SRAMT-S: A Hybrid Sender And Receiver-Based Adaptation Scheme For TCP Friendly Multicast Transmission Using Simulcast Approach

Ch. Bouras, A. Gkamas
Computer Technology Institute, Riga Feraiou 61, GR-26221 Patras, Greece
Computer Engineering and Informatics Dept., Univ. of Patras, GR-26500 Patras, Greece
Tel:+30-(0)610-{960375, 960355}
Fax:+30-(0)610-{996314, 960358}
e-mail: {bouras, gkamas}@cti.gr

Abstract: In this paper, we describe a hybrid sender and receiver-based adaptation scheme for multicast transmission of multimedia data using the simulcast approach, which we call SRAMT-S (Sender-Receiver based Adaptation scheme for Multicast Transmission using Simulcast). The most prominent features of SRAMT-S are its distributed (to sender and receivers) transmission rate estimation algorithm and its innovative RTT (Round Trip Time) estimation algorithm based on one-way delay measurements. SRAMT-S is using both an AIMD algorithm and a TCP model in order to estimate a TCP friendly bandwidth share. We evaluate SRAMT-S through a number of simulations in order to examine its behaviour to a heterogeneous group of receivers and its behaviour against TCP connections. Main conclusion of the simulation was that SRAMT-S has friendly behaviour against the dominant traffic types of today's Internet and treats a heterogeneous group of receivers with fairness.

Key words: Multimedia technologies and networking, Media streaming, Multicast Congestion Control, TCP friendliness.

1. INTRODUCTION

The multicast transmission of real time multimedia data is an important component of many current and future emerging Internet applications, like videoconference, distance learning and video-on-demand. The heterogeneous nature of the Internet makes the multicast transmission of real time multimedia data a challenge. Different receivers of the same multicast stream may have different processing capabilities, different loss tolerance and different bandwidth available in the paths leading to them.

In addition, any application that transmits data over the Internet should have a friendly behaviour towards the other flows that coexist in today's Internet and especially towards the TCP flows that comprise the majority of flows. We define as TCP friendly flow, a flow that consumes no more bandwidth than a TCP connection, which is traversing the same path with that flow ([14]).

The methods proposed for the multicast transmission of multimedia data over the Internet can be generally divided in three main categories, depending on the number of multicast streams used:

– The sender uses a single multicast stream for all receivers ([1], [4], [19]). This results to the most effective use of the network resources, but on the other hand the fairness problem among the receivers arises especially when the receivers have very different capabilities.
– Simulcast: The sender transmits versions of the same video, encoded in varying degrees of quality. This results to the creation of a small number of multicast streams with different transmission rates ([9], [5], [3]). Each receiver joins in the stream that carries the video quality, in terms of transmission rate, that it is capable of receiving.
– The sender uses layered encoded video, which is video that can be reconstructed from a number of discrete data layers, the basic layer and more additional layers, and transmits each layer into different multicast stream ([10], [18]). The basic layer provides the basic quality and the quality improves with each additional layer. The receivers subscribe to one or more multicast streams depending on the available bandwidth into the network path to the source.

The subject of transmission of TCP friendly flows over networks has engaged researchers all over the world ([14], [18], [19]). Various adaptation schemes deploy an analytical model of TCP ([14]) in order to estimate a TCP friendly bandwidth share. With the use of this model, the average bandwidth share of a TCP (r_{tcp}) connection is determined (in bytes/sec) with the following equation:

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

Where P is packet size, l is the packet loss rate, t_{out} is the TCP retransmission timeout, t_{RTT} is the RTT of the TCP connection and D the number of acknowledged TCP packets by each acknowledgment packet. SRAMT-S is using the above described analytical model of TCP, in order to estimate TCP friendly bandwidth shares. For the following of this paper 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).

In this paper, we propose an adaptation scheme for multicast transmission of multimedia data over best effort networks, like the Internet, which provides the most satisfaction to the group of receivers, with the current network conditions. We call this adaptation scheme SRAMT-S (Sender-Receiver based Adaptation scheme for Multicast Transmission using Simulcast) and it is a hybrid sender and receiver-based adaptation scheme. SRAMT-S is trying to transmit TCP friendly multicast flows with the use of simulcast approach. SRAMT-S creates n different multicast streams (in most network conditions a small number of different multicast streams is enough - typically 3 or 4 multicast streams), each one within certain bandwidth limits. All the multicast streams carry the same multimedia information, each one of them having a different quality and as result different transmission rate. In addition, SRAMT-S estimates the transmission rate of the multicast streams both with the use of an Additive Increase Multiple Decrease (AIMD) algorithm and with the use of an analytical model of TCP ([14]).

2. SRAMT-S DESCRIPTION

2.1 Architecture

With the use of SRAMT-S, the sender transmits multimedia data to a group of m receivers with the use of multicast. Sender is using the simulcast approach, and transmits the same multimedia information in n different multicast streams. The transmission rate within each multicast stream is adapting within its limits (each multicast stream has an upper and lower limit in its transmission rate) according to the capabilities of the receivers listening to it. The receivers join the multicast stream which suits better their requirements (available bandwidth between the sender and the receiver, etc) and if during the transmission of multimedia data the network conditions to the path between them and the sender change, the receivers have the capability to change the multicast stream that they listen and join a multicast stream that accomplish better their requirements. The communication between the sender and the receivers is based on RTP/RTCP sessions and the sender is using the RTP protocol to transmit the multimedia information and the participants (the sender and the receivers) use the RTCP protocol in order to exchange control messages.

In the following paragraphs, we give details about the different aspects of SRAMT-S.

2.2 Sender operation

Figure 1 shows the organisation and the architecture of the SRAMT-S's sender entity. The sender generates *n* different stream managers. In each stream manager an arbitrary number of receivers managers is assigned. Each receiver manager corresponds to a unique receiver that has joined the stream controlled by this stream manager. The synchronisation server is responsible for the management, synchronization and intercommunication between stream managers.

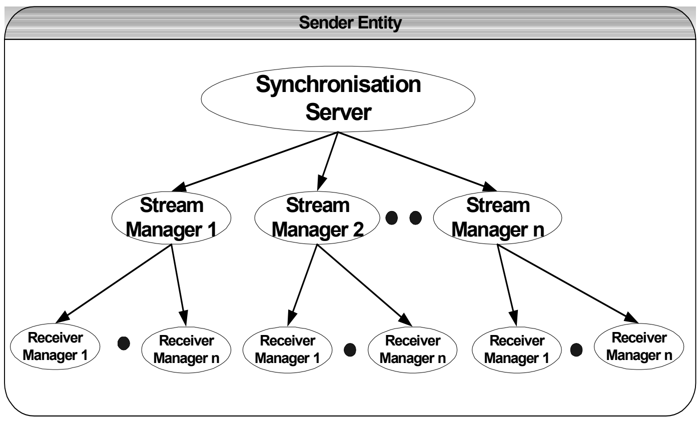

Figure 1. The architecture and the data flow of the Server.

The stream manager entity is responsible for the maintenance and the monitoring of one of the *n* different multicast streams that are generated by the sender. The stream manager creates a new receiver manager every time receives a RTCP report from a new receiver. Each receiver manager corresponds to a unique receiver. It processes the RTCP reports generated by the receiver and can be considered as a representative of the receiver at the side of the sender. It can interact only with one stream manager at a given time, the stream manager controlling the multicast stream from which the receiver is receiving the multimedia information. If the receiver manager does not receive RTCP reports from the receiver, which represents for long time, the stream manager suppresses its operation and releases its resources. Receiver manager based on the RTCP reports from the receiver, which represents, makes estimations about the TCP friendly bandwidth share to the path between the sender and the receiver.

With the use of RTCP adaptive feedback mechanism the receivers send their feedback to the sender in the form of RTCP receivers reports. We have added an application specific part (APP) to the RTP receiver report in order to include the receiver's estimation about the TCP friendly bandwidth share $r^i_{r\_tcp}$ (more information in section 2.3) in the path between the receiver and the sender.

When a receiver manager receives a RTCP receiver report from the receiver i (which represents) is using the packet loss rate to estimate the transmission rate r^i_{AIMD} of the receiver i with the use of an AIMD algorithm (which has been presented in [2]).

In addition, the receiver manager is using the analytical model of TCP in order to estimate a TCP friendly bandwidth share $r^i_{l\_tcp}$ in the path between the receiver and the sender: If the receiver experiences packet losses, a TCP friendly bandwidth share $r^i_{l\_tcp}$ (in bytes/sec) is estimated with the use of the equation (1) (where t^{r-i}_{RTT} is the sender estimation for RTT between that receiver and the sender (more information in section 2.5)), and l_i is the packet loss rate that the receiver i reports (more information in section 2.4)):

$$r^i_{l\_tcp} = \frac{P}{t^{r-i}_{RTT}\sqrt{\frac{2l_i}{3}} + 4t^{r-i}_{RTT}\min(1,3\sqrt{\frac{3l_i}{8}})l_i(1+32l_i^2)} \quad (2)$$

If the receiver does not experience packet losses, in order to estimate a TCP friendly bandwidth share $r^i_{l\_tcp}$, the $r^i_{l\_tcp}$ must not be increased more than a packet / RTT. For this reason receiver manager calculates the new value of $r^i_{l\_tcp}$ by adding (T_{rr} / t^{r-i}_{RTT}) packets (where T_{rr} is the time space between the current and the last receiver report of receiver i) to the previous value of $r^i_{l\_tcp}$ (the $r^i_{l\_tcp}$ is expressed in bytes/sec):

$$r^i_{l\_tcp} = r^i_{l\_tcp} + \frac{T_{\mathrm{rr}}}{(t^{r-i}_{RTT})^2}P \quad (3)$$

Then the receiver manager selects as receiver's i preferred transmission rate r^i the minimum of the $r^i_{r\_tcp}$, r^i_{AIMD}, $r^i_{l\_tcp}$:

$$r^i = \min(r^i_{r\_tcp}, r^i_{AIMD}, r^i_{l\_tcp}) \quad (4)$$

In addition each time one receiver manager receives a receiver report informs its sender manager to update the transmission rate. In order to update the transmission rate each stream manager polls the preferred transmission rates of all the receiver managers that correspond to a receiver receiving this stream and sets the transmission rate $r_{stream-j}$ of that stream to be the minimum preferred transmission rate of all the receivers receiving this stream:

$$r_{stream-j} = \min(r^i), \text{ for all receivers } i \text{ listening to stream j (5)}$$

In addition, the sender includes to all the RTP packets, which transmits, the transmission rate of all the streams. This information can be used from the receivers in order to change streams and accommodate better their requirements.

2.3 Receiver operation

Each receiver measures the characteristics of the path, which connects it with the sender and informs the sender with the use of receiver reports. Each receiver measures the following parameters of the path, which connects it with the sender:

– Packet loss rate (l_i): The receiver calculates the packet loss rate during the reception of sender multicast stream based on RTP packets sequence numbers (more information in section 2.4).
– RTT estimations (t^{e-i}_{RTT}): The receiver makes an estimation for the RTT between it and the sender based on one way delay measurements with the use of RTP packets timestamps (more information in section 2.5).

The receiver emulates the behavior of a TCP agent with the use of the analytical model of TCP and estimates a TCP friendly bandwidth share every RTT time t^{e-i}_{RTT} using equation (1). If the receiver experiences packet losses is using the following equation in order to estimate a TCP friendly bandwidth share (in bytes/sec):

$$r^i_{r\_tcp} = \frac{P}{t^{e-i}_{RTT}\sqrt{\frac{2l_i}{3}} + 4t^{e-i}_{RTT}\min(1, 3\sqrt{\frac{3l_i}{8}})l_i(1+32l_i^2)} \quad (6)$$

If the receiver does not experience packet losses, in order to estimate a TCP friendly bandwidth share $r^i_{r\_tcp}$, the $r^i_{r\_tcp}$ must not be increased more than a packet / RTT. For this reason receiver calculates the value of $r^i_{r\_tcp}$ with the following equation (in bytes/sec):

$$r^{i}_{r\_tcp} = r^{i}_{r\_tcp} + \frac{1}{t^{e-i}_{RTT}} P \quad (7)$$

Each time the receiver sends a receiver report to the sender includes the packet loss rate since the previous report and the average value of $r^{i}_{r\_tcp}$ since last receiver report.

In addition the receiver has the capability to change stream based on the information that gathers itself and the information that sender includes in to RTP packets (more information in section 0). The receivers' stream changes are synchronized at the end of a specific time period T_{epoch}, which we call epoch. There are two cases that will lead to a receiver's transition towards another stream (we assume that the receivers have been informed about the upper and lower limits of each stream during the setup of the connection):

– If the multicast stream from which the receiver is currently receiving video has already reached its lowest (or highest) transmission rate and the receivers TCP friendly bandwidth share estimation is less (or more) than the stream transmission rate, then the receiver stops listening to this stream and joins the stream of a lower quality stream (or higher quality stream), if such a stream exists.
– When a receiver that co-exists in a stream with low (or high) capacity receivers but is preferring better (or worse) quality video, so it has been unable to increase (or decrease) the video quality of the current stream. The mechanism used aims in making the SRAMT-S more conservative and operates by counting the number of consecutive epochs the receiver's TCP friendly bandwidth share estimation was greater (or lesser) than the transmission rate of the multicast stream. When this number exceeds a certain limit (for our simulations this number was set to 4 which results a minimum of 20 seconds between stream change which is a time space enough in order to take a justified decision.), we assume that the receiver has indeed higher (or lower) capabilities and move it to a better (or worse) quality stream.

We declare as unsuccessful stream change the situation when a receiver joins a stream with higher transmission rate (or a lower transmission rate) and after a sort time period (T_{change}) returns to the previous stream. During our performance evaluation, we observe that the unsuccessful stream changes by the receivers cause instability to the operation of SRAMT-S and must be avoided. In order to avoid unsuccessful stream changes by the receivers, when a receiver makes an unsuccessful stream change we avert the receiver to make the stream change, which was unsuccessful for the next $2^{k} * T_{change}$ time (where k the number of continuant unsuccessful stream changes since the last successful stream change). Due to fact that T_{change}

affects linearly the value $2^k * T_{change}$ time and the k affects the value of $2^k * T_{change}$ exponentially, we set T_{change} to 5 seconds but also other values of T_{change} can be used.

2.4 Packet loss rate estimation

Each receiver measures the packet loss rate based on RTP packets sequence numbers. In order to prevent a single spurious packet loss having an excessive effect on the packet loss estimation, receivers smooth the values of packet loss rate using the following filter, which computes the weighted average of the m most recent loss rate values (the following filter has been presented in ([19]) and has been evaluate to give a good estimation of packet loss rate):

$$l_k = \frac{\sum_{i=0}^{m-1} w_i l_{k-i}}{\sum_{i=0}^{m-i} w_i} \quad (8)$$

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. In our simulations we use m=8 and the following values for the weights w_i: $\{1,1,1,1,0.8,0.6,0.4,0.2\}$.

2.5 RTT estimations

When a receiver i receives a RTP packet from the sender, uses the following algorithm in order to estimate the Round Trip Time (RTT) between the sender and the receiver. If we assume that the sender and the receiver have synchronized clocks, receiver can use the timestamp of the RTP packet ($T_{timestamp}$) and the local time that receives that packet ($T_{receiver}$) in order to estimate the one way delay form sender to receiver (T_{oneway}):

$$T_{oneway} = T_{receiver} - T_{timestamp} \quad (9)$$

If the path between the source and the receiver was symmetric and the path had the same delay into both directions, the RTT between the sender and the receiver would be twice the T_{oneway}:

$$t_{RTT} = 2T_{oneway} \quad (10)$$

Until now, we have made two assumptions: (1) the sender and the receiver have synchronized clocks (2) the path between the sender and the receiver is symmetric. The above assumptions are not true for the Internet and as results in order to get accurate RTT estimations (t_{RTT}^{e-i}) receivers have to take the above assumptions into account. For this reason, we use a parameter a and we can write the equation (10) as:

$$t_{RTT}^{e-i} = (1+a)T_{oneway} \quad (11)$$

The parameter a is used in order to smooth the estimation of the RTT due to the potential unsynchronized clocks between the receiver and the sender and due to the potential asymmetry of the path between the sender and the receiver. In order to avoid solely phenomenon to affect the RTT estimations receivers pass the t_{RTT}^{e-i} values through a filter similar to the filter, which they use for filtering the values of packet loss rate (more information in section 2.4).

In order to estimate the value of parameter a, receivers need an effective estimation of RTT, which can be acquired, with the use of RTCP reports: The RTCP receiver report contains the t_{LSR} (the timestamp of most recent RTCP sender report from the sender) and t_{DLSR} (The delay between receiving the last sender report from sender and sending this receiver report) values. As result the sender can made an effective RTT measurement for the path between it and a receiver by using the following equation (where A is the time which the sender receives the receiver report from that receiver):

$$t_{RTT}^{r-i} = A - t_{LSR} - t_{DLSR} \quad (12)$$

The sender estimates an effective RTT measurement for a receiver i every time receives a receiver report from that receiver and includes this effective RTT measurement (with the id of the receiver) in to the next RTP packet of all the streams.

A receiver after receives an effective RTT measurement from the sender, estimates an appropriate value for the parameter a using the following equation:

$$a = \frac{t_{RTT}^{r-i}}{T_{oneway}} - 1 \quad (13)$$

Figure 2 shows the real values of RTT and the values, which obtained with the above RTT estimations algorithm during the transmission of multimedia data with the use of SRAMT-S over a 1Mbit link with background traffic produced by an on/off traffic generator using an exponential distribution with transmission rate of 0.5 Mbps during on times. This figure shows that in most of the cases the above algorithm give a good approximation of the real RTT values.

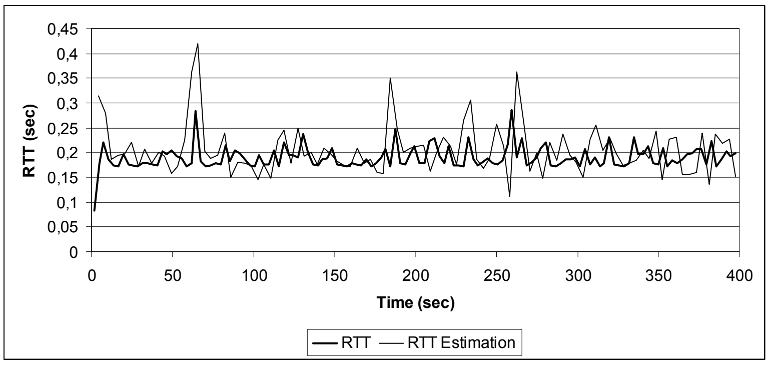

Figure 2. RTT Estimations

2.6 Extensions to RTP/RTCP

As we have already mentioned, the operation of SRAMT-S is based on the transmission with the use of RTP/RTCP. RTP provides an extension mechanism to allow individual implementations that require additional information to be carried in the RTP data packet header. SRAMT-S uses the extension mechanism of RTP in order to add to the following fields in to RTP header:

– T_{epoch} : The specific time period called epoch, which the sender adds to all RTP packets. At the end of the epoch the receivers have the capability to change stream.
– 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.
– Current transmission rates of sender streams $r_{stream-j}, j = 1,..,n$
– End of epoch flag: This flag is used in order the receiver to be informed about the end of an epoch and synchronize their stream change actions.

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 report an application specific part, which contains the average value of their estimations for TCP friendly bandwidth share $r^{i}_{r\_tcp}$ since last receiver report.

The RTP/RTCP protocols with the incorporation of the above described extensions can support in whole the operation of SRAMT-S.

2.7 Synchronization of stream changes

Similar research has shown ([11]) that, if the receivers synchronize their stream changes, the above problems can be minimized. For this reason the receivers' stream changes are synchronized in the end of each epoch. The sender marks the next RTP packet in all multicast streams after the end of an epoch with a special flag, which indicates the end of the epoch. However due to the network heterogeneity and packet losses, some receivers may not receive the special marked packet, or receive it in different time points. For this reason the sender includes the epoch duration T_{epoch} in all the RTP packets that transmits. Receivers can change streams either when receive a special marked packet or after $(T_{\text{epoch}} + T_{oneway})$ time after the end of the previous epoch (where T_{oneway} is the one way (sender to receiver) delay estimation of that receiver). During our simulation we set the T_{epoch} to be 5 seconds in order to allow receivers to quickly find the stream which fulfils better their requirements. The small value of T_{epoch} does not cause problems due to the tracing and suspending of unsuccessful stream changes (more information in section 2.3).

2.8 Scalability issues

The RTCP adaptive transmission mechanism defines as 5 sec the minimum value for RTCP report retransmission timeout. The RTCP adaptive transmission mechanism has as result the interval between the RTCP reports (each participant sends) to increase when the group of the participant increases.

In order to ensure that, when the group of the participants increases, the sender will collect feedback information representing all the receivers, we do the following modification to the RTCP adaptive transmission mechanism: When the RTCP adaptive transmission mechanism suggests a big retransmission interval more that $T_{suspent}$ (which means that the number of participants has increase too), the receivers is using the partial suppression method proposed in [13] to control the transmission of the RTCP reports. According to that partial suppression method, the receivers are using a truncated exponentially distributed retransmission timer in the interval $[0, T_{rand}]$ with density of:

$$T_{wait} = \frac{1}{\exp^{\lambda} - 1} * \frac{\lambda}{T_{rand}} \exp^{\frac{\lambda}{T_{rand}} z} \quad \text{if } 0 \le z \le T_{rand} \quad (14)$$

$$T_{wait} = 0 \text{ otherwise}$$

Each receiver schedules the RTCP retransmission timeout to be T_{wait}. If the receiver listens in the multicast session, a receiver report from an other receiver with TCP friendly bandwidth share $r^i_{r\_tcp}$ similar to its estimation of TCP friendly bandwidth share (we consider that two TCP friendly bandwidth shares are similar when they differ up to 2%), this receiver suspend the transmission of its receiver report. As [13] shows analytically, with the some selection of the equation (14) parameters (λ, T_{rand}), for 10.000 receivers less than 10 feedback messages are generated for each event the receivers are reporting on. During our simulations we set $T_{suspent}$ to 10 sec in order to ensure that the sender will always have feedback information, which represents all the receivers. With the above described mechanism, when the number of the receiver is small the sender collects information from all the receivers. When the number of the receivers is big the sender collects information from a part of receivers, which represents all the receivers.

3. MECHANISM EVALUATION

In this section, we present a number of simulations that we made in order to analyze the behavior of SRAMT-S, during the multicast transmission of multimedia data with the use of simulcast approach. We implemented SRAMT-S and run simulations in the LBNL network simulator ns-2 ([12]).

3.1 Heterogeneous multicast environments - TCP friendliness

In this simulation we investigate the performance of SRAMT-S in a heterogeneous multicast environment and its TCP friendliness. We choose to investigate the TCP friendliness of SRAMT-S in a multicast distribution tree without any shared links among the receivers. With this approach, we investigate the TCP-friendliness of SRAMT-S without having to consider the effects of interaction between different receivers, traversing multiple routers and different round trip delays among the receivers.

Figure 3 shows the topology of this simulation. The bandwidth of each link is given to the simulation topology and varies from 0.8 Mbps to 10.0 Mbps. All the links in the simulation topology are full duplex, have delay 10 ms and they use the RED (Random Early Drop) ([8]) policy to their queues. With the use of RED, we assure that all the flows receive the same loss ration and we avoid the synchronization among the flows. In this topology we have one sender (S), which transmits multimedia data with the use of SRAMT-S to a group of 6 receivers (R1-R6) with different capabilities. The sender transmits three multicast streams with the following limits: stream one: 100Kbps-600Kbps, stream two: 500Kbps-1100Kbps and stream three: 800Kbps-1600Kbps.In addition we have 3 TCP sources TCP1, TCP2, TCP3, which transmit data to Sink1, Sink2, Sink3. We model the TCP sources as “4.3BSD Tahoe TCP” ([17]) sources, which always have data to send during the simulation. In the simulation topology we have three bottleneck links (C1-C2, C1-C3 and C1-C4) and each router (C2, C3, C4) is shared between a sender multicast stream and a TCP stream with the same RTT time as the sender / receiver pair.

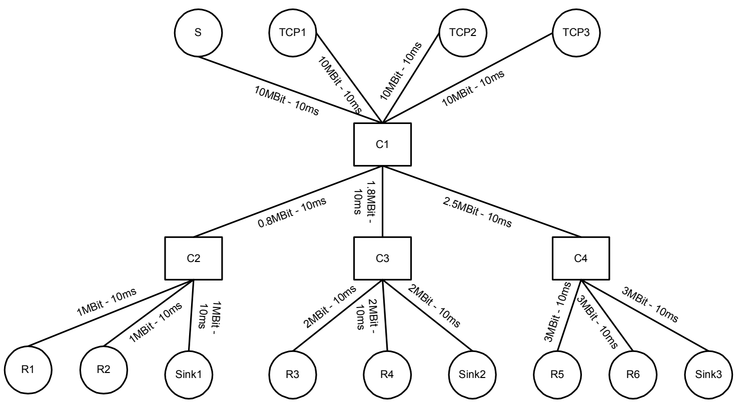

Figure 3. Topology with no share links

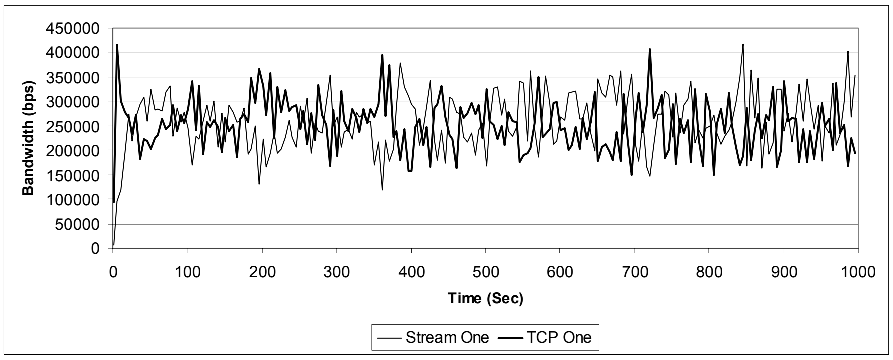

Figure 4. Bandwidth distribution on C1-C2 bottleneck link

We execute this simulation for 1000 seconds and the sender starts transmitting the stream one with transmission rate 100Kbps, the stream two with transmission rate 600Kbps and the stream three with transmission rate: 1100Kbps. Receivers R1 and R2 join the stream one, receivers R3 and R4 join the stream two and Receivers R5 and R6 join the stream three.

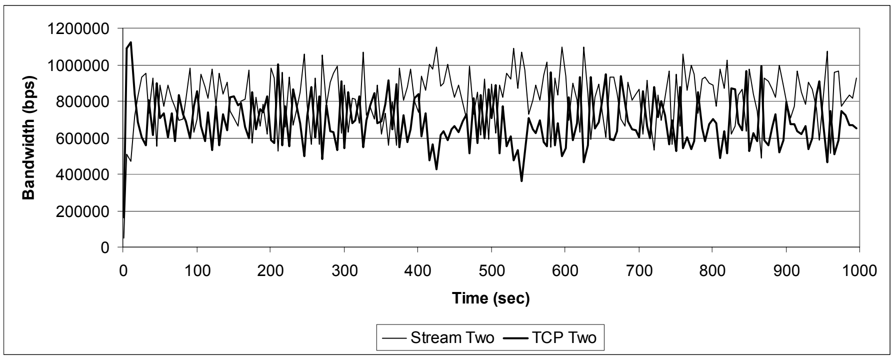

Figure 5. Bandwidth distribution on C1-C3 bottleneck link

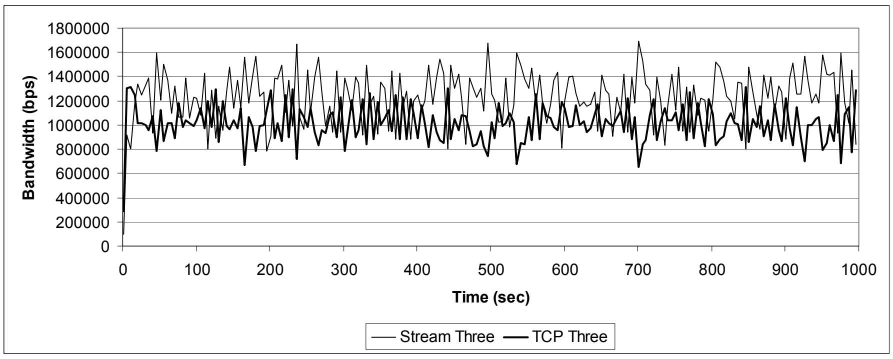

Figure 6. Bandwidth distribution on C1-C4 bottleneck link

Figure 4, Figure 5 and Figure 6 shows the bandwidth distribution to bottleneck links C1-C2, C1-C3 and C1-C4 respectively. These figures indicate that SRAMT-S is in general fair towards to TCP connections and treats the heterogeneous group of the receivers with fairness. In all bottleneck links SRAMT-S behaves as is expected, and shares the available bandwidth with the TCP connection with the same RTT delay. The behavior of SRAMT-S ("seeking" for available bandwidth and reaction to congestion) leads some times to get more bandwidth share that TCP and some times to get less bandwidth share that TCP, but in long term both the sender streams and the TCP flows get the approximately the same bandwidth share of the bottleneck links.

3.2 Multicast environments with share links

In this simulation we investigate the performance of SRAMT-S in a heterogeneous multicast environment with a multicast distribution tree that is shared among the receivers. With this approach, we investigate the behavior of SRAMT-S, when the actions of one receiver affect other receivers.

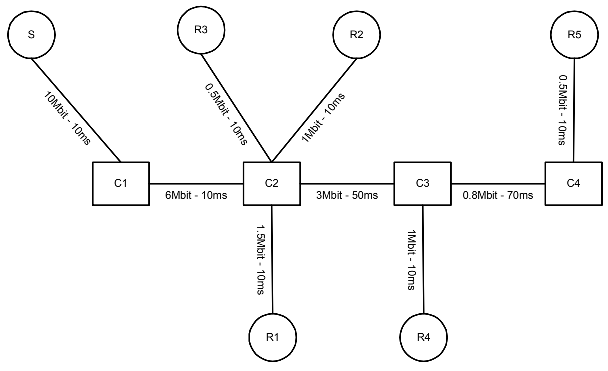

Figure 7. Topology with share links

Figure 7 shows the topology of this simulation. The bandwidth of each link is given to the simulation topology and varies form 0.5 Mbps to 10.0 Mbps. All the links in the simulation topology are full duplex, have delay, which varies form 10 ms to 70 ms, and they use again the RED policy to their queues. In this topology we have one sender (S), which transmits multimedia data with the use of SRAMT-S to a group of 5 receivers (R1-R5) with different capabilities. The sender transmits three multicast streams with the following limits: stream one: 100Kbps-600Kbps, stream two: 600Kbps-1100Kbps and stream three: 1100Kbps-1700Kbps.In addition each router (C2, C3, C4) is shared between the sender streams and an uncorrelated background traffic which consumes maximally the 60% of the router capacity. In order to produce the uncorrelated background traffic, we use a traffic generator with active and idle periods. During the active periods the transmission rate of the traffic generator follows a Pareto distribution with a scale factor of 1.1 and a mean of 20 packets. Active transfer phases are then followed by idle periods drawn by a Pareto distribution with a scale factor of

1.8 and a mean 0.5 seconds. As [15] suggests the above traffic generator models background web traffic.

We execute this simulation for 1000 seconds and the sender starts transmitting the stream one with transmission rate 100Kbps, the stream two with transmission rate 600Kbps and the stream three with transmission rate: 1100Kbps. In order to avoid synchronization, the receivers join randomly the stream one during the first 3 seconds of the simulation.

Figure 8 shows the bandwidth share of the receivers 1 to 5 during this simulation. As this figure suggests after some seconds each receives joins the stream, which we expect and receives also a bandwidth share close to the bandwidth share, which we expect. The receivers after some unsuccessful stream changes (during the first 100 seconds) have join the sender stream which fulfills better their capabilities and stay at that stream until the end of the simulation (due to the tracing of unsuccessful stream changes that SRAMT-S offers). In addition, due to the synchronization of stream changes the undesirable problems are minimal and in general the receivers actions does affect the bandwidth shares of the other receivers. For example the actions of receiver 4, which affects the bandwidth share of receiver 5 are minimal: The receiver 4 tries unsuccessfully to join stream three (57th second) and after some seconds (77th second) returns to stream two and keeps receiving this stream until the end of the simulation.

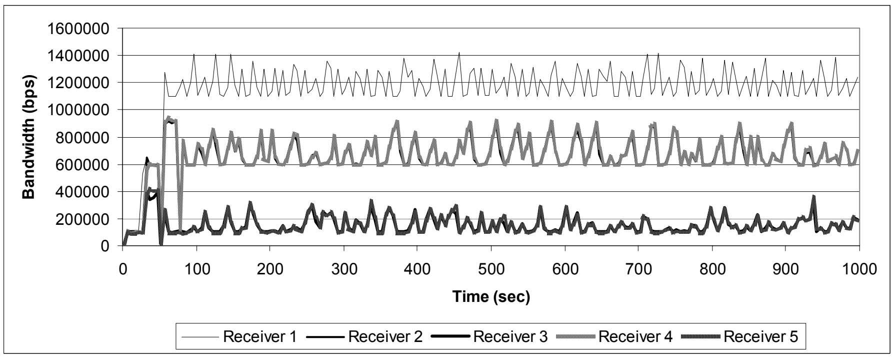

Figure 8. Bandwidth shares of Receiver 1 to Receiver 5

4. CONCLUSION - FUTURE WORK

In this paper, we present the behavior investigation of the SRAMT-S, a mechanism for multicast transmission of adaptive multimedia data in a heterogeneous group of receivers with the use of replicated streams. SRAMT-S is using a hybrid sender and receiver-based adaptation scheme and uses both an AIMD algorithm and a TCP model to estimate a TCP friendly bandwidth share. We investigate the behavior of SRAMT-S through a number of simulations. Main conclusion of the simulation was that SRAMT-S has friendly behavior against the dominant traffic types (TCP traffic) of today's Internet and good behavior during congestion condition. In addition SRAMT-S treats with fairness a heterogeneous group of Clients.

Our future work includes the investigation of dynamically adding more streams instead of the static number of streams that SRAMT-S supports now. In addition we will enhance the SRAMT-S in order to support multicast of layered encoded video and we will evaluate this new version. Moreover we plan to implement a prototype of SRAMT-S and evaluate its operation over the Internet.

5. REFERENCES

[1]. J.-C. Bolot, T. Turletti, I. Wakeman. "Scalable feedback control for multicast video distribution in the Internet" In Proceedings of SIGCOMM 1994, pp. 139-146, London, England, ACM SIGCOMM, August 1994.

[2]. Ch. Bouras, A. Gkamas, "Streaming Multimedia Data With Adaptive QoS Characteristics", Protocols for Multimedia Systems 2000, pp 129-139, Cracow, Poland, October 22-25, 2000.

[3]. Ch. Bouras, A. Gkamas, An. Karaliotas, K. Stamos, "Architecture And Performance Evaluation For Redundant Multicast Transmission Supporting Adaptive Qos" 2001 International Conference on Software, Telecommunications and Computer Networks (SoftCOM 2001) Split, Dubrovnik (Croatia) Ancona, Bari (Italy), October 09-12, 2001.

[4]. C. Bouras, A. Gkamas, "A Mechanism for Multicast Multimedia Data With Adaptive QoS Characteristics" 6th International Conference on Protocols for Multimedia Systems-PROMS 2001, pp. 74-88, Enschede, The Netherlands, October 17-19 2001.

[5]. S. Y. Cheung, M. Ammar, X. Li. "On the Use of Destination Set Grouping to Improve Fariness in Multicast Video Distribution", INFOCOM 96, San Francisco, March 1996.

[6]. C. Diot - Sprint Labs "On QoS & Traffic Engineering and SLS-related Work by Sprint", Workshop on Internet Design for SLS Delivery, Tulip Inn Tropen, Amsterdam, The Netherlands, 25 - 26 January 2001.

[7]. S. Floyd, K. Fall, "Promoting the Use of End-to-End Congestion Control in the Internet," IEEE/ACM Transactions on Networking, 1998.

[8]. S. Floyd, and V. Jacobson. 1993. Random Early Detection Gateways for Congestion Avoidance. IEEE/ACM Transactions on Networking, vol. 1,4: pp. 397-413.

[9]. T. Jiang, E. W. Zegura, M. Ammar, "Inter-receiver fair multicast communication over the Internet". In Proceedings of the 9th International Workshopon Network and Operating Systems Support for Digital Audio and Video (NOSSDAV), pp. 103-114, June 1999.

[10]. T. Kim, M. H. Ammar , "A comparison of layering and stream replication video multicast schemes", Proc. NOSSDAV'01, Port Jefferson, NY, June 25-26, 2001.

[11]. S. McCanne, V. Jacobson. Receiver-driven layered multicast. 1996 ACM Sigcomm Conference, pp. 117-130, August 1996.

[12]. S. McCanne, S. Floyd. The UCB/LBNL network simulator, software online. http://www.isi.edu/nsnam/ns/.

[13]. J. Nonnenmacher and Ernst W. Biersack, "Optimal multicast feedback, " in Proceedings of the Conference on Computer Communications (IEEE Infocom), San Francisco, USA, Mar. 1998.

[14]. 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.

[15]. K. Park, G. Kim, and M. Crovella, "On the Relationship Between File Sizes, Transport Protocols, and Self-Similar Network Traffic", In Proceedings of the International Conference on Network Protocols, pp 171-180, October, 1996.

[16]. H. Shculzrinne, S. Casner, R. Frederick, V. Jacobson, RTP: A Transport Protocol for Real-Time Applications, RFC 1889, IETF, January 1996.

[17]. W. Stevens, TCP Slow Start, Congestion Avoidance, Fast Retransmit and Fast Recovery Algorithms, RFC 2001, January 1997.

[18]. D. Sisalem and A. Wolisz, "MLDA: A TCP-friendly congestion control framework for heterogeneous multicast environments," in Eighth International Workshop on Quality of Service (IWQoS 2000), Pittsburgh, PA, June 2000.

[19]. J. Widmer and M. Handley, "Extending equation-based congestion control to multicast applications," in ACM SIGCOMM, August 2001.