Adaptive Smooth Simulcast Protocol for Multimedia Transmission

Christos Bouras, Apostolos Gkamas and Georgios Kioumourtzis
Research Academic Computer Technology Institute, N. Kazantzaki Str., University of Patras, 26500 Rion, Greece
&
Computer Engineering and Informatics Department, University of Patras, 26500 Rion, Patras, Greece
Tel: +30 2610 {960375, 960465, 960316}
Fax: +30 2610 960358
email: bouras@cti.gr, gkamas@cti.gr, gkioumou@ceid.upatras.gr

Abstract

We introduce Adaptive Smooth Simulcast Protocol (ASSP) for simulcast transmission over best-effort networks. ASSP is a new multiple-rate protocol that implements a single rate TCP-friendly protocol as the underlying congestion control mechanism for each simulcast stream. ASSP is build on top of the RTP/RTCP protocol and exploits the RTCP sender and receiver reports for the dissemination of feedback information. The key attributes of ASSP are: a) TCP-friendly behavior, b) adaptive per-stream transmission rates, c) adaptive scalability to large sets of receivers and finally d) smooth transmission rates that are suitable for multimedia applications. We evaluate the performance of ASSP and investigate its behavior through simulations conducted with the network simulator software (ns2).

1. Introduction-related Work

Multicast transmission is a preferable solution for multimedia data dissemination in respect of the nature of multimedia applications in which in many cases the same information is delivered to a group of receivers instead of a single receiver. We can transmit the same multimedia data with a finite and small number of different streams that differ in information quality, and hence they require different bandwidth capabilities.

The research community has provided a sufficient number of new and promising proposals for congestion or flow control. PLM [2] is an earlier works in which receivers are able to join different multicast groups, in accordance with the observed congestion in the network. FLID-DL [3] tries to mitigate known drawbacks related to long IGMP leave latencies, which leave the network in a congested state. Fine-grained layered multicast [4] addresses the drawbacks of cumulative layering regarding the coarse-grained adaptation of receivers from previous congestion control proposals. STAIR (Simulate TCP's Additive Increase/multiplicative decrease with Rate-based) [5], further minimizes the IGMP join and leave requests with the concept of "stair layers". SMCC (Smooth Multi-rate Multicast Congestion Control) [6] employs cumulative layered multicast and is based on TFMCC [7] for the adaptation of the individual layers.

Table 1. Mathematical notations

Symbol	Meaning
$rx_i^{inst}(t)$	Instantaneous TCP-Friendly bandwidth share at receiver i at time (t)
$tx_j^{inst}(t)$	Instantaneous transmission rate of stream j at time (t)
$avg_{tx}^{j}(t)$	Average transmission rate of stream j at time (t)
$avg_{rx}^{i}(t)$	Average receiving rate of receiver i at time (t)
$\Phi()$	EWMA averaging function
α	Exponential averaging factor
β	Transmission rate factor
κ	Current stream leave factor
λ	Higher stream join factor
Δt	Time period over which join or leave decisions are made
j	stream j
threshold	low BW limit of stream j

We present in this work Adaptive Smooth Simulcast Protocol (ASSP), a new multi-rate transport protocol for multicast transmission over best-effort networks. The building block of ASSP is based on our previous work that comprises a single-rate multicast congestion control protocol named ASMP ([10], [11]). ASSP is the extension of ASMP from the single-rate multicast congestion control schemes to simulcast transmission. As a result the transmission of each multimedia stream in the context of ASSP is based on the underlying

congestion control mechanism. The ASSP itself is responsible to handle all the issues related to simulcast transmission and the management and synchronization of the multiple multicast streams. ASSP exploits the concept of "*smooth transmission*" to avoid high oscillations of the transmission rate of each individual stream and minimizes the join and leave attempts to various multicast streams.

The rest of this paper is organized as follows: In the next section we provide a detailed description of ASSP explaining its internal functions. Results from simulations conducted with ns-2 [12] are presented in section 3. We conclude our paper in section 4.

2. ASSMP description

In ASSP the sender transmits a number of different multicast streams that carry the same multimedia data which differ in quality. Streams are adaptive so that they can accommodate a fair large number of receivers that have similar bandwidth capabilities. A second innovation in ASSP lays in the way the receivers make the decision to join or leave a higher or lower quality stream, based on network statistical measurements and a "strict" decision making algorithm. A high level overview of the functionality of ASSP is presented below: (a) The receiver measures a smooth TCP-friendly bandwidth share with the use of the analytical model of TCP and statistical data that is related to network conditions, (b) the receiver compares this TCP-friendly bandwidth share with both the sender's transmission rates in all streams and the limits of the upper and lower stream. In predefined time slots the receiver can leave and join a lower or higher capacity stream based on a decision making algorithm (5), (c) the sender gathers the RTCP [13] receiver reports and performs per-stream transmission rate adaptations based on the reported $rx_i^{inst}(t)$ values (equation 2), (d) the sender adds the average transmission rate of each stream $avg_{tx}^{j}(t)$ in the application part of the RTCP sender reports, (e) in fixed time intervals the sender notifies all receivers, so that join and leave requests are synchronized.

2.1. Sender's feedback functions

Sender's feedback functions are implemented via stream managers that gather the RTCP receiver reports and exploit the received feedback reports of each stream. The sender performs the following procedures in the event of a newly arrived RTCP receiver report:

Receive RTCP packet:

$$compare(rx_i^{inst}(t)) \quad (1)$$

$$adjustTransmissionRate()$$

Subroutine Compare():

$$tx_j^{inst}(t) = \min(rx_1^{inst}(t)....rx_i^{inst}(t)) \quad (2)$$

In other words the stream manager of stream j compares the reported $rx_i^{inst}(t)$ at time t from receiver i with all previous reported values from all receivers that belong to stream j. The new transmission rate (set by the subroutine *adjustTransmissionRate()*) of stream j is the lowest reported $rx_i^{inst}(t)$ value. Following, the sender adds the $tx_j^{inst}(t)$ value in a list and in the event of feedback timer timeout it averages the transmission rate of steam j with the use of Equation (3):

$$avg_{tx}^{j}(t) = \frac{1}{\Delta t}\int_{t}^{t+\Delta t} tx_j^{inst}(t)\,dt \quad (3)$$

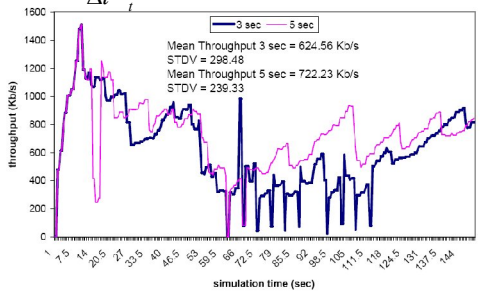

Figure 1. Effects of sender's average timeout interval.

The sender adds the average transmission rate of each stream in the application part of the sender RTCP report. In this work, we set the timeout interval for averaging the transmission rates to 5 sec based on simulation results which show that the above value is a good compromise between responsiveness and subscription level accuracy. Lower timeout values do not provide the desired confidence level for such decisions. Figure 1 presents simulation results of different timeout values.

We use for our experiments the network topology in Figure 9 and plot the achieved throughput of receiver R5 with different timeout intervals. We observe that short timeout values (3 seconds) create oscillatory behavior as this sampling window provides short term statistical information with low accuracy level. Values from 5 sec and above provide the required statistical accuracy so that we prevent oscillations. Therefore, for implementing ASSP we suggest a timeout value of 5 seconds which provide high statistical accuracy without challenging the responsiveness of ASSP due to network dynamics.

2.2. Receiver's functions

The ASSP receiver is responsible for monitoring the reported transmission rates from the sender and adjusts its subscription level[1] based on a decision making algorithm. Upon the arrival of a new RTCP packet the receiver checks the join flag. We use an application part of the RTCP packet to set a flag in order to provide receivers with a notification concerning the join/leave requests to a higher or lower capacity stream. When the flag in true, receivers measure the average receiving rate over a period Δt. However, when a receiver leaves its current stream and joins a higher or lower stream there is a period in which the receiver does not receive any data packet. If the measurements for the receiving rate were based only on instantaneous values the receiver would appear to have zero receiving rates during this leave period. This situation will lead to oscillatory behavior when receivers change their subscription level. Therefore, we use an Exponentially Weighted Moving-Average (EWMA) function $\Phi(\ )$, with averaging factor $0 < \alpha < 1$. The following operations take place when a data RTCP packet is received at receiver i:

Receive RTCP packet:

$$\begin{aligned} &if\ (J_{flag} = 1)\ then \\ &compare(avg_{rx}^{i}(t)) \\ &else\ do\ nothing \\ &end\ if \end{aligned} \tag{4}$$

Subroutine Compare():

$$\begin{aligned} &if\ (avg_{rx}^{i}(t) > avg_{tx}^{j+1}(t)\cdot\beta\ \&\ avg_{rx}^{i}(t) > threshold_{J+1}\cdot\lambda) \\ &\quad leave\ j \\ &\quad join\ j+1 \\ &else\ if\ (avg_{rx}^{i}(t) < threshold_{j}\cdot\kappa) \\ &\ leave\ j \\ &\ join\ j\text{-}1 \end{aligned} \tag{5}$$

The average receiving rate $avg_{rx}^{i}(t)$ at time t is defined as follows:

$$\begin{aligned} avg_{rx}^{i}(t) &= \Phi(avg_{rx}^{i}(t_0), \alpha) \\ avg_{rx}^{i}(t) &= (1-a)\cdot avg_{rx}^{i}(t_0) + a\cdot avg_{rx}^{i}(t_1) \end{aligned} \tag{6}$$

In our implementation we set $a = 0.3$ based on experimental results. The different values of a define the level of receiver's responsiveness to network changes. We do not let instantaneous high or low $avg_{rx}^{i}(t)$ values play a central role in the decision making algorithm. However, as we can see from algorithm (5) the decision for leaving a lower capacity stream and joining a higher capacity stream is not only based on measuring the average receiving rate. The receiver is not only required to have an average receiving rate that is higher of the lower limit of the higher stream. It has to be able to follow "similar" receiving rates with those of the set of receivers in this higher stream. Thus, β takes the following values:

$$0.7 \le \beta \le 1 \tag{7}$$

The higher stream join factor λ is bounded between 1 and 2

$$1 \le \lambda \le 2 \tag{8}$$

meaning that the average receiving rate $avg_{rx}^{i}(t)$ should be at least equal or higher of the lower limit of the higher capacity stream.

Lastly, κ is bounded between $0.8 \le \kappa \le 1$. Receivers can remain in the current stream j even though its average receiving rates $avg_{rx}^{i}(t)$ are at least 80% of the low limit of the current stream. In our simulations we set $\beta = 0.7$, $\kappa = 0.8$ and $\lambda = 1.2$ based on various experimentations with different network topologies which are not presented in this paper due to space limitations.

3. Performance evaluation

We conduct a number of simulations under different scenarios to investigate:

- The accuracy of ASSP in subscribing the correct stream in respect of the receiver's bandwidth capacity.
- The TCP-friendly behavior of ASSP.
- The responsiveness of ASSP to network dynamics.

3.1. Stream Subscription Accuracy

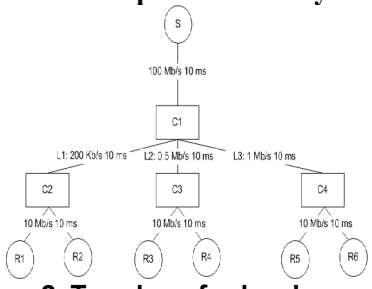

Figure 2. Topology for level accuracy subscription.

[1] The subscription level is the stream in which the receiver makes a join request.

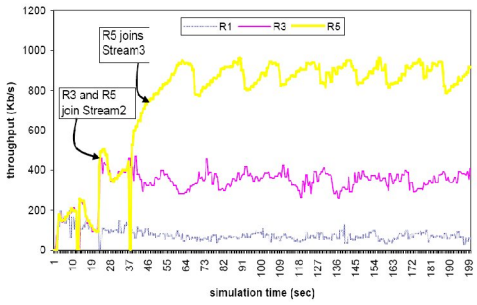

Figure 3. Join attempts of multicast receivers

In this simulation we investigate the accuracy of ASSP in terms of stream subscription level and also the convergence time of the protocol to reach a stable state. Our simulation scenario consists of one multicast sender (S) and six multicast receivers R1 to R6 (figure 2). C1 to C4 stand for the network routers. We set up Drop Tail queues in the routers C1 to C4 and set the one way delay in all paths to 30 ms. We run the simulation for 200 seconds and average the results of ten random simulation runs. The sender transmits three multicast streams within the following limits: `Stream1` (100 Kb/s-200 Kb/s), `Stream2` (200 Kb/s-500 Kb/s), and `Stream3` (600 Kb/s-1 Mb/s). Multicast receivers are connected with links that differ in capacity. Therefore, receivers have different receiving capabilities. The sender initially sets the lower limit of each stream as the initial transmission rate. At start time all receivers join `Stream1`, which is the stream with the lowest transmission rate. For easier observation we present in the simulation graphs the results of only one receiver of each multicast pair. We observe from the simulation results (figure 3) that at the 20th simulation second R3 and R5 join the next higher capacity stream, `Stream2`. Therefore, it takes receiver R3 four runs (20 seconds) to reach a stable state. By that time `R3` and `R5` estimate average receiving rates 1.2 times more than the lower limit of `Stream2` (200 Kb/s). Their average receiving rates are also within the limits of the average transmission rate of `Stream2` (0.8 times 200 Kb/s). It takes R5 four additional runs to join `Stream3` at the 40th simulation second. For the remaining of the simulation and in the lack of any changes to the network conditions the status of simulcast receivers remain unchanged.

It is interesting to notice that receivers present stable behavior without "jumping" from lower to higher streams and vice versa throughout the simulation lifetime.

3.2. TCP-fairness

In this simulation we evaluate the fairness of ASSP towards TCP traffic. We use a bottleneck scenario (figure 4) in which a multicast sender shares multiple bottleneck links with TCP sources. S and R represent the multicast sender and receiver, whereas TCP and Sink stand for the TCP sender and receiver, respectively. C1 to C4 stand for the network routers. The multicast sender transmits three different streams within the following limits: `Stream1`: 100Kb/s-300Kb/s, `Stream2`: 300Kb/s-500Kb/s, and `Stream3`: 600Kb/s-1.5Mb/s. We use in our simulations Drop Tail queue in the routers and set the access link capacity of all agents to 10 Mb/s with an access delay of 10 ms.

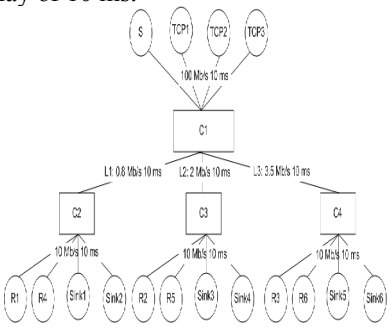

Figure 4. TCP-fairness network topology.

According to proportional fairness the bottleneck links should be equally shared by multicast flow and TCP traffic which means that each multicast receiver must not consume more than 33.3% of the bottleneck bandwidth (each bottleneck link, L1, L2 and L3 is shared by two TCP connections and one multicast flow). As a result we expect each flow to receive 100%/3 = 33.3% of the bottleneck bandwidth. Therefore, multicast receivers in the low capacity link (L1=0.8 Mb/s) must not consume more than 266.66 Kb/s, in the middle capacity link (L2=2 Mb/s) must not consume more than 682 Kb/s, and in the higher capacity link (L3=3.5 Mb/s) not more than 1.66 Mb/s. The rest of the available bandwidth should be consumed by TCP connections and it is expected that each TCP connection will receive at least the same bandwidth share with the multicast flow. Figure 5 presents the throughput of R1, R2 and R3. Figures 6, 7 and 8 depict the achieved throughput of multicast and TCP receivers. R1 consumes on average 70%, while TCP is close to 101% of its share. In the higher capacity link (L2), R2 has an average reception rate of approximately 486 Kb/s, which is 71% of its fair bandwidth share. TCP in L2 enjoys again higher bandwidth share than ASSP and consumes 110% of its bandwidth share (figure 7).

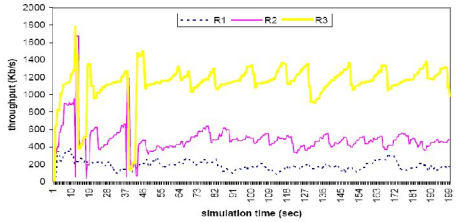

Figure 5. Receiving rates of multicast receivers.

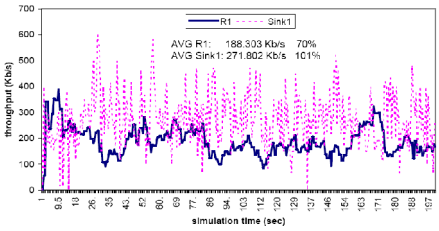

Figure 6. ASSP vs TCP traffic in L1.

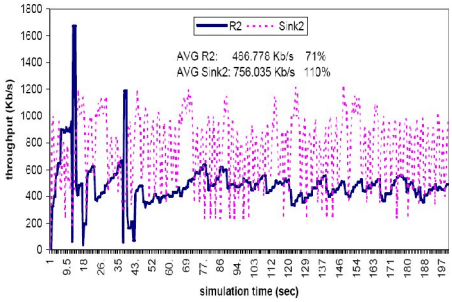

Figure 7. ASSP vs TCP traffic in L2.

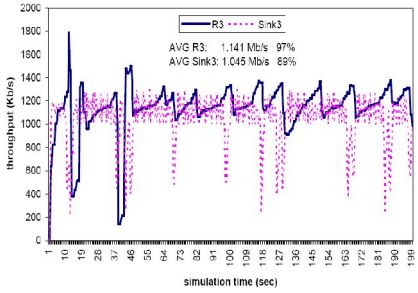

Figure 8. ASSP vs TCP traffic in L3.

However, ASSP, in the higher capacity link L3, enjoys higher throughput than TCP. R3 consumes 97% of its bandwidth share (figure 8). The explanation is that ASSP is very conservative due to its smoothing functions. In this simulation scenario packet losses mainly occurred in low capacity links L1 and L2. TCP can better react to packet losses and reaches its upper transmission limit faster than ASSP. The smooth operations in ASSP make it slower to dynamics of network changes. With the above simulation results we verify that ASSP is indeed a TCP-friendly protocol. ASSP also presents smooth and steady behavior in the light of competing TCP traffic.

3.3. Responsiveness to dynamics of competing traffic

In the last simulation scenario we investigate the behavior of ASSP in the light of other competing traffic that causes congestion at certain periods during the simulation lifetime. We use a network topology in which the number of connected TCP sources and receivers change over time. In this scenario (figure 9) each bottleneck link (L1, L2 and L3) is shared by two ASSP and two TCP receivers. R1 to R6 stand for the ASSP receivers whereas Sink1 to Sink8 stand for the TCP receivers.

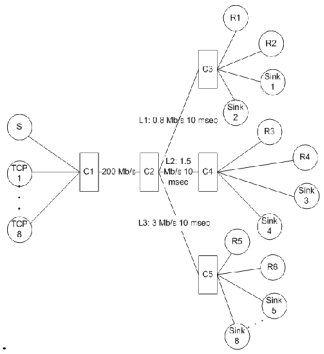

Figure 9. Heterogeneous dynamic network.

At the 100th simulation second three additional TCP sources start their transmission and we connect these sources with three TCP receivers in link L3. At the 125th simulation second the three additional TCP sources stop their transmission. This procedure is replicated between the 300th and the 325th simulation seconds. The multicast sender transmits again three different streams within the following limits: `Stream1`: 100Kb/s-300Kb/s, `Stream2`: 300Kb/s-500Kb/s, and `Stream3`: 500Kb/s-1.5Mb/s. We run the simulation for 500 seconds. We observe in figure 10 that ASSP receivers in links L1 and L2 present smooth and stable behavior as they do not observe any changes in the links L1 and L2.

Figure 11 presents the throughput of ASSP receiver R5, versus the TCP receiver Sink5. We observe that when the additional TCP traffic flows through link L3, Sink5 almost immediately backs-off. R5 reacts to congestion by changing its subscription level from `Stream3` to `Stream2`. R5 reacts slower to congestion than Sink5 and this is something that we expected. At the 126th simulation second Sink5 reaches its previous rates while ASSP rejoins Stream3 at the 145th simulation second. This time is roughly twenty seconds after the additional TCP traffic stopped flowing via link L3. Therefore, by comparing these results with the first simulation scenario we verify that

the delay of ASSP for selecting the suitable data rate is four runs (20 seconds). We regard this delay as an acceptable and rather good response time when taking into account that ASSP regains its previous receiving rates in a smooth way without creating any oscillations. We observe the same behavior between the 300th and the 335th simulation seconds.

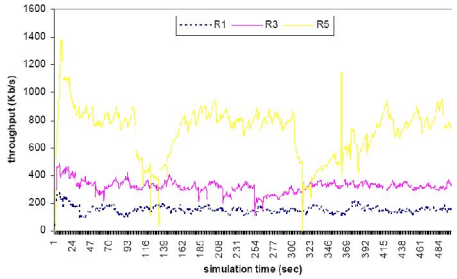

Figure 10. Receiving rates of multicast receivers.

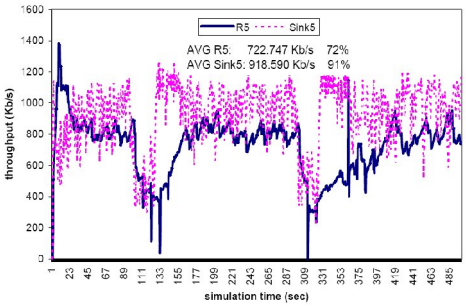

Figure 11. ASSP vs. TCP traffic in L3

4. Conclusions-future work

We presented in this paper a new multi-rate multicast protocol for simulcast transmission. Our main objective was to provide a solution for multimedia data transmission that was TCP-friendly with adaptive per-stream functionality, in order to serve a large number of receivers with different capabilities. With a "strict" decision making algorithm that was based on network statistical measurements we managed to prevent oscillatory behavior that would lead to congestion due to frequent join and leave attempts. Simulation results verified our design objectives.

ASSP presented stable and smooth behavior in the light of bandwidth degradation, due to other competing traffic. ASSP demonstrated its TCP-friendly behavior by allowing TCP traffic to achieve almost 100% of its bandwidth share when it flowed through the same bottleneck links with ASSP traffic. A penalty of the smooth and steady behavior of ASSP was lower bandwidth utilization when compared to TCP throughput. However, there has been always a trade-off between smooth transmission rates and high bandwidth utilization. Our future work includes the comparison and the performance evaluation of ASSP against other multi-rate solutions in a simulation environment with real video transmission, by taking into accounts not only network metrics but also multimedia related metrics. The dynamic creation of additional streams based on receiver feedbacks would also increase the Quality of Service (QoS) that is offered to the end users in a multicast environment. Finally, sources, documentation, simulation scripts and simulation results are available in [14].

5. References

[1] ITU-T Recommendation H.264, "Advanced video coding for generic audiovisual services" July 2007.

[2] A. Legout, E. Biersack, "PLM: fast convergence for cumulative layered multicast transmission schemes", in Proceedings of ACM SIGMETRICS, 2000.

[3] J.W. Byers et al., "FLID-DL congestion control for layered multicast", in Proceedings of NGC, 2000.

[4] J. Byers, M. Luby, M. Mitzenmacher, " Fine-grained layered multicast" in Proceedings of INFOCOM 2001, Volume 2, Issue, 2001 Page(s):1143 – 1151.

[5] J. Byers, G. Kwon, "STAIR: Practical aimed multirate multicast congestion control", in: Proceedings of NGC, 2001.

[6] G.-I. Kwon, J. Byers, "Smooth multirate multicast congestion control," in: Proceedings of IEEE INFOCOM, 2003.

[7] RFC 4654, "On TCP-friendly Multicast Congestion Control (TFMCC)", J. Widmer, M. Handley.

[8] C. Bouras, A. Gkamas, "SRAMT-S: A hybrid sender and receiver-based adaptation scheme for TCP friendly multicast transmission using simulcast approach", 1st IFIP Workshop on Internet Technologies, Applications and Social Impact (WITASI-02), Wroclaw, Poland, C. Bouras, A. Gkamas, 10 - 11 October 2002, pp. 105 – 122.

[9] Jiangchuan Liu, Bo Li, "Optimal stream replication for video simulating", 10th IEEE International Conference on Network Protocols, 2002.

[10] C. Bouras, A. Gkamas, G. Kioumourtzis, "Adaptive Smooth Multicast Protocol for Multimedia Data Transmission", 2008 International Symposium on Performance Evaluation of Computer and Telecommunication Systems – SPECTS 2008, Edinburgh, UK, 16 - 18 June 2008.

[11] C. Bouras, A. Gkamas, G. Kioumourtzis, " Comparison of Single-Rate Multicast Congestion Control Protocols vs. ASMP", 16th Annual Meeting of the IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS)– MASCOTS 2008, Baltimore, MD, USA, 8-10 Sep. 2008.

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

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

[14] http://nsnam.isi.edu/nsnam/index.php/Contributed_Code#Transport