Priority Based Dynamic Rate Control for VoIP Traffic

Fariza Sabrina
CSIRO ICT Centre, Sydney, Australia
Email: Fariza.sabrina@csiro.au

Jean-Marc Valin
Octasic Inc., Montreal, Quebec, Canada
Email:jmvalin@ieee.org

Jesper Kjeldskov
CSIRO ICT Centre, Sydney, Australia
Email:Jesper.Kjeldskov@csiro.au

***Abstract*—This paper presents a novel mechanism for dynamic rate control of prioritised Voice Over IP (VoIP) traffic in real time. The system uses our proposed variable bit rate speech codec called Speex, which can dynamically adjust the encoding bit rate (and hence the voice quality) based on the feedback information about the network congestion, flow priority, and the instantaneous speech properties. Our extensive NS2 simulation results along with results from ITU-T standard of speech quality evaluation tool (PESQ) show that the proposed system indeed provides highest quality speech while maximising the bandwidth utilisation and reducing the network congestion.**

I. Introduction

Voice over IP (VoIP), which enables real-time delivery of voice traffic in packets between two or more parties across networks, has become one of the most popular IP based real-time communication applications in recent years. To support VoIP applications over the Internet, two conflicting requirements need to be met [1]. On one hand, applications are sensitive to delay, packet loss and bandwidth, so it is required to minimise the effect of network impairments on voice quality. On the other hand, since the Internet is a shared environment, resource utilisation needs to be controlled so that resource usage is optimized and congestion is avoided. This leads existing research to focus on achieving highest quality voice while both maximising the network resource utilisation and avoiding congestion.

Recently, several researchers have focused on rate/quality control for multimedia flows on the Internet. Bolot et. al [2] have demonstrated a packet loss feedback based rate control scheme. However, since packet loss does not necessarily mean network congestion, algorithm based on that may not always be accurate. Mahlo et. al. [3] proposed a different approach to adapting bit-rate for VoIP flows based on an enhanced TCP-friendly rate control (TFRC) protocol that adapts the coding and packetisation to optimize the VoIP quality.

In [4], researchers have also proposed a bit-rate control mechanism for VoIP application based on individual network parameters like packet loss and delay or on the predicted, perceived speech quality. In the proposed system the feedback information was sent via RTCP reports. Currently, the most common transport method for VoIP is the real-time transport protocol (RTP). Unfortunately, the cost in terms of overhead is large. Considering that most speech codecs use a frame size of 20 ms, the sum of the IP, UDP and RTP headers (20+8+12=40 bytes), sent 50 times per second represents 16 kbit/s. This means that typically at least half of the traffic in a VoIP conversation is headers. For many applications, we have many VoIP conversations being carried over the same links. An example of this is a company with large offices in two different cities with several parallel VoIP connections. For these applications, it is possible to aggregate the traffic of all sources and thus greatly reduce the overhead caused by the headers, as is already done in the IAX2 protocol [5].

We propose to take the aggregation approach one step further. Instead of assigning the same bit-rate to every conversation, we can optimise the bit allocation such that conversations that require more bits at a certain point in time can "steal" bits from other conversations. As an example, it would be desirable to reduce the bit-rate when a source is silent so that a higher bit-rate can be used for other conversations. At the same time, the average bit-rate per source needs to be controlled by the link bandwidth and the amount of traffic on that link. This leads to control of the bit-rate based on both the source signals and the channel capacity.

The problem of achieving highest quality for VoIP traffic gets complicated when different flows have different priorities. Some voice conversation might have high priority, others have medium or low priority. Allocating bit rate to traffic flows based on the priority is a challenging research issue. In our preliminary work [6] we focused on cases where there is no priority. In this paper, we propose a new priority based VoIP system that employs an adaptive variable rate speech codec called Speex and a feedback based flow controller called Explicit Control Protocol (XCP) [7]. XCP calculates the allowable aggregate rate and this rate is distributed among the flows based on their priority. We have classified all the flows in three priorities. All the flows with high priority belong to class 1, flows with medium priority belong to class 2 and the low priority flows belong to class 3. We use a weight based rate allocation method to distribute the available bits among the three classes of flows. This allocated rate is then interpreted as a quality level. The level is then linked to one of 8 possible bit-rates of the speech codec. When there is congestion we reduce the encoding bit rate adaptively, and increase the rate when there is spare bandwidth.

Our work differs from previous works on VoIP rate control in three ways. **First**, instead of relying on loss or delay information, we use an explicit feedback-based system which estimates the exact rate that the network can allow without

congestion. **Second**, our approach not only adapts the rate based on network conditions, but also on the source properties. **Third**, we ensure that flows rates are allocated based on the flow priority and thus ensure the quality of service. Our contributions are as follow:

- We present a novel variable bit-rate speech encoding, as part of the Speex codec.
- We use explicit feedback about network condition for adapting to the encoding bit-rates.
- We present a novel mechanism for adjusting the encoding bit rate based on
  - the feedback about the network condition,
  - the instantaneous speech properties,
  - the flow priority.

The paper is organised as follows. A brief introduction to the Speex codec and XCP is given in Sections II and III respectively. The Proposed framework is given in Section IV and detailed simulation result is given in Section V. Conclusions are drawn in Section VI.

II. Speex Codec

For this work, we proposed a novel variable bit-rate speech encoding, as part of the Speex codec. Speex is an open-source speech codec, which we developed and that is described in [8]. This choice is based on the fact that the Speex codec supports rates ranging from 2.15 to 24.6 kbit/s for narrowband (8 kHz telephony standard rate), while supporting source-controlled variable bit-rate. Speex is designed mainly for VoIP and encodes speech in frames of 20 ms (160 samples at 8 kHz). The bit-rates supported (after rounding up to an integer number of bytes per frame) are 2.4 kbit/s, 4 kbit/s, 6 kbit/s, 8 kbit/s, 11.2 kbit/s, 15.2 kbit/s and 18.4 kbit/s.

Like most other modern speech codecs, Speex is based on the Code-Excited Linear Prediction (CELP) [9] algorithm. It is however different from many other CELP codecs in that it supports multiple rates and allows changing of the bit-rate for every 20 ms frame. The rate being used is encoded in the first 5 bits of the of each compressed payload.

A. Source-Controlled Variable Bit-Rate

Because CELP is an inherently fixed-bit-rate algorithm, implementing Variable Bit Rate (VBR) is not as easy as it is for general audio codecs (such as MP3, Vorbis, and AAC). In Speex, source-controlled VBR is implemented using an open-loop approach where the bit-rate is determined before quantisation for each input frame. For each audio frame f, a decision is based on a combination of *scores* computed from simple heuristics:

- Voiced segments require a higher bit-rate $s_v(f)$;
- Onsets (increasing power) require a higher bit-rate $s_o(f)$;
- Regions with very low power require a lower bit-rate $s_l(f)$;
- Regions with decreasing power require a lower bit-rate $s_d(f)$.

Table I
VARIABLE BIT-RATE THRESHOLDS ON $s(f)$ FOR $Q=9$.

rate(kb/s)	0	2.4	4	6	8	11.2	15.2	18.4	24.8
$s(f)$	-1	-0.8	1	2.3	3.5	4	6	7	9.8

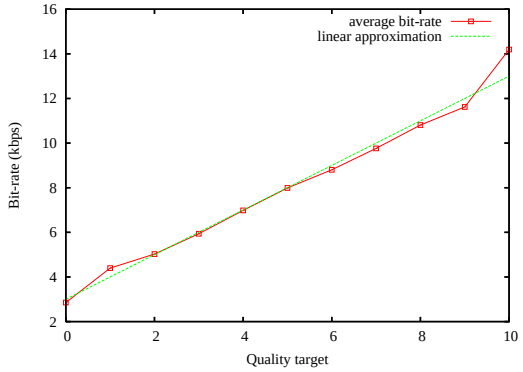

Figure 1. Average bit-rate as a function of the quality target Q.

A final score, representing how difficult the current frame is to encode, is computed as:

$$s(f) = s_v(f) + s_o(f) + s_l(f) + s_d(f) \quad (1)$$

Based on the final score $s(f)$ ranging from -1 to 10, and a quality control parameter Q ranging from 0 to 10 (continuous scale), we can determine what bit-rate to use for frame f. As an example, we show for $Q=9$ the thresholds on $s(f)$ required for each bit-rate in Table I. For instance, if $s(f)=5.5$, we would be choosing to encode the current frame at 11.2 kbit/s. A different set of threshold is defined for all integer values of Q and the thresholds are linearly interpolated for non-integer values of Q. The threshold values were hand-tuned to maximise speech quality. Details of these could be found in [10].

In Fig. 1, we show how the average bit-rate varies as a function of the VBR quality control parameter Q. The results have been taken from actual experimentation of Speex codec. The red curve is the actual characteristics of Speex and the green line is the linear approximation of that. Although only integer values of Q are shown, any real value is allowed. It is important to stress that the average rates in Fig. 1 are data-dependent and only hold for large number of sources (or large time frames). To make the curve more useful for use with XCP, we use the following linear approximation for the average bit-rate b:

$$b \approx (3000 + 1000Q)\, bit/s \quad (2)$$

This makes it easy to compute how a change in bit-rate should affect the quality control parameter Q.

III. Explicit Control Protocol (XCP)

In this paper we used a modified version of XCP (as presented in section IV) for calculating the allowable aggregated rate. XCP [7], a recently developed feedback-based flow control protocol has attracted much attention in the Internet

research community because of its capability of achieving high link utilisation while maintaining low queues in routers. In XCP, each packet carries a small congestion header. The routers compute per-flow bandwidth allocation and this information is sent to the senders as a feedback. The senders change their sending rate based on this explicit feedback. Brief description of XCP is given below.

The congestion header for each XCP packet consists of three fields, such as sender's current throughput, the estimate of sender's current round-trip time (RTT) and a field for feedback. The sender fills in the first two fields and they do not get modified in transit. The other field, *feedback* is initialized by the sender and modified/updated by the routers as the packet traverse the network.

The sender keeps information about the current congestion window (*cwnd*), and an estimated round trip time (*rtt*). When a packet is sent, the sender sets the *throughput* field to $cwnd/rtt$ and *rtt* to its current *rtt*. The sender requests for the desired thoughput increase by initialising the *feedback* field. Whenever a new acknowledgment arrives, positive feedback increases the senders *cwnd* and negative feedback reduces it:

$$cwnd = max(cwnd + feedback * rtt, s), \tag{3}$$

where s is the packet size, and *rtt* is the most recent estimated round trip time.

When a packet reaches the receiver, the receiver copies the congestion header from the data packet and sends it as an acknowledgment.

The router is responsible for computing the feedback; it uses an efficiency controller and a fairness controller to do so. The efficiency controller's purpose is to maximise link utilisation by controlling the aggregate traffic. The aggregate feedback ϕ is computed at the control point for each control interval as:

$$\phi = \alpha B - \beta \frac{q}{r} \tag{4}$$

In (4), ϕ is the total amount of desired change in input traffic. α and β are constant parameters, whose values set to 0.4 and 0.226 (respectively) based on the analysis presented in [7] and r is the average RTT, q is the persistent queue size, and B is the spare bandwidth. B is derived from $B = X - y$, where X is the link capacity and y is the total input traffic rate. When, $B \geq 0$, the tunnel is under-utilized and then a positive feedback will be sent to the source, and when the tunnel is congested (i.e. $B < 0$), a negative feedback will be sent to the source. The efficiency controller deals only with the aggregate behavior, how exactly the feedback is divided among the packets (and hence the flows) is dealt by the fairness controller. The fairness controller (FC) distributes the feedback to individual packet as below: If $\phi > 0$, FC allocates it among all the flows equally. If $\phi < 0$, FC allocates it so that the decrease in throughput of a flow is proportional to its current throughput.

Feedback assigned to packet i is computed as the combination of two elements, positive (P_i) and negative (N_i) feedback.

$$feedback_i = P_i - N_i \tag{5}$$

For the case where the aggregate feedback is positive ($\phi > 0$), the throughput of all flows should be increased equally. Thus the throughput of any flow i should be changed proportional to its reserved rate, so $\Delta r_i \propto constant$. Here, Δr_i is the change in rate. To convert this desired change of the throughput to per-packet feedback, the total change in throughput is divided by the number of packets from flow i that the router sees in a control interval d. This number is proportional to the flow's throughput divided by its packet size, $\frac{r_i}{s_i}$. Since d is constant for the duration of the control interval, the per-packet positive feedback is inversely proportional to its throughput divided by its packet size, (i.e., $P_i \propto \frac{1}{r_i/s_i}$). Thus the positive feedback P_i is derived as: $P_i = \varepsilon_p \frac{s_i}{r_i}$. where, ε_p is a constant. The total increase in the aggregate traffic rate ($h + max(\phi, 0)$, where $max(\phi, 0)$ ensures that it is positive feedback) is equal to the sum of the increase in the rates of all flows in the aggregate, and so:

$$h + max(\phi, 0) = \sum^{M} P_i \tag{6}$$

where M is the number of packets seen by the router during the control interval d. From this, ε_p can be derived as:

$$\varepsilon_p = \frac{h + max(\phi, 0)}{\sum \frac{s_i}{r_i}} \tag{7}$$

Similarly per-packet negative feedback is calculated when the aggregate feedback $\phi < 0$. For this case, the throughput of flow i is decreased proportionate to its current throughput (i.e., $\Delta r_i \propto r_i$). To calculate the per-packet negative feedback, the desired change in throughput is divided by the expected number of packets from this flow that the router sees in an interval d. Here d is proportional to $\frac{r_i}{s_i}$. Thus, the per-packet negative feedback is calculated as: $N_i = \varepsilon_n . s_i$. where, ε_n is a constant. The total decrease in the aggregate traffic rate is the sum of the decrease in the rates of all flows:

$$h + max(-\phi, 0) = \sum^{M} N_i \tag{8}$$

ε_n is derived as:

$$\varepsilon_n = \frac{h + max(-\phi, 0)}{\sum s_i} \tag{9}$$

This P_i and N_i is used to compute $feedback$ in equation (5). All the parameters used in the feedback calculations can easily be obtained at the router, for example, the throughput, r_i, is in the congestion header, the packet size, s_i is in the IP header, and the aggregate traffic rate y, and the average RTT, d are measured by the router. The Fairness controller keeps track of the total amounts of positive and negative feedback allocation. It allocates positive feedback until the sum of the total allocated positive feedback reaches the target of $h + \max(\phi, 0)$, and also stops allocating negative feedback when the sum of the negative feedback (allocated since the beginning of the interval) reaches the target of $h + \max(-\phi, 0)$.

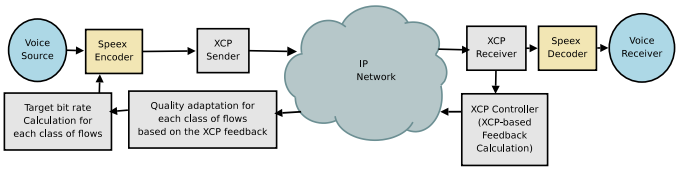

Figure 2. Framework for the Network Aware Rate Adaptation

IV. Framework for Network Aware Dynamic Rate Adaptation

The main focus of our work is to optimise the voice quality (or bit rate) adaptively based on the network condition, flow priority and speech properties. This is done **firstly**, by reducing the encoding bit rate when the network is congested and increasing it if there is available bandwidth. This ensures that rather than dropping voice frames, voice quality is graciously reduced when the network is congested and when there is available bandwidth, the voice quality is upgraded. **Secondly**, the available bitrate is allocated to different flows based on their priorities. As we mentioned before, flows are classified into three classes based on their priorities. 50% of the total available bandwidth is allocated to the highest priority class and 30% to the next priority and the rest to the third priority class. **thirdly**, we adapt the source bit-rate to achieve the highest speech quality. We optimise the bit allocation by allocating more bits for voice data that requires a higher bit-rate to achieve good quality, while reducing the bit rate for silent sources or for sounds that need less bitrate to achieve good quality. The motivation of the proposed combined system is to improve the perceived speech quality by combining rate-adaptive encoder and feedback based flow control schemes.

Our integrated framework of the Speex codec and a rate controller (modified XCP) is shown in Fig. 2. It consists of voice sources, Speex encoder, XCP sender, XCP controller, XCP receiver, Speex decoder and voice receiver. Since VoIP involves two way communication, all the sender and the receiver side modules need to be implemented on both sides. For clarity we show the sending side modules on the left side and the receiver side modules on the right side in the figure.

In our framework, we have gateways on both sending and receiving sides. The gateways are the places where the modules are integrated. The clients send the speech frame to the gateway, and the gateway sets the quality parameter based on the feedback information received from XCP controller. In other words, the encoding bit rate of the codec is adjusted based on feedback information received from the XCP controller. The encoding bit rate is allocated among the 3 classes of flows are based on the following relationship. As shown in Fig. 2, the quality control module is used to adapt the quality level in accordance to the feedback information received from the XCP controller. We assign 50% of the bit rate to class 1 flows (flows with high priority) and 30% of the bit rate to the class 2 flow priority and the rest (20%) to the class 3. We use the following equation to calculate the bit rate for the flows of a particular class:

$$x = \frac{a * y}{n}; \tag{10}$$

where, $a =$ weight for the priority class (0.5 for class 1, 0.3 for class 2 and 0.2 for class 3), $y =$ total aggregated bit rate for all the flows of all classes as calculated by XCP, and $n =$ the total number of flows for each particular class.

Once we have calculated the bit rate, the new target bit-rate is computed and converted to a new quality control parameter Q based on equation (2). The new value of Q is then used by the Speex encoder. The encoder then chooses the bit-rate for each source based on both Q and the content of each source. After encoding the frame and the encoded frames are put in a buffer.

The XCP sender aggregates encoded frames from all sources until either the packet size reaches 1500 bytes (including the headers), or the oldest frame aggregated is more than 20 ms old. We use one byte to indicate the source index, which typically represents an overhead of 0.4 kb/s. Even when counting the XCP overhead, the overhead is much smaller than the 16 kb/s overhead due to IP/UDP/RTP headers in a non-aggregated VoIP traffic. The flow diagram is shown in Figure 3.

On an arrival of an XCP packet, the system (receiver side) updates the XCP controller's control variables (e.g. input traffic rate, average *rtt* etc) and puts the packets on a FIFO packet queue. It also calculates the new aggregate rate. At each average *rtt* interval, one of the timer handlers is scheduled to calculate the XCP controller variables such as positive and negative feedback related variables as explained in Section III. Before departure of each XCP packet, the feedback is calculated and the congestion header is updated.

When an XCP packet reaches the receiver side, the frames are decoded and sent to the corresponding destinations. The XCP controller calculates the feedback based on the information received in the XCP packet header and the network condition. This feedback is sent to the sender side. Since XCP does not keep any per-flow state information, the complexity added by XCP in the router is negligible.

We have modified the XCP sender in our framework. Here, the XCP sender creates packet which consists of encoded voice frames. We also modified the XCP controller for our system. Here, XCP controller is used to calculate the target aggregate rate and based on that rate we set the quality control parameter and also the encoding bit rate. The transmission rate is not controlled by the XCP controller. We send one packet in every 20 ms or once the accumulated encoded frame size is greater than 1460 (whichever comes first). It should be noted here that the packet size is not fixed. The modification on XCP receiver is as follows, in addition to sending the acknowledgment packet with feedback information to the sender, it sends the XCP packet to the Speex decoder for decoding.

V. Simulation

This section presents the detailed simulation results.

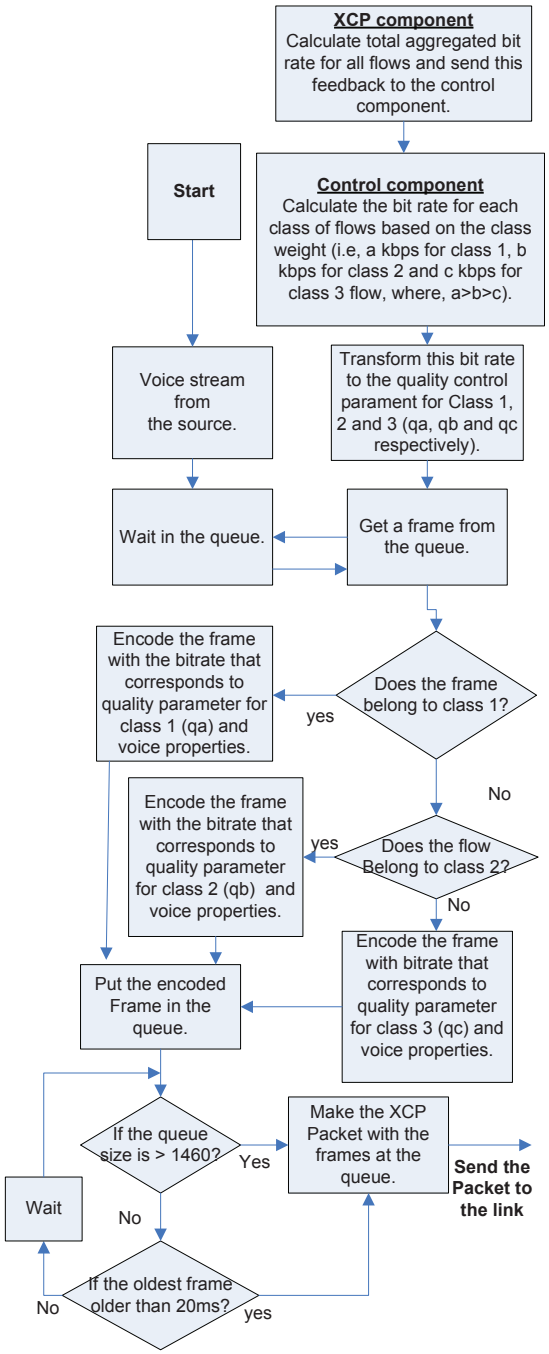

Figure 3. Work Flow at the sender side

A. Simulation Setup

Our simulation environment (Fig. 4) integrates NS2 network simulator [11] and Speex codec and consists of three main parts: (1) An NS2 simulation of 100 incoming and outgoing flows (voice traffic) on each sides. (2) a VoIP simulation system to simulate VoIP flow which uses the Speex encoder and decoder and control modules, and (3) a modified XCP controller which determines the allowable transmission rate. We have added new C++ classes in NS2 simulator to represent the gateway and clients (source and destination), and also modified XCP congestion control algorithm. The simulation was run for 100 sec and for various link capacities ranging from 0.5 to 1.5 Mbit/s. All the sources were attached with a CBR traffic generator where the sending interval was set 20 ms (i.e. each source is sending one frame in every 20 ms). All the encoded frames are stored in a buffer transmitted in an XCP packet when it reaches 1500 bytes.

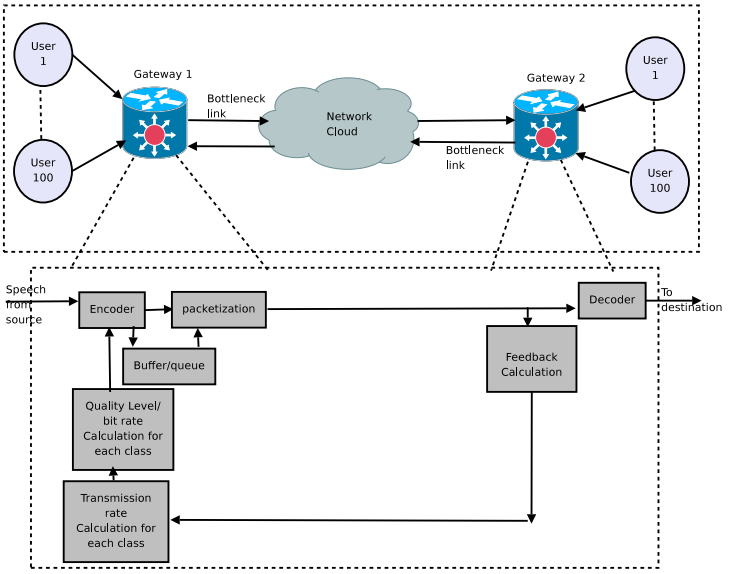

Figure 4. Simulation System

B. Simulation Results: Scenario 1

In this scenario we used 100 sources on each side and the sources were started sequentially in 5ms intervals. Among these 100 sources, 35 sources were of class 1 (highest priority), 35 sources were of class 2 (medium priority) and the rest (30 sources) were of class 3 (low priority). We investigated the performance of the system under different network congestion scenarios. We measured link utilisation, packet loss, delay and also quality level for all these scenarios. The respective results are given in the following sub sections.

1) Quality Control Parameter (Q): We measured the quality control parameter for all 3 classes for cases with different link capacities (i.e, 0.5, 0.8, 1.0, 1.2, & 1.5Mbit/s). The comparative results (presented in Fig. 5 and Table II) demonstrates how the quality level is adjusted with different available link capacities and class priority. Table II also shows that, class 1 flows always get the higher bit rate so the quality level is higher than the lower priority classes.

2) Link Utilisation and Achieved Bit Rate: We have measured the link utilisation for various scenarios (congested, lightly congested, and non-congested scenario) with different link capacity as presented in Fig. 6. The results were satisfactory, and in most of the cases the link utilisation was 93%. It proves that when the link capacity was low the coding bit rate was reduced and when the link capacity was high, the

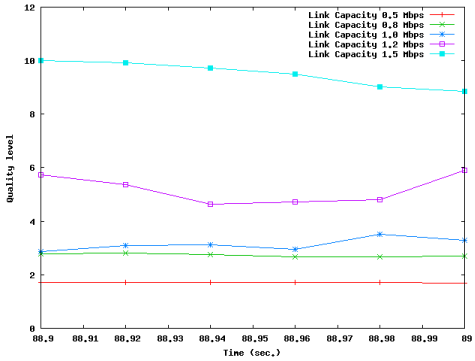

Figure 5. Quality Level for Various Link Capacities for class 1 flows.

Table II
QUALITY LEVEL FOR DIFFERENT LINK CAPACITIES AND FLOW CLASSES

Link Capacity (Mbit/s)	Flow class (Priority)	Quality Level			
		Avg.	Max.	Min.	Std. dev.
0.5	1	2.5	10	0	1.9
	2	0.5	5.8	0	0.7
	3	0.1	1.8	0	0.2
0.8	1	6.7	10	2.6	1.6
	2	2.8	5.6	0.4	0.9
	3	1.5	3.7	0	0.7
1.0	1	8.6	10.0	4.5	1.4
	2	4.3	10.0	1.5	1.4
	3	2.7	6.4	0.5	1.1
1.2	1	9.4	10.0	5.3	0.9
	2	6.2	10.0	2.0	2.0
	3	4.1	10.0	0.9	1.7
1.5	1	9.5	10.0	2.2	0.0
	2	8.4	10.0	0.0	2.6
	3	6.9	10.0	0.0	2.8

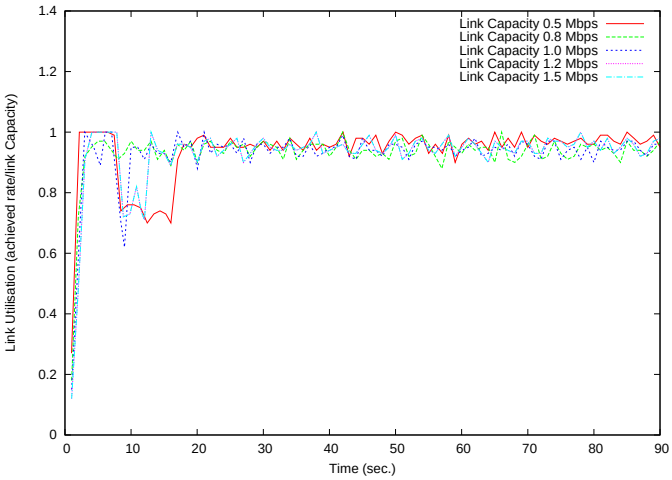

Figure 6. Link Utilisation at Gateway 1

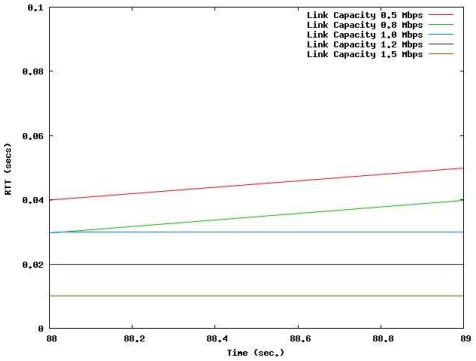

Figure 7. Average Round Trip Time.

coding rate was increased so that the link utilisation could be maximised.

We also measured the effective voice data rate and XCP overhead for various link capacities (as shown in Table III). For a particular case where the link capacity was 1.2Mbit/s, the results show that the average achieved rate was 1.12 Mbit/s (93% of the link capacity). The effective voice data rate was 1.02 Mbit/s (85% of the total traffic), and the rest (8% or 0.10 Mbit/s) is the XCP overhead .

3) Packet Loss: One of the critical issues in VoIP system is the degradation of speech quality due to packet loss. In our simulation, the packet loss was almost zero for most of the cases. We only noticed a few lost packets for the case of link capacity 1.5 Mbit/s.

4) Delay Measurement: VoIP flows are sensitive to delay and delay jitters. In our simulation we have measured the round trip time (RTT). The detailed delay result for all 5 cases are shown in Table IV. The RTT for all cases were satisfactory and the low standard deviation (5^{th} column of Table IV) proves that delay behaviour was consistent. Fig. 7 shows the RTT for last second of simulation time for all five cases with various link capacity. RTT is the total time taken for the following actions: sending the frame from source to gateway, encoding the frame and packetising the frames, transmitting the packet to the other end, creating the acknowledgement (ack) packet, and sending the ack packet back to the gateway.

Table III
ACHIEVED RATE AND OVERHEAD FOR VARIOUS LINK CAPACITIES.

Link Capacity (Mbit/s)	Link Utilisation (Avg.)	Effective voice data (Avg.)	XCP overhead (Avg.)
0.5	94 %	80%	14%
0.8	93%	84%	9%
1.0	93%	84%	9%
1.2	93%	85%	8%
1.5	91%	84%	7%

Table IV
ROUND TRIP TIME (RTT) FOR DIFFERENT LINK CAPACITIES.

Link Capacity (Mbit/s)	RTT (sec.)			
	Avg.	Max.	Min.	Std. dev.
0.5	0.04	0.1	0.02	0.017
0.8	0.03	0.08	0.02	0.01
1.0	0.03	0.06	0.02	0.01
1.2	0.02	0.05	0.01	0.007
1.5	0.03	0.19	0.01	0.39

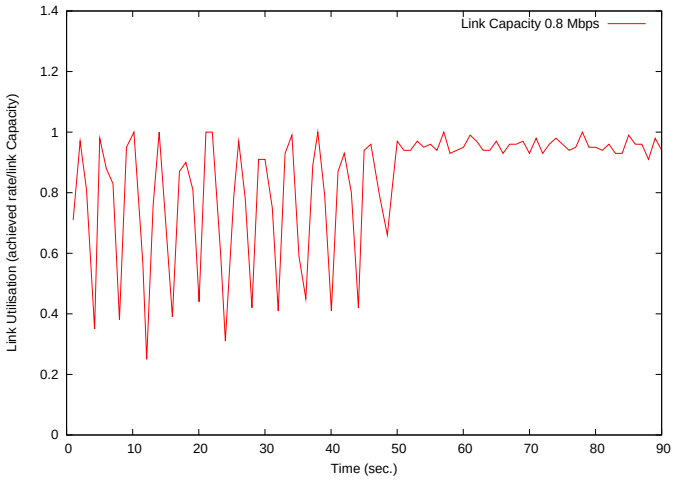

Figure 8. Link Utilisation at Gateway 1 (Scenario 2).

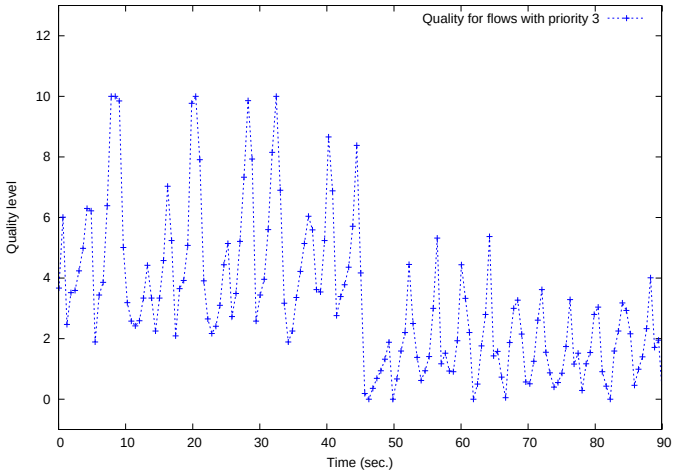

Figure 9. Quality Level (Scenario 2).

C. Simulation Results: Scenario 2

For this scenario, we started 50 sources (18 of them had highest priority, 17 had medium priority and 15 had low priority) in the beginning and then added 50 more sources (17 with highest priority, 18 with medium priority and 15 with low priority) after 45^{th} secs. The link capacity was 0.8 Mbit/s and we measured link utilisation, achieved rate, RTT and quality level. Due to space limitation, only key graphs are presented here. Fig. 8 shows the total link utilisation. In the beginning when there was only 50 sources on, there was some fluctuation in the link utilisation curve but after 45^{th} second 50 more sources were made active and the link utilisation were almost 97%. Fig. 9 shows the quality level for class 3 flows. The quality was higher while the 0.8 Mbit/s link was shared by 50 sources. After 45 sec, the link was shared by 100 sources, so the quality control parameter Q was lowered accordingly. This proves that our system can handle dynamic traffic and can adapt the quality accordingly. Other classes have also shown the same behaviour.

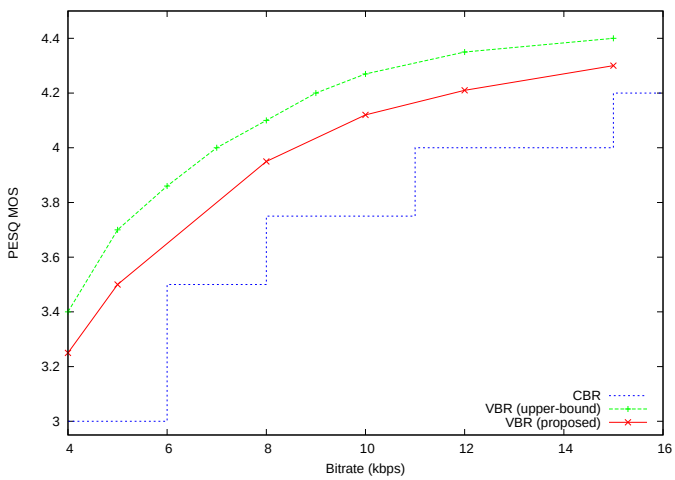

Figure 10. PESQ MOS quality as a function of the payload bit-rate (All flows have the same priority).

D. Audio Quality

The ultimate goal of the proposed source-channel-controlled VBR approach is to maximise the quality of the transmitted speech. To assess the quality, we use the PESQ tool [12], which approximates the mean opinion score (MOS) as it would be rated by a human listener.

As an upper-bound, we use the average bandwidth and speech quality obtained by setting Q to a constant value for an entire conversation (assuming no bandwidth limitation). As a lower bound, we have the constant bit-rate (CBR) case, where the MOS quality varies in steps as encoding changes from one fixed bit-rate to another when the bandwidth increases (e.g. with 12 kbit/s available, we need to use 11.2 kbit/s because CBR rates are not continuous). Here, we show the audio quality for both single priority and multiple priority case in Figure 10 and 11 respectively. Figure 10 shows that the proposed approach provides a significant improvement over the use of constant bit-rate compression when all flows have the same priority. We also see that there is still room for improvement before reaching the upper-bound. We improved this situation for flows that require very high quality by prioritising the flows. As shown in Figure 11, the quality of the audio streams with highest priority almost reached the upper bound. Streams with priority 2 also performs better than the CBR case. The quality of priority 3 streams was similar to CBR. Comparing Figure 10 and 11, we prove that by prioritising the streams we could provide much better quality voice for high priority voice traffics.

E. Performance with background TCP traffic

We have tested the system with background TCP traffic. In this scenario, the simulation was run for five different link capacities (i.e., 1 Mbps, 1.6 Mbps, 2 Mbps, 2.4 Mbps, 3 Mbps). The experiment was designed so that half of the traffic was VOIP traffic and the other half was background TCP traffic. We have calculated round trip time trip and evaluated the quality of audio using PESQ tool. Table V shows that the

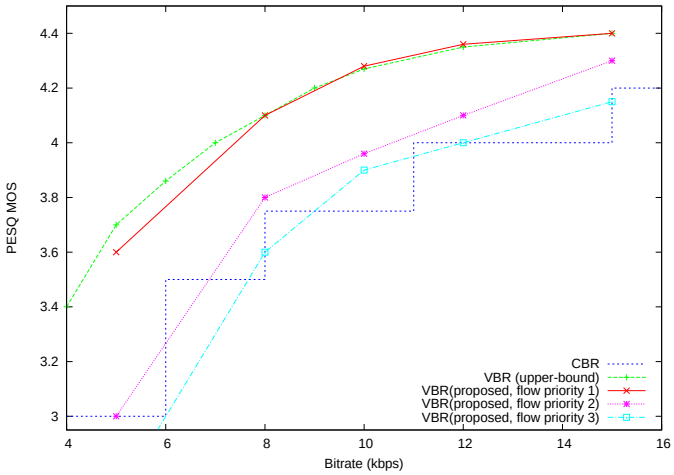

Figure 11. PESQ MOS quality as a function of the payload bit-rate (Multiple priority case).

Table V
ROUND TRIP TIME (RTT) FOR DIFFERENT LINK CAPACITIES.

Link Capacity (Mbit/s)	RTT (sec.)			
	Avg.	Max.	Min.	Std. dev.
1.0	0.017	0.05	0.009	0.014
1.6	0.017	0.05	0.008	0.014
2.0	0.016	0.05	0.008	0.015
2.4	0.011	0.05	0.007	0.009
3.0	0.011	0.05	0.006	0.013

delay behaviour was very good with very low average delay. Figure 12 shows the audio quality for class 1 flow with and without background traffic. This figure shows that, the audio quality for flows with priority 1 was better than CBR with background TCP traffic too. But we observe that the audio quality was a bit better without the background traffic.

VI. CONCLUSION AND FUTURE WORK

In this paper, we presented a novel technique for controlling the bit-rate of aggregated VoIP traffic based on the network conditions, source properties, and also the traffic priorities. We investigated perceived speech quality, delay behaviour and the link utilisation for various network scenarios by integrating NS-2 simulator with a real adaptive speech codec (Speex) and a perceived quality evaluation tool called PESQ. The goal of this combined system is to achieve best possible QoS under any network condition. Our simulation results demonstrated that our system achieved this.

Currently we are working on further improving the performance of our system with background traffic. We will also implement this work on our real world testbed. In future, our explicit rate control system will be extended for multimedia flows, where the traffic contains voice, video and data traffic.

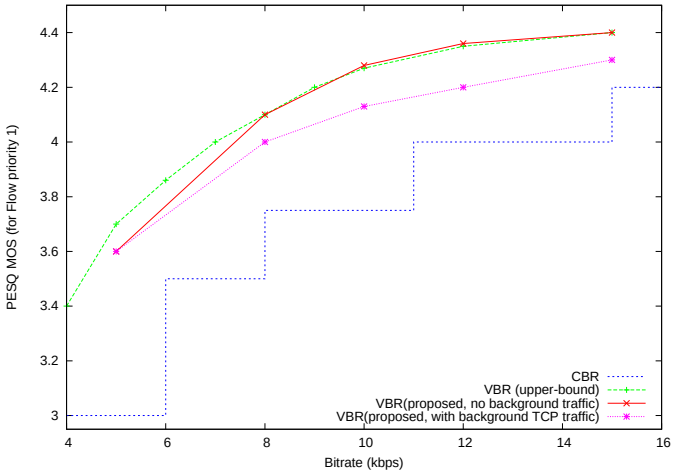

Figure 12. PESQ MOS quality as a function of the payload bit-rate (For all flows).

REFERENCES

[1] R. Rejai, M. Handley, and D. Estrin, "Layered quality adaptation for internet video streaming," *IEEE Journal on selected area of communications*, vol. 18, no. 12, pp. 2530–2543, 2000.

[2] J.-C. Bolot, S. Fosse-Parisis, and D. F. Towsley, "Adaptive FEC-based error control for internet telephony," in *Proc. INFOCOM*, 1999, pp. 1453–1460.

[3] C. Mahlo, C. Hoene, A. Rostami, and A. Wolisz, "Adaptive coding and packet rates for TCP-friendly VoIP flows," in *Proc. of 3rd International Symposium on Telecommunications (IST2005)*, 2005.

[4] Z. Qiao, L. Sun, N. Heilemann, and E. Ifeachor, "A new method for voip quality of service control use combined adaptive sender rate and priority marking," in *Proc. ICC*, 2004, pp. 1473–1477.

[5] M. Spencer, B. Capouch, E. Guy, F. Miller, and K. Shumard, "IAX2: Inter-Asterisk eXchange version 2," 2007. [Online]. Available: http://www.ietf.org/internet-drafts/draft-guy-iax-03.txt

[6] F. Sabrina and J.-M. Valin, "Adaptive rate control for aggregated voip traffic." in *In Proceedings of GLOBECOM 2008*, 2008, pp. 1405–1410.

[7] D. Katabi, "Decoupling congestion control from the bandwidth allocation policy and its application to high bandwidth-delay product networks," PhD Thesis, MIT, March, 2003.

[8] J.-M. Valin, "The speex codec manual." [Online]. Available: http://www.speex.org/docs/

[9] M. Schroeder and B. Atal, "Code-excited linear prediction(CELP): High-quality speech at very low bit rates," in *Proc. ICASSP*, 1984, pp. 937–940.

[10] J.-M. Valin, "The speex codec." [Online]. Available: http://www.speex.org/downloads/

[11] "The network simulator - ns-2." [Online]. Available: http://www.isi.edu/nsnam/ns/

[12] *Recommendation P.862: Perceptual evaluation of speech quality (PESQ): An objective method for end-to-end speech quality assessment of narrow-band telephone networks and speech codecs*, International Telecommunications Union (ITU-T), 2001.