VoIP Congestion Control by changing packet size – Simulation

Alejandro Avella (aavella@cisco.com) and Srikanth Kakani (Srikanth.Kakani@efi.com)

ABSTRACT

The dynamics of the patent described in [5] regarding a "codec-independent technique for modulating bandwidth in packet networks" are investigated by means of simulations. The patent idea is that senders adapt their rate, by changing packet size, based on network load. Two sender-based rate algorithms, 2 RTCP versions, 2 codecs and several bandwidth sizes are studied in several scenarios combining all the variables. We evaluated these scenarios with an evaluation criteria composed of several items including voice quality via PESQ. We showed that the idea presented in [5] controls congestion, but that a mechanism to prevent oscillations is necessary at the sender-based rate adaptation algorithms studied.

Keywords

RTP, RTCP, sender-based rate adaptation, simulation.

1. INTRODUCTION

Voice architecture group at Cisco Systems had a need to better understand the dynamics of the idea defined in [5]. Simulation using ns-2 was proposed as a tool to understand the dynamics of the algorithm.

Many codecs provide rate adaptation inside the codec itself such as G.729 and iLBC [1], but these functions are not used as part of [5]. There are other mechanisms to adapt the rate at the sender such as layered encodings, reducing or increasing Forward Error Correction (FEC) and changing codec in use, but only changing the packetization time is considered as part of this simulation as a codec-independent method to modulate the bandwidth.

RTP/RTCP [10] is a transport protocol that has been used successfully to carry voice traffic over the Internet with the use of the profile for voice and video [11]. Section 10 of [10] shows that voice traffic is an inelastic source that can't adapt its rate during the sessions in any direction (up or down). The idea suggested in [5] shows how this rate adaptation can be achieved on by changing the packet size to control congestion.

2. PRIOR WORK

The patent [5] describes a sender-based rate adaptation algorithm. Multiple algorithms have been proposed in this area, such are the one described in [2], which is used as a comparison point in the simulations. Since this is a simulation paper we did not attempt to propose any new sender-based rate adaptation algorithm as part of this work, instead we investigated these two algorithms and compared them as a basis for a future proposal.

Existing research proposes a VoIP system that changes payload sizes depending on changing network QoS conditions using transmission efficiency and voice quality as the parameters to control the system [14]. Another paper investigates changing the sender's rate based on RTCP reports containing MOS scores calculated via PESQ; on top of that DiffServ marking is applied to the outgoing packets [15].

TCP Friendly Rate Adaptation Control (TFRC) [18] is a congestion control algorithm that shares the available bandwidth with TCP applications. VoIP may use TFRC because it offers low variation in throughput, which is desired in this kind of applications. TFRC-Voice is a variant of TFRC [16] for audio applications that have a fixed sending rate but vary their packet size in response to congestion. This latter effort is related to work to the simulation presented in this paper, where ongoing work is being done as the writing of this paper.

Karam and Tobagi [7] studied optimal packet sizes for wired VoIP for a good compromise between low delay and efficient network utilization. These results are used as reference to decide what packetization time to use in the simulations (10 ms to 80ms in the case of G.729 and 10 to 30 ms in the case of G711). We decided to use 20 ms for G711 and 30 ms for G729 based on the results of this paper and on industry experience.

RTP is a transport protocol and as such it needs to address congestion control [3], however RTP specification ([10], [11]) have vague statements about congestion control and nothing is specified in detail. See for example section 2 of [11].

IETF IAB has expressed concerns regarding congestion control for voice traffic in the Internet [4]. To prevent congestion collapse, there are several options such as: a) over engineer the network by adding more bandwidth to the links; b) Terminate calls in case of congestion; or c) adapt the rate of the senders

depending on network load. This later option is the one studied in this simulation work.

RTCP-based feedback [9] abolishes the 5 sec minimum RTCP interval and introduces the use of NACKs immediately after packet loss occur in immediate mode. This internet-draft has been proposed as a standard earlier in 2005 and will be used as a comparison point with RFC-3550 [10]. This earlier feedback can be used more effectively by the sender-based rate adaptation algorithms to control congestion.

RTCP Extended Reports [17] adds new statistics to the reports as part of VoIP Metrics Report Block that could be used to control the RTP sender-rate algorithms but these reports are not used in this simulation and are suggested as future work.

3. MATHEMATICAL ANALYSIS

A fundamental relationship that needs to be understood is the budget delay:

$$D = T_f + l + \sum_{h \in Path} (T_h + Q_h + P_h) + T_j \leq 150\,ms$$

Equation – 3.1 – – > [7]

Where,
D =Total Budget Delay
T_f =Formation time
l = Look-ahead (Codec dependent)
T_h = Transmission Delay at hop h
Q_h =Queuing Delay at hop h
P_h = Propagation Delay at hop h
T_j = Time at the jitter buffer

According to ITU-T G.114, the maximum delay for voice interactivity is 150 ms, if the delay is longer than that people will start complaining. Then, it can be seen that selecting an arbitrarily large packet size is not an option. For example, if G729 is in use the look-ahead time is 7.5 ms; for a US continental call we can approximate the third factor in the equation 3.1 to about 30 ms; typical jitter buffers will buffer a couple packets before playing them out to the user when they are configured in static mode, which is 60 ms extra. That is a total of 97.5 ms; the difference which is 52.5 ms could be using forming a packet, then in this case you could choose a packetization time say of 30 ms and it will be ok, but a packetization time of 80 ms won't be ok due to the maximum acceptable delay D_{max}. For further analysis refer to [7].

Delay Assumptions:

- Transmission Delay (T_h) is bytes per packet over bandwidth of the link. For example, a G711 20ms call will have T_h of 135 us over a single T1 link. Supposing there are 10 hops, this quantity is about 2 ms.
- The Propagation Delay (P_h) is about 30 ms one-way for a call across the US, which is the scenario we are simulating.
- The Queuing Delay (Q_h) at intermediate routers is assumed to be zero, causing zero Jitter for IP phones. Queuing delay is left for future study.
- For simplicity, given the above considerations for Transmission Time, Propagation Delay and Queuing Delay we assume a constant Network Delay (T_n) of 30 ms. The WAN link has varying capacity depending on the scenario.
- Look-head time is half of the formation time for G729 and zero for G711.
- We assume variable jitter buffer, where the depth is set to twice the formation time automatically based on the variation delay and the changing size of the received packets. The means of changing the depth of the jitter buffer depth are out of the scope of this simulation.
- Silence Suppression is not in use at the transmitter, eliminating variability due to the random events of talk spurts and silence periods.

Using these delay assumptions, the following relations can be derived for the maximum formation time allowed under these conditions:

$$D_{G729} = T_f + \tfrac{1}{2} T_f + 30\,ms + 2 * T_f \leq 150\,ms$$
$$T_{fG729} \leq 34\,ms$$
$$D_{G711} = T_f + 30\,ms + 2 * T_f \leq 150\,ms$$
$$T_{fG711} \leq 40\,ms$$

Equation – 3.2

There is a trade-off between the throughput constraint, which favors larger packets and the delay constraint, which favors smaller packets. It is to note, however, that the optimal packet size won't be achieved during congestion periods using the idea proposed in the Patent [5]. The algorithms attempt to use either optimal [5] or minimal [2] packet size when there is no congestion.

Another reason packet size cannot be arbitrarily long is that when a packet is lost users can notice this more than if a small packet is lost. On top of that, Packet Loss Concealment (PLC) algorithms won't work as well when a large packet is lost.

Figure 1 shows the idea that during times of congestion endpoints can increase packet size, to decrease the sender's rate and alleviate congestion. For example, at G711 ptime 20 ms the rate is 80Kbps and it could be decreased to about 75 Kbps in 30 ms is in use.

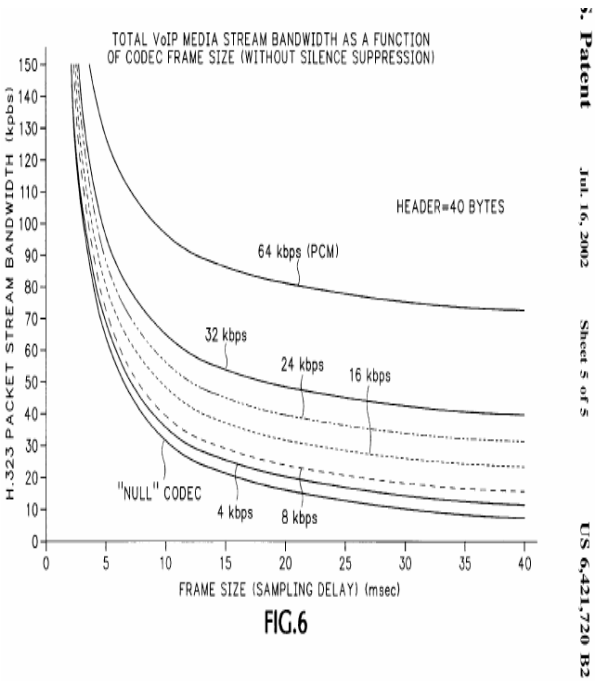


Figure 1 – Summary of Patent idea

Table 1 shows the bandwidth requirements in bits per second for two codecs: G711 and G729 in bits per second using the following relationship:

$$\text{bps} = \frac{(\text{bytes} * 8)}{\text{ptime}} = \frac{(40 + \text{payload}) * 8}{\text{ptime}}$$

Equation 3.3

A header of 40 bytes is used in this calculation, disregarding the data link layer header. This helped us with the comparisons with the Patent [5] that also ignore the data link layer. The amount of bytes that we put in the **payload** is what we vary to change the sender's rate which decreases as the packetization time increases (ptime).

Codec	ptime (ms)	payload (bytes)	pps	bps
G711	10	80	100.00	96,000
G711	20	160	50.00	80,000
G711	30	240	33.33	74,667
G729	10	10	100.00	40,000
G729	20	20	50.00	24,000
G729	30	30	33.33	18,667
G729	40	40	25.00	16,000
G729	50	50	20.00	14,400
G729	60	60	16.67	13,333
G729	70	70	14.29	12,571
G729	80	80	12.50	12,000

Table 1 – bps for G711 and G729

4. SIMULATION ENVIRONMENT

4.1 Network Simulator version 2 (ns-2)

The simulations were conducted using ns-2 [8] on top of Cygwin or Linux. The scenarios are set-up using TCL. Using these TCL scripts we specified the network topologies (nodes and links, bandwidths, queue sizes or error rates for links) and the parameters of the new RTP agents developed, which are the protocol configurations.

4.2 Network Topologies

There are two locations: San Jose, CA and New York, NY. IP Phones are connected to a 100 Mbps Ethernet switch via point-to-point connections at each location. In each location there is a router with PPP WAN with a 30 ms delay with a bandwidth barely enough, around capacity and with enough capacity to handle the offered call load. Figure 2 depicts a graphical representation of the network in use. Table 2 shows the actual bandwidth used depending on the codecs in use.

Phones are paired with phones in other locations and calls are started at the beginning of the simulation and RTP packets are sent between origination to destination unidirectional**.** In other words, only RTCP packets flow in the reverse direction.

The low number of calls and the use of traffic in one direction help us in the analysis and visualization of results.

Codec Combination	WAN Link (Kbps)
4 G711	200, 300, 500
4 G729	40, 70, 200
2 G711 + 2 G729	100, 180, 300

Table 2 – WAN links

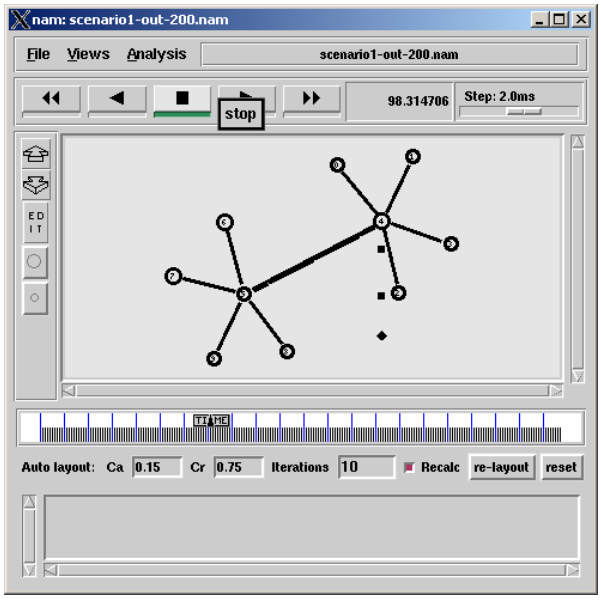


Figure 2 – Network Topology

4.3 Agents

Ns-2 currently has an incomplete implementation of the RTP/RTCP protocols and it was developed partially in C++ and partially in oTCL. For example, timestamps, packet loss, delay and jitter calculations are not included in today's ns-2 implementation for RTP/RTCP. We implemented some of the missing items in the simulator. These agents are attached each of the endpoint nodes defined in the network topologies defined in section 4.2.

A single RTP agent was created that implements the sender-based rate adaptation algorithms described in [2], [5]. Only the unicast portion of the existing RTP agent in the simulator was used, while ignoring the multicast code available in the existing code.

We added packet loss statistic in RTCP reports which was not present in the original code.

According to [9], RTCP reports are considered to be too slow for congestion control algorithms to work properly. We implemented this new RTCP agent as defined in [9] in ns-2. This new RTCP agent sends NACKs as soon packet loss is detected in the "immediate mode" of operation. The new Audio-visual Profile (AVP) enables receivers to provide more immediate feedback to the senders and thus allow for short-term adaptation.

4.4 Scenarios

There will a number of variables in the simulation that will be changed as follows during a reasonable timescale:

4.4.1 Sender-based rate adaptation algorithms

Sender-based rate adaptation algorithms in study [2], [5] will vary the packetization times from 10 to 80 ms. Algorithms will determine what packet size to use and when to make the packet changes. These two algorithms are compared to the case in which there is NO sender-based rate adaptation as is the case in current VoIP deployments.

4.4.2 RTCP version

Whether the RTCP implementation is the one defined in RFC-3550 [10] or the one defined in [9]. We do not implement the Timer Reconsideration algorithm defined in RFC-3550, since this deals with large multicast groups, which are not part of this simulation. Instead we use the minimum allowed interval of 5 sec as constant for all scenarios using this version of RTCP.

4.4.3 Codec combinations

The distribution of codecs in use will be varied as follows: G711 only, G729 only and G729 and G711 combined scenarios. These variations will explore the algorithms while using different packet sizes on each channel. It is important to note that we did not change codecs on the fly as a reaction to congestion.

4.4.4 Congestion

Packets are dropped by overflow in routers' queues during congestion periods. Congestion is generated by making more VoIP calls than the ones supported on the WAN link.

4.4.5 Non-ideal channel conditions

Ideal channel will be assumed in these experiments for 802.3 LANs and the WAN link, since bit error rates on current wired networks is very low.

4.4.6 Scenarios summary

Table 5 summarizes the 54 scenarios studied as part of this simulation project. After the simulation is done in about 2 hours in a 2.4 GHz PC for all of the scenarios, about 1,800 files are generated for a total of about 1.5 GBytes of data that is later analyzed with 9 graphs per scenario or about 450 graphs total.

Two sender-based rate adaptation algorithms are compared with no adaptation [2] [5]. While RTCP version described in [10] is compared with the quicker version described in [9]. The mix of codecs is varied according to pure G711, pure G729 and a combination of the two codecs with total of 4 channels going over a WAN link as defined in Table 2. For further details, refer to Table 5 where the 54 scenarios studied are summarized.

4.5 Implementation details

Our goal was to understand the dynamics of the patent [5], which suggested one rate-based adaptation algorithm. We proposed comparing this algorithm with the one in [2]. A third comparison is done with the case where no rate adaptation is done.

In order to achieve this we needed to implement the RTP and RTCP agents as per [9] and [10]. We implemented those parts of the above papers and RFCs that were necessary to carry out our comparison. We started with existing implementation in C++ available in the ns-2 [8] simulator and we modified and enhanced it to fit our needs.

This section describes some implementation details used during the simulation for the two classes involved RTPAgent77 and RTCPAgent77 and the related TCL scripts. The definition of the 54 scenarios was done in TCL scripts with the addition of a shell script that automatically launches the simulation of all them without human intervention.

4.5.1 RTPAgent77 in C++

RTPAgent77::RTPAgent77()

1. In the constructor the object variables are initialized and with the use of the function *"bind"*, some of those variables are available from TCL scripts.

RTPAgent77::start()

1. Sends the first packet and schedules for sending a new packet in *interval_* secs.

RTPAgent77::timeout()

1. *interval_* is the variable where we keep the current packetization time for the endpoint. The timeout is re-scheduled to this interval_ (e.g. 20 ms) to send a new RTP packet.

RTPAgent77::sendpkt() & RTPAgent77::makepkt()

1. Sends a new RTP packet of current size with new sequence number incremented by one.
2. Added code to send the packet based on the current size stored as a global variable in the agent.
3. For voice quality analysis we filled the payload of the packet with actual voice data read from an input file.

RTPAgent77::recv()

1. Reception of RTP packet and storage on degraded output file. Once the simulation is completed we compare this output file with the input file using PESQ and provide a MOS score for the scenario.
2. In the RTCP "Quick" implementation we needed to find out for missing packets. We chose to do this in the RTP Recv. The algorithm is to identify a missing packet and call the RTCP Agent to send a NACK to the remote node.
3. The second modification to this function consists of finding out a sequence of packets missing from the received packets. For this purpose we maintained a histogram and filled up whenever we found the gap between the packets to be greater than 0.

RTPAgent77::command()

1. Definition of miscellaneous functions that can be called from TCL scripts. We added the commands to initialize names for files needed during simulation.

4.5.2 RTCPAgent77 in C++

This section describes how quickly the RTCP sender sends the RTCP Receiver Reports or NACKs and what the behavior of the RTCP receiver is when it receives one of those.

RTCPAgent77::RTCPAgent77()

1. Initialization and binding of variables in the constructor.

RTCPAgent77::start()

1. Schedules the transmission of the first RTCP packet after the RTCP *interval_* expires.

RTCPAgent77::calcFracLost()

1. The fraction of packet loss calculation was implemented as per section A.3 of RFC-3550. This number is calculated before sending an RTCP Receiver Report each 5 seconds.

RTCPAgent77:: sendpkt

Depending on the scenario a different version of RTCP will be in use. [9] describes three modes of RTCP traffic, two of which are implemented in this project.

1. RTCP 3550[10] or RTCP Regular mode: In this mode the RTCP receiver reports were sent at a regular interval based on the total bandwidth available to the user which in the case of unicast was the minimum which is 5 seconds. The Receiver reports contained one important field called fraction lost which was used in the RTCP recv algorithm.
2. Immediate RTCP Mode: In this mode, if the RTCP bandwidth allows (5% of the current RTP bandwidth), we allocated for each 5 second interval some number of NACKs to be sent. If the allocated number of NACKs is exceeded we switch back to regular mode. This was an implementation of the immediate mode in [9].

RTCPAgent77::recv()

On the reception on RTCP Receiver Reports, 3 possible reactions are possible based on the scenario plus there is the case of the reception of a NACK:

1. RTCPAgent77::rtcpRecvPatent() - Sender-Based Rate Adaptation based on Patent [5]

Figure 3 depicts the algorithm implemented and Table 3 shows the packetization times used for different levels of congestion. This table shows the packetization time that should be in use for a given packet loss in the network. Whenever the packet loss is within an acceptable range 0-2% the optimal packet size is in use (20 ms for G711 and 30 ms for G729). If the packet loss received is greater than the packet loss range in the table, then the ptime is incremented by 10ms intervals *progressively* each time a RTCP report is received until it gets to the value specified in the table.

Packet Loss to Ptime (Patent)

Packet Loss (%)	Ptime G711 (ms)	Packet Loss (%)	Ptime G729 (ms)
0-2	20	0-2	30
2-100	30	2-4	40
		4-6	50
		6-8	60
		8-10	70
		10-100	80

Table 3 – Packet Loss to Packetization Time (Patent [5])

Sender Based Rate Adaptation

Patent

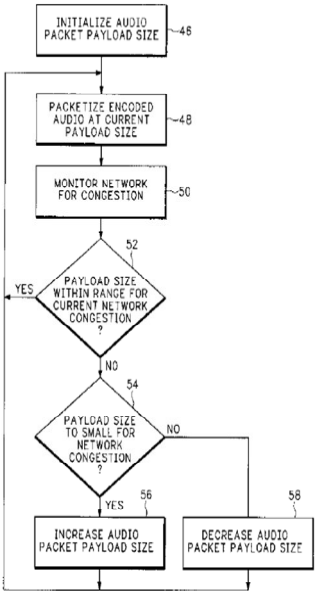


Figure 3 – Patent Sender-Based Rate Adaptation

2. RTCPAgent77::rtcpRecvPaper1996() - Sender-Based Rate Adaptation based on paper [2]

In this mode a low pass filter is used to delay the reaction to RTCP receiver reports using a value of $\alpha = 0.3$ as suggested in that paper (see Figure 4). This approach gives more importance to the historical data on the packet loss than to new received reports creating the net effect of delaying the reaction to packet loss experienced in the network. λ is the smoothed packet loss and it is calculated as:

$$\lambda = 0.7 * \lambda + 0.3 * b$$

where b is the received packet loss percentage.

This calculated smoothed packet loss λ is compared with a range where λ_u is set to 2% while λ_c is set to 4%. If λ lies in the range 0-2% then the packet size is decreased which minimizes the end-to-end delay; if λ lies within 2 and 4 % then NO changes to the packet size are done; and finally if λ is greater than 4% packet size is incremented in an attempt to control congestion.

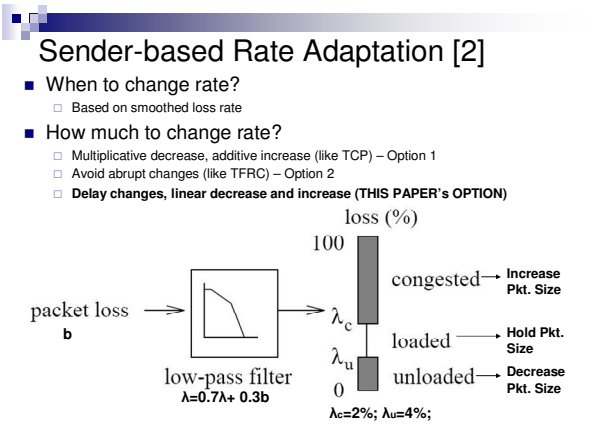


Figure 4 – Sender-based rate adaptation [2]

3. RTCPAgent77::rtcpNone() - No adaptation (current behavior)

In this scenario, no packet size changes are done. This is done for comparison purposes.

4. RTCPAgent77::rtcpHandleNack() - NACK reception

For the scenarios implementing [9], on reception of a NACK, packet size is always increased by 10 ms up to the maximum allowed by the codec.

RTCPAgent77::sendnack()

1. If a gap is recognized by the RTPAgent77::recv sendnack() is called and a NACK is ONLY sent if the allowed number of events is greater than zero. This controls the number of NACKs never exceeds 2.5% of the available bandwidth.

RTCPAgent77::getNoEventsPer10Sec()

1. Assures that the number of NACKs never exceed 2.5% of the allowed bandwidth.

RTCPAgent77::timeout()

1. For both modes of operation, the timeout is re-scheduled to 5 sec as the minimum possible specified in [10].
2. For immediate mode we calculate the allowed number of events per second, to verify if the immediate mode with NACKs is permitted.

RTCPAgent77::command()

1. Miscellaneous commands are defined. We added "*rtcpToRtp"* which connects the RTCP77 and RTP77 agents in a node with a pointer.

4.5.3 Codecs G711 and G729

The codecs were simulated based on the size of the payload per each 10 ms step in each of the codec. For instance, if a particular agent was using G711, each 10ms of voice would result in 80 bytes of data. And for G729 based senders each 10ms of voice would imply 10 bytes of data.

4.5.4 TCL Scenario scripts

In the TCL scripts we define that each node in the topology simulating a phone is running a RTPAgent77 and RTCPAgent77 agent as defined in the previous sections. This is done via the "*attach-agent*" command. We connect in each node the RTP and RTCP agents using the command *"rtcpToRtp"*

The simulation starts at 0 sec where node 0 starts, other nodes start in 5 sec intervals (randomized). All phones start using the optimal packet size for the codec.

Each endpoint node is associated with a pair node in the remote location. Figure 5 shows a couple of phones communicating where the relationship between agents is shown with connecting arrows. RTP sender sends packets and RTP receiver stores them in a file for later analysis, every 5 sec the RTCP sender calculates the fraction lost and sends RTCP Receiver Reports. The RTCP receiver on originating node extracts this feedback and changes the packet size depending on the sender-based rate adaptation in use by the algorithm. If NACKs are in use those are sent as soon as gaps are detected in the incoming RTP stream, the RTCP receiver increases the packet size on reception of a NACK.

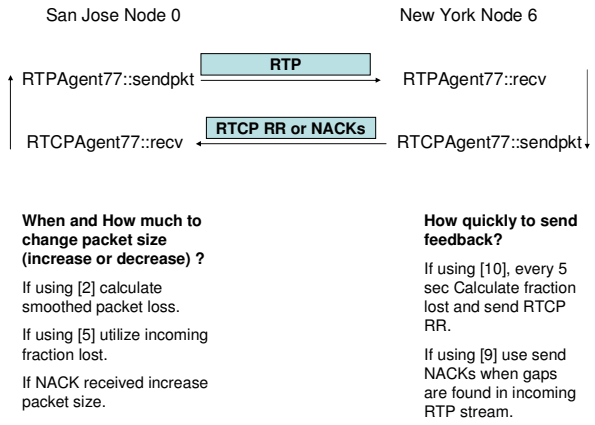


Figure 5 – ns-agent relationships

The *record.lib* library is a sub-routine that creates the files that are used later to create the following graphs using *xGraph and imagemagic.org*. This function is called at regular intervals in order to create the input files for the following graphs.

1. Bandwidth (Kbps) over time (sec)
2. Packet loss (%) over time (sec)
3. Smoothed Packet Loss (%) over time (sec)
4. End-to-End delay (sec) over time (sec)

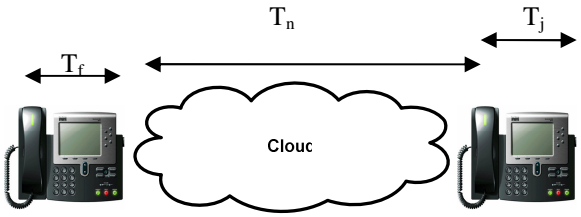
We developed a shell script that automatically runs all 54 scenarios. It has 4 nested loops to cover all the variations under study. For each script 9 graphs are generated, plus the associated trace files, a log file and animation file for *nam* later play. We run the simulations for 300 sec (5 min).

If there is a need to extend the number of endpoints in the simulation the Sources variable could be increased and the wanBW could be increased as well reach higher link capacities such as T1 or T3.

For troubleshooting purposes the function *gen-map* was extremely useful to identify all the nodes and components in the simulation.

Delay calculation

The end-to-end delay is the packetization time (T_f) plus the network delay (T_n) plus the Time at the jitter buffer (T_j). The packetization time is varied over time, the network delay is a constant of 30ms and the Time at the jitter buffer is assumed that adapts an on average is $2*T_f$. Tn is calculated as Time at sender – Time at the receiver since we are assuming synchronized clocks, but for these simulations is fixed to 30 ms.



5. EVALUATION CRITERIA

Scenarios will be evaluated based on the following criteria:

IF the following items are met

- MOS > 3.6, calculated by using PESQ [13].
- Packet loss is less than 2% over the simulation period.
- End-to-end delay is less than 150 ms over the simulation period (ITU G.114).
- Reaction time to congestion should minimal.
- QoS Oscillations should not be present.
- Duration of clips less than 50 ms.

THEN,

The scenario contains a combination of Sender-based rate algorithm, RTCP version and codec combination that creates an acceptable solution.

6. SIMULATION QUESTIONS

When we started our simulation these were the questions that we were trying to get an answer to:

a) Is the "new" RTCP profile proposed in [9] necessary for this system to work? If it is not necessary, will the current RFC-3550 RTCP [10] work?

b) Are any of the two sender-based rate adaptation algorithms [2], [5] viable solutions to alleviate congestion?

c) Which scenarios result in an acceptable solution for all the variables under simulation based on the evaluation criteria?

d) Understand the algorithms behavior over time. For example, once congestion is detected how quickly the senders change their rates. Are there any QoS oscillations created by the continuous packet size changes? In the event that the congestion is controlled, how quickly the senders come back to their original rates.

This question was answered, in part, by analysis of the scenarios over time using "nam" (ns graphical user interface).

7. SIMULATION ANALYSIS

This section includes analysis of the results obtained for each of the 54 scenarios studied.

7.1 Scenario 1

Sender	RTCP Mode	Codec	Bandwidth
1996 [2]	3550 [10]	G711	200

In this scenario the bandwidth available in the WAN link is not enough to accommodate the load offered by 4 channels. As a result there is high packet loss in all four channels as high as 75%.

The smoothed packet loss shows how the high frequency components on the packet loss changes over time are filtered delaying the times at which sender-based algorithm proposed in the 1996 paper [2] tries to reduce the rate by increasing to the packetization time to 30 ms, but the algorithm can't control packet loss and rate stabilizes just below 300 Kbps (the aggregate bandwidth for 4 G711 channels). Note: The maximum packetization time for G711 is 30 ms.

120 ms end-to-end delay was measured in this scenario, which is acceptable for all four channels. There are some oscillations on packetization time in use. Every so often the smoothed packet loss falls below 2 and the ptime is changed to 20 ms which causes more packet loss and the cycle continues. This instability is not desirable.

It is to note that RTCP reports are sent every 5 seconds in this scenario, making it slower to react to the packet loss created around 20 sec when the 4th call is offered to the system. It is only after about 25 seconds, that all the channels start increasing their packet size to 30 ms in an attempt to control congestion. This delay reacting to congestion is not desirable.

There are hundreds of clips of 2 to 4 packets which is as much as 120 ms of lost speech. These clips even though are not as long will be noticeable to most users. Few clips of longer duration were observed. The frequency is unacceptable also, since during the 300 sec simulation the maximum allowed number of clips was 5 clips and we measured hundreds of clips.

The voice quality on average for all 4 channels was 3.14, which is not acceptable. It is to note that we only send a file for the first 15 sec of the call on each channel. Since not all the channels start at the same time, the voice quality measurements will differ since for the first 20 sec or so there is not packet loss in the system.

We didn't expect the algorithms to be able to handle this scenario correctly, since the load offered is much higher than the bandwidth available and even increasing the packet size to the maximum on all channels would not decrease the rate enough to control congestion. As a result, this scenario can't be controlled by this combination of RTCP and sender-based rate algorithm.

7.2 Scenario 2

Sender	RTCP Mode	Codec	Bandwidth
1996 [2]	3550 [10]	G711	300

The Offered load received at the originating router escalates in steps of 80 Kbps for 20ms or 96Kbps if 10 ms is in use. Initially, 20 ms is used for all the calls, but if a RTCP Receiver Report is received with the packet fraction lost field equal to zero, then the smoothed packet loss is also zero and the algorithm decreases the packetization time to 10 ms for that channel. That is why some channels are at 80 Kbps and some at 96 Kbps (See Figure 10).

4 G711 channels at 20 ms ptime using 40-byte header create a load of 320 Kbps. We selected a bandwidth size of 300 Kbps which will create congestion at the originating router when the four calls are in progress (about 20 sec into the simulation). At around this time, the sender-based rate adaptation algorithms reacts and 30 ms packets are used instead for the channels where the smoothed packet loss rate is over 4% (See Figures 6 to 11).

With the increase in packet size congestion is controlled momentarily and packet loss decreases. Once smoothed packet loss goes below 2%, the algorithm will increase the rate by decreasing the packet size to 20 ms or even 10 ms, which causes packet loss again. Once the receiver reports contain unacceptable lost rates, the sender-based rate adaptation algorithm increases the packet rate to attempt controlling congestion one more time. This cycle repeats over time creating QoS oscillations.

As a result, of these oscillations on the packet loss experienced by the receiver, the packetization time varies from 10 ms to 20 ms to 30 ms, as the packet loss thresholds are crossed in the sender-based rate adaptation algorithm.

Only short clips of one or two packets are observed for all channels but the number of clips is extremely high,

around 800 per channel in a 5 simulation minute period.

Voice quality was 3.97 which is acceptable, but note that the oscillation effect is not fully measured since PESQ is only calculated for the first 15 sec on each call.

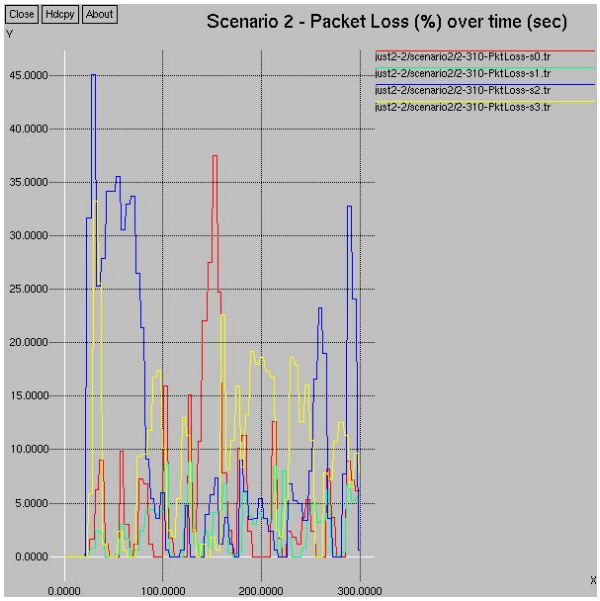


Figure 6 - Scenario 2 - Packet Loss (%) over time (sec) BW=300 Kbps

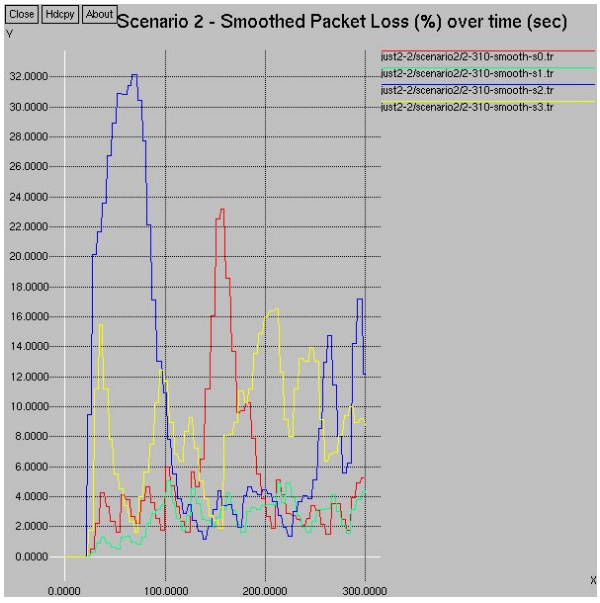


Figure 7 - Scenario 2 - Smoothed - Packet Loss (%) over time (sec) BW=300Kbps

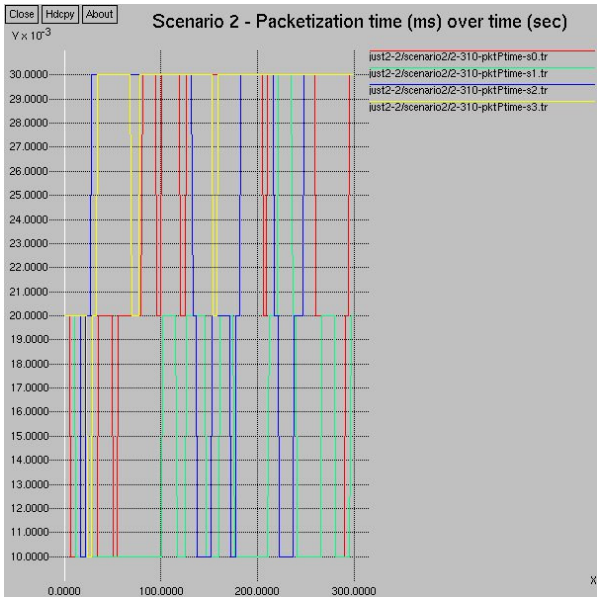


Figure 8 - Scenario2 - Packetization time (ms) over time (sec) BW=300 Kbps

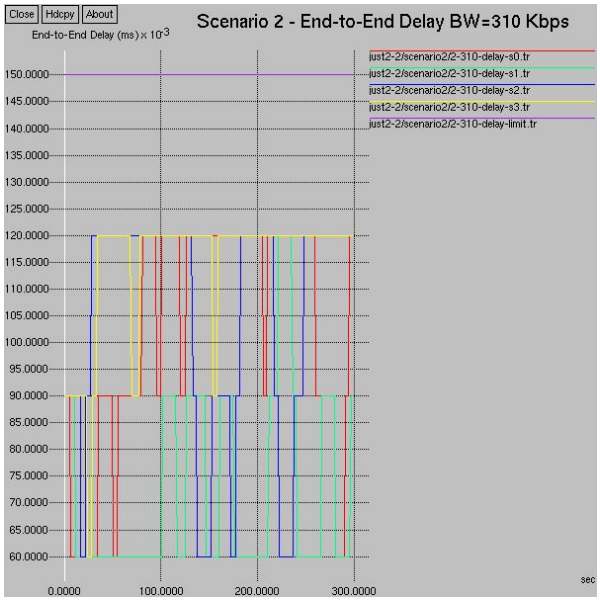


Figure 9 - Scenario 2 - End-to-End Delay (ms) over time (sec) BW=300 Kbps

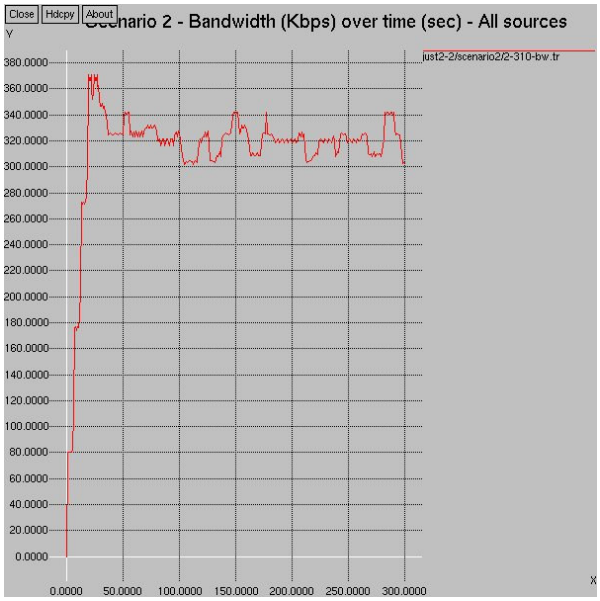


Figure 10 – Scenario 2 – Bandwidth (Kbps) over time (sec) All sources BW=300 Kbps

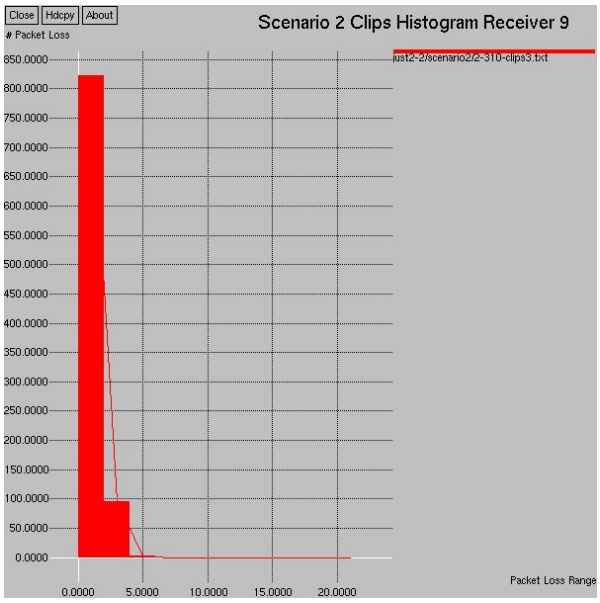


Figure 11 – Scenario 2 – Clips Histogram for receiver 9

7.3 Scenario 3

Sender	RTCP Mode	Codec	Bandwidth
1996 [2]	3550 [10]	G711	500

In this scenario, there is plenty of bandwidth to accommodate all 4 calls; as a result the packet loss is zero for all channels, which also results in the zero smoothed packet loss for all the phones. The sender-based rate adaptation algorithm described in [2] observing decreases the packetization time to 10 ms on all the phones independently (see Figure 12) in order to minimize the end-to-end delay which is 60 ms for each channel. The Bandwidth for 4 G711 at 10 ms packetization time stabilizes at 384 Kbps as shown in Figure 13. Voice Quality is at 4.5, which is excellent, but please note that we are not considering the decrease in quality introduced by the codec itself.

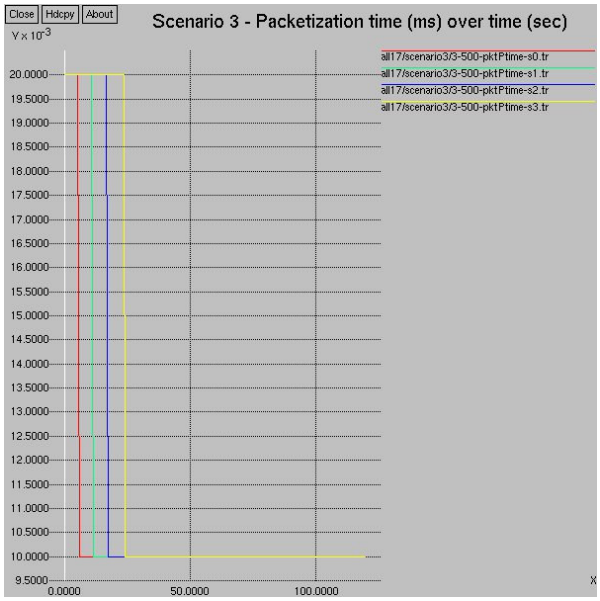


Figure 12 – Scenario 3 – Packetization time (ms) over time (sec)

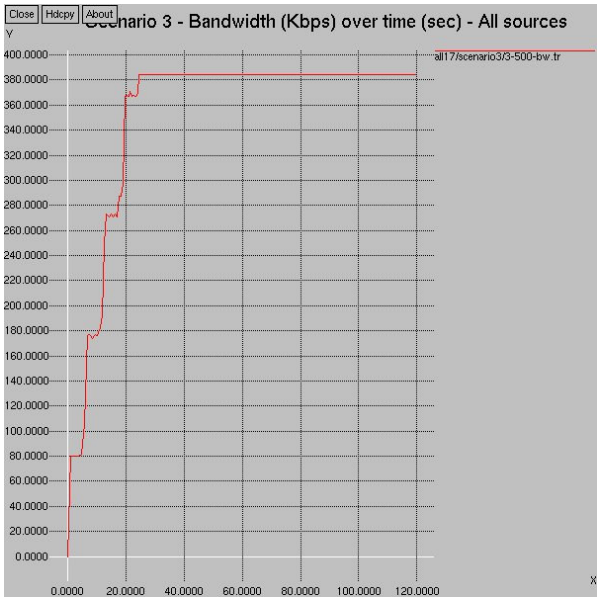


Figure 13 – Scenario 3 – Bandwidth (Kbps) over time (sec)

7.4 Scenario 4

Sender	RTCP Mode	Codec	Bandwidth
1996 [2]	3550 [10]	G729	40

The packetization time is set progressively up to 80 ms, which creates an end-to-end delay of 270 ms. For interactivity, a maximum of 150 ms one way delay is acceptable according to G.114, so this delay is unacceptable. The algorithm is unable to control congestion even using the maximum packetization time and the load offered to the system tends to 48 Kbps (4 G729 phones at 80 ms packetization time). Figure 11 shows packetization time changes up to 80 ms.

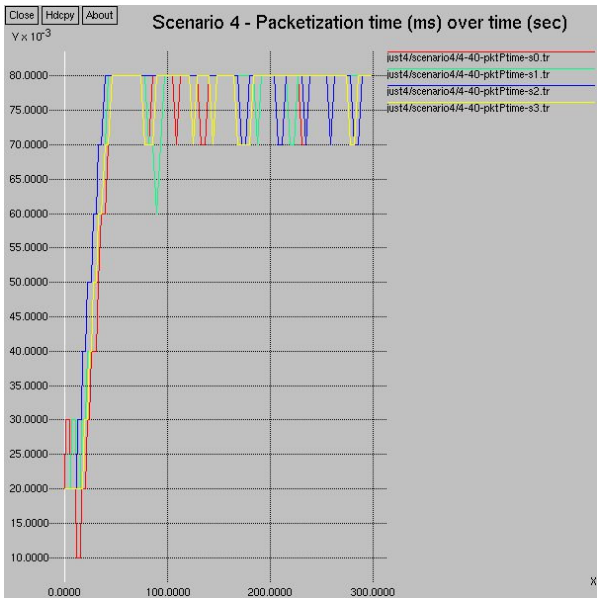


Figure 14 – Scenario 4 – Packetization time (ms) over time (sec)

7.5 Scenario 5

Sender	RTCP Mode	Codec	Bandwidth
1996 [2]	3550 [10]	G729	70

The packet loss is controlled for brief periods of times (See Figure 12). Decreases in packet size, cause an increase in rate, which causes congestion and QoS oscillations. Not all the channels are using the same packet size at any point in time and end-to-end is unacceptable for some channels.

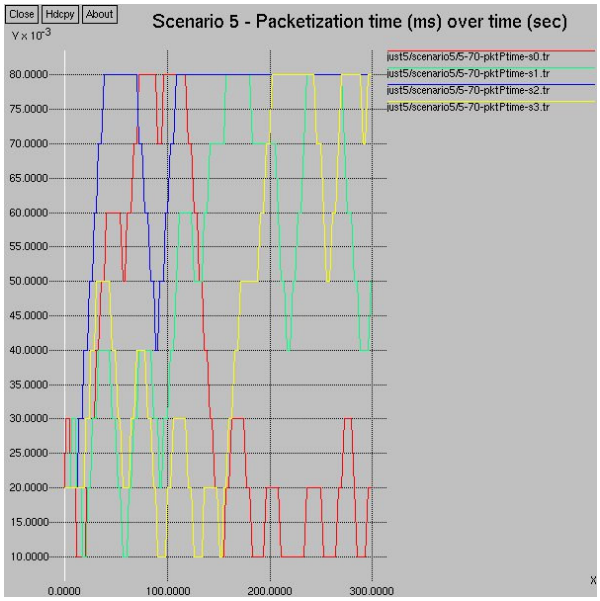


Figure 15 – Scenario 5 – Packetization time (ms) over time (sec)

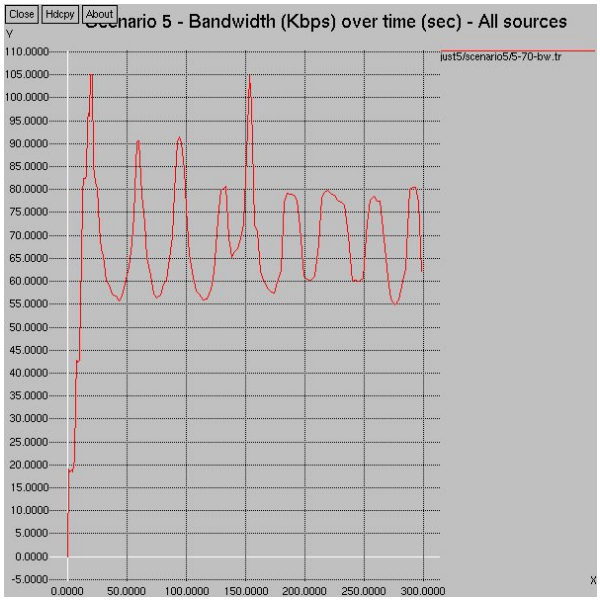


Figure 16 – Scenario 5 – Bandwidth (Kbps) over time (sec) – All sources

7.6 Scenario 6

Sender	RTCP Mode	Codec	Bandwidth
1996 [2]	3550 [10]	G729	200

This scenario minimizes to the end-to-end delay by setting the packetization time to 10 ms for all the phones creating an acceptable end-to-end delay of 60 ms. There is not packet loss and the offered loss stabilizes at 160 Kbps for the 4 G729 channels. This

scenario is similar to scenario 3 where the congestion control algorithm is NOT in use.

7.7 Scenario 7

Sender	RTCP Mode	Codec	Bandwidth
1996 [2]	3550 [10]	Both	100

The algorithms try to react and set the packetization equal to 80 ms for G729 and to 30 ms for G711. The end-to-end delay is 270 ms for G729 which is not acceptable. Longer clips are noticed on this combination of codecs and bandwidth available (See Figure 14). The increase in clips is due to the fact that in this case the offered load is much higher than the available bandwidth.

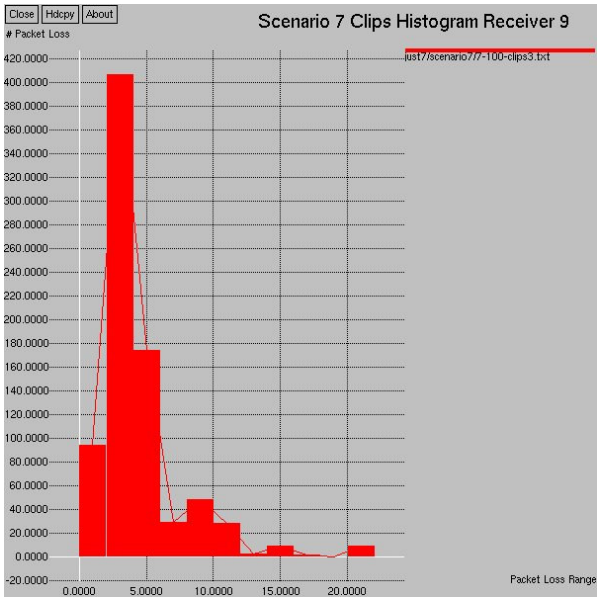


Figure 17 – Scenario 7 – Clips Histogram Receiver 9

7.8 Scenario 8

Sender	RTCP Mode	Codec	Bandwidth
1996 [2]	3550 [10]	Both	180

Nothing changes in comparison with scenarios 2 and 4. The main observation is that there are QoS oscillations also when codecs are combined.

7.9 Scenario 9

Sender	RTCP Mode	Codec	Bandwidth
1996 [2]	3550 [10]	Both	300

Similarly as in scenarios 3 and 6 end-to-end delay is minimized to 60 ms for all phones. No congestion control is invoked.

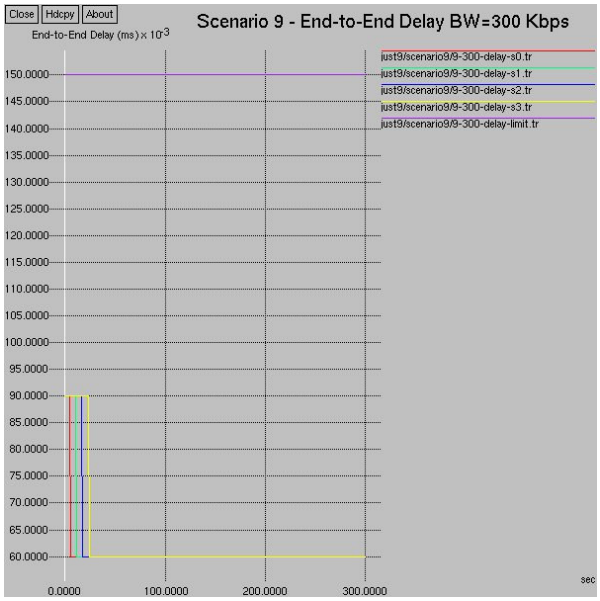


Figure 18 – Scenario 9 – End-to-End Delay BW=300 Kbps

7.10 Scenario 10

Sender	RTCP Mode	Codec	Bandwidth
1996 [2]	Quick [9]	G711	200

Even though the reaction to congestion is immediate, the algorithm cannot control congestion. Bandwidth tends to 300 Kbps which is 4 G711 calls at 30 ms.

7.11 Scenario 11

Sender	RTCP Mode	Codec	Bandwidth
1996 [2]	Quick [9]	G711	300

The reaction to packet loss is a lot quicker (60 ms for this network topology), which causes the frequency of the oscillations to be much higher. The sender-based rate algorithm proposed in [2] is the cause of the oscillations, not the fact that the feedback was given a lot quicker. Figures 16 and 17 clearly show that in this scenario the frequency of the oscillations is much higher. Compare these two figures with Figure 7 where the oscillation is slower due to the fact the RTCP reports is each 5 secs.

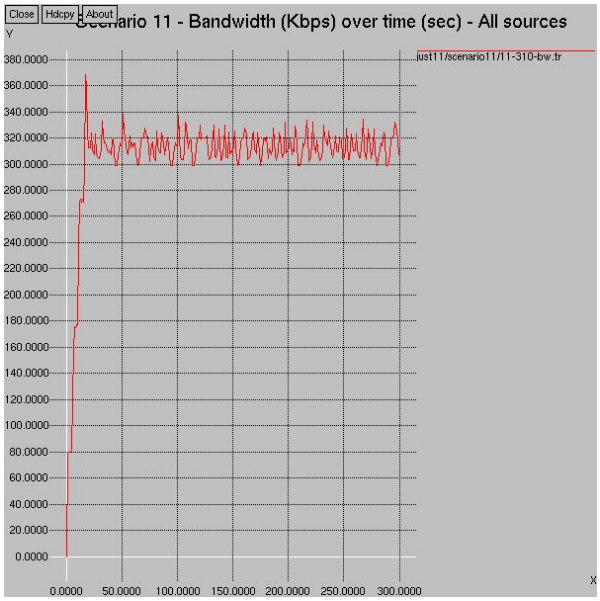


Figure 19 – Scenario 11 – Bandwidth (Kbps) over time (sec) – All sources

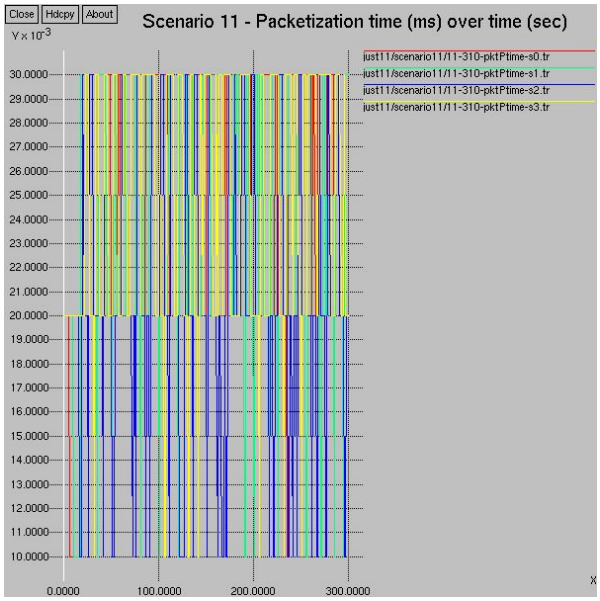


Figure 20 – Scenario 11 – Packetization time (ms) over time (sec)

7.12 Scenario 12

Sender	RTCP Mode	Codec	Bandwidth
1996 [2]	Quick [9]	G711	500

As in previous scenarios where bandwidth is abundant, the system is stable and no congestion control is invoked.

7.13 Scenario 13

Sender	RTCP Mode	Codec	Bandwidth
1996 [2]	Quick [9]	G729	40

The algorithm can't control congestion as in other scenarios where bandwidth is scarce.

7.14 Scenario 14

Sender	RTCP Mode	Codec	Bandwidth
1996 [2]	Quick [9]	G729	70

There are also oscillations as in other similar scenarios where there is not enough bandwidth, but the algorithm can control congestion momentarily. An interesting observation that is seen clearly in this scenario is that not all the channels perceive the same packet loss and as a consequence the packetization time in use in each channel is different. See Figure 21.

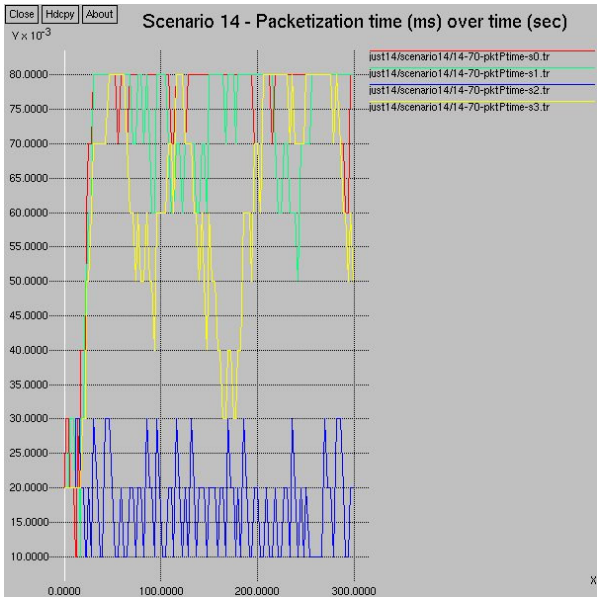


Figure 21 – Scenario 14 – Packetization time (ms) over time (sec)

7.15 Scenario 15

Sender	RTCP Mode	Codec	Bandwidth
1996 [2]	Quick [9]	G729	200

This is a scenario where there is enough bandwidth and no congestion control is necessary.

7.16 Scenario 16

Sender	RTCP Mode	Codec	Bandwidth
1996 [2]	Quick [9]	Both	100

Algorithms can't react to such offered load to control congestion. G711 calls use 30 ms while G729 calls use 80 ms and the bandwidth tends to 175 Kbps as

expected. The end-to-end delay is unacceptable for G729 calls. No oscillations were observed in this scenario because the smoothed packet loss never went below 2%. Packet loss appears as spikes, which different from other scenarios, it was not clear to us why that was the case.

7.17 Scenario 17

Sender	RTCP Mode	Codec	Bandwidth
1996 [2]	Quick [9]	Both	180

Several times during this scenario NACKs were not sent because the number of allowed events to report exceeded the maximum allowed, relying then on the regularly scheduled RTCP reports. Oscillations are present in this scenario. For the G711 phones, the packetization varies from 10 to 30 ms, while for the G729 phones the packetization time varies from 10 to 80 ms.

7.18 Scenario 18

Sender	RTCP Mode	Codec	Bandwidth
1996 [2]	Quick [9]	Both	300

There is no congestion and all channels use the lowest possible packetization time (10 ms) to minimize the end-to-end delay (60 ms) and the total offered load tends to 272 Kbps as expected (see Table 6 – Codec and Packetization Time Summary Table for 4 channels)

7.19 Scenario 19

Sender	RTCP Mode	Codec	Bandwidth
Patent [5]	3550 [10]	G711	200

This algorithm depends directly on the percentage packet loss received on the RTCP Reports (there is not smoothed packet loss as in [2]). The offered load tends to 300 Kbps. Packet loss is above 2% for the first 3 channels, which based on the patent algorithm force those phones to use 30 ms packetization time. For the 4th channel, the packet loss reaches zero every 20 sec or so which causes the use of 20 ms and packet loss in the immediate 5 sec interval, but that increases the rate above the available bandwidth and packet loss occurs and the cycle repeats. See Figure 22

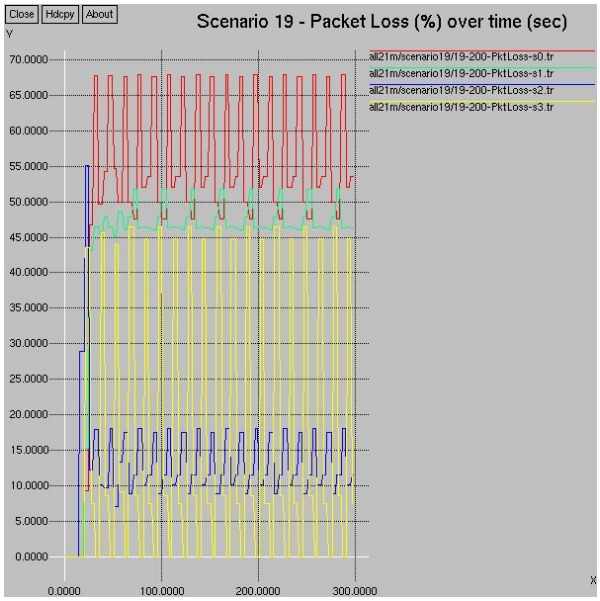


Figure 22 – Scenario 19 – Packet Loss (%) over time (sec)

7.20 Scenario 20

Sender	RTCP Mode	Codec	Bandwidth
Patent [5]	3550 [10]	G711	300

In this scenario, only one of the phones experience packet loss and for that channel the packetization time is 30 ms, while the other 3 stayed at 20 ms. This shows the possibility that not all the channels experience the same amount of loss and as result the sender-based rate adaptation works different per each channel. See Figure 23.

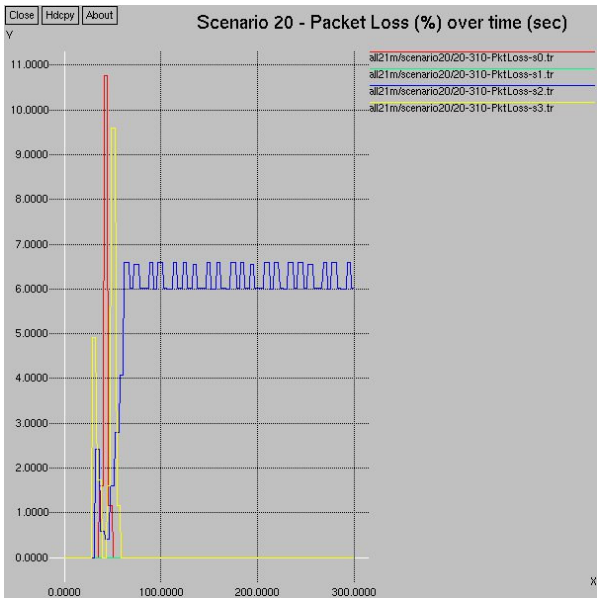


Figure 23 – Scenario 20 – Packet Loss (%) over time (sec)

7.21 Scenario 21

Sender	RTCP Mode	Codec	Bandwidth
Patent [5]	3550 [10]	G711	500

The important fact to note in this scenario is that with patent algorithm, the minimum packetization time in use is the optimal which creates a larger end-to-end delay than the algorithm in [5]. However, this end-to-end delay is still acceptable (90 ms). There is not packet loss and the bandwidth in use after the four calls is 320 Kbps all using the G711 codec.

7.22 Scenario 22

Sender	RTCP Mode	Codec	Bandwidth
Patent [5]	3550 [10]	G729	40

Over time all the channels tend to use 80 ms, then the best the algorithm can do is then is to get the bandwidth to 48 Kbps, which is not enough. Oscillations occur when the packet loss goes below 2% momentarily.

7.23 Scenario 23

Sender	RTCP Mode	Codec	Bandwidth
Patent [5]	3550 [10]	G729	70

This scenario shows how the increment on packet size is done gradually in steps of 10 ms, but it does not get to the maximum of 80 ms since the congestion is controlled momentarily, but the oscillations also occur. See Figure 24

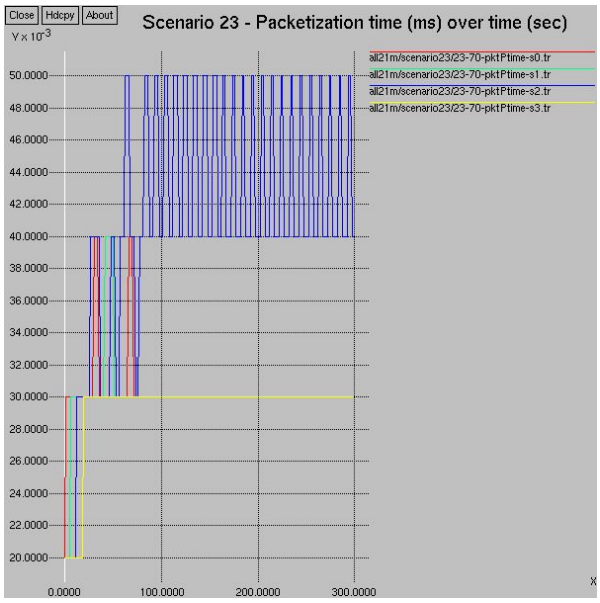


Figure 24 – Scenario 23 – Packetization time (%) over time (sec)

7.24 Scenario 24

Sender	RTCP Mode	Codec	Bandwidth
Patent [5]	3550 [10]	G729	200

This scenario correct operation of the algorithm when there is enough bandwidth to support all 4 calls.

7.25 Scenario 25

Sender	RTCP Mode	Codec	Bandwidth
Patent [5]	3550 [10]	Both	100

2 G711 channels use 30 ms with an end-to-end delay of 120 ms. 2 G729 channels use 80 ms with an unacceptable end-to-end delay of 270 ms. See Figure 25. There are no oscillations on this scenario, which is attributed to the fact that in order to support the 4 calls 240 Kbps are required which is a lot more than 100 Kbps, which creates very high packet loss. After the packet size adjustments the offered load is about 175 Kbps, which is much bigger than 100 Kbps and the packet loss is always above 20%. As a result, the maximum packet size is always in use.

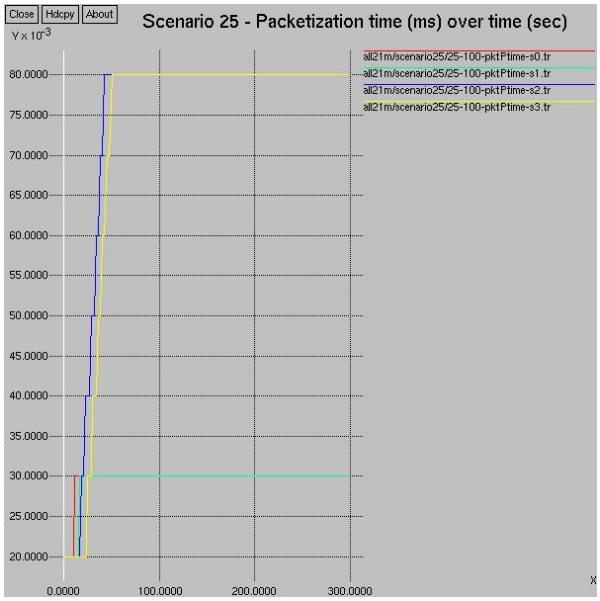


Figure 25 – Scenario 25 – Packetization time (ms) over time (sec)

7.26 Scenario 26

Sender	RTCP Mode	Codec	Bandwidth
Patent [5]	3550 [10]	Both	180

The fact that two codecs are in use did not change the fact that there are oscillations. Two channels did not

have packet loss while two channels had (one G711 and one G729).

7.27 Scenario 27

Sender	RTCP Mode	Codec	Bandwidth
Patent [5]	3550 [10]	Both	300

When there is plentiful bandwidth the optimal packet size is in use for each codec. End-to-End delay is acceptable for all 4 calls and there is no packet loss.

7.28 Scenario 28

Sender	RTCP Mode	Codec	Bandwidth
Patent [5]	Quick [9]	G711	200

Even though the reaction to congestion is immediate, the algorithm cannot control congestion. Bandwidth tends to 300 Kbps which is 4 G711 calls at 30 ms.

7.29 Scenario 29

Sender	RTCP Mode	Codec	Bandwidth
Patent [5]	Quick [9]	G711	300

In this scenario oscillations occur between 20 and 30 ms, packet sizes. This is due to the quick reaction to the network bandwidth changes. The congestion is controlled momentarily however as soon as the senders discover available bandwidth the packet size is decreased, causing congestion all over again, leading to oscillations. Again as seen in Scenario 20 most packet loss occurs in one channel, but this algorithm is fairer than Scenario 20.

7.30 Scenario 30

Sender	RTCP Mode	Codec	Bandwidth
Patent [5]	Quick [9]	G711	500

When there is plentiful bandwidth the optimal packet size is in use for each codec. End-to-End delay is acceptable for all 4 calls and there is no packet loss.

7.31 Scenario 31

Sender	RTCP Mode	Codec	Bandwidth
Patent [5]	Quick [9]	G729	40

Over time all the channels tend to use 80 ms, then the best the algorithm can do is then is to get the bandwidth to 48 Kbps, which is not enough. Oscillations occur when the packet loss goes below 2% momentarily. See Figure 26.

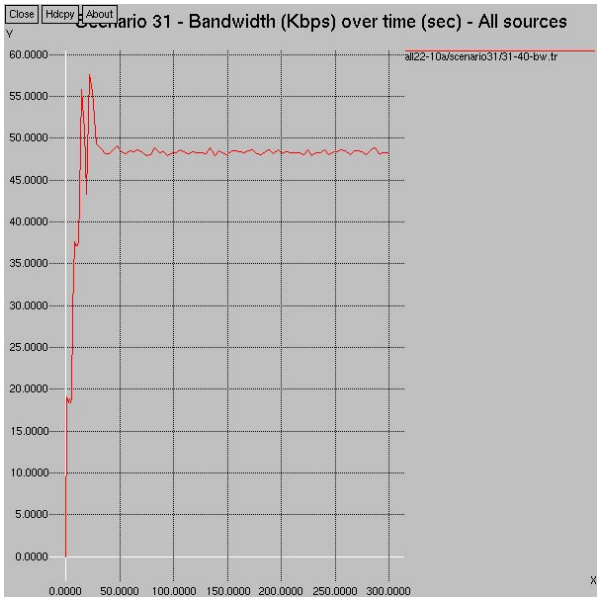


Figure 26 – Scenario 31 – Bandwidth (Kbps) over time (sec)

7.32 Scenario 32

Sender	RTCP Mode	Codec	Bandwidth
Patent [5]	Quick [9]	G729	70

In this scenario unlike Scenario23 the oscillations are seen for all the channels, it appears that the Quick algorithm is fair to all the channels; this is due to all the channels quickly reacting to the decreasing and increasing bandwidth situations.

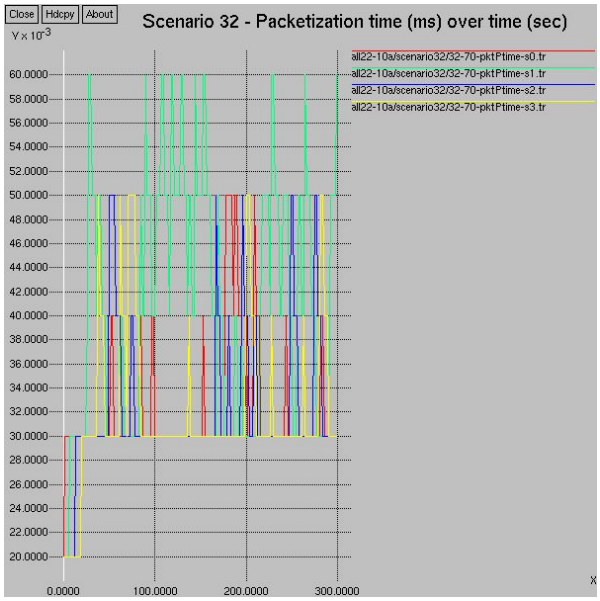


Figure 27 – Scenario 32 – Packetization time (ms) over time (sec)

7.33 Scenario 33

Sender	RTCP Mode	Codec	Bandwidth
Patent [5]	Quick [9]	G729	200

Again in this scenario all the channels have plentiful bandwidth and do not react.

7.34 Scenario 34

Sender	RTCP Mode	Codec	Bandwidth
Patent [5]	Quick [9]	Both	100

All the channels reach the maximum packet sizes. Oscillations are not seen because the bandwidth does not even reach close to 2% of the maximum allowed. End-to-end delay is not acceptable for G729 channels.

7.35 Scenario 35

Sender	RTCP Mode	Codec	Bandwidth
Patent [5]	Quick [9]	Both	180

There are large oscillations when ever the bandwidth is momentarily controlled in both G711 and G729 channels. This again is due to the quick nature of the reaction to Congestion.

7.36 Scenario 36

Sender	RTCP Mode	Codec	Bandwidth
Patent [5]	Quick [9]	G711	300

Lot of available bandwidth implies no congestion control.

The next 18 scenarios do not use sender based rate adaptation; however the packet loss is an interesting aspect to note.

7.37 Scenario 37

Sender	RTCP Mode	Codec	Bandwidth
None	3550 [10]	G711	200

A lot of packet loss occurs in one channel, 50% which is unfair and is unacceptable.

7.38 Scenario 38

Sender	RTCP Mode	Codec	Bandwidth
None	3550 [10]	G711	300

Again about 28% packet loss is observed consistently on one channel.

7.39 Scenario 37

Sender	RTCP Mode	Codec	Bandwidth
None	3550 [10]	G711	500

Enough bandwidth available, so no loss.

7.40 Scenario 40

Sender	RTCP Mode	Codec	Bandwidth
None	3550 [10]	G729	40

Packet losses ranging from 25 to 65% on all four channels are observed. This will cause degradation in voice data.

7.41 Scenario 41

Sender	RTCP Mode	Codec	Bandwidth
None	3550 [10]	G729	70

Packet losses are observed on all channels. If the packet losses are spread over and not continuous it still might be acceptable to the users.

7.42 Scenario 42

Sender	RTCP Mode	Codec	Bandwidth
None	3550 [10]	G729	200

No packet loss was observed, this is due to a lot of bandwidth available to all channels.

7.43 Scenario 43

Sender	RTCP Mode	Codec	Bandwidth
None	3550 [10]	Both	100

Over 50% packet loss seen on some channels this is clearly unacceptable for voice communications.

7.44 Scenario 44

Sender	RTCP Mode	Codec	Bandwidth
None	3550 [10]	Both	180

Similar to scenario 41, but again the loss may be acceptable or unacceptable based on how the loss is distributed.

7.45 Scenario 45

Sender	RTCP Mode	Codec	Bandwidth
None	3550 [10]	Both	300

Plenty of available bandwidth and hence no packet loss

7.46 Scenario 46

Sender	RTCP Mode	Codec	Bandwidth
None	Quick [9]	G711	200

There is a total packet loss of 385 seconds which is large for each channel simulated for an average of little less than 300 seconds. This scenario compares well with scenario37 where the packet losses seen are much smaller. The reason for this being RTCP Quick uses

5% of current RTP bandwidth and that results in packet losses even on RTP.

7.47 Scenario 47

Sender	RTCP Mode	Codec	Bandwidth
None	Quick [9]	G711	300

This again compares with Scenario 38 and the total packet loss was a little greater.

7.48 Scenario 48

Sender	RTCP Mode	Codec	Bandwidth
None	Quick [9]	G711	500

No losses observed

7.49 Scenario 49

Sender	RTCP Mode	Codec	Bandwidth
None	Quick [9]	G729	40

Packet loss was again a little greater than Scenario 40

7.50 Scenario 50

Sender	RTCP Mode	Codec	Bandwidth
None	Quick [9]	G729	70

Packet loss observed may be acceptable but it was greater than Scenario 41.

7.51 Scenario 51

Sender	RTCP Mode	Codec	Bandwidth
None	Quick [9]	G729	200

No losses observed

7.52 Scenario 52

Sender	RTCP Mode	Codec	Bandwidth
None	Quick [9]	Both	100

Overall packet loss greater than Scenario 43 due to bandwidth usage by RTCP Quick

7.53 Scenario 53

Sender	RTCP Mode	Codec	Bandwidth
None	Quick [9]	Both	180

This follows the same pattern as all the None + "Quick" algorithms.

7.54 Scenario 54

Sender	RTCP Mode	Codec	Bandwidth
None	Quick [9]	Both	300

No issues with any of the channels

7.55 Scenario Summary

In addition to the scenario graphs we also worked on obtaining the summary of all the scenarios in a table.

We identified three key data that will enable us to do so. The following tables give us a very good idea about the performance of each algorithm in comparison to others having the same parameters. The three data that are important are:

7.55.1 Voice Quality for all scenarios

We used a standard voice quality measurement tool called PESQ to compare the output wave file that we obtained from each of the channels with the input file. The PESQ software generates numbers from 0-4.5 where 4.5 is the highest quality and lower this number the worse the quality of the file is. Based on this we obtained the following numbers:

What is interesting to note is the case [Patent + Quick] with the middle tier in the bandwidth, the numbers really show that the advantage of converging quickly to the optimal bandwidth.

Scenarios 37 and 46 provide voice quality results that do not match expectations. We consider this be incorrect, but couldn't find the root cause of the anomaly.Scenarios where bandwidth is abundant result in 4.5 PESQ score, 0 packet loss and 0 clips, which proves correctness of our implementation.

7.55.2 Total Packet Loss per scenario groups

The way this was computed was to record the number of packets lost in ms and add them up for each channel and then sum up all the four channels. The numbers shown below are in seconds. The reason to choose seconds instead of bytes of data was to be able to effectively compare the G711 and G729 codecs.

Please note that the packet loss in case of the middle tier of bandwidth where the sender based rate adaptation has any effect has the least losses in the case of [Patent+Quick] being used. This again confirms the results seen fro the above table. The packet losses are reduced from 86 seconds to 5 seconds when both codecs are used.

7.55.3 Clip length per scenario groups

This value is obtained by dividing the average packet loss by the number of clips. This number will show if the clips are discernable by the user.

In case of G711 and G729 most of the clips for the middle tier of bandwidth are fine with the users.

Voice Quality (PESQ Score) for all Scenarios

	G711	G711	G711	G729	G729	G729	Both	Both	Both
Bw	**200**	**300**	**500**	**40**	**70**	**200**	**100**	**180**	**300**
1-9 [1996 + 3550]	3.1	3.7	4.5	2.1	2.9	4.5	2.3	3.3	4.5
10-18 [1996 + Quick]	3.3	4.3	4.5	2.6	3.8	4.5	2.3	4.0	4.5
19-27 [Patent + 3550]	3.6	4.5	4.5	2.3	3.7	4.5	2.2	4.0	4.5
28-36 [Patent + Quick]	3.2	**4.5**	4.5	2.4	**4.3**	4.5	2.0	4.1	4.5
37-45 [None + 3550]	*3.8**	3.9	4.5	1.7	3.4	4.5	1.6	3.7	4.5
46-54 [None + Quick]	*3.4**	4.1	4.5	1.6	3.3	4.5	2.6	3.1	4.5

Average Clip (ms) for all Scenarios

	G711	G711	G711	G729	G729	G729	Both	Both	Both
Bw	**200**	**300**	**500**	**40**	**70**	**200**	**100**	**180**	**300**
1-9 [1996 + 3550]	45	19	0	95	30	0	76	28	0
10-18 [1996 + Quick]	51	23	0	91	32	0	76	27	0
19-27 [Patent + 3550]	43	24	0	87	32	0	68	29	0
28-36 [Patent + Quick]	56	24	0	93	32	0	79	28	0
37-45 [None + 3550]	20	20	0	73	31	0	60	30	0
46-54 [None + Quick]	26	20	0	74	31	0	50	47	0

Total Packet Loss (sec) for all Scenarios

	G711	G711	G711	G729	G729	G729	Both	Both	Both
Bw	**200**	**300**	**500**	**40**	**70**	**200**	**100**	**180**	**300**
1-9 [1996 + 3550]	141	25	0	92	25	0	196	21	0
10-18 [1996 + Quick]	136	7	0	73	6	0	198	8	0
19-27 [Patent + 3550]	138	15	0	83	7	0	180	17	0
28-36 [Patent + Quick]	136	**6**	0	73	**2**	0	184	5	0
37-45 [None + 3550]	*53**	25	0	194	23	0	204	59	0
46-54 [None + Quick]	*111**	25	0	194	24	0	213	86	0

Table 4 – Summary Voice Quality, Clips and Packet Loss for all 54 scenarios

* Experiment anomaly. Results are incorrect.

8. CONCLUSIONS

By analyzing the results obtained via simulation we can make the following conclusions in regards to the idea described in the patent [5] and related topics:

- Both sender-based rate algorithms studied showed QoS oscillations, even by smoothing the packet loss as suggested in the 1996 paper [2]. As a result, we suggest that a mechanism to prevent oscillations should be added to these algorithms.
- A faster packet loss reporting as described in [9] appears to be a better option where interactivity is important for the application such as is the case for VoIP calls. However, the oscillations were also more frequent with this option. Comparing [Patent+Quick] and [Patent + 3550] or [1996 +Quick] and [1996 + 3550] we find that the "Quick" option aids these algorithms to converge more quickly and hence outperforms the 3550 by a large margin. See rows 1 and 2; also 3 and 4 in Table 4.
- The combination of sender-based rate adaptation algorithm described in [5] combined with Immediate RTCP feedback [9] produce the best results. The early-feedback combined with no delay in reaction after receiving RTCP packet proved to be better than other combinations. See scenario 32 in Table 4.
- We confirmed that increasing the packet size as described in the patent [5] is independent from the codec in use (G711 or G729). Performance of the algorithms was the same for both of them or even when codecs where combined. See for example, the behavior observed on scenarios (1,2,3) vs. (4,5,6) vs. (7,8,9).
- In cases where the load offered is much higher than the bandwidth available, increasing the packet size to the maximum on all channels would not decrease the rate enough to control congestion, however using VoIP congestion control will improve the voice quality perceived by subscribers even with the undesirable oscillations. See for example, scenarios 10, 13 and 16.
- The sender-based rate adaptation algorithm presented in [2] minimizes end-to-end delay during periods of no congestion, while the algorithm presented in the patent [5] uses the optimal packet size during the periods of no congestion, which does not minimize the end-to-end delay. See Figure 12.
- Comparing the summary tables for voice quality and packet loss, we find that between [1996+3550] and [Patent + 3550] the [Patent + 3550] outperforms the [1996 + 3550] by a large amount. This makes us conclude that Patent is a better option because of the fast convergence. The patent sender-based rate adaptation algorithm reacts quicker, since it does not have low pass filter implemented. See rows 1 and 3 in Table 4 in the case where the bandwidth is just enough for the calls.
- Comparing any scenario with no rate adaptation and with any rate adaptation there is a minimal to substantial improvement in the quality of the voice depending on the available bandwidth. This forces us to conclude that the Rate adaptation algorithms should be used.
- In case no rate adaptation is used it is the best to use RTCP specified in RFC 3550, since the use of Immediate Feedback will only consume bandwidth and the feedback in the form of NACKs will not be used by the sender.

9. FUTURE WORK

As future work we have identified the following items:

- Algorithms should be developed aimed at reduction of QOS oscillations in sender based rate adaptation.
- Impact of changing packet size on jitter buffer depth is a practical question that needs to be understood, before implementing this idea and it will be left for further research. We assumed that the jitter buffer automatically adapts its size based on the size of the packets received.
- Change LAN from Ethernet to 802.11 where bandwidth varies and is smaller and bit error rates are higher. Hole and Tobagi [6] may be used as a starting point on this area.
- Instead of using packet loss as indicator of congestion which is a reactive measure, use jitter measurements as a proactive method to indicate congestion.

- Investigate the gradual support scenario where some endpoints support the idea proposed in this paper while some others do not.
- Implement in endpoints and test over a real network.
- Perform the simulation with competing TCP sessions including the TCP Friendly Rate Control (TFRC) Voice [16] as the congestion control mechanism.
- Use MOS score from RTCP XR as the feedback metric to control the sender-based rate algorithms.
- Simulate using higher bandwidth on the WAN such as T1, 2xT1 and T3.
- Simulate bi-directional transmission of RTP packets.
- Perform a literature survey of sender-based rate algorithms to find one with better properties for QoS oscillations.
- Study the effect of choosing different λ limits and the use of a different α in [2].
- Introduce queuing delay at intermediate routers and repeat the experiments.

10. ACKNOWLEDGMENTS

The authors would like to thank Rajesh Kumar for suggesting the simulation project and Cary Fitzgerald for his support explaining the patent idea.

11. REFERENCES

[1] Andersen, S., Duric A., Astrom, H., Hagen R., Kleijn, W, Linden J., *Internet Low Bit Rate Codec (iLBC),* RFC 3951, December 2004.

[2] Busse, I., Deffner, B. and Schulzrinne H., *Dynamic QoS Control of Multimedia Applications Based on RTP*, Computer Communications , vol. 19, pp. 49--58, January 1996.

[3] Floyd, S., *Congestion Control Principles,* BCP 41, RFC 2914, September 2000.

[4] Floyd, S., Kempf, J. IAB Concerns Regarding Congestion Control for Voice Traffic in the Internet, RFC-3714, March 2004.

[5] Fitzgerald, C. *Codec-Independent technique for modulating bandwidth in packet network,* Patent No. US 6,421,720, 1992.

[6] Hole, D., Tobagi, F. *Capacity of an IEEE 802.11b Wireless LAN supporting VoIP*, ICC, December 2004.

[7] Karam, M., Tobagi, F. *Analysis of delay and jitter of voice traffic in the Internet,* Computer Networks (40) 2002, p711-726.

[8] Network Simulator Version 2 - ns-2, available from http://www.isi.edu/nsnam/ns

[9] Ott J., Wenger S., Sato N., Burmeister C., and Rey J., "Extended RTP Profile for RTCP-based Feedback", Internet Draft, draft-ietf-avt-rtcp-feedback-11.txt, Proposed Standard, August 2004

[10] Schulzrinne, H., Casner, S., Frederick, R., Jacobson, V., *RTP: A transport protocol for Real-Time Applications,* RFC-3550, July 2003.

[11] Schulzrinne, H., Casner, S., *RTP Profile for Audio and Video Conferences with Minimal Control,* RFC-3551, July 2003.

[12] Monarch Project, Wireless and Mobility extensions to ns-2, available from http://www.monarch.cs.cmu.edu/cmu-ns.html

[13] ITU-T 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

[14] Oouchi, H., Takenaga, T., Sugawara, H., Masugi, M. *"Study on appropriate voice data length of IP packets for VoIP network adjustment"*, GLOBECOM 2002 - IEEE Global Telecommunications Conference, November 2002

[15] Qiao, Z., Sun, L., Heilemann, N., Ifeachor, E. *"A new method for VoIP quality of service control use combined adaptive sender rate and priority marking"*, ICC 2004 - IEEE International Conference on Communications, vol. 27, no. 1, June 2004

[16] Floyd S., Kohler, E. *"TCP Friendly Rate Adaptation for Voice: VoIP variant and Faster Restart"*, draft-ietf-dccp-tfrc-voip-01.txt

[17] Friedman, T., Caceres R., Clark A., *RTP Control Protocol Extended Reports (RTCP XR)*, RFC-3611, November 2003

[18] Handley M., Floyd S., Padhye J., Widmer J., *TCP Friendly Rate Control (TFRC): Protocol Specification*, RFC-3448, January 2003

12. ACRONYMS

802.11	Wireless LAN
802,3	Ethernet
AVP	Audio Video Profile
C++	Object-oriented programming.language
DiffServ	Differentiated Services
FEC	Forward Error Correction
G711	Pulse Code Modulation Codec
G729	CS-ACELP – Conjugate Structure – Algebraic Code-Book Excited Linear Prediction codec
G114	End-to-End Delay requirement
IAB	Internet Architecture Board
IETF	Internet Engineering Task Force
ITU-T	International Telecommunications Union - Telecommunications
IP	Internet Protocol
LAN	Local Area Network
MOS	Mean Opinion Score
NACKs	Negative Acknowledgements
ns-2	Network Simulator
PESQ	Perceptual Evaluation of Speech Quality
PPP	Point-to-Point Protocol
QoS	Quality of Service
RFC	Request for Comments
RTCP	Real-Time Congestion Protocol
RTP	Real-Time Protocol
T1	1.544 Mbps
TCL	Tool Command Language
TCP	Transmission Control Protocol
TFRC	TCP Friendly Rate Adaptation Control VOIP
WAN	Wide Area Network
WAV	Audio file format

Scenario	Sender	RTCP Mode	Codec	Bandwidth
Scenario1	1996 [2]	3550 [10]	G711	200
Scenario2	1996 [2]	3550 [10]	G711	300
Scenario3	1996 [2]	3550 [10]	G711	500
Scenario4	1996 [2]	3550 [10]	G729	40
Scenario5	1996 [2]	3550 [10]	G729	70
Scenario6	1996 [2]	3550 [10]	G729	200
Scenario7	1996 [2]	3550 [10]	Both	100
Scenario8	1996 [2]	3550 [10]	Both	180
Scenario9	1996 [2]	3550 [10]	Both	300
Scenario10	1996 [2]	Quick [9]	G711	200
Scenario11	1996 [2]	Quick [9]	G711	300
Scenario12	1996 [2]	Quick [9]	G711	500
Scenario13	1996 [2]	Quick [9]	G729	40
Scenario14	1996 [2]	Quick [9]	G729	70
Scenario15	1996 [2]	Quick [9]	G729	200
Scenario16	1996 [2]	Quick [9]	Both	100
Scenario17	1996 [2]	Quick [9]	Both	180
Scenario18	1996 [2]	Quick [9]	Both	300
Scenario19	Patent [5]	3550 [10]	G711	200
Scenario20	Patent [5]	3550 [10]	G711	300
Scenario21	Patent [5]	3550 [10]	G711	500
Scenario22	Patent [5]	3550 [10]	G729	40
Scenario23	Patent [5]	3550 [10]	G729	70
Scenario24	Patent [5]	3550 [10]	G729	200
Scenario25	Patent [5]	3550 [10]	Both	100
Scenario26	Patent [5]	3550 [10]	Both	180
Scenario27	Patent [5]	3550 [10]	Both	300
Scenario28	Patent [5]	Quick [9]	G711	200
Scenario29	Patent [5]	Quick [9]	G711	300
Scenario30	Patent [5]	Quick [9]	G711	500
Scenario31	Patent [5]	Quick [9]	G729	40
Scenario32	Patent [5]	Quick [9]	G729	70
Scenario33	Patent [5]	Quick [9]	G729	200
Scenario34	Patent [5]	Quick [9]	Both	100
Scenario35	Patent [5]	Quick [9]	Both	180
Scenario36	Patent [5]	Quick [9]	Both	300
Scenario37	None	3550 [10]	G711	200
Scenario38	None	3550 [10]	G711	300
Scenario39	None	3550 [10]	G711	500
Scenario40	None	3550 [10]	G729	40
Scenario41	None	3550 [10]	G729	70
Scenario42	None	3550 [10]	G729	200
Scenario43	None	3550 [10]	Both	100

Scenario	Sender	RTCP Mode	Codec	Bandwidth
Scenario44	None	3550 [10]	Both	180
Scenario45	None	3550 [10]	Both	300
Scenario46	None	Quick [9]	G711	200
Scenario47	None	Quick [9]	G711	300
Scenario48	None	Quick [9]	G711	500
Scenario49	None	Quick [9]	G729	40
Scenario50	None	Quick [9]	G729	70
Scenario51	None	Quick [9]	G729	200
Scenario52	None	Quick [9]	Both	100
Scenario53	None	Quick [9]	Both	180
Scenario54	None	Quick [9]	Both	300

Table 5 – Simulation Scenarios

Codec	Ptime (ms)	pps	SBw (bps)	4 ch.	Bw (bps)	2.5% SBw	Events/s*
G711	10	100.00	96,000	384,000	300,000	2,400	3
G711	20	50.00	80,000	320,000	300,000	2,000	2
G711	30	33.33	74,667	298,667	300,000	1,867	2
G729a	10	100.00	40,000	160,000	90,000	1,000	1
G729a	20	50.00	24,000	96,000	90,000	600	0.8
G729a	30	33.33	18,667	74,667	90,000	467	0.6
G729a	40	25.00	16,000	64,000	90,000	400	0.5
G729a	50	20.00	14,400	57,600	90,000	360	0.5
G729a	60	16.67	13,333	53,333	90,000	333	0.4
G729a	70	14.29	12,571	50,286	90,000	314	0.4
G729a	80	12.50	12,000	48,000	90,000	300	0.4

Table 6 – Codec and Packetization Time Summary Table for 4 channels

* Average RTCP size = 96 bytes = 768 bits