear delay 10 to 20 ms smaller than with other methods. This shows that N2 succeeds in trading delay for losses while keeping the residual packet loss rate acceptable. If delay is important (Fig. 5 (a)), N2 manages to keep the mouth-to-ear delay much lower than other methods (20 to 30 ms lower) at the price of a slightly higher residual packet loss rate (see Figure 5 (c)). On the other hand, if delay issues are less crucial



(a) (b)

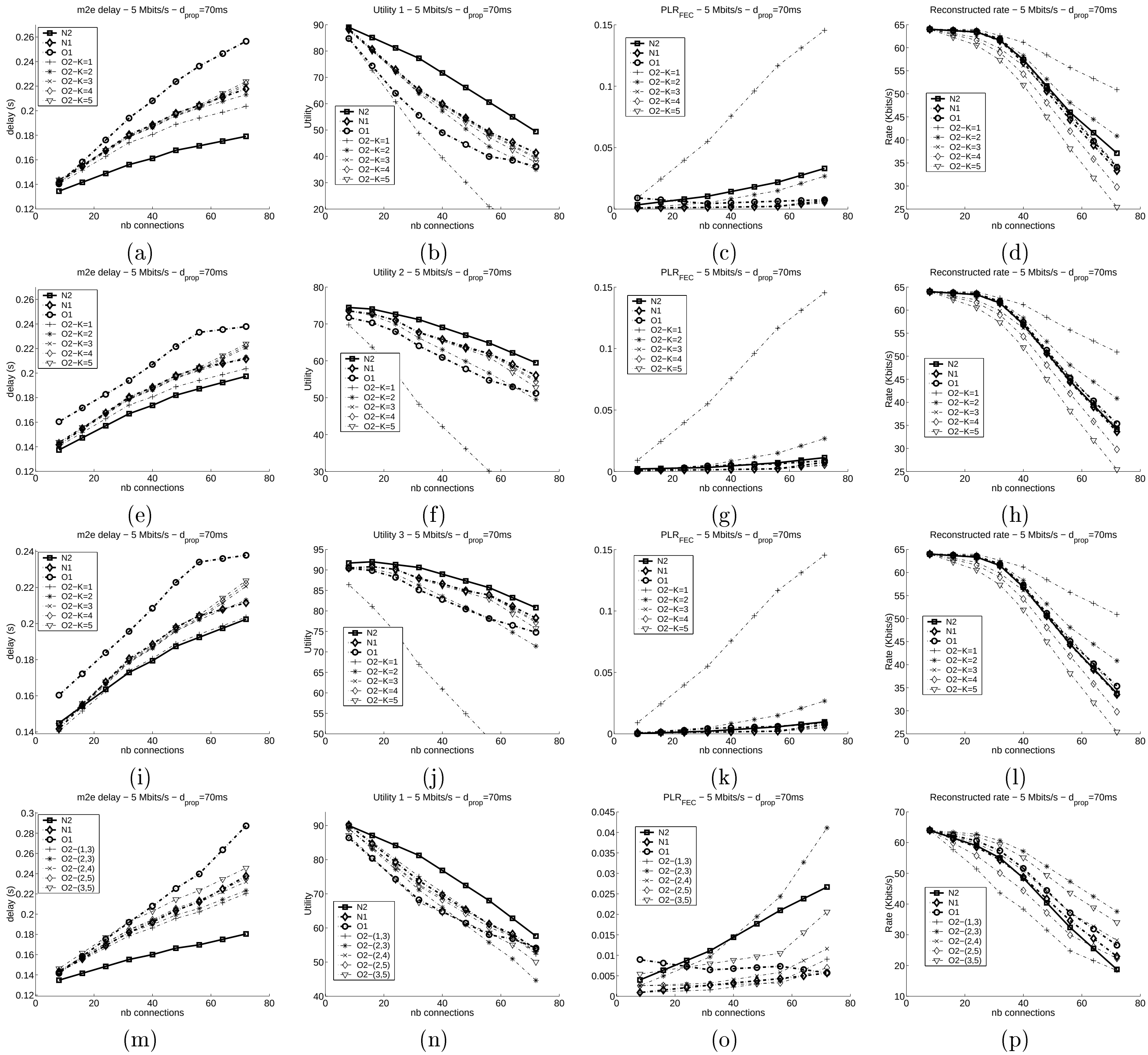
(c)

Figure 4: *(a) TCP-Friendly rate constraint. (b) Gilbert parameter p. (c) Gilbert Parameter q as a function of the number of connections sharing the link.*

for the user, it accepts to increase its playout in order to reduce the residual packet loss rate (Fig. 5 (i) and (k)).

- method N1 gives results equivalent to the best combination of existing methods. When method O2 is used with Signal Processing FEC, the optimal parameter setting is $K = 3$ (K being the total number of copies sent) for a wide range of network conditions; with Reed-Solomon (n, k), a paradoxale result is that the setting $k = 1$ gives very good result. With this setting, Reed-Solomon is equivalent to a Signal Processing FEC with all the copies encoded with the same quality.

The same simulations were performed with smaller values of propagation delay and results corresponding to a propagation delay of 25 ms are presented in Appendix B.2. In the case where the propagation delay is very small, the method N2 still outperforms other methods, but the results obtained with method N2 are very close to the one obtained



(a) (b) (c) (d)

(e) (f) (g) (h)

(i) (j) (k) (l)

(m) (n) (o) (p)

Figure 5: *Performances of the different methods N1, N2 (Play First), O1 and O2 (with different FEC parameter settings). Mouth-to-ear delay, utility measured at the destination, residual packet loss rate and reconstructed rate of audio for SP FEC with utlity f_1 from (a) to (d), f_2 from (e) to (h), f_3 from (i) to (l) and for Reed-Solomon with utility f_1 from (m) to (p).*

with method N1. This is mainly due to the fact that when delay is very small, method N2 allows the delay to increase more rapidly, in order to reduce the packet loss rate after reconstruction (that has a greater impact on the quality than the delay, when the latter is very small) and thus, the delay obtained with N2 comes closer to the delay obtained with N1[4].

Figures 6 (a) to (d) compare the performances of the play first and play best strategies when used with N2 and Signal Processing FEC. In all our simulations, we observed that the two strategies lead to similar result. Consequently, we recommend the use of the play first strategy, which is more simple.

Figure 7 (a) shows the average number of copies of a given packet that was sent by the source when using the methods N2 and O1 with utility f_2. Figure 7 (b) shows the probability that the last copy of given packet is discarded at destination because it arrived too late. Figure 7 (b) shows that the optimal solution does not always wait for all the FEC packets to arrive (in 40 to 50% of the cases, method N2 does not wait for the last FEC packet to be received). The playout adjustment scheme used by O1 leads thus to excessive playout delay (as can be seen in Figure 5 (a), (e) and (i)).

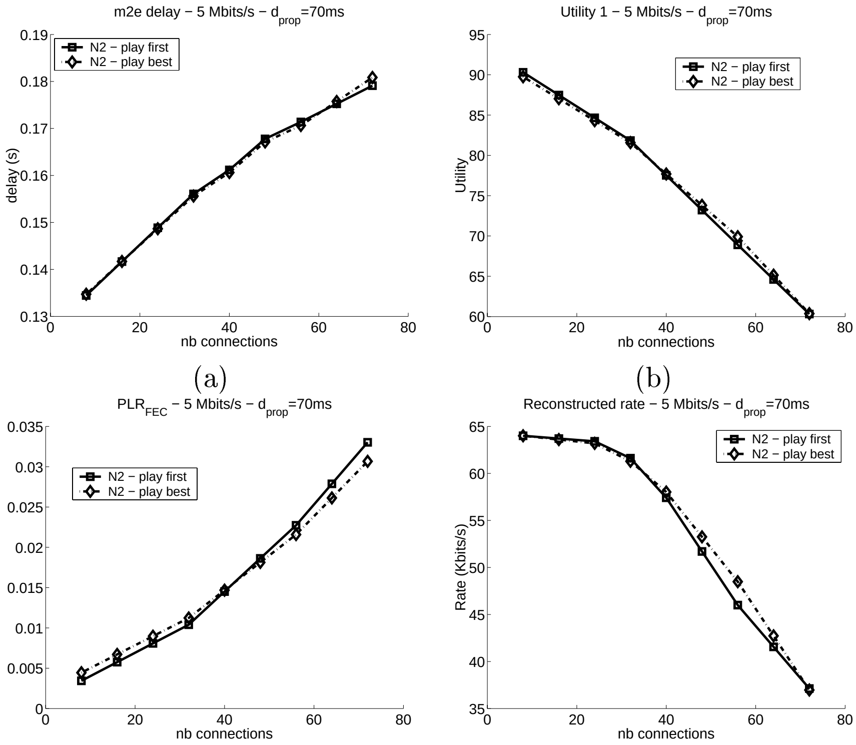
6 Conclusions

We designed a method for joint control of delay-aware FEC and playout for interactive audio applications over the Internet. We have shown that, in cases where delay matters (i.e. around a threshold effect), there is real benefit in using the joint method. We have also shown that the improvement brought by delay aware FEC cannot be obtained if the delay aware FEC control is simply piggybacked onto existing adaptive playout control methods; in contrast, it should be incorporated in a complete joint optimization of both FEC and playout. We have implemented our method in ns2 and made it available for public use. Our control method uses a channel model for both delay and loss that is fairly simplistic. Further work will address whether there is any benefit in using more sophisticated models.

References

[1] C. S. Perkins, O. Hodson, and V. Hardman, “A survey of packet-loss recovery techniques for streaming audio,” September/October 1998.

[4] It is important to notice that, in the simulations presented in Appendix B.2, since the propagation delay is smaller, the losses experienced by the audio sources (see Fig. 9) are higher than the ones presented in Fig. 4 (in order to keep a fair share of the bottleneck bandwidth - according to the TCP-Friendly principle).



(a) (b)

(c) (d)

Figure 6: *Comparison of play first and play best strategies: both strategies achieve equivalent performances.*

[2] J-C. Bolot, S. Fosse Parisis, and D. Towsley, “Adaptive FEC-based error control for internet telephony,” in *Infocom’99*, March 1999.

[3] T. J. Kostas, M. S. Borella, I. Sidhu, G. M. Schuster, J. Grabiec, and J. Mahler, “Real-time voice over packet-switched networks,” *IEEE Network*, pp. 18–27, January/February 1998.

[4] C. Boutremans and J.-Y. Le Boudec, “Adaptive delay aware error control for internet telephony,” in *2nd IP-Telephony workshop*, April 2001, pp. 81–92, Columbia University, New York.

[5] “Robust audio tool,” http://www-mice.cs.ucl.ac.uk/multimedia/software/rat/.

[6] “Freephone,” http:www-sop.inria.fr/rodeo/index.html.

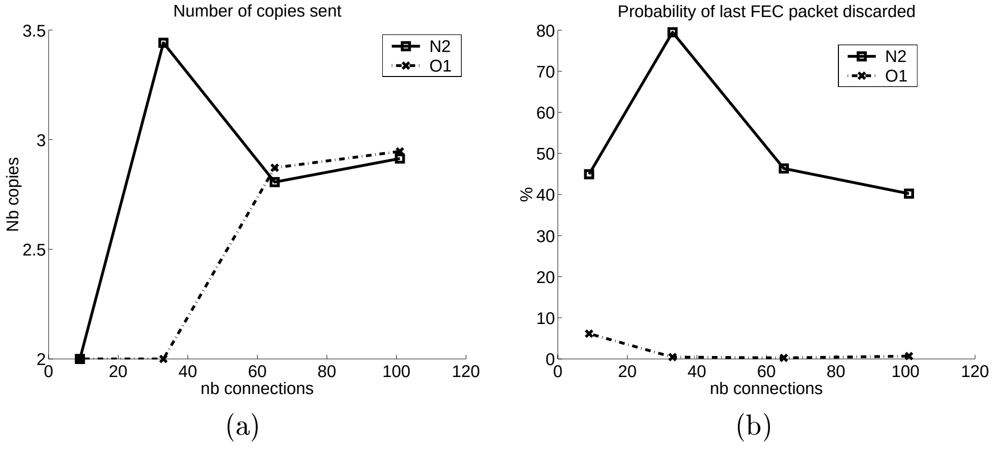


Figure 7: *The optimal solution (N2) does not always wait for all FEC packets to arrive.*

[7] J. Rosenberg, L. Qiu, and H. Schulzrinne, "Integrating packet FEC into adaptive voice playout buffer algorithms on the internet," in *Proc. IEEE Infocom'2000*, March 2000.

[8] M. Podolsky, C. Romer, and S. McCanne, "Simulation of fec-based error control for packet audio in the internet," in *IEEE Infocom'98*, April 1998.

[9] J-C. Bolot and A. Vega Garcia, "Control mechanisms for packet audio in the internet," in *IEEE Infocom'96*, March 1996.

[10] "Network simulator," Available from http:www-mash.cs.berkeley.edu.ns.

[11] V. Hardman, A. Sasse, M. Handley, and A. Watson, "Reliable audio for use over the internet," in *Proceedings of INET'95*, June 1995.

[12] H. Sanneck, "Concealment of lost speech packets using adaptive packetization," in *Proceedings IEEE Multimedia Systems 1998*, Austin, TX, June 1998.

[13] "Coding of speech at 8 kbit/s using conjugate-structure algebraic-code-excited linear-prediction (cs-acelp)," March 1996.

[14] H. Sanneck and N. Le, "Speech property-based FEC for Internet Telephony applications," in *Proceedings of the SPIE/ACM SIGMM Multimedia Computing and Networking Conference (MMCN)*, San Jose, CA, January 2000, pp. 38–51, ftp://ftp.fokus.gmd.de/pub/glone/papers/Sann0001:Speech-FEC.ps.gz.

[15] R. Blahut, *Theory and Practice of Error control codes*, Addison-Wesley, 1993.

[16] N. Shacham and P. Mc Kenney, "Packet recovery in high-speed networks using coding and buffer management," in *Proc. IEEE Infocom'1990*, June 1990, vol. 1, pp. 124–131.

[17] D. R. Figuereido and E. de Souza e Silva, "Efficient mechanisms for recovering voice packets in the internet," in *Proc. IEEE Globecom'99*, 1999, pp. 1830–1836.

[18] J-C. Bolot and A. Vega Garcia, "The case for FEC-based error control for packet audio in the internet," in *to appear in ACM Multimedia Systems.*

[19] V. Paxson, "End-to-end internet packet dynamics," in *Proc. ACM Sigcomm'97*, september 1997, Cannes, France.

[20] J.-C. Bolot, "End-to-end packet delay and loss behavior in the internet," in *Proc. ACM Sigcomm'93*, August 1993, pp. 189–199, San Francisco, CA.

[21] M. Yajnik, S. Moon, J. Kurose, and D. Towsley, "Measurement and modelling of temporal dependence in packet loss," in *Proc. IEEE Infocom'99*, March 1999, New York, NY.

[22] R. Ramjee, J. Kurose, D. Towsley, and H. Schulzrinne, "Adaptive playout mechanisms for packetized audio applications in wide-area networks," in *Proc. IEEE Infocom'94*, 1994.

[23] S. B. Moon, J. Kurose, and D. Towsley, "Packet audio playout delay adjustment: Performance bounds and algorithms," *ACM/Springer Multimedia Systems*, vol. 5, pp. 17–28, Januray 1998.

[24] J. Mahdavi and S. Floyd, "TCP-Friendly unicast rate-based flow control," in *Draft posted on end2end mailing list*, Jannuary 1997, http://www.psc.edu/networking/papers/tcp_friendly.html.

[25] I. Rhee, V. Ozdemir, and Y. Yi, "TEAR: TCP emulation at receivers - flow control for multimedia streaming," Technical report, Department of Computer Science, NCSU, April 2000.

[26] R. Rejaie, M. Handley, and D. Estrin, "RAP: An end-to-end rate-based congestion control mechanism for realtime streams in the internet," in *Proc. IEEE Infocom'99*, March 1999, New York.

[27] D. Sisalem and H. Schulzrinne, "The loss-delay adjustment algorithm: A TCP-Friendly adaptation scheme," in *NOSSDAV'98*, July 1998, Cambridge,UK.

[28] Sally Floyd, Mark Handley, Jitendra Padhye, and Joerg Widmer, "Equation-based congestion control for unicast applications," in *SIGCOMM'2000*, August 2000.

[29] W. Tan and A. Zakhor, "Error resilient packet video for the internet," in *ICIP'98*, October 1998, vol. 3, pp. 458–462.

[30] J-C. Bolot and T. Turletti, "Experience with rate control mechanisms for packet video in the internet," *ACM Computer Communication Review*, vol. 28, no. 21, January 1998.

[31] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: A transport protocol for real-time applications," RFC 1889, 1996.

[32] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose, "Modeling TCP throughput: A simple model and its empirical validation," in *Proc. SIGCOMM'98*, 1998.

[33] D. de Vleeeschauwer, J. Janssen, G. H. Petit, and F. Poppe, "Quality bounds for packetized voice transport," *Alcatel Telecom Review*, pp. 19–24, January 2000.

[34] "The e-model, a computational model for use in transmission plannning," ITU-T Recommendation G.107, May 2000.

[35] "Speech processing, transmission and quality aspects (STQ); overall transmission plan aspects for telephony in private networks," ETSI Guide 201 050, February 1999.

[36] "Provisional planning values for equipment impairment factor Ie," Appendix to ITU-T Recommendation G.113, February 2001.

[37] W. Jiang and H. Schulzrinne, "Comparison and optimization of packet loss repair methods on voip perceived quality under bursty loss," in *NOSSDAV'02*, May 2002, pp. 73–81.

[38] Athina Markopoulou, Fouad Tobagi, and Mansour Karam, "Assessment of VoIP quality over internet backbones," in *to appear in Proc. Infocom'02*, June 2002.

[39] V. A. Vaishampayan, "Design of multiple description scalar quantizers," *IEEE Transactions on Information Theory*, vol. 39, no. 3, pp. 821–834, May 1993.

[40] R.G. Cole and J.H. Rosenbluth, "Voice over ip performance monitoring," *Computer Communication Review*, vol. 31, no. 2, pp. 9–24, April 2001.

[41] R. V. Cox and P. Kroon, "Low bit-rate speech coders for multimedia communication," *IEEE Communications Magazine*, pp. 34–41, December 1996.

[42] R. Rajkumar, C. Lee, J. Lehoczky, and D. Siewiorek, “A resource allocation model for qos management,” in *IEEE Real-Time Systems Symposium*, December 1997.

[43] Pascal Frossard, “Fec performance in multimedia streaming,” *IEEE Communications Letters*, vol. 5, no. 3, pp. 122–124, March 2001.

A Reed-Solomon FEC: computation of PLR_{FEC}

This section gives the detailed computation of the residual packet loss rate after reconstruction when (1) a Reed-Solomon code (n, k) is used, (2) packets are sent over a channel that complies to the assumptions made in Section 4.1 and (3) the playout delay is D.

$$PLR_{FEC} = \frac{1}{k}\sum_{i=1}^{k}\overline{P}(i) \tag{16}$$

where $\overline{P}(i)$ is the probability that the ith packet in the block is lost for the application i.e. that the ith packet is either (1) dropped in the network and could not be reconstructed or (2) received after its playout time and could not be reconstructed:

$$\begin{aligned}
\overline{P}(i) &= Pr(\{Y_i = 1\} \cap \{\text{packet } i \text{ can not be reconstructed }\}) \\
&\quad + Pr(\{\{Y_i = 0\} \cap \{D_i > D\}\} \bigcap \{\text{packet } i \text{ can not be reconstructed }\}) \\
&= Pr(\{\text{packet } i \text{ can not be reconstructed}\} | \{Y_i = 1\}) \times Pr(\{Y_i = 1\}) \\
&\quad + Pr(\{\text{packet } i \text{ can not be reconstructed}\} | \{\{Y_i = 0\} \cap \{D_i > D\}\}) \\
&\quad \times Pr(\{\{Y_i = 0\} \cap \{D_i > D\}\}) \\
&= (1 - Pr(\{\text{packet } i \text{ can be reconstructed}\} | \{Y_i = 1\})) \times Pr(\{Y_i = 1\}) \\
&\quad + (1 - Pr(\{\text{packet } i \text{ can be reconstructed}\} | \{\{Y_i = 0\} \cap \{D_i > D\}\})) \\
&\quad \times Pr(\{\{Y_i = 0\} \cap \{D_i > D\}\})
\end{aligned} \tag{17}$$

With a Reed-Solomon code (n, k), any missing (dropped or received late) packet in a block can be reconstructed if and only if at least k packets among the n packets in this block are received before its playout time:

$$\begin{gathered}
\{\text{packet } i \text{ can be reconstructed}\} \\
\Longleftrightarrow \\
\{\exists\ k'\ (k' \geq k) \text{ packets in the block } \{1, \ldots, n\} | \{\{Y_j = 0\} \cap \{D_j \leq D + (i-j)T\}\,,\, j \in \{1, \ldots, n\}\}\}
\end{gathered}$$

From Property 1, if packet i $(i \leq k)$ arrives after its playout time, then packets j $(i < j \leq n)$ will also arrive after i's playout time:

$$Pr(\{D_j \leq D + (i-j)T\} | D_i > D) = 0 \quad \forall j\,,\, i < j \leq n$$

This implies that, if packet i ($i \leq k$) is received after its playout time, it will not be reconstructed, since the following packets will also arrive after its playout time and we need k packets to be received before i's playout time in order to reconstruct it:

$$Pr(\{\text{packet } i \text{ can be reconstructed}\} | \{\{Y_i = 0\} \cap \{D_i > D\}\})) = 0$$

Hence, Equation (17) can be re-written:

$$\begin{aligned}
\overline{P}(i) &= Pr(\{Y_i = 1\}) \times (1 - Pr(\{\text{packet } i \text{ can be reconstructed}\} | \{Y_i = 1\})) \\
&\quad + Pr(\{\{Y_i = 0\} \cap \{D_i > D\}\}) \\
&= \frac{p}{p+q}(1 - P_{REC}(i)) + \frac{q}{p+q}(1 - F_D(D)) \qquad (18)
\end{aligned}$$

where $P_{REC}(i)$ is the probability to reconstruct the ith packet in the block, given that it was lost in the network. $P_{REC}(i)$ is expressed as follows:

$$\begin{aligned}
P_{REC}(i) &= Pr(\{\exists\ k'\ (k' \geq k) \text{ packets in the block } \{1, \ldots, n\} \text{ such that} \\
&\quad \{Y_j = 0\} \cap \{D_j \leq D + (i-j)T\}, j \in \{1 \ldots n\}\} | \{Y_i = 1\}) \qquad (19)
\end{aligned}$$

We compute $P_{REC}(i)$ conditionally to the events $\{A_j, j = 0, \ldots, n\}$ on the delays of packets in the block:

$$\begin{cases}
A_0 = \{d_1 > D - (1-i)T\} \\
A_j = \{\{d_j \leq D - (j-i)T\} \cap \{d_{j+1} > D - (j+1-i)T\}\} \quad \text{for } j = 1 \ldots n-1 \\
A_n = \{d_n \leq D - (n-i)T\}
\end{cases}$$

where d_j is the delay experienced by the jth packet in the block. From Properties 1 and 2, $P(\bigcup_{i=0}^{n} A_j) = 1$ and $A_i \cap A_j = \emptyset$ for $i \neq j$. Hence, from the Total Probability Theorem, $P_{REC}(i)$ can be computed as follows:

$$\begin{aligned}
P_{REC}(i) &= \sum_{g=0}^{n-1} P_{REC}(i|A_g) \times P(A_g) + P_{REC}(i|A_n) \times P(A_n) \\
&= \sum_{g=k+1}^{n-1} P_{REC}(i|A_g) \times P(A_g) + P_{REC}(i|A_n) \times P(A_n) \\
&\quad (\text{since } P_{REC}(i|A_g) = 0\,, \forall g \leq k) \\
&= \sum_{g=k+1}^{n-1} P_{REC}(i|A_g)[F_D(D - (g-i)T) - F_D(D - (g+1-i)T)] \\
&\quad + P_{REC}(i|A_n) F_D(D - (n-i)T)
\end{aligned}$$

$P_{REC}(i|A_g)$, $(g = k+1, \dots, n)$ is the probability to receive at least k packets in the block, given that we can wait until we have received the gth packet in the block and that we lost the ith packet, namely, it is the probability to receive at least k packets among the g first packets in a block, given that we lost the ith packet. We will denote this probability by $P_{PAR}(k, g, i)$.

$$\begin{aligned} P_{REC}(i|A_g)) &= Pr(\{\exists\ k'\ (k' \geq k) \text{ packets in the set } \{1, \dots, g\} \text{ such that} \\ &\quad \{\{Y_j = 0\}, j \in \{1 \dots g\}\}|\{Y_i = 1\}) \\ &= P_{PAR}(k, g, i) \end{aligned}$$

$P_{PAR}(k, g, i)$ is the probability to loose between 1 and $g - k$ packets in $\{1, \dots, g\}$, $(g \geq k+1)$, given that the ith packet is lost.

Now, let $p_R(i)$ denote the probability to receive exactly $i - 1$ packets until the next packet loss, conditionally to the fact the we just lost packet: $p_R(i) = Pr(\{Y_j = 0, \forall j \in \{1, \dots, i-1\}\} \cap \{Y_i = 1\}|\{Y_0 = 1\})$. Similarly, let $P_R(i)$ denote the probability that at least $i-1$ packets will be received correctly until the next packet loss, conditionally to the fact that we just lost a packet: $P_R(i) = Pr(\{Y_j = 0, \forall j \in \{1, \dots, i-1\}\}|\{Y_0 = 1\})$. For a Gilbert loss process, these probabilities are expressed as follows:

$$p_R(i) = \begin{cases} 1 - q & \text{if } i = 1 \\ q(1-p)^{(i-2)}p & \text{otherwise} \end{cases}$$

and

$$P_R(i) = \begin{cases} 1 & \text{if } i = 1 \\ q(1-p)^{(i-2)} & \text{otherwise} \end{cases}$$

Using these probabilities, the probability $R(m, n)$ that $m - 1$ packets are lost in the next $n - 1$ packets following a packet loss (conditionally to the fact that we just lost a packet) can easily be computed by recurrence [43]:

$$R(m, n) = \begin{cases} P_R(n) & \text{for } m = 1 \text{ and } n \geq 1 \\ \displaystyle\sum_{i=1}^{n-m+1} p_r(i) R(m-1, n-i) & \text{for } 2 \leq m \leq n \end{cases}$$

Similarly, we can define the 'backwards' probabilities $\tilde{p}_R(i)$ to be the probability to have received exactly $i-1$ packets since the last packet loss, conditionally to the fact the we just lost packet: $\tilde{p}_R(i) = Pr(\{Y_{-j} = 0, \forall j \in \{1, \dots, i-1\}\} \cap \{Y_{-i} = 1\}|\{Y_0 = 1\})$; $\tilde{P}_R(i)$ to be the probability that at least $i-1$ packets were received correctly since the last packet loss, conditionally to the fact that we just lost a packet: $\tilde{P}_R(i) = Pr(\{Y_{-j} =$

$0, \forall j \in \{1, \ldots, i-1\}\}|\{Y_0 = 1\})$ and $\tilde{R}(m,n)$ to be the probability that $m-1$ packets are lost in the last $n-1$ packets preceeding a packet loss (conditionally to the fact that we just lost a packet). The reversibility of the Gilbert loss process allows to write: $\tilde{p}_R(i) = p_R(i)$, $\tilde{P}_R(i) = P_R(i)$ and $\tilde{R}(m,n) = R(m,n)$.

The probability $P_{PAR}(k,g,i)$ to loose between 1 and $g-k$ packets in $\{1,\ldots,g\}$, $(g \geq k+1)$, conditionally to the fact that the ith packet is lost is now easy to compute:

$$\begin{aligned} P_{PAR}(k,g,i) &= \sum_{l=1}^{g-k} \sum_{m=0}^{\min(l-1,i-1)} \tilde{R}(m+1,i)R(l-m,g-i+1) \\ &= \sum_{l=1}^{g-k} \sum_{m=0}^{\min(l-1,i-1)} R(m+1,i)R(l-m,g-i+1) \qquad (20) \end{aligned}$$

B Further Simulation Results

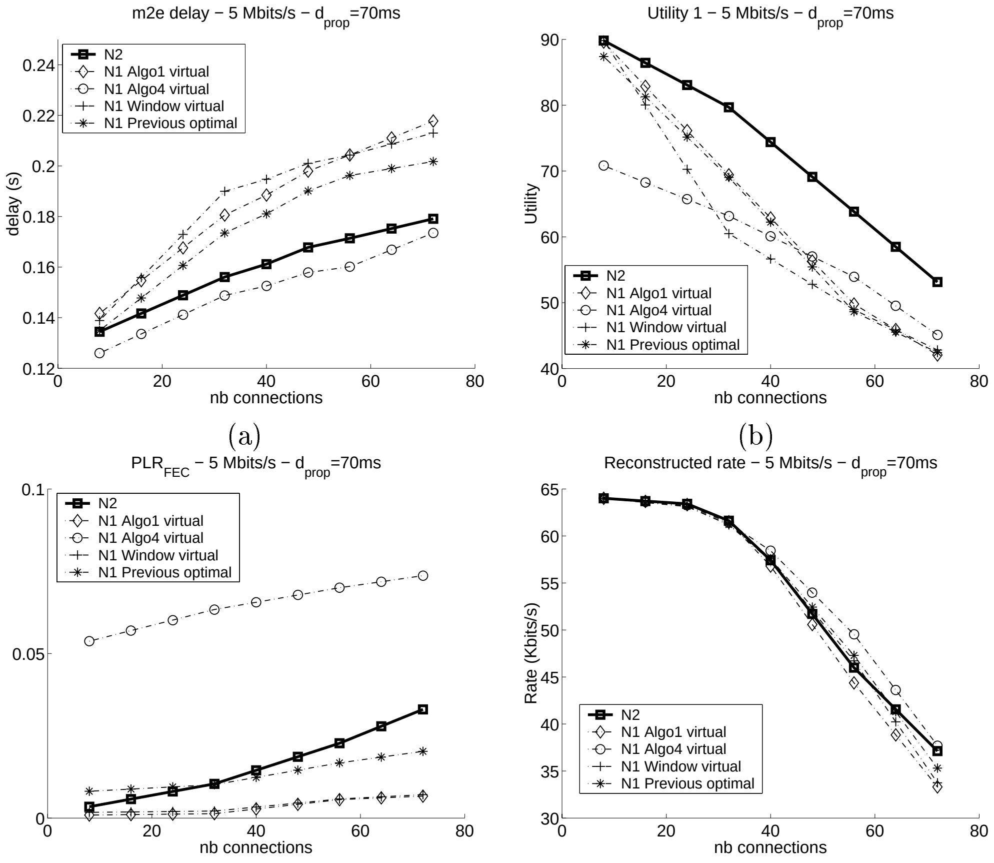
B.1 Comparison between Different Playout Algorithms

Figures 8 (a) to (d) compare the performances obtained with method N2 to the ones obtained with method N1 using different playout adjustment algorithms at the receiver and with Signal Processing FEC. Utility f_1 is used and the channel characteristics are the same as shown in Fig. 4. We compare the *virtual* [7] versions of Algorithms 1 and 4[5] from [22], to the *virtual* version of the algorithm proposed in [23] (that we call 'Window' in Fig. 8) and to the *previous optimal* algorithm proposed in [7]. Figure 8 (a) shows that the delays obtained with the virtual Algorithm 1, the previous optimal and the virtual 'window' algorithm are quite close to each other. Figure 8 (b) shows that the virtual version of Algorithm 1 gives good results in a wide range of network conditions. Even though the virtual version of Algorithm 4 had a very small playout delay, it got a very low utility compared to other algorithms (see Fig. 8 (b)). This is mainly due to the poor performances of this algorithm regarding the packet loss rate after reconstruction. Indeed, this algorithm leads to higher loss probabilities than other algorithms because it attempts to track the network delays too closely and it looses a lot of packets whenever the delay estimate is small (our results concerning Algorithm 4 confirm the results obtained in [7]).

B.2 Case of Small End-to-End Delays

Figures 9 (a), (b) and (c) show respectively the TCP-Friendly rate constraint R_{max}, the Gilbert parameter p and the Gilbert parameter q as a function of the number of connections

[5] We omit results obtained with algorithms 2 and 3, since they do not outperform the other algorithms.



(a)

(b)

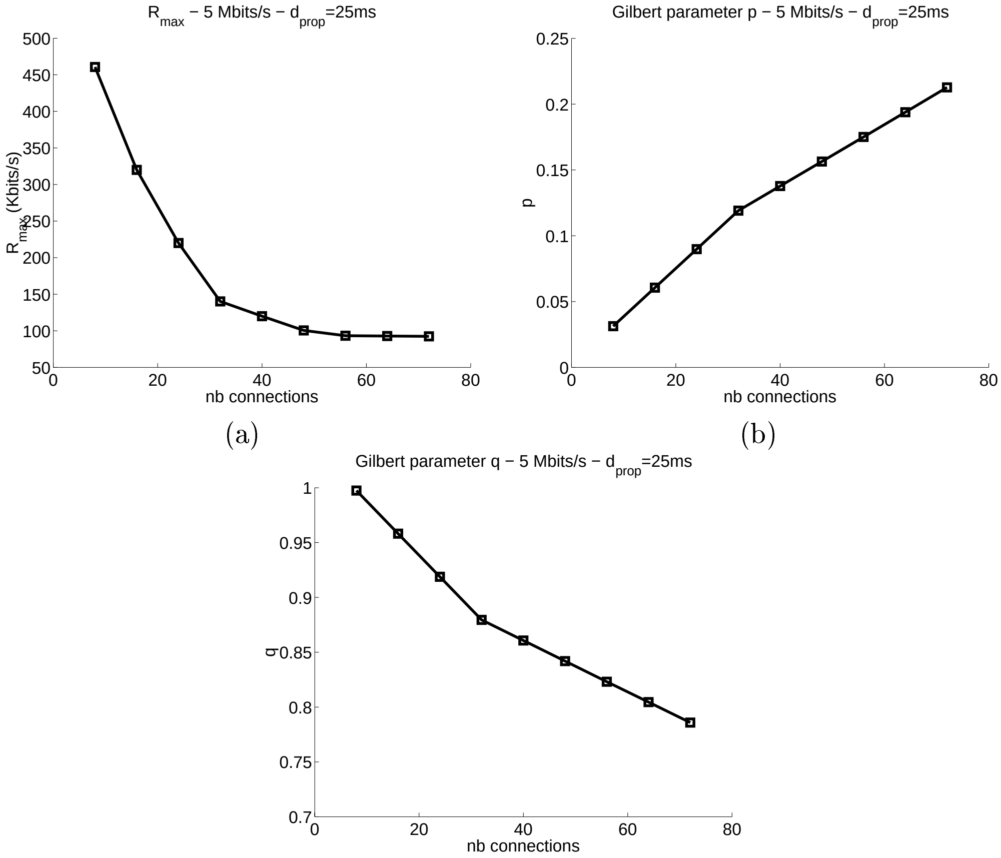
(c)

(d)

Figure 8: *Performances of the different methods N2 and N1 using different playout adjustment algorithms: (a) Mouth-to-ear delay, (b) utility measured at the destination, (c) residual packet loss rate and (d) reconstructed rate of audio for SP FEC with utlity* f_1

sharing the link for a bottleneck bandwidth of 5Mbits/s and a one-way propagation delay (without queuing) of 25 ms.

Figure 10 shows the performances (in terms -from left to right- of mouth-to-ear delay, utility measured at the receiver, residual packet loss rate and rate of the reconstructed audio stream at the receiver) for the different methods N1, N2 (Play First), O1 and O2 as a function of the number of connections sharing the link. Figures 10 (a) to (d), (e) to (h) and (i) to (l) correspond respectively to the results obtained with utility functions f_1, f_2 and f_3 with a Signal Processing FEC coding scheme. Figures 10 (m) to (p) were obtained with utility function f_1 and a Reed-Solomon error coding. Algorithm 1 in [22] was used to adjust the playout delay with method O1 and its virtual version was used with N1 and O2. For clarity of the figure, we only show the best curves obtained with method O2. These curves were obtained with $K = 3$ in the case of Signal Processing FEC and with $(k, n) = (1, 3)$ and $(k, n) = (2, 3)$ in the case of Reed-Solomon FEC.

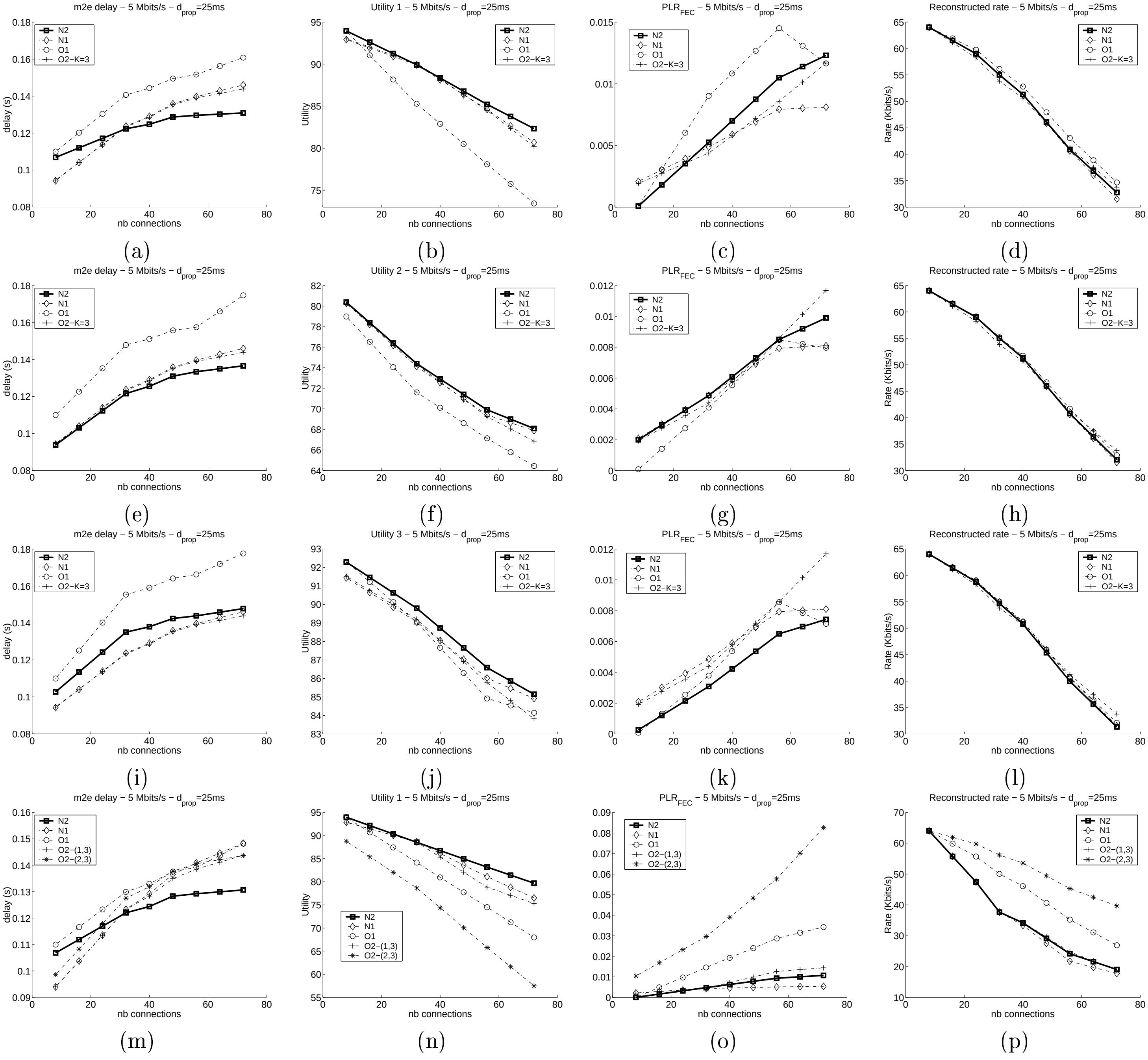


(a)

(b)

(c)

Figure 9: *a) TCP-Friendly rate constraint. (b) Gilbert parameter p. (c) Gilbert Parameter q as a function of the number of connections sharing the link.*



(a) (b) (c) (d)

(e) (f) (g) (h)

(i) (j) (k) (l)

(m) (n) (o) (p)

Figure 10: *Performances of the different methods N1, N2, O1 and O2 (with different FEC parameter settings). Mouth-to-ear delay, utility measured at the destination, residual packet loss rate and reconstructed rate of audio for SP FEC with utlity f_1 from (a) to (d), f_2 from (e) to (h), f_3 from (i) to (l) and for Reed-Solomon with utility f_1 from (m) to (p).*