Smooth Multicast Congestion Control for Adaptive Multimedia Transmission

Christos Bouras[1, 2], Apostolos Gkamas[1] and Georgios Kioumourtzis[2]

[1]Research Academic Computer Technology Institute, Patras, GR26500, Greece
[2]Computer Engineering and Informatics Department, University of Patras, Patras, GR26500, Greece

Abstract— **In this paper we introduce an equation-based smooth multicast congestion control for adaptive multimedia transmission over best-effort wired networks. Target of the proposed schema is (a) smooth transmission rate, in order to minimize the Audio-Video (AV) encoding and decoding distortion and (b) TCP friendly transmission. The "smoothness" lays in the way the TCP-Friendly transmission rate is filtered. We integrate the congestion control functions in the RTP protocol and use the RTCP sender and receiver reports to provide the necessary feedback information for the sender's adaptive transmission rate. The performance evaluation of the smooth adaptation and TCP-friendliness is conducted through a number of simulations with the network simulator software (ns2). Our intention is to use this congestion control in the context of a proposed framework for multimedia transmission over wired and wireless networks.**

Index Terms— **Congestion Control, Multicast, Multimedia transmission, network simulator (ns2).**

I. Introduction- Related work

CONGESTION control for multicast transmission of multimedia data is a challenging research area. In any proposed solution, one has to find the balance between the attributes of multimedia applications (bandwidth consuming applications, tolerant to packet losses, sensitive to delays) and the need for TCP-friendly behavior. Although, congestion control for multimedia data transmission involves various contradictory requirements, we believe that at least the three following requirements should be satisfied:

- Any proposed congestion control should prevent oscillations, as much as possible, in order to minimize the Audio-Video (AV) encoding and decoding distortion.
- Inter-arrival jitter delay should be small in order to meet the multimedia application's requirements.
- Packet losses should be minimized and when exist they should have minimal negative results in end user's perception.

Up to now there are promising approaches in the field of the single layer multicast congestion controls in bibliography. TFMCC [1], extends the basic mechanisms of TFRC [2] to support single layer multicast congestion control. The most important attribute of TFMCC is the suppression of feedback receiver reports. TFMCC is using the receiver with the lowest receiving capacity to act as the representative of the multicast group. PGMCC [3] uses a window-based TCP controller based on positive ACKs between the sender and the group representative. TBRCA [4] targets at maximizing the overall amount of multimedia data to the whole set of receivers. With the use of a bandwidth rate control algorithm it dynamically controls the output rate of the video coder. LDA+ [5] employs a TCP equation based congestion control for measuring the TCP friendly bandwidth share in the event of packet losses. LDA+ uses the RTP protocol [6] for collecting loss and delay statistics from the receivers. Our work is using RTP/RTCP protocols for the transmission of the multimedia data and it is based on the following concept: Exploitation of existing standard protocols in order to gather feedback information from the receivers. The main drawback in this approach is the high value of the time interval between two consecutive RTCP feedback reports. As a result, the sender does not have fast reactions in the light of rapid changes of the network conditions. However, the main objective under the RTP-based adaptation scheme is to adjust the sender's transmission rate to the average available bandwidth and prevent high oscillations of the transmission rate. This is the case in multimedia data transmission, when very frequent changes in the transmission rate may result to serious distortions in AV encoder/decoder. This may lead to infeasible adaptation rates for the AV encoder.

We present in this paper the performance evaluation of a smooth multicast congestion control protocol. Our intention is to use this congestion control as the rate control protocol in our proposed framework. We restrict our evaluation in this work only in the wired portion of the network, as the protocol has to be modified and enhanced in order to support wireless receivers. The rest of this paper is organized as follows. Our proposed protocol and framework are presented in section 2. Simulation results are presented in section 3. Conclusions and future work are discussed in section 4.

II. Proposed Framework – Smooth Multicast Congestion Control

In this section initially, we briefly describe the proposed framework for cross-layer adaptation over wired and wireless networks in the context of which the proposed smooth congestion control protocol will operate. Following we present

Digital Object Identifier inserted by IEEE

the details of the proposed smooth congestion control protocol.

A. Proposed Framework

The proposed framework consists of four entities: The sender, which represents the multimedia server, the proxy, which is located at the edge of the wired network, the AP, which co-located with the proxy and can be integrated with the proxy, and finally the receivers; wired and/or wireless. This proposed framework separates the wired from the wireless part of the network by introducing a new entity named "*proxy*" between the sender and the wireless receivers. The sender transmits multimedia data to wired receivers (*proxy* is also a wired receiver). The *proxy* is responsible to transmit the multimedia data received by the sender to wireless receivers. The transmission of multimedia data in the wired portion of the network is performed with a single or a multi-rate multicast stream. The *proxy* collects estimations from the wireless users concerning the conditions of the wireless medium. It then sends these estimations to the multimedia sender so that the sender can adjust accordingly its transmission rate. The interested reader can find a more detailed description in [7]. The proposed smooth congestion control protocol will be used for single stream multicast of multimedia data in the wired part of the proposed framework (e.g. multimedia transmission among sender, wired receivers and proxies).

B. Smooth Multicast Congestion Control

Our proposed scheme consists of a single rate multicast congestion control mechanism, which takes advantage of the RTCP feedback sender and receiver reports. The innovation in this work is the way that the multimedia sender adjusts its transmission rate based on the receiving feedback reports. Our main objective is to adjust the transmission rate in such way that oscillations are reduced, following a smooth fashion. A second important characteristic is the long term TCP-friendliness, meaning that the multimedia stream consumes no more bandwidth than a TCP connection, which is traversing the same path with the multimedia stream. Finally, with the use of the feedback RTCP reports we provide better scalability, as the amount of these feedback reports in the network are controlled by the RTCP protocol and they cannot exceed a specified threshold as percentage of the total available bandwidth [6].

A short overview of the functionality of the proposed congestion control is presented below:

- Each receiver measures the loss event rate based on the RTP packet sequence numbers (more information in paragraph II.C).
- The receiver measures the RTT to the sender based on the receiver's one-way time measurements and the sender RTCP feedback reports (more information in paragraph II.D).
- The receiver measures the TCP friendly bandwidth share with the use of the analytical model of TCP (more information in paragraph II.E).
- The sender adapts its transmission rate based on the RTCP feedback reports sent by all receivers that join the session.

In the following paragraphs we include a detailed description of the above functions.

C. Measuring the Loss Event Rate

The method for measuring the loss event rate is crucial for the TCP-friendly rate estimation. The wired receiver measures the packet losses during an RTT interval, based on the functionality provided by the RTP protocol. The sequence numbers in the RTP packet header provide a straightforward way for the discovery of lost packets. In order to prevent a single spurious packet loss from having an excessive effect on the packet loss estimation, the wired receivers smooth the values of packet loss rate by using the following filter that computes the weighted average of the m most recent loss rate values l_i^m. The following filter has been presented and evaluated in [8] and provides a good estimation of the packet loss rate:

$$l_i^m = \frac{\sum_{i=0}^{m-1} w_i l_i}{\sum_{i=0}^{m-i} w_i} \quad \text{for wired receiver } i \tag{1}$$

Where, l_i^m is the smooth value of packet loss rate. The weights w_i are chosen so that very recent packet loss rates receive the same high weights, while the weights gradually decrease to 0 for older packet loss rate values. We use m=8 and the following values for the weights

w_i: {1,1,1,1,0.8,0.6,0.4,0.2}.

The above values provide satisfactory packet loss filtering according to [8].

D. Measuring the Round Trip Time (RTT)

When a wired receiver i receives a RTP packet from the sender, it uses the algorithm described below in order to estimate the RTT between the sender and the wired receiver. Assuming that the sender and the wired receiver have synchronized clocks, the wired receiver can use the timestamp of the RTP packet ($T_{timestamp}$) and the local time when it receives that packet ($T_{receiver}$) to estimate the one way delay in the path between the sender and the wired receiver (T_{oneway}):

$$T_{oneway} = T_{receiver} - T_{timestamp} \tag{2}$$

If this path were symmetric and had the same delay in both directions, the RTT between the sender and the wired receiver would be twice T_{oneway}:

$$t_{RTT} = 2T_{oneway} \tag{3}$$

However, this assumption is not true for the Internet. Therefore, we use a parameter α in order to perform accurate RTT estimations (t_{RTT}^{e-l}) and we can write equation (3) as:

$$t_{RTT}^{e-l} = (1+a)T_{oneway} \tag{4}$$

The parameter α is used to smooth the estimation of the

RTT due to the potential unsynchronized clocks and the asymmetry of the path between the sender and the wired receiver.

To estimate the value of parameter α, the wired receivers need an effective estimation of the RTT, which can be acquired with the use of the RTCP reports. Thus, the RTCP receiver report contains two additional fields; the t_{LSR} (the timestamp of the most recent RTCP sender report) and the t_{DLSR} (the delay between the reception of the last sender report and the transmission of the receiver report). As a result the sender can make an effective RTT measurement for the path between the sender and a wired receiver by using the following equation. A is the time when the sender receives the receiver report from the given wired receiver:

$$t_{RTT}^{r-i} = A - t_{LSR} - t_{DLSR} \quad (5)$$

The sender estimates an effective RTT measurement for a wired receiver i, every time it receives a receiver report from that wired receiver. It then includes this effective RTT measurement (with the id of the receiver) in the next RTCP sender report.

When a wired receiver receives the effective RTT measurement from the sender, it estimates an appropriate value for the parameter α by using the following equation:

$$a = \frac{t_{RTT}^{r-i}}{T_{oneway}} - 1 \quad (6)$$

Furthermore, in order to avoid solely phenomenon of an instant RTT high value, which will affect the RTT estimations, we use an exponentially weighted average value.

$$t_{RTT}^{e-l} = t_{RTT}^{r-i} \cdot \beta + (1-\beta) \cdot t_{RTT}^{e-l} \quad (7)$$

Where t_{RTT}^{r-i} is the instantaneous RTT measurement made by the wired receiver and β=0.5. A high value of β gives more gravity to the instantaneous RTT measurement and results in more accurate TCP measurements. The trade-off is the oscillations of the calculated TCP-friendly rates and those oscillations are not preferable by multimedia applications. More on the filter β values can be found in [1] and [2]. Our performance evaluation shows that the selection of β=0.5 offers a reasonable compromise between smooth transmission rate and responsiveness to network changes

E. TCP-friendly Bandwidth Share Estimation

The wired receiver emulates the behavior of a TCP agent and as such when packet losses occur, it estimates a TCP friendly bandwidth share r^{i}_{tcp} every RTCP report interval with the use of the following analytical model presented in [9]

$$r^{i}_{tcp} = \frac{P}{t_{RTT}\sqrt{\frac{2Dl}{3}} + t_{out}\min(1, 3\sqrt{\frac{3Dl}{8}})l(1+32l^2)} \quad (8)$$

Where, r^{i}_{tcp} is the receiver's i estimation (in bytes/sec), P is packet size in bytes, l is the packet loss rate, t_{out} is the TCP retransmission timeout, t_{RTT} is the Round Trip Time (RTT) of the TCP connection and D is the number of acknowledged TCP packets by each acknowledgment. In our implementation we assume that $D=1$ (each acknowledgment packet acknowledges one TCP packet) and $t_{out} = 4t_{RTT}$ (the TCP retransmission timeout is set to be four time the RTT).

As we have already mentioned, the operation of our proposed smooth multicast congestion control is based on the functionality provided by the RTP and its associate RTCP protocols. RTP provides an extension mechanism to allow individual implementations that allow additional information to be carried in the RTP data packet header. We use the extension mechanism of RTP in order to add the following fields in to RTP header: t_{RTT}^{r-i} and receiver id: With this field the sender informs the receiver i about the effective RTT measurement between this receiver and the sender in order the receiver to use this value in the above equation.

If the wired receiver has not experienced any packet losses since the previous RTCP report, r^{i}_{tcp} must not be increased more than one *P/RTT*. For this reason the wired receiver calculates the new r^{i}_{tcp} value from the following equation (in bytes/sec):

$$r^{i}_{tcp} = r^{i}_{tcp} + \frac{1}{t_{RTT}^{e-i}} P \quad (9)$$

In addition, RTCP gives the capability to the participants to include in the RTCP reports an application specific part (APP) intended for experimental use. The receivers add to their receiver reports an application specific part, which contains their estimations for TCP friendly bandwidth share r^{i}_{tcp}.

The sender selects as the next transmission rate the minimum TCP friendly bandwidth share estimations received by the receivers, with the use the aforementioned extensions to RTCP.

The sender uses an additional filter in order to further smooth the calculated bandwidth share, which provides a smoother reaction to network changes.

$$r_{tcp} = r_{tcp}^{inst} \cdot \gamma + (1-\gamma) \cdot r_{tcp} \quad (10)$$

Where, r_{tcp}^{inst} is the instantaneous estimation of the slowest transmission rate reported by the receivers and γ a predefined value between 0 and 1.

$$0 < \gamma < 1 \quad (11)$$

The value of γ should be carefully chosen as high values make the algorithm more insensitive to changing network conditions. This is again a trade-off between responsiveness to rapid network changes and smooth oscillations. Our main objective is to prevent fast oscillations of the transmission rate. In this way we are able to better adapt and harmonize the transmission rate with the AV coder. We will elaborate the behavior of this proposal through simulation results and will observe how this algorithm behaves under competing traffic

sources that increase network congestion in the next section.

F. Modifications to the RTP code in ns2

The ns2 [10] code for the RTP/RTCP implementation is very generic and provides only the methods for the creation of sessions between the sender and the receivers. The RTCP sender and receiver feedback reports do not provide any functionality except for the basic API for further development. In this work we extended the ns2 code to include:

- A smooth TCP friendly multicast congestion control based on the analytical TCP model described in [9]. The control functions have been integrated inside the existed RTP source code in ns2, and make use of the RTCP sender and receiver reports for the external signaling between the sender and the wired receivers.
- Extensions to the RTP and RTCP packet headers to include the necessary fields for the feedback reports.
- Additional functionality for the generation of the RTCP sender and receiver reports.

With the above modification we have fully implement the RFC 3550 specification in ns-2 simulator in terms of QoS measurements.

III. Performance Evaluation

A. Simulation Environment and Network Topology set-up

The topology that is used for the evaluation of the proposed protocol is a Local Area Network (LAN), which consists of one multimedia sender and six heterogeneous wired receivers (figure 1). The heterogeneity of the receivers lays in the variation of the link capacity, which connects the receivers with the sender.

We have intentionally created a "bottleneck" between routers 1 and 2, in order to create two different sets of wired receivers. The first set of receivers (Nodes 1, 2, 3 "fast receivers") is able to receive at higher bit rates than the second set (Nodes 4, 5, 6 "slow receivers"). The server transmits a single multicast stream to the set of all the wired receivers (fast and slow receivers) that join the session. The initial transmission rate was set to 150 Kb/s. All the receivers join the multicast stream at the same time. Figure 1 depicts the network topology for the simulated scenario.

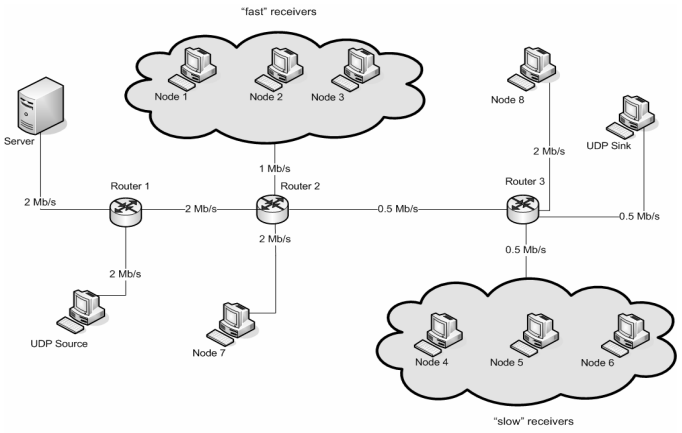

Fig. 1. Simulated Network Topology.

We evaluate the behavior of the proposed multicast congestion control with a number of simulations to investigate:

- The TCP-friendly behavior towards competing TCP traffic.
- The smooth transmission rates.
- The suitability of our proposed congestion control for multimedia data transmission.
- The responsiveness to network changes due to congestion that is caused by other competing traffics.

B. TCP-fairness

In this simulation, we analyze the fairness towards competing TCP traffic without using the smooth function that is expressed in Equation (10). Node 7 (TCP Agent) starts transmitting TCP traffic to Node 8 (TCPSink), through the congested path from router 2 to router 3. The initial sending rate for the TCP traffic is set to 150 Kb/s.

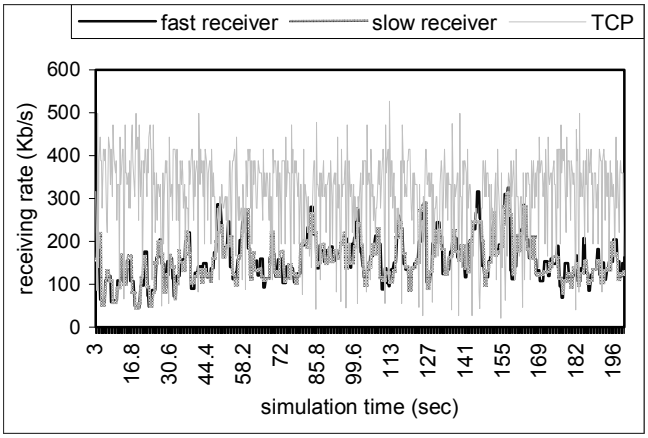

Fig. 2. TCP-fairness - Receiving rates

We use in our evaluation Random Early Drop (RED) queue in the routers to avoid synchronization in the routers' queues that will affect the evaluation results. With this approach we ensure that all the transmitted streams will receiver similar packet losses. A RED gateway, detects upcoming congestion by computing the average queue size. When the average queue size exceeds a preset minimum threshold the router drops each incoming packet with some probability. Exceeding a second maximum threshold leads to dropping all arriving packets. This approach not only keeps the average queue length low but also ensures that all flows receive the same loss ratio and avoids synchronization among the flows.

As we observe in the simulation results (Figure 2), in the beginning of the simulation, the server backs off and reduces its transmission rate, as it encounters the first packet losses. In the remaining of the simulation time we observe that the TCP traffic is transmitted with even higher rates than the initial rate, confirming the TCP-friendly behavior of our congestion control protocol.

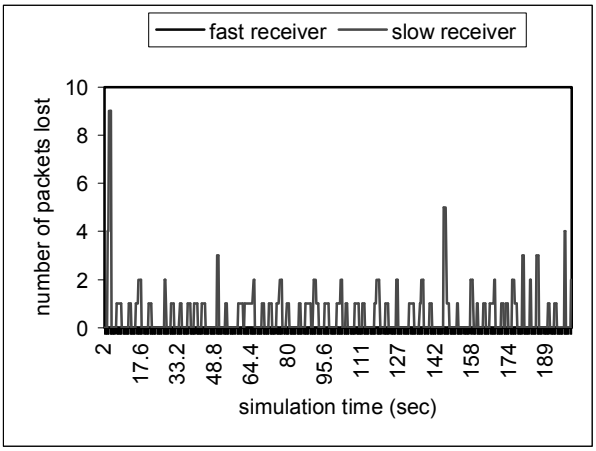

Fig. 3. TCP-fairness - Packet losses

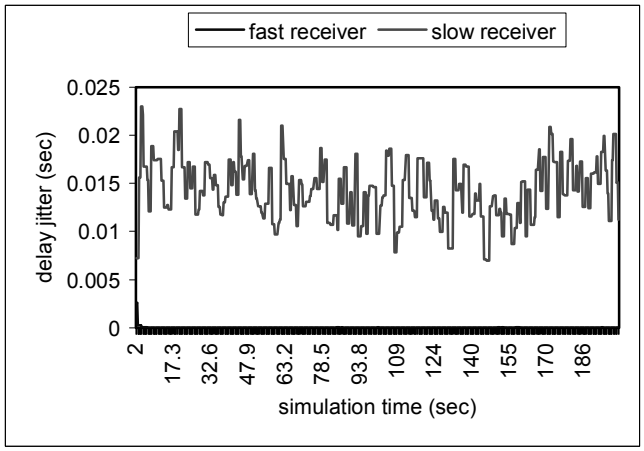

Fig. 4. TCP-fairness - Delay jitter

Fast and slow receivers enjoy the same receiving rates. However, our proposed solution is more than TCP-friendly that it ought to be. As we have previously explained, we try to integrate a TCP-friendly behavior based on RTCP sender and receiver feedback reports. The RTCP feedback reports are transmitted roughly every 5 seconds, making the sender react slowly to rapid networks changes. TCP responses are faster than our protocol and as a result TCP occupies a larger portion of the available bandwidth.

However, we confirm that in the event of competing traffic the server transmits multimedia data to the wired receivers at rates that provide acceptable receiving rates for multimedia data in terms of QoS. Figure 3 depicts the observed packet losses in the two representative nodes from the "fast" and "low" receiver's sets. Fast receivers present zero packet losses, whereas slow receivers present very low packet losses. Delay jitter in fast receivers is close to zero (it cannot even been projected in the simulation result chart), while in slow receivers the jitter values are between 1 to 23 milliseconds (figure 4).

C. Smooth Behavior

In this simulation we investigate whether or not our proposed solution for smooth transmission rates, can indeed meet our design objectives, without challenging the performance of TCP traffic. This simulation scenario has exact the same network attributes as our previous simulation in order to achieve a fair comparison.

We observe from the simulation results (Figure 5) that TCP traffic enjoys steady and high receiving rates. This is a first encouraged indication that a smooth adaptive transmission rate is a desired attribute not only for the multimedia server and the serving receivers but also for TCP traffic. We observe also in the same figure that the receiving rates in slow and fast receivers are smoother without seriously increasing the packet losses (Figure 6). We regard the amount of packets lost as low for a multimedia application. Forward Error Correction (FEC) can address to this amount of packet losses. Delay jitter values are similar to the previous simulation (figure 7). Fast receivers enjoy almost zero jitter delays.

Next in figure 8 we present the comparison of the transmission rates with and without the smooth rate

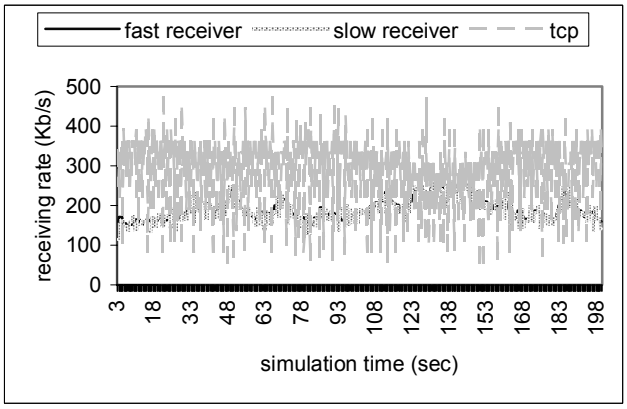

Fig. 5. Smooth behavior - Receiving rates

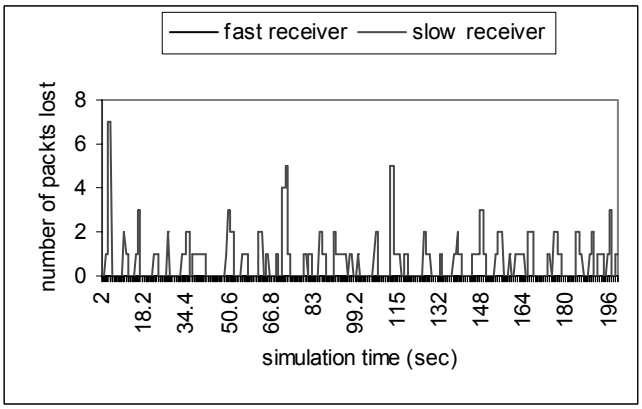

Fig. 6. Smooth behavior - Packet losses

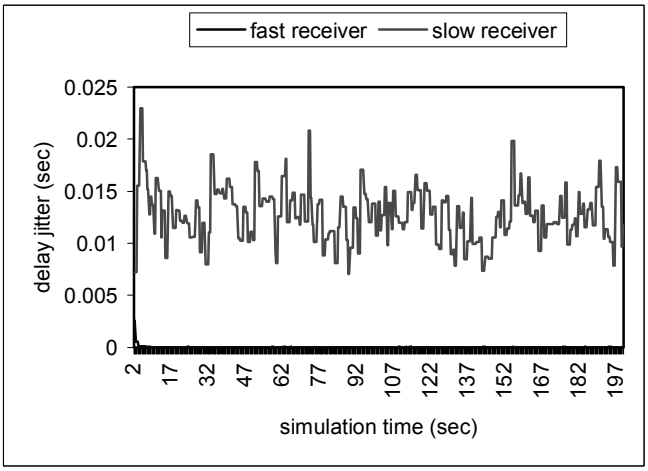

Fig. 7. Smooth behavior - Delay jitter

adaptation. It is clearly shown that, smooth transmission rates prevent oscillations and as such they are more suitable for multimedia data transmission, due to the minimal distortion in AV encoding and decoding. The only drawback one can see is the slightest increase of packet losses due to the fact that the protocol cannot respond rapidly to instantaneous network changes.

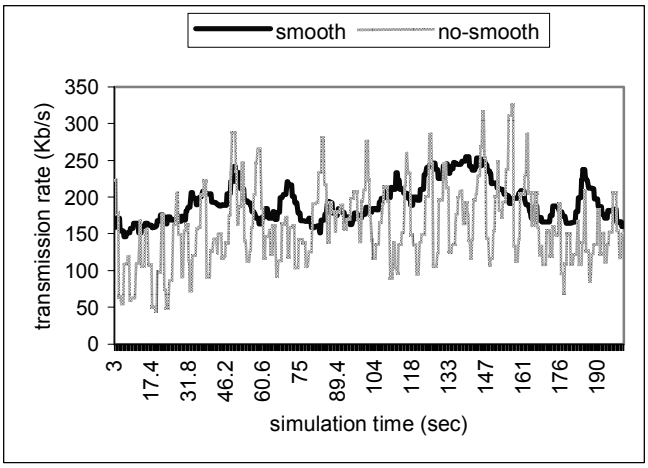

Fig. 8. Smooth rate vs non-smooth

D. Responsiveness to Changes

An important attribute of any congestion control algorithm is its responsiveness to changes of the network conditions (e.g. congestion created by competing traffic, transmission errors). This behavior is investigated through a new set of simulations in which we add competing Constant Bit Rate (CBR) traffic. We use the same simulation scenario with the previous simulation, but this time in addition to RTP and TCP traffic we transmit UDP packets at a rate of 150 Kb/s via the "bottleneck" path between routers 1 and 2. The UDP source starts transmitting at the 30th simulation second and then stops the transmission at the 100th simulation second. Our objective is to investigate how the range of values between 0 and 1 of filter γ affect the protocol's behavior at the start and the end of UDP transmission. We run various simulation sets with different values for γ. We present, though, for easier observation, only a subset of these results by taking two representative values for γ. We observe in figure 9 that for small γ values (γ=0.3) the protocol reacts faster, adopting its transmission rates more rapidly than it does for large γ values (γ=0.8). We notice however, that large γ values provide some "resistance" to competing UDP traffic. Moreover, for the same large γ values the protocol's reaction time is not as high as we expected to be when the UDP source stops its transmission. The results are very encouraging as we can assume that we managed to tune the protocol in such way that it adapts the network changes in a smooth manner without seriously reducing its responsiveness. Packet losses are not so that dramatically different (figure 10). We observe of course that large γ values (γ=0.8) cause more packet losses but the total amount is still acceptable for multimedia data transmission. Delay jitter is smooth and low for both large and small γ values (figure 11). The case when γ=0.8, presents an even smoother delay jitter, which is a desired attribute in multimedia applications.

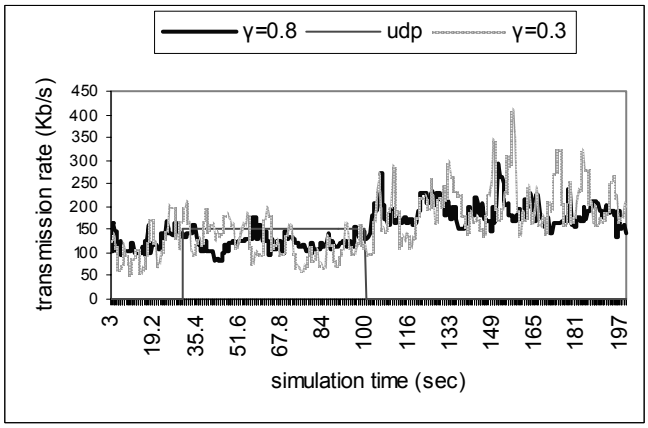

Fig. 9. Responsiveness –transmission rates

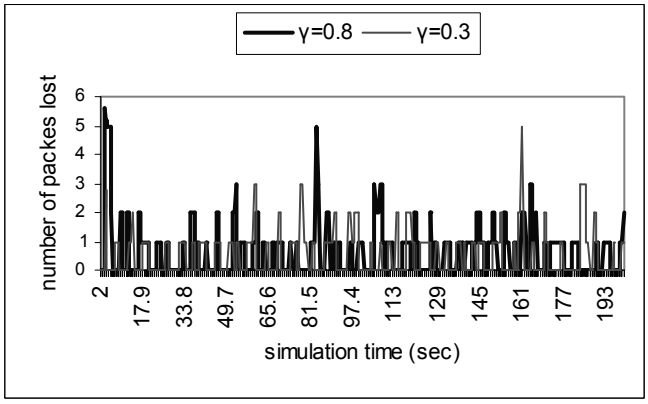

Fig. 10. Responsiveness – Packet losses

E. Selecting γ values

We have seen in the previous simulations how the selection

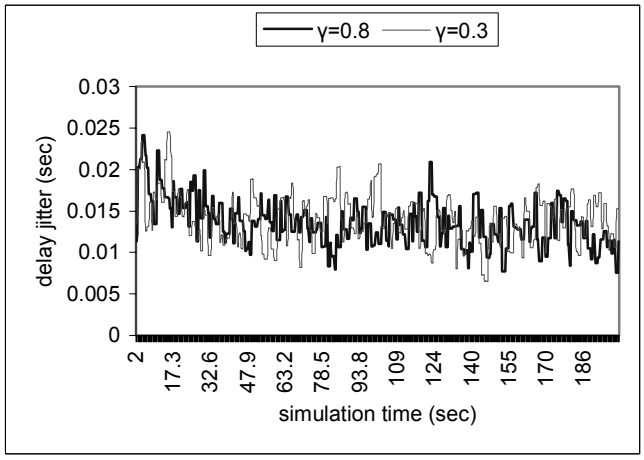

Fig. 11. Responsiveness – Delay jitter

of the value γ affects the behavior of our proposed protocol. In this work we have used a heuristic method for the selection of γ values based on simulation results. We have run several simulation sets in an effort to find a value for γ that presents an optimal performance in terms of packet losses, delay jitter, responsiveness to network changes, TCP fairness and finally smooth behavior. A good compromise was found for values between 0.5 and 0.8, in which the protocol seems to perform

better and keeps a smooth transmission rate, preventing high oscillations and packet losses. We can see in figure 12 that for γ=0.8 we have a steady transmission rate with all the characteristics mentioned above.

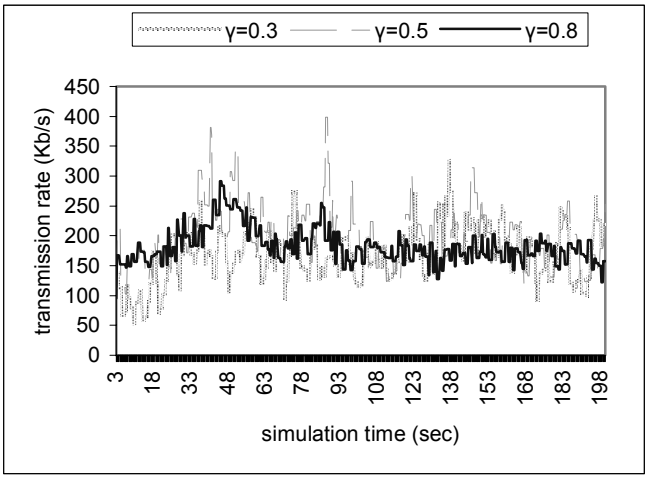

Fig. 12. Filter (γ) values

And optimized method for defining the value for filter γ would have been to dynamically adjust that value to the network changes. This method would emulate Equation (1) that is used for the estimation of the packet smooth loss rate. The weighted *m* values could be found by collecting statistical data concerning the network conditions. This part was left for future work.

IV. Conclusions - Future Work

In this paper we presented an algorithm for adaptive multimedia transmission that is TCP-friendly. Our solution relies on the RTCP sender and receiver reports, which eliminate the need for additional feedback reports. The outcome of this approach is higher bandwidth utilization for user data. We have implemented a smooth adaptive algorithm to minimize oscillations of the transmission rates that may lead to infeasible adaptation of the AV coders.

Measurements and simulation results suggested that our proposed solution maintains its TCP-friendly behavior, although the feedback reports (e.g. RTCP reports in our proposal) are transmitted on much slower scale than other TCP-friendly solutions.

Even though our proposed solution cannot antagonize other multicast control schemes, due to infrequent feedback reports, the whole concept for smoothing the transmission rates may be suitable to be used in a multicast control framework, especially if someone focus on end user perception and minimal AV encoding and decoding distortion.

In our future work we will work on protocol enhancements so that the value of filter γ will be dynamically chosen based on network statistics. We will also investigate deeper the effect of "smoothens" on other competing traffic types and loss error schemes. Finally, it is our intention to use our solution as part of the congestion control mechanism in our proposed framework presented in [7]

V. Acnowledgement

We thank the anonymous EURO-NGI 2008 reviewers for their helpful comments.

References

[1] J. Widmer and M. Handley, "Extending equation-based congestion control to multicast applications," in Proc. of ACM SIGCOMM '01, 2001.

[2] RFC 3448, M. Handley, S. Floyd, J. Padhye, J. Widmer, "TCP Friendly Rate Control (TFRC)", Network Working Group, January 2003.

[3] L. Rizzo, "pgmcc: A TCP-friendly single-rate multicast congestion control scheme," in Proc. of ACM SIGCOMM '00, 2000.

[4] Smith, H., Mutka, M., Rover, D. A Feedback based Rate Control Algorithm for Multicast Transmitted Video Conferencing, Accepted for publication in the Journal of High Speed Networks.

[5] Sisalem D., Wolisz A., "LDA + TCP - Friendly Adaptation: A Measurement and Comparison Study," in the 10th International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV'2000), June 25 - 28, 2000, Chapel Hill,NC, USA.

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

[7] C. Bouras, A. Gkamas, G. Kioumourtzis, "A Framework for Cross Layer Adaptation for Multimedia Transmission over Wired and Wireless Networks", The 2007 International Conference on Internet Computing (ICOMP'07), Las Vegas, Nevada, USA, 25 - 28 June 2007.

[8] L. Vicisiano, L. Rizzo, J. Crowcroft, "TCP - like congestion control for layered multicast data transfer", in IEEE INFOCOM, March 1998, pp. 996 - 1003.

[9] J. Pandhye, J. Kurose, D. Towsley, R. Koodli, "A model based TCP friendly rate control protocol", Proc. International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV), Basking Ridge, NJ, June 1999.

[10] http://www.isi.edu/nsnam/ns/