Rate-control for Conversational Video Communication in Heterogeneous Networks

Varun Singh[†], Jörg Ott[†], Igor D. D. Curcio[‡]
[†]*Comnet, Aalto University, Espoo, Finland*
[‡]*Nokia Research Center, Tampere, Finland*
{*varun,jo*}*@comnet.tkk.fi, igor.curcio@nokia.com*

***Abstract*—Heterogeneous networks are made up of wireless and wired links. The wireless link quality may vary due to interference and fading phenomena and, peered with radio and link layer reconfigurations lead to varying error rates, latencies, and, most importantly, changes in the available bit rate. In both fixed and wireless networks, varying amounts of cross traffic from other nodes may lead to fluctuations in queue size (reflected again in a path latency) and to congestion (reflected in packet drops from router queues). We propose a rate-adaptation algorithm for conversational video communication in a heterogeneous environment and use the standardized RTCP extensions as a starting point. We use metrics such as PSNR, loss rate, bandwidth utilization and fairness to evaluate the proposed algorithm.**

I. Introduction

Video communication is a delay-sensitive real-time application and it requires the end-points to adapt quickly to the changes in the network conditions. The end-points rely on congestion indicators (e.g. loss, RTT, jitter, etc.) to make adjustments to the media encoding-rate so that they do not exceed the end-to-end (e2e) channel capacity.

In heterogeneous networks, traditional congestion indicators, such as, packet losses are not applicable because air interface (bit-error) losses and congestion losses may be hard to differentiate [1]. Even if the losses are congestion-induced, they may be problematic for video communication because the receiver may discard packets that arrive later than the acceptable playout delay, i.e., even before congestion-induced losses appear. Furthermore, flows compete with each other for bandwidth on a shared bottleneck link and non-adaptive video flows can be unfair to other flows, such as, TCP. More importantly, a non-adaptive video flow may create congestion in the network leading to additional queuing delays, which may degrade the video quality at the receiver. Therefore, a video sender has to anticipate upcoming congestion from various cues—including but not limited to the per-packet delay used in many delay-based congestion control algorithms—to prevent network queues from building up in the first place. This requires extreme sensitivity while maintaing stability to the reported transmission characteristics.

Differences in the physical nature and the network architecture of 3G mobile services, WLAN and ADSL-type Internet access make congestion control in heterogeneous environments challenging because the application-defined maximum delay and the minimal network-incurred latency may leave only very little room for a congestion control algorithm to operate. In this paper, we take the characteristics of different congestion indicators into account to propose a rate-adaptation algorithm that can operate in heterogeneous environments. We show that endpoints are able to adapt to the varying path latencies and channel capacities. Also that video calls are able to share bandwidth on a common bottleneck path with other video calls. Lastly, we note that playout delay, discarded packets also indicate congestion apart from loss, jitter and RTT.

II. Background

The Real-time Transport Protocol (RTP) [2] is used to deliver real-time content and its associated RTP Control Protocol (RTCP) forms the control channel between the endpoints. RTCP Sender Report (SR) and Receiver Report (RR) carry important information related to media playout and link conditions, such as, jitter, loss rate, RTT, etc. RTCP reports are sent periodically but not often, the time interval is in the order of seconds. [3] allows the RTCP to use up to 2.5% of the media rate for each end-point in a point-to-point scenario, which is quicker than the $5 \pm 2.5s$ restriction described in [2]. To quickly adapt to congestion, Garudadri et. al. [4] schedule RTCP feedback packets every *200ms-380ms* and use non-compound RTCP reports [5] to conserve RTCP bandwidth. Typically, RTCP calculation assumes a fixed media rate; dynamically adjusting RTCP reporting to a variable bit rate media stream has not received much attention so far.

Currently, the end-points can use one or a combination of the following RTCP extensions for implementing congestion control. RTCP Extension Report (RTCP-XR) [6] defines receiver summary statistics containing max, min, average and standard-deviation of jitter, RTT, lost, and duplicate packets in an RTCP interval, which along with the information in a standard RTCP RR can be fed into TCP Friendly Rate Control (TFRC) [7]. However, TFRC requires ACKing once per RTT, which may be shorter than the minimum RTCP reporting interval [8]. Furthermore, TFRC is not well suited for multimedia applications because it underutilizes the link [9][10][11]. RTCP Explicit Congestion Notification (ECN) feedback packet [12] reports the packets that would have been dropped by an intermediate router,

978-1-4673-1239-4/12$31.00 ©2012 IEEE

this information can be used by the sender for adapting the media bit rate. However, this would require intermediate routers to implement ECN. Temporary Maximum Media Stream Bit Rate Request (TMMBR) [13] can be used by the receiver to request a specific media bit rate from the sender. The receiver generates the TMMBR requests based on the packet inter-arrival time, RTP buffer size, packet losses and packet discards [11]. This type of receiver-driven mechanism is found to be conservative [11] and may not be able to compete with cross-traffic. Receiver Buffer Signaling is proposed in 3GPP [14] for Packet-switched Streaming Service (PSS). [11] uses the same 3GPP signaling mechanism to implement video communication in a 3G environment. On the other hand, [4] use uplink and downlink token buckets to adapt video in a 3G environment. In this case, the receiver explicitly signals the current receiver buffer and receiver rate.

Rate adaptation schemes are mechanisms to signal congestion cues between the sender and the receiver and also to react based upon them. The sender responds to these by adapting the sending rate to meet the end-to-end link capacity. We note that there are many RTCP extensions that can be used to make rate-control decisions in combination with the normal RTCP report. However, each additional extension adds to the signaling overhead and that can adversely affect the RTCP interval by making it longer.

III. Metrics for Rate Adaptation

In this section, we introduce metrics for evaluating rate-control algorithms. The sender's goal is to minimize losses at the receiver while maintaining throughput. Losses are caused by congestion or bit-errors and are detrimental to video quality. Although, video communication is tolerant to small amount of losses, they should be avoided. Bit-errors are due to the physical properties of the network and cannot be predicted ahead of time. Congestion losses are due to over-utilization of the links and may cause long delays or congestion-induced drops at the router.

Rate-control algorithms observe the congestion cues, and react to the changes in the cues, by modifying the encoding/sending rate to match the available end-to-end bit rate. Exceeding the available path bit rate may lead to a temporary increase in per-packet delay until the rate adaptation measures take effect and, optionally, to packet losses if the queue capacity is exceeded.

For the delay, we define three values:

- Threshold 1 refers to the mean one-way delay (OWD) observed under normal operating conditions; this value may be defined statically according to expectations for a certain environment, or determined dynamically. This reflects the mouth-to-ear or camera-to-eye delay. Threshold 1 may be 25–$200ms$.
- Threshold 2 defines the maximum acceptable one-way delay for a certain scenario after which rendering of the received video packets is no longer meaningful and packets arriving later than threshold 2 will be considered lost. Threshold 2 may be, e.g., 200–$500ms$ for video, since the human eye is more tolerant to video glitches [15].
- The short-term delay peak reflects the maximum delay peak encountered during a rate adaptation operation.

For losses, we consider two values:

- Packets lost in the network due to bit errors and/or increased queue lengths or overflows (e.g., caused by drop-tail or RED queue management).
- Packets discarded at the receiver because their arrival delay violated threshold 2.

Additionally, we also consider, instantaneous and average encoder rate, receiver rate and goodput, Average bandwidth utilization (ABU) and Peak Signal-to-Noise Ratio (PSNR).

IV. Algorithms

In this section, we introduce the algorithms and enhancements we developed for rate-control in video communication.

A. Controlling the RTCP Reporting Interval

The end-to-end channel capacity in a 3G or heterogeneous network can change dramatically at any moment in time, therefore over-utilization by the sender will cause congestion. The end-to-end latency can also vary in heterogeneous networks due to diverse network paths. To overcome small variations the receiver is able to adapt by changing the playout time [16], but dramatic changes will result in packets being discarded at the receiver due to late arrival. The late-arriving packets may also indicate congestion due to increase in network queues. Therefore, the receiver should report the current network conditions to the sender in a timely fashion; reporting congestion early may lead to under-utilization and reporting too late may cause severe congestion.

At the receiver, if packet loss and discards occur close together then the receiver identifies these as congestion induced losses. If only few packet are lost and no packets are discarded, then initially, the receiver identifies these losses as bit-error losses. However, if sustained packet losses occur over consecutive reporting intervals then the receiver identifies them as congestion induced losses and reduces the reporting interval to send more frequent feedback.

At the beginning of the session, the endpoints start with the reporting interval set to $5 \pm 2.5s$ [2]. If congestion occurs, the receiver is capable of shortening its interval all the way down to the minimum RTCP Interval defined in RFC4550 [3] (Algorithm 1 line: 12). Specifically, Algorithm 1 records the number of bad packets in an interval; conversational video communication is tolerant to occasional packet loss because the Group of Pictures (GOP) is small. To reduce the next reporting interval, we assume that the number of bad packets in an interval must exceed $\Gamma_{bpr} = \frac{GOP}{FPS} = 0.33$ (Algorithm 1 lines: 5–17).

Algorithm 1 Algorithm for adapting the of RTCP RR Interval

```
Require: RTCP RR timeout
Ensure: Compliance to RTCP Timing as defined in [RFC4585].
 1: Set $rtcp\_interval_{new} \leftarrow rtcp\_interval_{prev}$
 2:
 3: $min\_rtcp\_intvl = Min4585RTCPInterval()$
 4: if ($losses$ or $discards$) then
 5:     Calculate $good\_packet\_rate = \frac{received-discarded}{received+lost}$
 6:     Calculate $bad\_packet\_rate = 1 - good\_packet\_rate$
 7:     $\Gamma_{bpr} = \frac{GOP}{FPS}$; we use $\Gamma_{bpr} = 0.33$
 8:     if ($bad\_packet\_rate > \Gamma_{bpr}$) then
 9:         set $rtcp\_interval_{new} = rtcp\_interval_{prev} \times (1 - \Gamma_{bpr})$
10:         if ($rtcp\_interval_{new} < min\_rtcp\_intvl$) then
11:             set $rtcp\_interval_{new} = min\_rtcp\_intvl$
12:         end if
13:     end if
14: else
15:     set $rtcp\_interval_{new} = min(rtcp\_interval_{prev} \times \frac{3}{2}, Std3550Interval())$
16: end if
17: if (no RTP packet received in one RTCP interval) then
18:     set $rtcp\_interval_{new} = min\_rtcp\_intvl$
19: end if
20: set $rtcp\_interval_{prev} = rtcp\_interval_{new}$
```

B. Playout delay and Time Since Receiving Last Packet, (ΔT_{HSN})

The sender transmits a real-time media stream and each packet number includes a sequence number and timestamps (TS_S) so that the receiver knows the baseline for rendering each packet. The sender also keeps a ring buffer of the sent packet information between two receiver reports. Apart from the RTCP RR the receiver also reports Next Application Data Unit (NADU) [14] packet and the number of bytes discarded [17] in the last interval.

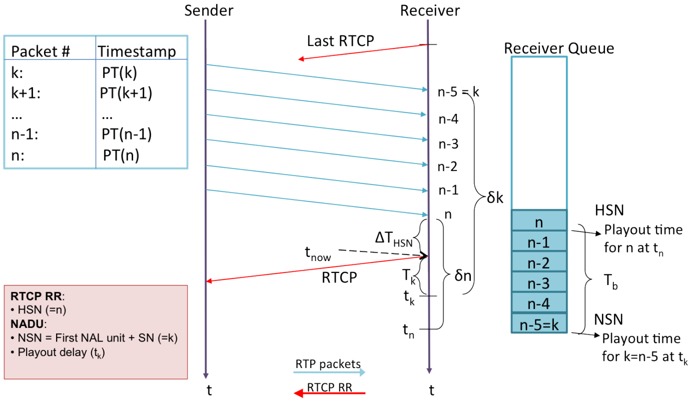


Figure 1: Time Since Receiving Last Packet (ΔT_{HSN})

From the received NADU [14] packet, the sender learns about the receiver state:

1) which packet is the first one in the receiver buffer (NSN = $n - 5 = k$ in Figure 1);
2) which Network Abstraction Layer (NAL) unit inside this packet is going to be rendered next; and
3) when the specific NAL unit will be rendered (playout delay).

This information allows the sender to reconstruct the contents of the receiver buffer in the absence of packet losses: it knows the first packet's sequence number (NSN) from NADU and the highest sequence number (HSN) from the RR packet. Thus, the sender knows which packets are currently buffered ($PiB = HSN - NSN + 1$). Furthermore, from its local table (see the left hand side in Figure 1), the sender can determine the duration of real-time media by computing the difference in timestamps of the first packet ($TS_S[NSN]$) and the last packet ($TS_S[HSN]$). This yields the playout time of all the media contained in the receiver buffer T_b and defines when a decoder under-run will occur and impact the perceived media quality.

$$T_b = BFL_{in\_ms} = \quad TS_S[HSN] - TS_S[NSN] \ s$$

But this is insufficient to determine the total playout delay of the last received packet (T_n). For calculating T_n accurately, the period between the reception of the last packet (HSN) in the buffer n (received at $t_r(n)$) and its playout point needs to be determined: $\delta_n = t_n - t_r(n)$. The sender knows the duration of the contents in the buffer, T_b and the playout delay of the first packet, T_k. The playout delay for the last packet received is thus $T_n > T_k + T_b$ or $T_n = T_k + T_b + \Delta T_{HSN}$. ΔT_{HSN} represents the time between the reception of packet n at the receiver and the generation of the RTCP receiver report. Depending on the latency of the network, the rate at which media packets are generated and the jitter they experience inside the network, this value may be significant or, at least, non-negligible. This can be summarized as follows:

$$\begin{aligned} delay_{decoding} &= time\ spent\ in\ the\ decoder's\ queue \\ &= Time_{render}[k] - Time_{decoder}[k] \\ &= order\ of\ a\ few\ to\ tens\ of\ ms. \end{aligned}$$

$$\begin{aligned} PD_k &= Time_{render}[k] - Time_{now} \\ &= (delay_{decoding}) + Time_{decoder}[k] - Time_{now} \end{aligned}$$

$$\begin{aligned} PD_{HSN} &= Time_{render}[HSN] - Time_{now} \\ &= (PD_{NSN} + BFL_{in\_ms}) \end{aligned}$$

$$\begin{aligned} delay_{HSN} &= Time_{render}[HSN] - TS_S[HSN] \\ &= \quad PD_{HSN} \ + \ OWD \ + \underbrace{Time_{now} - TS_R[HSN]} \\ &= \overbrace{PD_{NSN} + BFL_{in\_ms}} + OWD \ + \quad \Delta T_{HSN} \end{aligned}$$

$$delay_{HSN} = PD_{NSN} \ + \ BFL_{in\_ms} \ + \ \frac{RTT}{2} \ + \ \Delta T_{HSN}$$

$$< delay_{max}$$

$$delay_{HSN} \quad should\ be \quad < \ delay_{max}$$

To summarize, variation in bandwidth affects both buffer fill-level and RTT. The total delay experienced by an HSN packet ($delay_{HSN}$) indicates congestion and underflow. We conclude that this is an important parameter for estimating the underflow point and should be signaled in addition to the NADU. We will denote the algorithm and signaling scheme as C-NADU, which stands for Conversational NADU.

Table I: Summary of Congestion Indicators.

Indicator	Type of cue	Observation	Possible usage
RTT	Long-term cue	Fluctuates when the network queues build-up and reduce. The RTT variation is smooth depending on adaptation	Sender needs to collect history to observe a trend, it can be used for fairness
Jitter	Monitoring, values also depend on cross-traffic	Spikes in fixed networks can be correlated but not in mobile networks	A receiver or sender needs to compare jitter values over long time scales
Loss Rate	Slow and late	Hard to distinguish bit-error losses from congestion losses.	If link are already congested, the sender should undershoot the current channel capacity to offset the loading
Discard Rate	Early	Discard indicates congestion along the path, packet loss followed by discards may suggest congestion as opposed to bit-error losses.	Undershoot the receiver rate to decongest the link.
Playout Delay	Early	Can indicate underflow and overflow, better indicator than RTT but dependent on maximum allowed delay.	Fine tunes the sending rate
Frame Inter Arrival Time (FIAT) or Packet delay variation (PDV)	Early	better indicator than jitter in case of conversational video in 3G because of quicker feedback and larger variation in Bandwidth.	Needs to be filtered else it is quite aggressive in ramp-up and ramp-down
ΔT_{HSN}	slow	Very good for low delay networks, conservative for high latency networks	Can cause rapid oscillations in sending rates, needs to be dampened

C. Sender-side Algorithm

While conducting the rate-control experiments, we also studied the behavior of the congestion indicators. Table I summarizes our observations and describes the possible application of each congestion indicator. In the Table, the "Type of cue" column denotes the timeliness of the cue to indicate congestion. This can be classified as, *early*, *late*, *slow*, and *long-term*. "Early" means it appears at the onset of congestion, while "late" signifies that the congestion cannot be avoided any more. "Long-term" requires capturing the variation of the cue over time (history) to make decisions and "slow" means that it takes time for the cue to indicate congestion. The "observation" column describes the behavior of the cue and the "Possible usage" column describes the application of the cue. Based on these observations, we apply the congestion indicators in our rate-control algorithm.

Jitter, Loss rate and RTT describe the queues along the path, these may not necessarily be ideal indicators for congestion. Packets discarded at the receiver indicate an increase in network queuing and possibly the onset of congestion. Additionally, Playout delay describes the amount of data in terms of content length and not size of buffer. For heterogeneous environments, we apply enhancements to the base algorithm proposed in [11] and the modifications are as follows:

- **Discards**: If discards occur then the encoding rate drops to the calculated goodput. (Algorithm 2 lines: 10–12)
- **Loss**: If losses and discards occur in the same reporting interval or if the losses occur in the interval just after discards then the encoding rate drops to the goodput, in other case it may be ignored as bit-error losses. (Algorithm 2 lines: 10–12 and 16–22)
- **Average Frame Inter-Arrival Time**: The average inter arrival time between video frames. If the frame is fragmented over multiple RTP packets then it is reconstructed and the arrival time of the first RTP packet is used to calculate the difference in arrival times of the frames. The difference is calculated at the receiver for each incoming packet in an interval by subtracting the *arrival time of the last packet of the previous frame* from the *arrival time of first packet of the current frame*. The average is calculated for all the complete frames received in this reporting interval [6]. Partial frames are carried over to the next reporting interval. The sender compares the sent and received average frame inter-arrival time to calculate the new encoding rate. (Algorithm 2 line: 23)
- **Playout Delay**: The sender compares the $delay_{HSN}$ to $delay_{max}$ to calculate the new encoding rate. (Algorithm 2 line: 23, also read Sec. IV-B)
- **TCP Friendly Rate (TFRC)**: The sender calculates the TFRC rate based on the fractional loss reported in the RR and RTT. If the sending rate in the last few intervals is much higher than the calculated TFRC rate and the receiver is reporting losses/discards then the sending rate should be limited to the TFRC rate.

In the Algorithm 2: line 11, $\delta_{undershoot}$ is calculated only on the first appearance of a loss and discard event; this is done to quickly mitigate congestion because of higher rate packets in transit. In cases of extreme congestion when no conclusive information is available we use constants (α, β) to reduce the rate by a fraction (See *lines 8, 14* of

Algorithm 2 Sender-side Rate Adaptation Algorithm

```
Require: Encoder maintains a ring-buffer with sizes of packets sent since the HSN
    of Last RR
Ensure: Reception of Latest RR from receiver
 1: Calculate $ReceiverRate_{estimated}, GoodPut_{estimated}, PD_{HSN}$
 2: if the same HSN is reported for two RTCP intervals then
 3:     $NewBw \leftarrow CurrentBw \times \alpha$; we use $\alpha = 0.5$
 4: else if ($losses$ and $discards$) then
 5:     if ($CurrentBw > GoodPut_{estimated}$) then
 6:         $NewBw \leftarrow GoodPut_{estimated} \times \delta_{undershoot}$;
 7:         $\forall \delta_{undershoot} \in (0, 1]$
 8:     else
 9:         $NewBw \leftarrow CurrentBw \times \beta$; we use $\beta = \frac{\sqrt{2}}{2}$
10:     end if
11: else if (only losses and no discards) then
12:     if ($CurrentBw < ReceiverRate_{estimated}$) then
13:         $NewBw \leftarrow ReceiverRate_{estimated}$
14:     else
15:         $NewBw \leftarrow CurrentBw \times \Psi$; we use $\Psi = 1.1$
16:     end if
17: else
18:     $NewBw \leftarrow CurrentBw \times MIN(\frac{\frac{1000}{FPS_{curr}}}{FIAT_{avg}}, \frac{delay_{max}}{delay_{HSN}})$
19: end if
```

Algorithm 2). On line 33, $\Psi = 1.1$ so that the sender can probe for higher bandwidth, when no packets are discarded in the current or previous intervals. In this case we assume that the packet loss may have occurred due to bit-error loss.

V. Simulation Environment

Network Simulator 2 (*ns-2*) [18] is used to simulate the wired and heterogeneous scenarios. *Ns-2* is a discrete event simulator, wherein each event in the system is maintained and scheduled in a chronological order. In order to simulate a real scenario, we use Nokia's H.264/AVC codec [19] for encoding and decoding video. The media packets flow between the codec and the simulator using the *REAL interface*. The interface synchronizes the media clock of the codec with that of the timer in *ns-2* to provide real-time emulation.

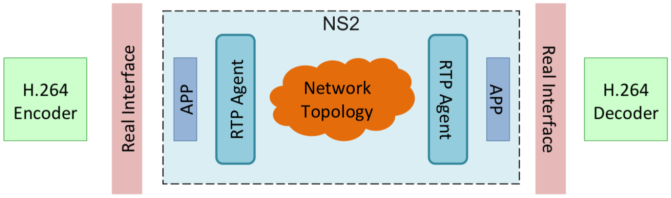


Figure 2: Simulator System Overview

Figure 2 shows the system overview of the simulator. The codec packetizes the H.264 [20] video frames into RTP packets based on [21] and sends it to the REAL Interface. Additionally, the codec provides APIs to change the encoding, quality and network parameters. The REAL interface uses these APIs to configure the codec. The REAL interface also maintains a buffer for retransmitting packets at the sender and a de-jitter buffer to reorder out-of-order packets at the receiver. The REAL interface also implements the communication interface with the simulator for sending/receiving RTP packet, control messages, and synchronizing the clock with the simulator.

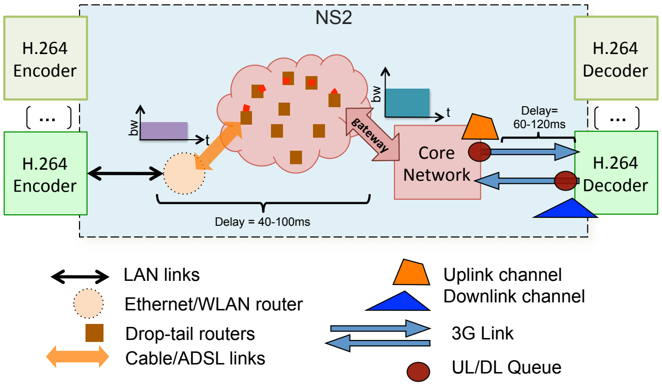


Figure 3: Components of a heterogeneous simulation environment

The *Application* module in *ns-2* is the entry and exit point for the packets to and from the codec. The *RTP Agent module* is responsible for scheduling and transmitting the RTCP reports and conforms to the RTCP timing rules described in [2][3]. The *Network topology module* simulates the behavior of the links in a specific multimedia environment.

Nowadays, calls originate on the Internet and may terminate on a mobile phone or vice versa; this simulation scenario attempts to emulate the real world. We combine the knowledge of the applicability of the congestion cues from the wired scenario to the to the heterogeneous scenario. Herein, two or more senders share a bottleneck link while the receivers are on independent 3G links undergoing different link conditions. The limiting link changes between the fairness on the bottleneck link to throughput of the 3G links. Figure 3 shows a simplified illustration of this scenario. The bottleneck link has a capacity of 1Mbps and the fan-out link to each receiver and sender is 10Mbps. The delay on the bottleneck link is 40-80ms and the fanout links have 20-40ms delay. The 3G links conform to the behavior described in [22]. The Radio Link Control (RLC) [23] controls the scheduling and the amount of data (inclusive of the headers) that flows on the 3G link. The RLC is configured to unacknowledged mode to keep link layer delays to a minimum. The 3G RLC frame sizes and scheduling are based on real-world traces [24] and the RTT varies between *60-120ms*.

A. Simulation Settings

A typical conversational mobile multimedia system, such as MTSI [25], requires that capture-to-display delay does not exceed 400 ms [15] to provide acceptable media quality and a good user experience. The 400ms delay includes the decoding and rendering delay; therefore the 400ms is the upper bound of the delay-budget ($delay_{max}$). We assume a suitable lower bound to be 150ms; this should provide the opportunity for the decoder to cache at most 2-6 frames of a 15 FPS video. So, packets arriving later than the current delay-budget are discarded upon reception.

We use a medium-motion media sequence ("Foreman" sequence) encoded at $15FPS$. Furthermore, in all the scenarios, the sender begins with an initial sending rate of 128 $kbps$[1] and the encoder is not restricted by a maximum encoding rate. As discussed earlier, interactive video does not allow sufficient buffering (in time) to make use of B-frames[2]. Therefore, the encoder is configured to only produce I- and P-frames. To quickly overcome the bit-error losses in wireless networks, conversational video communication uses short Group of Pictures (GOP), we use $N = 5$, i.e., *"I P P P P I"*, for a 15 FPS video this strategy would create 3-4 I-Frames in a $1s$ interval.

VI. Performance Evaluation

In this section, we evaluate the performance of the proposed algorithm in a heterogeneous environment. We conducted experiments in different configurations and parameters but due to lack of space we only report one scenario run over 5 simulation runs. The scenario simulates a real-world scenario, where the user on the Internet initiates a video call with a user on a 3G connection, or vice-versa. Moreover, the video call also competes for capacity on a bottleneck link with other similar video calls. The purpose of the setup is to evaluate the behavior of C-NADU in a heterogeneous environment with similar cross-traffic (5 calls) and variable latency and channel capacity.

The five callers are connected to the Internet by $10Mbps$ links and the five callees are using the 3G mobile network. The channel of each callee is separate and undergoes different type of fading and interference, which results in different channel bandwidth and latency for each user. The capacity of each 3G channel can vary from about $16kbps$ to $512kbps$. The 3G channel capacity for each callee is described in Table II. The end-to-end path latency for the Call 2 and 4 is $100ms \pm 25\%$ and Call 1, 3 and 5 is $240ms \pm 25\%$. The 3G channel capacity and latency varies at $1s$ intervals.

Table II: Link variation for the 3G end-points

	Bandwidth Sequences
Call 1	Excellent-Poor-Elevator
Call 2	Good-Good-Poor
Call 3	Poor-Poor-Poor
Call 4	Fair-Fair-Poor
Call 5	Excellent-Elevator-Poor

Each pattern is $60s$ long and the average bandwidth for the pattern titled "Excellent" is $400kbps$, "Elevator" is $300kbps$, "Good" is $250kbps$, "Fair" is $200kbps$, and "Poor" is $60kbps$[3]. The patterns are based on 3GPP RAN traces [24]. We also use 3G link packet traces [26] for simulating 0.5% link-layer bit-error rate (BER). The two networks are inter-connected by a $1Mbps$ bottleneck link. Depending on the behavior of the 3G link, the constraining link for each call can alternate between the 3G link and the bottleneck link. Therefore, the rate-control algorithm has to take the overall path characteristic into account and not only the individual links. Figures 4 (a)–(d) show the channel utilization of each call with reference to Call 1 and Table III presents the average goodput, loss rate and Average Bandwidth Utilization (ABU) for each call.

Table III: Scenario: Five calls in a heterogeneous network with high delay

	$Goodput_{avg}$ (kbps)	Loss Rate (%)	$PSNR_{avg}$ (dB)	ABU (%)
Call 1	140.10	2.15%	31.4 ($\sigma = 0.39$)	70.1%
Call 2	133.55	1.61%	31.9 ($\sigma = 0.62$)	66.8%
Call 3	35.18	1.55%	22.2 ($\sigma = 1.13$)	17.59%
Call 4	114.96	2.75%	31.1 ($\sigma = 0.75$)	57.5%
Call 5	130.23	2.25%	31.3 ($\sigma = 0.13$)	65.1%

The end-to-end capacity for a call depends on the fair sharing of capacity on the bottleneck link and the capacity of the 3G link. Call 3 shows that during the whole simulation, the 3G link was the constraint and the average goodput of the call is $\approx 35kbps$. In the first $60s$ of the simulation, the average channel capacity on the remaining four 3G links is greater than $250kbps$ and in this period the associated calls exhibit fair usage of the bottleneck link ($\approx 180 - 220kbps$) and try to use up all the capacity on the bottleneck link (See Figure 5). Furthermore, in the last $60s$ of the simulation, when four out of five 3G links have poor connectivity, in this period Call 1 is transitioning from poor to better connectivity and is able to quickly adapt its encoding rate to occupy more of the end-to-end capacity.

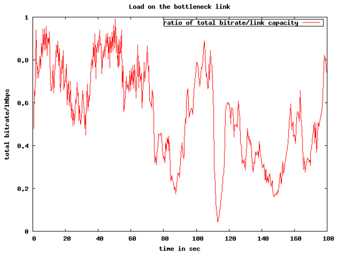


Figure 5: Plot shows the capacity usage on the bottleneck link (1Mbps). The usage drops low when the bandwidth on the 3G links is a constraint. Note that the major and minor X-axis in each graph is 180 seconds long and the Y-axis measures bandwidth and ranges from 0 to 1Mbps.

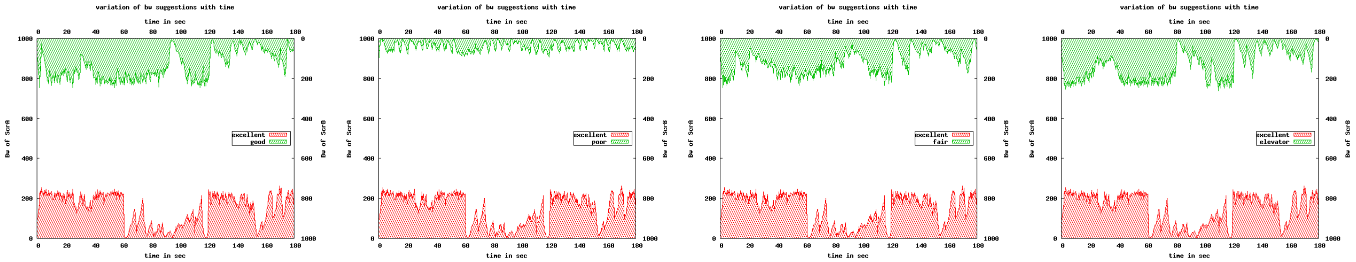
VII. Conclusions and Future Work

In this paper, we took a wide range of queues into account to evaluate the rate-control algorithm in a heterogeneous environment. In the simulated scenario, C-NADU is able to adapt to the constraints imposed by the shared bottleneck link and the variable capacity of the 3G links. Moreover, it is fair to other video calls on the shared bottleneck,

[1] Starting rate is an open problem, and is not tackled here.

[2] B-frames predict based on past and future packets.

[3] Note that the names of the traces only describe the bandwidth value and not the quality of the link.



(a) Call 1 vs Call 2

(b) Call 1 vs Call 3

(c) Call 1 vs Call 4

(d) Call 1 vs Call 5

Figure 4: Plot shows goodput of five calls competing for bandwidth on a shared bottleneck in a heterogeneous network. At some instances of time the 3G link is the constraint and at other times the bottleneck link. Each calls needs to adapt quickly to changes in 3G link capacity and fairly share the bottleneck link. Note that the major and minor X-axis in each graph is 180 seconds long and the Y-axis measures bandwidth and ranges from 0 to 1Mbps.

yet aggressively utilizes the available end-to-end channel capacity.

To minimize the transmission overhead, non-compound RTCP packets [5] may be used. The rate-control algorithm may also exploit network-assistance, e.g, ECN for RTP [12]. The RTCP ECN feedback from the receiver would accurately indicate congestion unlike reporting discarded packets by the receiver which may be due to increase in RTT or congestion.

References

[1] S. Cen, P. Cosman, and G. Voelker, "End-to-end differentiation of congestion and wireless losses," *IEEE/ACM TON*, vol. 11, no. 5, 2003.

[2] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications," RFC 3550, Jul. 2003.

[3] J. Ott, S. Wenger, N. Sato, C. Burmeister, and J. Rey, "Extended RTP Profile for RTCP-Based Feedback (RTP/AVPF)," RFC 4585, 2006.

[4] H. Garudadri, H. Chung, N. Srinivasamurthy, and P. Sagetong, "Rate adaptation for video telephony in 3G networks," *Packet Video '07*.

[5] I. Johansson and M. Westerlund, "Support for Reduced-Size RTCP: Opportunities and Consequences," IETF, RFC 5506, 2009.

[6] T. Friedman, R. Caceres, and A. Clark, "RTP Control Protocol Extended Reports (RTCP XR)," RFC 3611, 2003.

[7] S. Floyd, M. Handley, J. Padhye, and J. Widmer, "TCP Friendly Rate Control (TFRC): Protocol Specification," RFC 5348, Sep. 2008.

[8] L. Gharai and C. Perkins, "RTP with TCP Friendly Rate Control," March 2011, (Work in progress).

[9] ——, "Implementing congestion control in the real world," 2002.

[10] H. Vlad Balan, L. Eggert, S. Niccolini, and M. Brunner, "An Experimental Evaluation of Voice Quality Over the Datagram Congestion Control Protocol," in *IEEE INFOCOM*, 2007.

[11] V. Singh, J. Ott, and I. Curcio, "Rate adaptation for conversational 3G video," in *INFOCOM Workshop on MoViD*, Rio de Janeiro, BR, 2009.

[12] M. Westerlund, I. Johansson, C. Perkins, P. O'Hanlon, and K. Carlberg, "Explicit Congestion Notification (ECN) for RTP over UDP," (work in progress), March 2010.

[13] S. Wenger, U. Chandra, M. Westerlund, and B. Burman, "Codec Control Messages in the RTP Audio-Visual Profile with Feedback (AVPF)," RFC 5104, 2008.

[14] 3GPP TS 26.234, "Transparent end-to-end Packet-switched Streaming Service (PSS); Protocols and codecs."

[15] 3GPP S4-080771, "MTSI Video Dynamic Rate Adaptation: Evaluation Framework," Oct. 2008.

[16] C. Perkins, *RTP: audio and video for the internet*, 1st ed. Addison-Wesley Professional, 2003.

[17] J. Ott, I. Curcio, and V. Singh, "Real-time Transport Control Protocol Extension Report for Run Length Encoding of Discarded Packets," March 2011, (Work in progress).

[18] "Network Simulator (ns2) ," http://nsnam.isi.edu/nsnam.

[19] Nokia, "Homepage of H.264 codec." [Online]. Available: http://research.nokia.com/page/4988

[20] ITU-T Rec. H.264, "Advanced video coding for generic audiovisual services."

[21] S. Wenger, M. Hannuksela, T. Stockhammer, M. Westerlund, and D. Singer, "RTP Payload Format for H.264 Video," IETF, RFC 3984, 2005.

[22] 3GPP TR 26.902, "Video codec performance," Jan. 2008.

[23] 3GPP TS 25.322, "Radio Link Control (RLC) protocol spec," Sep. 2008.

[24] 3GPP R1-081955, "LTE Link Level Throughput Data for SA4 Evaluation Framework." May 2008.

[25] 3GPP TS 26.114, "IP Multimedia Subsystem (IMS): Multimedia telephony; media handling and interaction."

[26] 3GPP S4-050560, "Software Simulator for MBMS Streaming over UTRAN and GERAN," Sep. 2005.