TCP-Friendly Transmission of Voice over IP

F. BERITELLI, G. RUGGERI, G. SCHEMBRA
Dipartimento di Ingegneria Informatica e delle Telecomunicazioni
University of Catania
Viale A. Doria 6 - 95125 Catania
E-mail: (beritelli, gruggeri, schembra)@diit.unict.it

Abstract. In the last few years an increasing amount of attention has been paid to technologies for the transmission of voice over IP (VoIP). At present, the UDP transport protocol is used to provide this service. However, when the same bottleneck link is shared with TCP flows, and in the presence of a high network load and congestion, UDP sources capture most of the bandwidth, strongly penalizing TCP sources. To solve this problem some congestion control should be introduced for UDP traffic as well, in such a way that this traffic becomes "TCP-friendly". In this perspective, several TCP-friendly algorithms have been proposed in the literature. Among them, the most promising candidates for the immediate future are RAP and TFRC. However, although these algorithms were introduced to support real-time applications on the Internet, up to now the only target in optimizing them has been that of achieving fairness with TCP flows in the network. No attention has been paid to the applications using them, and in particular, to the quality of service (QoS) perceived by their users. The target of this paper is to analyze the problem of transmitting voice over IP when voice sources use one of the above-mentioned TCP-friendly algorithms. With this aim, a VoIP system architecture is introduced and the characteristics of each its elements are discussed. To optimize the system, a multirate voice encoder is used to be feasible to work over a TCP layer, and a modification of both RAP and TFRC is proposed. Finally, in order to analyze the performance of the proposed system architecture and to compare the modified RAP and TFRC with the original algorithms, the sources have been modeled with an arrival process modulated by a Markov chain, and the model has been used to generate traffic in a simulation study performed with the ns-2 network simulator.

1 INTRODUCTION

The spread of the Internet in the last few years and the ever-growing amount of available transmission bandwidth are changing the Internet application scenario. Whereas up to a few years ago the most common applications were only TCP-based, like e-mail, FTP and WEB, recently real-time applications have become more and more popular. In this context, an increasing amount of attention has been paid to technologies for the transmission of voice. Various network operators have already started to offer Voice over IP (VoIP) services and in the next few years most traditional public telephone services, currently based on circuit switched (CS) networks, are likely to migrate towards packet switching (PS) technologies. The TCP transport protocol introduced by Jacobson in [1] is not suitable for these applications because of its complex retransmission mechanism, and the excessive burstiness of the traffic transmitted. For this reason, all VoIP applications at present transmit via the UDP transport protocol. This evolution is affecting the telecommunications scenario of the past [2], which featured traffic stability and fairness. In fact, when the same bottleneck link is shared between TCP and UDP flows, in the presence of a high network load and congestion, TCP sources reduce their output traffic due to their intrinsic window-based congestion control mechanism, while UDP sources continue to transmit in the same way, as if the network were underloaded. In this way, UDP sources capture most of the bandwidth, strongly penalizing TCP sources. To solve this problem some congestion control should be introduced for UDP traffic as well, in such a way that this traffic becomes "TCP-friendly": it has to receive the same average bandwidth over the timescale of a session as a TCP flow along the same path affected by the same delay and packet loss conditions [3]. In other words, the target of TCP-friendly rate control mechanisms is to make UDP traffic sources behave like "good citizens" towards TCP sources. The application of TCP-friendly algorithms by real-time traffic has two important advantages: it provides TCP-friendliness, and gives to the source a mechanism to learn by itself the available bandwidth in the network, in such a way that it can adapt its rate to the current situation. As far as the first issue is concerned, let us note that following the TCP dynamics on short-term time scales
may be deleterious for real-time sources, because of the short-term transmission oscillations of TCP sources, and frequent loss occurrences. The TCP-friendliness concept can therefore be relaxed, capturing it on medium-term time scales only. The second issue is very relevant and constitutes the most important advantage of applying these algorithms in real-time transmission on the Internet, especially in the current best-effort scenario. In this perspective, several TCP-friendly algorithms have been proposed in the literature (see the TCP-friendly WEB site [3] for a list of related works). One of the best known algorithms is the rate adaptation protocol (RAP) [4]. RAP adopts an additive increasing multiplicative decreasing (AIMD) algorithm for rate adaptation to obtain TCP friendliness. It performs loss-based rate control and does not need any explicit congestion signal from the network since packet loss is considered to be the only feasible implicit feedback signal on the Internet due to the presence of TCP traffic. Another very important TCP-friendly algorithm proposed in the literature is TCP-Friendly Rate Control (TFRC) [5]. It is an equation-based congestion control since it uses a control equation that explicitly gives the maximum acceptable sending rate as a function of the recent loss event rate. The sender adapts its sending rate, guided by this control equation, in response to feedback from the receiver. More specifically, in order to be fair to TCP sources which share the same bottleneck link, TFRC uses the TCP response function characterizing the steady-state sending rate of TCP as a function of the round-trip time and steady-state loss rate. The problem dealt with in this paper arises from the fact that although the above algorithms were introduced to support real-time applications on the Internet, the only target in optimizing them was to achieve fairness towards TCP flows in the network. No attention has been paid to the applications using them, and in particular, to the quality of service (QoS) perceived by their users. The target of this paper is to analyze the problem of transmitting voice over IP when voice sources use one of the above-mentioned TCP-friendly rate-control algorithms. With this aim, a VoIP system architecture is introduced and the characteristics of each of its elements are discussed. More specifically, the following three issues are considered:

- the TCP-friendly algorithm;
- the voice encoding technique;

As regards the first issue, both the RAP and the TFRC have been considered, and a modification has been introduced to reduce the high percentage of losses which is intrinsic in the original specifications. As far as the second issue is concerned, after an overview of the most common standardized encoding techniques for voice transmission over IP, it is concluded that the most suitable one in a TCP-friendly scenario is a multirate encoding technique. Finally, regarding the third issue, an encoder controller device has been introduced into the voice transmission system to select the encoding modes in the encoder according to the bandwidth indications provided by the TCP-friendly algorithm. In order to analyze the performance of the proposed system architecture and to compare the modified RAP and TFRC with the original algorithms, the sources have been modeled with an emission process modulated by a Markov chain, and the model has been used to generate traffic in a simulation study performed with the ns-2 network simulator [6].

The paper is structured as follows. Section 2 provides a general overview of the most important aspects of rate-control algorithms and a brief description of both the TCP-friendly algorithms considered, i.e. RAP and TFRC. Section 3 analyzes the problem of voice encoding for voice transmission over IP, taking into account four VBR speech coding types: ON-OFF, multimode, multirate and scalable. Section 4 describes the voice transmission system, analyzing each of its components. Section 5 analyzes the performance of the voice transmission system, comparing the TCP-friendly algorithms in terms of the subjective quality of the encoding resulting from the available bandwidth variations, loss probability and end-to-end delay. Finally, Section 6 concludes the paper.

2 TCP-FRIENDLY RATE CONTROL MECHANISMS

In this paper we will consider the two most important TCP-friendly rate control mechanisms defined in the literature: the rate adaptation protocol (RAP) and TCP Friendly Rate Control (TFRC). For the sake of completeness, in Sections 2.1 and 2.2 we will give a brief description of them.

2.1 RAP

The RAP protocol is located over the UDP layer in the UDP/IP protocol stack, with the objective of controlling the output rate of the source using it in order for this source to receive the same bandwidth as all the TCP sources sharing the same bottleneck links. For a more detailed description the reader is referred to [4].

The RAP protocol is mainly implemented at the source. The destination only acknowledges each packet, providing end-to-end feedback about the state of congestion of the network. Using the feedback, the RAP source implements a rate-adaptation mechanism, which varies the output rate with the objective of being fair to the TCP sources present in the network. According to [7], a rate-adaptation mechanism is characterized by three functions: the decision function, the increase/decrease algorithm, and the decision frequency. The decision function consists of detecting a congestion or a non-congestion state, in order to increase or decrease the bit rate. Like the congestion avoidance algorithm in TCP
[1], the RAP source searches for available bandwidth on the bottleneck link by periodically increasing its transmission rate. It decreases its rate, on the other hand, when congestion is detected. Like TCP, RAP considers timeouts and gaps in the sequence space as congestion signals, and maintains an estimate of the round-trip time (RTT), called SRTT. Moreover, like TCP, it uses the Jacobson/Karel algorithm to calculate the timeout as follows:

$$\text{Timeout}_{n+1} = \mu \cdot SRTT_{n+1} + \phi \cdot S_{n+1}$$ (1)

where:
- $SRTT_{n+1}$ is the new estimated RTT, which is calculated by a weighted averaging of the previous estimated RTT, $SRTT_n$, and the new sampled RTT, $RTT_{n+1}$, as follows:

$$SRTT_{n+1} = \Delta \cdot RTT_{n+1} + (1 - \Delta) \cdot SRTT_n$$ (2)

- $\sigma_{n+1}$ is the new estimated variance of the RTT, which is calculated as follows:

$$\sigma_{n+1} = \Delta \cdot |RTT_{n+1} - SRTT_n| + (1 - \Delta) \cdot \sigma_n$$ (3)

- $\Delta$, $\mu$, and $\phi$ are suitable weights values.

Unlike TCP, RAP is not ack-clocked given that a RAP source may send several packets before receiving any new ACK to update the RTT estimate. The RAP source maintains a transmission history, made up of one record for each transmitted packet not yet acked. Each record contains the sequence number, the departure time, the transmission rate and a status flag of the relating packet. Before sending a new packet, the source checks for a potential timeout among the packets in the history using the updated value of the SRTT estimate, obtained as in (2). In this way the RAP source considers all expired packets to be lost. The RAP protocol, like TCP, uses another mechanism in parallel to detect congestion in the network, called three duplicated acks: if the RAP source receives an ACK that implies delivery of three packets after a missing one, the packet is considered as lost.

The increase/decrease algorithm is additive increasing, multiplicative decreasing (AIMD). Three kinds of events determining rate variation are envisaged:

- an ack arrival event, which can determine a rate decrease;
- a packet transmission event, which can determine a rate decrease;
- RTT timer expiry, which can determine a rate increase.

More specifically the transmission rate, expressed in byte/s, is increased in a step-like fashion:

$$R_{n+1} = R_n + \alpha$$ (4)

where $\alpha$ is the step height. In the presence of congestion, on the contrary, the transmission rate is decreased in a multiplicative fashion:

$$R_{n+1} = R_n \cdot \beta, \quad 0 < \beta < 1$$ (5)

where $\beta$ is the decreasing factor. Both transmission rate variations are obtained by varying the inter-packet gap (IPG), which is linked to the transmission rate as follows:

$$IPG_n = \frac{s}{R_n}$$ (6)

where $s$ is the packet size expressed in bytes. The values of $\alpha$ and $\beta$ proposed in [4] are:

$$\alpha = \frac{s}{SRTT}$$

$$\beta = 0.5$$ (7)

where SRTT is the last estimated RTT.

Finally, the decision frequency specifies how often to change the rate. In [3] it is suggested that in ack-based schemes the rate be adjusted not more than once per RTT, because changing the rate too often results in oscillation, whereas infrequent changes of rate lead to unresponsive behavior. For these reasons RAP adjusts the rate once every round-trip time. The time between two subsequent adjusting points is called a step. Given the random nature of the RTT signal [8], using the most recent RTT sample as the step length may determine inefficiencies. Therefore for the step length RAP uses the most recent SRTT value, which is the smoothed version of RTT.

### 2.2 TFRC

The TFRC protocol, like the RAP protocol, is located over the UDP layer and has the same target of controlling the output rate of a UDP source in order for this source to receive the same bandwidth as all the TCP sources sharing the same bottleneck link. For a more detailed description the reader is referred to [5]. The main difference as compared with AIMD TCP-friendly protocols like RAP is that TFRC uses a control equation calculating the maximum bit rate which can be used by the source as a function of the loss rate and the RTT. For this reason it is an equation-based TCP-friendly protocol, and is not so drastic as AIMD protocols when network congestion occurs. TFRC therefore presents less responsiveness to congestion situations than RAP, but has a smoother output rate.

The TFRC protocol can operate in the following four ways, as described in Fig. 1:

- **Init RTT mode**: to determine the initial RTT, packets are sent at a very low bit rate until the receiver report arrives and the RTT can be initialized. The mode is switched to **slow start**.
- **Slow start mode**: the sending rate is doubled every RTT to determine the initial sending rate for the congestion control mode. The sender terminates **slow**
The TFRC receiver calculates the loss rate, bandwidth variations and output rate stability. The TFRC receiver calculates the loss rate, considering a loss event to be not only the loss of one packet, but also the loss of a cluster of packets in a time interval equal to the round-trip time. The loss rate is the inverse of the average loss interval. The average loss interval, $\hat{\tau}$, is calculated at each loss occurrence, as the weighted average of the last $n$ loss intervals, $l_n, ..., l_2, l_1$, respectively sorted according to their occurrence instants:

$$\hat{\tau} = \frac{\sum_{i=1}^{n} w_i l_i}{\sum_{i=1}^{n} w_i}$$  \hspace{1cm} (10)

where $w_i$, for each $i \in 1, n$, are the weights of the averaging operation, which determine the tradeoff between output rate responsiveness and smoothness. This problem was analyzed in [5] and the values were calculated as follows:

$$w_i = \begin{cases} 1 & 1 \leq i \leq \frac{n}{2} \\ \frac{1}{i} - \frac{1}{n} & \frac{n}{2} < i \leq n \end{cases}$$  \hspace{1cm} (11)

where $n = 8$ was demonstrated to provide the best tradeoff [5].

The drawback to this method for calculating the average loss interval is that it does not calculate the value of the interval since the most recent loss as it is not between two loss events. A high value for this parameter should indicate an actual decrease in congestion events and so it must be taken into account. To do so, an auxiliary parameter is defined by averaging the last $n - 1$ events, $l_{n-1},...,l_1$, with the interval since the most recent loss, $l_0$:

$$\hat{\tau} = \frac{\sum_{i=0}^{n-1} w_{i+1} l_i}{\sum_{i=1}^{n} w_i}$$  \hspace{1cm} (12)

Finally the loss rate calculated by the receiver and sent to the TFRC sender is:

$$p = \frac{1}{\max(\hat{\tau}, \hat{\tau}')}$$  \hspace{1cm} (13)

3 VARIABLE RATE SPEECH CODING FOR VOICE OVER IP

In an adaptive IP voice transmission system there are two basic aspects closely linked to QoS over which particular care needs to be taken: choice of the type of adaptive speech coding and definition of the rate control mechanism. A Variable Bit Rate (VBR) speech coder chooses the most appropriate encoding mode from a pre-defined set. The choice can be driven either by the source or by the network [9]. In the former case the codec exploits the various features of speech, choosing the most appropriate coding model for each phonetic class. In the latter case, it is the network which imposes the output rate, and therefore determines the choice of the most suitable coding scheme,
through a pre-defined rate control mechanism, irrespective of the phonetic contents of the frame. We can distinguish between four different types of VBR speech coding [10]:

- **ON-OFF**;
- **Multimode**;
- **Multirate**;
- **Scalable**.

The **ON-OFF coding mode** is the easiest source-driven coding mode. It is realized by combining a CBR codec and a Voice Activity Detector (VAD). In this case the transmission is discontinuous, featuring talkspurt periods (ON) and periods of silence or background noise (OFF) in which the source does not transmit anything. The codec further classifies the ON and OFF classes into the relative phonetic subclasses (i.e., voiced/unvoiced (ON), stationary/transient noise (OFF)), a more efficient source-driven coding mode is obtained, called **multimode coding**, which adapts the coding model to the local signal features [11]. The **multirate coding mode**, on the other hand, is network-driven because it is constituted by a number of CBR coding schemes, each with a different bit rate; according to the network conditions, it changes the coding scheme in order to present an output rate which is never greater than the available bandwidth.

Finally, the **scalable coding mode** uses an embedded structure in which the data packet obtained by coding each single frame comprises a very low bit-rate core (e.g., 2 kbit/s) to which a series of enhancement stages, marked with low priority, are added to increase the quality of the reconstructed signal. In this way, in the event of congestion, if the network is capable of managing different levels of priority, some enhancement stage may be dropped in any network node, but the receiver is guaranteed at least the basic quality. For this reason, the **scalable coding mode** is network-driven, i.e., it is still capable of adapting to network bandwidth variations. Generally, both multirate and scalable coding techniques also adopt a VAD for silence suppression.

In [12] and [10] a comparison between the four techniques outlined above is made in order to analyze their behavior in a VoIP application scenario. More specifically, besides comparing the inherent quality assured by the four different methods in both clean and noisy conditions, the impact of the network protocol is analyzed in terms of both frame loss and the overhead introduced by packetizing. Comparing the four methods in terms of requested bandwidth, it was observed that the best technique is multirate coding, in that by suppressing silences and adapting the peak bit rate according to network requirements it reduces the network bandwidth required. The multirate technique thus has the best overall impact on the network. In addition, as regards interoperability with the most commonly used coding schemes, the transcoding complexity required is comparable with that required by most currently used low-bit rate encoding schemes, like G.729 [13]. For these reasons we will use a multirate encoder from now on.

### 4 Voice Transmission System

The proposed system is general and applicable to any network scenario, either if voice traffic shares resources with other traffic in a best-effort scenario, or if bandwidth is reserved to it, e.g., in a DiffServ scenario. In the latter case, the TCP-friendly rate control algorithm is used to allow sources to discover the available bandwidth, whereas in the former case, an additional aim is to guarantee fairness to TCP sources.

More specifically in this section we will describe the system we propose in this paper to transmit voice over IP in a TCP-friendly fashion. It is shown in Fig. 2. The sender is made up of the voice source, the voice encoder, the encoder controller, the packetizer, and the TCP-friendly rate control protocol over the UDP/IP protocol stack. The receiver has a symmetric architecture. The output of the voice source is sent to the voice encoder. Due to what was said in the previous section, a multirate voice encoder was chosen so as to be able to adapt the output rate to the bandwidth imposed by the TCP-friendly rate-control mechanism. The voice encoder emits a number of bytes for each frame interval, $d_f$, according to the encoding mode selected to encode the frame. The encoder output is sent to the packetizer which emits packets with an emission period, $d_p$, a multiple of the frame interval, $d_f$. The term $d_p$ is the so-called packetization delay. The TCP-friendly sender is the layer implementing the sender part of one of the two rate-control mechanisms described in Section 2, either RAP or TFRC.

The **Encoder controller** plays a fundamental role because it pilots the encoding modes of the multirate voice encoder according to the information received by the TCP-friendly layer. Let us note that the RAP mechanism, as defined in [4], modifies the rate by varying the inter-packet gap. The problem here is that, unlike video sources which produce several packets per frame, the number of which
is variable in time, voice sources produce one packet for each \( K \equiv d_P / d_f \) frames, and this rate is constant in time. In this case, therefore, the IPG value is constant and equal to the packetization delay, and the output rate can only be changed by varying the packet size. So the value of \( s \) in (6) and (7) cannot be the size of the packets actually sent over the network, which is variable. For this reason, we consider \( s \) as the size of a reference packet which, according to (4) and (7), represents the increment granularity and therefore influences the smoothness and stability of the output rate. Similar reasoning can be followed for the TFRC protocol. In fact, although TFRC directly provides the output rate to be used by the source, it depends on the packet size, \( s \). Likewise, the value of \( s \) is a reference value with the same meaning as it has in RAP, while the actual packet size depends on the encoding mode, \( M \), selected from the set of possible encoding modes. In both RAP and TFRC, the number of bytes emitted in the \( i \)-th frame interval during the preparation of the \( j \)-th packet is \( \tilde{s}_{F,i,j} \), and the number of bytes constituting the generic \( j \)-th packet is \( \tilde{s}_{P,j} \). We have:

\[
\tilde{s}_{F,i,j} = R(M_{j,i}) \cdot d_f
\]

(14)

\[
\tilde{s}_{P,j} = \left( \sum_{i=1}^{K} \tilde{s}_{F,i,j} \right) + H
\]

(15)

where \( H \) is the size of the combined IP and UDP headers, and \( R(M_{j,i}) \) is the encoder output rate associated with the selected encoder mode \( M_{j,i} \) used to encode the \( i \)-th frame of the \( j \)-th packet. Finally, the encoder controller decides the encoding mode, in order to respect the rate indicated by the TCP-friendly mechanism according to (4) and (5) if RAP is used, or (8) if TFRC is used.

Let \( R(M) \) be the rate associated with encoding mode \( M \), and \( R \) be the rate imposed by the TCP-friendly algorithm. Here two alternatives are possible:

- selecting the encoding mode in such a way that the relating rate never exceeds \( R \), that is:

\[
\mathcal{M} = \max \{ R(M) \leq R \}
\]

(16)

- selecting the encoding mode in such a way that the medium-term source throughput is equal to the throughput of the TCP sources sharing the same bottleneck links. This can be achieved by selecting the encoding mode which produces the rate closest to \( R \):

\[
\mathcal{M} = \min \{ R(M) - R \}
\]

(17)

Let us now discuss a problem of both the TCP-friendly protocols considered, which we will solve by proposing to modify them. Although both protocols were introduced to support transmission of real-time traffic over the Internet, the only specification target which has been considered in the literature up to now is fairness to TCP, whilst the quality of service real-time sources are provided with has been neglected. In fact, like TCP, they increase their output rate until some loss occurs, or the RTT exceeds the timeout. In both TCP-friendly protocols the loss rate is high, and may not be acceptable for voice transmission applications. The solution we propose to overcome this problem is based on taking into account not only the loss rate and estimated delay, but also the delay variation: when the delay variation exceeds a given threshold, indicating that a congestion situation is imminent, a source has to reduce its transmission rate. By so doing, congestion is detected early and losses are reduced. More specifically, let us define the delay variation function as follows:

\[
\Delta \text{SRTT}_{n+1} = \frac{\text{SRTT}_{n+1} - \text{SRTT}_n}{\text{SRTT}_n}
\]

(18)

and introduce a threshold parameter \( \xi \), representing the guard threshold of incoming congestion.

So we can define a modified version of the RAP, called RAP\(_\Delta \), and a modified version of the TFRC, called TFRC\(_\Delta \). RAP\(_\Delta \) behaves exactly as the original rate control algorithm, except that it halves the source emission rate not only after a loss or a timeout expiry, but also when \( \Delta \text{SRTT}_{n+1} > \xi \). Likewise TFRC\(_\Delta \) works as described in Section 2.2, that is, it changes the transmission rate according to (8), but in addition it halves the output rate and starts the “slow-start phase” again each time \( \Delta \text{SRTT}_{n+1} > \xi \).

5 PERFORMANCE ANALYSIS

Given that the most common approach to providing QoS for VoIP is to use a QoS architecture like DiffServ, which is able to reserve a certain amount of bandwidth for VoIP traffic, in this section we analyze the case of a network, the topology of which is shown in Fig. 4, where traffic is due to voice sources only.

More specifically let us now evaluate the performance of the voice transmission system of Fig. 2 and compare the four TCP-friendly algorithms, RAP, TFRC, RAP\(_\Delta \), and TFRC\(_\Delta \). As regards the voice encoding technique, due to the observations made in Section 3, we used one of the most significant multirate coders today, the Adaptive Multi-Rate (AMR) coder, which was recently devised by ETSI for the third-generation mobile system (UMTS) [14]. The AMR coder is an analysis-by-synthesis speech coding technique which, like ITU-T standard G.729 [13], is based on CELP coding. The AMR uses eight different bit rates for talkspurt coding (4.75 kbit/s, 5.15 kbit/s, 5.90 kbit/s, 6.70 kbit/s, 7.4 kbit/s, 7.95 kbit/s, 10.2 kbit/s, 12.2 kbit/s). Silence periods are revealed by a VAD, so the output rate is null during these periods. As a reference for comparison, we considered the CBR G.729 combined with a VAD. In this case the output rate is constant and equal to 8 kbit/s during speech periods, and null during silence periods. Due to the different applications for which the coders were designed, they use frames of different lengths: \( d_f = 10 \) ms.
for G.729 and \( d_f = 20 \) ms for the AMR.

The system was simulated using the ns-2 network simulator [6]. In order to generate voice traffic we modeled the sources with the 2-state discrete-time Markov modulated frame emission process [15] shown in Fig. 3. The slot duration was chosen as equal to the frame duration \( d_f \) of the encoder being considered. The two states correspond to the phonetic classes revealed by the VAD: ON and OFF.

The probabilities of transition between the two states, \( p \) and \( q \), are defined as follows:

\[
p = \frac{d_f}{T_{off}} \quad \text{and} \quad q = \frac{d_f}{T_{on}}
\]  

where \( T_{off} \) and \( T_{on} \) are the average duration of the OFF- and ON-periods revealed by the VAD. The probabilities \( p \) and \( q \) therefore depend on the VAD alone. The emission process is defined as follows:

- when the Markov chain is in the OFF state, it does not emit;
- when the Markov chain is in the ON state, it emits one frame of \( \tilde{s}_f \) bytes, as defined in (15), according to the encoding mode, \( M \), chosen by the encoder controller.

The above emission process is able to model both the G.729 and the AMR coder emission processes. Given that the output rate of the G.729 coder is 8 kbit/s and its frame size \( d_f = 10 \) ms, it uses a constant frame size of \( \tilde{s}_f = 80 \) bits. If the AMR coder is used, the \( \tilde{s}_f \) frame sizes are those listed in Table 2. The traces were extracted from a database containing sequences uttered by both male and female speakers, linearly quantized at 16 bits and sampled at 8 kHz. Each sequence lasts 3 minutes, and has 40% speech activity (active frames), which is on average the typical activity percentage in a telephone conversation. To assess the behavior of the various codecs when different languages are spoken, the sequences were uttered by native speakers in Italian, English, French and German. Three different signal power levels (-16, -26, -36 dB), different types of noise (Car, Office, Train, Restaurant, Street) and different signal-to-noise ratios (SNR) (0, 10, 20) were also used, giving a total of 648 minutes of speech. In this way it was possible to obtain a set of model parameters for several operating conditions.

<table>
<thead>
<tr>
<th>Languages</th>
<th>Italian, French, German, English</th>
</tr>
</thead>
<tbody>
<tr>
<td>Levels (dBovl)</td>
<td>–16, –26, –36</td>
</tr>
<tr>
<td>SNR (dB)</td>
<td>0, 10, 20</td>
</tr>
<tr>
<td>Noises</td>
<td>Car, Office, Train, Restaurant, Street</td>
</tr>
</tbody>
</table>

The ns-2 modules implementing the above voice sources are available in [16], together with the trace database and the values of \( p \) and \( q \) characterizing all the traces considered.

To simulate the encoding system we considered the simple network topology shown in Fig. 4: a single link of capacity \( e = 150 \) kbit/s loaded by \( N \) homogeneous rate-controlled voice sources. Performance was calculated as a function of the number of sources. The source operating conditions (language, level, SNR, noise) were uniformly chosen at random from those available in the database illustrated in Table 1. The queue size was set in such a way as to have a maximum delay of 60 ms. To obtain the best trade-off between the payload/overhead ratio and the packetization delay delay \( d_f = 40 \) ms [17] was chosen, which means \( K = 2 \) frames per packet for the AMR and \( K = 4 \) frames per packet for the G.729. Varying the number of sources sharing the link, the parameters most affecting the quality of the speech signal were calculated for each control mechanism: delay, loss, average transmission rate, and the channel utilization coefficient.

Starting from the CMOS (Comparison Mean Opinion Scores) values [18] of the AMR versus G.729 (see Table 2), obtained via a series of informal listening tests using the database shown in Table 1, the CMOS values for the rate-controlled AMR sources were determined. Thanks to the experience accumulated after a large number of simulations it was also decided to set \( \xi \) to 0.1 and \( \beta \) to 0.5.

In order to evaluate the inherent encoding quality obtained when the RAP and the TFRC control algorithms are used, and to compare that obtained when the original RAP and TFRC algorithms are used, Fig. 9 shows the resulting average CMOS values. The reference value refers to the G.729 encoding, which does not use any control algorithm. In this figure we can observe that the algorithms behave very similarly for both a low and a high number of sources.
Table 2: AMR encoder characteristics

<table>
<thead>
<tr>
<th>Coding Modes</th>
<th>AMR Coding Rate (kbit/s)</th>
<th>CMOS AMR vs G.729</th>
<th>Frame size (bit)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>12.2</td>
<td>0.3</td>
<td>244</td>
</tr>
<tr>
<td>2</td>
<td>10.2</td>
<td>0.2</td>
<td>204</td>
</tr>
<tr>
<td>3</td>
<td>7.95</td>
<td>0.1</td>
<td>159</td>
</tr>
<tr>
<td>4</td>
<td>7.4</td>
<td>0</td>
<td>148</td>
</tr>
<tr>
<td>5</td>
<td>6.70</td>
<td>-0.1</td>
<td>134</td>
</tr>
<tr>
<td>6</td>
<td>5.90</td>
<td>-0.2</td>
<td>118</td>
</tr>
<tr>
<td>7</td>
<td>5.15</td>
<td>-0.3</td>
<td>103</td>
</tr>
<tr>
<td>8</td>
<td>4.75</td>
<td>-0.4</td>
<td>95</td>
</tr>
</tbody>
</table>

Figure 5: Loss rate comparison varying the number of multiplexed sources

Figure 6: Average delay [ms] comparison varying the number of multiplexed sources

6 CONCLUSIONS

The paper deals with the problem of transmitting voice over IP (VoIP) in a best-effort scenario, when real-time sources apply a TCP-friendly rate control algorithm to adapt their output flow rate to the bandwidth available in the network. To this end a VoIP system architecture has been introduced, whose main elements are the TCP-friendly algorithm, the voice encoder and the encoder controller. As regards the TCP-friendly algorithm, both RAP and TFRC were considered, and a modification has been proposed to reduce the high percentage of losses which is intrinsic in the original specifications. As far as the voice encoder is concerned, a multirate encoder, demonstrated to be the most suitable in a TCP-friendly scenario was chosen. Finally, the encoder controller device has been introduced into the voice transmission system to select the encoding modes according to the bandwidth indications provided by the TCP-friendly algorithm. Through a simulation study the performance of the proposed system architecture has been evaluated, and the modified RAP and TFRC algorithms have been compared with the original ones. Numerical results have demonstrated that the TCP-friendly rate control algorithms considered behave very similarly for both a low and a high number of sources. The most important differences are for a medium number of sources, where the non-modified protocols, RAP and TFRC, behave better. This is due to the fact that the CMOS parameter shown in the figure does not consider what happens in the network (losses and delays). To consider this point better, Figs. 5 and 6 show the loss probability and delay measured during simulation for all the TCP-friendly algorithms considered, and when G.729 encoding is used. In these figures we can observe, as expected, that the performance provided by the proposed RAP and TFRC control algorithms is absolutely the best. For example, the loss rate obtained with RAP and TFRC is in some cases over one order of magnitude less than that obtained with the others. This derives from the fact that they are able to load the network more lightly than the others, as can be observed in Figs. 7 and 8, where the average emission rate for each source and the utilization link coefficient are shown. Therefore, the slight superiority of TCP and TFRC in terms of CMOS is fully offset by the better results in the network. In order to consider these aspects jointly, we encoded a real trace with the AMR encoder, then marked in this trace the positions of the frames lost in the network (obtained via simulation), and finally played the resulting sequence obtained via the AMR decoder. The lost frames were reconstructed using the frame repetition technique proposed in the AMR standard [14]. The subjective analysis was carried out through a series of listening tests according to the ITU-T standard P.800 [18]. From the subjective analysis, we obtained that the best algorithm is RAP, immediately followed by TFRC, while the RAP and TFRC algorithms present worse performance. The sequences obtained are downloadable from [16].
TCP-Friendly Transmission of Voice over IP

TFRC_A control algorithms is absolutely the best. This is also confirmed by a series of informal listening tests at destination. Summarizing, we have obtained that the best algorithm is RAP_A, immediately followed by TFRC_A, while the RAP and TFRC algorithms present worse performance. Finally, let us note that the complexity of our approach, as regards both the rate control and the speech coding parts, is comparable with that of the traditional solutions currently adopted. In fact, the rate control algorithm has a complexity comparable with that of the TCP protocol, while that of the AMR speech codec is slightly higher than the complexity of the EFR (Enhanced Full Rate) speech coder, adopted in many low-cost systems such as 2G cellular devices. The architecture proposed in the paper can be extended to work in an Internet scenario where routers use active queueing management (AQM) techniques to prevent congestion situations, explicitly communicating to the sources the rate variations to be applied. In this context it will be important to evaluate the responsiveness of the voice transmission system proposed in the paper to rate variations imposed by the main AQM techniques, such as RED [2].

REFERENCES


[9] E. Paksoy, K. Srinivasan, and A. Gersho, “Variable Bit-Rate CELP Coding of Speech with Phonetic Classification,” Manuscript received on September 13, 2001


