Simulcast Transmission for Video Applications: Performance Evaluation with an Integrated Simulation Environment

Christos Bouras, Georgios Kioumourtzis
Computer Engineering and Informatics Department
University of Patras
Rion, Greece
bouras@cti.gr , gkioumou@ceid.upatras.gr

Apostolos Gkamas
Research Academic Computer Technology Institute
Rion, Greece
gkamas@cti.gr

Abstract— **This paper presents the performance evaluation of a multi-rate multicast protocol named Adaptive Smooth Simulcast Protocol (ASSP) for simulcast video transmission. ASSP implements a single rate TCP-friendly protocol as the underlying congestion control mechanism for each simulcast stream. ASSP is built 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 under an integrated simulation environment, which extends ns-2 and Evalvid-RA into the multicast domain with the use of RTP/RTCP protocols. Simulations conducted under this environment combine the measurements of network metrics along with objective evaluation criteria on the perceived video quality by the end user.**

Keywords-component; Multicast; congestion control; multimedia transmission; ns2; simulation

I. INTRODUCTION

Simulcast is an efficient way for multimedia data transmission mainly because users in today's internet follow a clustered distribution either because they use standard access interfaces or they may be behind the same bottleneck links. Therefore, we can deliver multimedia data to a fairly large number of user groups with a finite and small number of different streams that differ in multimedia quality, and hence they have different bandwidth requirements. The main advantage of simulcast is that it does not require sophisticated scalable encoders and does not incur the extra overhead of scalable encoding [1]. The disadvantage is that the multiple streams take up more hard disk space on servers and are streamed over the network in multiple copies. Therefore, it is important to minimize the number of the transmitted streams to free up network resources. This can be feasible only if the transmitted streams are adaptive, so that they can serve a fairly large number of users with similar receiving capabilities. Another major issue related to the transmission of multimedia data is the "TCP-friendly" behavior of the underlying transport protocols so that TCP-based applications will not starve.

The research community has provided a sufficient number of new and promising proposals for congestion or flow control mechanisms. Receiver-driven Layered Multicast (RLM) [2], PLM [3] and RLC [4] are earlier works in which receivers are able to join different multicast groups, in accordance with the observed congestion in the network. FLID-DL [5] tries to mitigate known drawbacks related to long IGMP leave latencies, which leave the network in a congested state. Fine-grained layered multicast [6] addresses the drawbacks of cumulative layering regarding the coarse-grained adaptation of receivers. STAIR (Simulate TCP's Additive Increase/multiplicative decrease with Rate-based) [7], further minimizes the IGMP join and leave requests with the concept of "stair layers". SMCC (Smooth Multi-rate Multicast Congestion Control) [8] employs cumulative layered multicasting and is based on TFMCC [9] for the adaptation of the individual layers. SRAMT-S [10] is another proposal in the area of simulcast transmission in which the transmission rate of each multicast stream is based on joint sender and receiver estimations. A study on the mismatch between a receiver's capacity and the stream bandwidth in simulcast transmission is cited in [11].

We present in this work Adaptive Smooth Simulcast Protocol (ASSP), a new multi-rate transport protocol for simulcast transmission over best-effort networks. The key attributes of our proposal 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. The building block of ASSP is based on our previous work that comprises a single-rate multicast congestion control protocol named ASMP ([12], [13]). 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 for handling all the issues related to simulcast transmission as well as the management and synchronization of the multiple multicast streams.

ASSP exploits the concept of "smooth transmission" to avoid high oscillations of the transmission rates of each

individual stream, and minimizes the join and leave attempts to the available multicast streams. This is an important attribute as frequent join and leave requests will introduce network congestion due to long IGMP leave latencies. Furthermore, frequent join and leave requests lead to high oscillations that cause instability. Another important attribute of ASSP is TCP-friendliness as each individual receiver calculates a TCP-friendly bandwidth share with the use of the TCP analytical model presented in [14]. Lastly, ASSP is a pure end-to-end solution that does not require any network support except for IP-multicast. This is an important attribute as it allows easy deployment over different administrative domains.

We focus in this work on a detailed evaluation which is based both on multimedia metrics and network metrics. This "joint" performance evaluation process provides the set of metrics to better understand the benefits and limitations of our proposal for multimedia data transmission.

The rest of this paper is organized as follows: The next section describes the functionality of ASSP. The simulation environment for our experiments is discussed in Section 3. In Section 4 we present performance evaluation results. We conclude our paper in section 5.

TABLE I. 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
$threshold_j$	low BW limit of stream j

II. ASSP DESCRIPTION

In ASSP the sender transmits a number of different multicast streams that carry the same multimedia data which differ in quality. Streams with lower transmission rates offer lower multimedia quality. The innovation in ASSP is that each stream is not transmitted at a fixed rate. Streams are adaptive so that they can accommodate a fair large number of receivers with similar bandwidth capabilities. In this way we can minimize the number of required streams in a simulcast transmission in order to save network resources. The streams however, need to be bounded between a predefined lower and upper limit. 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. ASSP is a pure end-to-end solution that is implemented up to the application layer. The sender is not required to perform complex calculations other than transmission rate adaptations and simple averaging functions per multicast stream. 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 which is expressed in (5), (c) the sender gathers the RTCP [15] receiver reports and performs per-stream transmission rate adaptations based on the reported values (equation 2), (d) the sender sums the average transmission rate of each stream 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. In the following paragraphs we present all the issues related to simulcast transmission as well as the management and synchronization of the multiple multicast streams. The transmission of each single multicast stream is controlled by the ASMP protocol ([12], [13]).

A. 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:
$compare(rx_i^{inst}(t))$ (1)
$adjustTransmissionRate()$

Subroutine Compare():
$tx_j^{inst}(t) = \min(rx_1^{inst}(t)....rx_i^{inst}(t))$ (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 of stream j is the lowest reported $rx_i^{inst}(t)$ value. Next, 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 stream j with the use of Equation (3):

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

The sender adds the average transmission rate of each stream in the application part of the sender RTCP report. Therefore, each receiver at any given time has knowledge of the average transmission rates of each multicast stream. In this work, we set the timeout interval for averaging the transmission rates to 5 seconds 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.

B. Receier'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 is 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 $0 < \alpha < 1$ to minimize the leave/join side effects. The following operations take place when a sender RTCP packet is received at receiver i (we assume that the sender informs each receiver about the thresholds of each transmitted stream during the connection to the service. This feature can be easily implemented with the use of application defined (APP) RTCP packets):

Receive RTCP packet:

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

Subroutine Compare():

$$\begin{aligned} &if\ (avg^i_{rx}(t) > avg^{j+1}_{tx}(t) \cdot \beta\ \&\ avg^i_{rx}(t) > threshold_{j+1} \cdot \lambda) \\ &\quad leave\ j \\ &\quad join\ j+1 \\ &else\ if\ (avg^i_{rx}(t) < threshold_j \cdot \kappa) \\ &\ leave\ j \\ &\ join\ j\text{-}1 \end{aligned} \qquad (5)$$

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

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

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. High values make the receiver respond faster but they create more leave and join requests. Our objective is however, to stabilize receiver's behavior and avoid frequent join and leave requests. Therefore, we do not let instantaneous high or low $avg^i_{rx}(t)$ 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 than 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:

$$0.7 \le \beta \le 1 \qquad (7)$$

which means that the receiver should be able to achieve receiving rates al least 70% of the average transmission rate $tx^{inst}_{j+1}(t)$ in order to join the higher capacity stream $j+1$. The higher stream join factor λ is bounded between 1 and 2

$$1 \le \lambda \le 2 \qquad (8)$$

meaning that the average receiving rate $avg^i_{rx}(t)$ should be at least equal or higher than 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 their average receiving rates $avg^i_{rx}(t)$ are at least 80% of the lower 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. The meaning of the above filtering functions is that receivers build a certain level of confidence when they decide to join a higher capacity stream. On the other hand, receivers can still participate in the current session of stream j even though their receiving capacities are within an acceptable value that is lower than the low limit of the current stream. Therefore, we prevent oscillatory behavior and ensure stability. However, the penalty we pay is lower bandwidth utilization as receivers do not rapidly join or leave a higher or lower capacity stream based on current network conditions. We argue however that in many cases stable reception rates close to optimal values are more important than higher throughput as a result of stream changes to higher transmission rates.

III. Simulation Environment

Most of the related work has been evaluated through simulations conducted with the ns-2 [16] simulator software. Those simulations were not based on any multimedia traffic generation model and in the best case trace files were used instead. Therefore, the only quality indicators were purely based on "typical" network metrics. However, different multimedia encodings can result in different perceived video quality, although the transmission is done with exactly the same set of protocols and under the same network conditions. Therefore, it is important to study the performance of any proposed solution by using real video files and to associate simulation results with video QoS metrics. For the purpose of this work we have extended a previous work named Evalvid-RA [17] in order to integrate the required sources into ns-2 and

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

thus enabling us to conduct a number of realistic experiments with real video files. This simulation environment consists of three parts and is depicted in Figure 1. During the pre-processing, a raw video file which is usually stored in YUV format is encoded with the desired video encoder into 30 different encoded MPEG-4 video clips with quantizer scale values in the range 2 to 31. Quantizer scale 2 provides an encoded video with the highest quality. We use the `ffmpeg`[2] free video encoder for the creation of the video clips. For our simulations all video clips have temporal resolution of 25 frames per second and GoP (Group of Pictures) pattern IBPBPBPBPBPB, with a size of 12 frames. The frame size of all clips is 352 x 288 pixels, which is known as the Common Intermediate Format (CIF). The encoded video files are then traced to produce 30 frame size trace files. At the end of the pre-processing phase we have 30 `m4v` files with their associated frame size files.

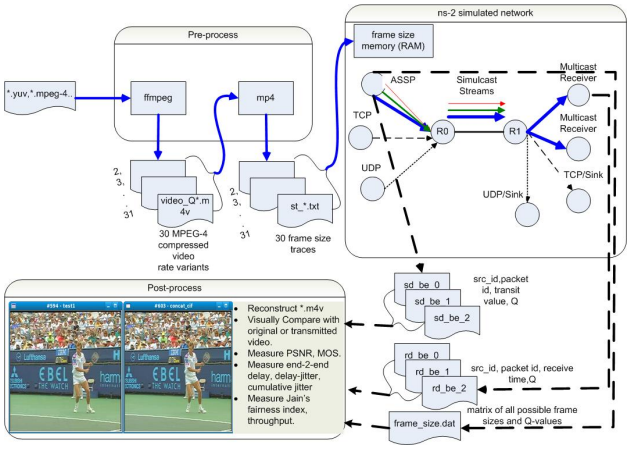

Figure 1. Overview of the simulation environment (pre-processing, ns-2, post-processing)

Next the ns-2 creates the simulated network. The video file is transmitted from the server by using a number of predefined simulcast streams to groups of receivers. There are no limitations on the number of the transmitted simulcast streams but a high number of streams consume more network resources. In our simulations we use high, medium and low quality streams. During the simulation time we store the traces of all transmitted simulcast streams from the sender and also the traces from the multicast receivers. The third part of the simulation environment consists of the reconstruction of the transmitted video and the measurement of the performance evaluation metrics. The reconstruction of the received video traces is implemented off-line by comparing the transmitted and the received traces with traces from the original video sequence of all the transmitted simulcast streams. The following metrics are stored and calculated:

- PSNR/MOS values.
- End-to-end delay.
- Packet delay variations.
- Inter-frame cumulative jitter.
- Delay jitter.
- Frame loss rate.
- Throughput per flow.
- Jain's fairness index.

IV. Performance Evaluation

A. *Preliminary Test (level subscription accuracy)*

In this simulation we investigate the accuracy of ASSP in terms of stream level subscription and also its stability. The simulation scenario consists of one multicast sender (S) and six multicast receivers, ASSP1 to ASSP6 (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 50 ms. We choose a high-motion video sequence (highway) which consists of 2000 frames. This high-motion sequence is challenging as it provides low PSNR values. We run the simulation for 80 seconds and average the results of ten random simulation runs. The sender transmits three multicast streams with different quality. To do so, at start time the server starts the transmission of the different streams with different quantizer scales. During the simulation the sender adjusts the transmission rate of each individual stream based on feedback reports from the receivers. The interested reader can refer to [17] for more details on the dependencies between the transmission rates and the video quality. As for the multicast receivers, they are connected with links that differ in capacity. Therefore, we create three different groups of receivers and each group has different receiving capabilities. At start time all receivers join the stream with the lowest quality. For easier observation we present in the simulation graphs the results of only one receiver of each multicast pair. We observe that at the first join attempt, ASSP3 and ASSP5 join the medium capacity stream and ASSP5 then quickly joins the high capacity stream, (figure 3). It is interesting to notice that in the lack of any changes to the network conditions, the receivers present stable behavior without "jumping" from lower to higher streams and vice versa, throughout the simulation lifetime. The average throughput of each group is 83 Kb/s, 221 Kb/s and 360 Kb/s, with a frame loss ratio of 0.016, 0.001 and 0.002 for the low, medium and high quality stream, respectively. We notice this rather low link utilization which is inherited by the *smoothing* property of the underlying congestion control mechanism. Our design target is to minimize oscillations as they tend to increase packet losses and lead to an unsatisfactory video experience at the user side. However, there is always a trade-off between maximizing throughput and stability.

We measure the stability of ASSP by using the coefficients of variation (CoV)[3] of the throughput values and plot the results in figure 4. We observe that ASSP shows good stability. Delay jitter measurements (figure 5) present low values which

[2] Static quantizer scale values ranging from 2-31 are supported by `ffmpeg`

[3] Coefficient of Variation (CoV) is the standard deviation divided by the mean.

are suitable also for conversational applications as they are much lower than 150 ms. Cumulative jitter (figure 6) presents high values in the case of the low quality stream, which is an indication of a congested link between the sender and the low capacity receivers. These values require a higher playback buffer in the receivers to smooth the negative effects of late-arrived frames.

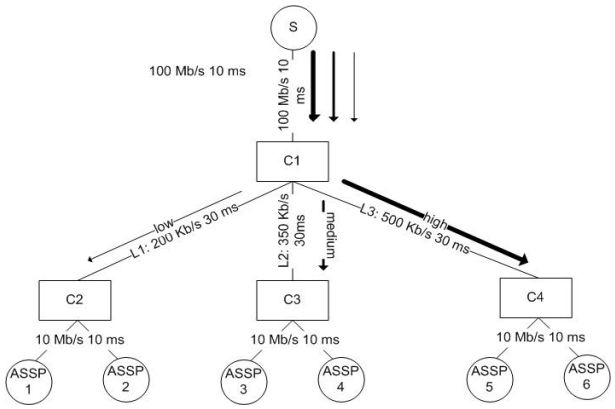

Figure 2. Topology for stream accuracy subscription

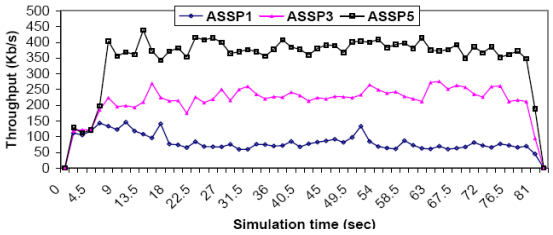

Figure 3. Achieved throughput of the three multicast receivers (ASSP1: low connection (83 Kb/s), ASSP3: medium (221 Kb/s), ASSP5: high (360 Kb/s)

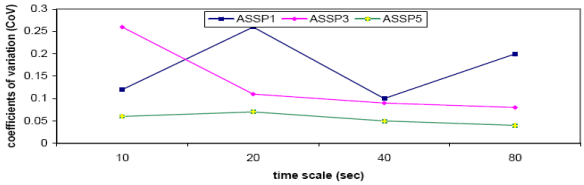

Figure 4. CoV of the three ASSP flows

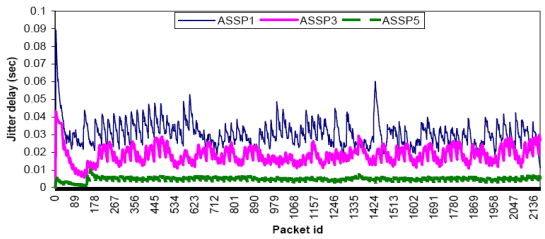

Figure 5. Delay jitter of the three ASSP flows

Next the average PSNR values of the received video are depicted in figure 7. We observe that on average all of the received video files have average PSNR values above 25 dB although in this simulation we use low capacity links that do not allow the transmission of high quality video files. The average MOS grading for the three flows is 3. The best quality video with higher PSNR values could have been obtained using links with capacities of at least 3 Mb/s. However, in this simulation we intentionally test ASSP in low capacity links to investigate its limitations. Our assessment from this experiment is that ASSP presents a stable behavior, without creating oscillations. The low frame loss ratio is an indication of the correctness of the underlying congestion control mechanism.

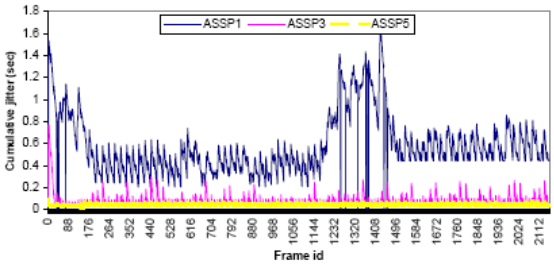

Figure 6. Cumulative jitter of the three ASSP flows

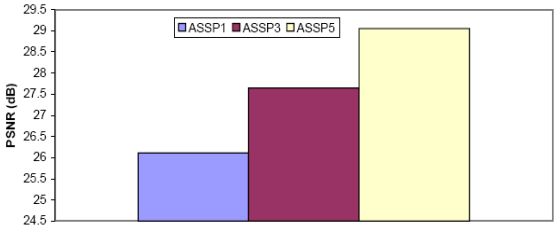

Figure 7. PSNR values of the received MPEG-4 video (Average PSNR: ASSP1=26.11 dB, ASSP3=27.64 dB, ASSP5=29.05 dB)

B. Fairnes

We investigate the fairness of ASSP in two different occasions: a) TCP-friendliness, and b) fairness between flows of the same protocol. We regard that a flow is TCP-friendly when it does not consume more bandwidth than a TCP connection, which is traversing the same path with this flow. Fairness measurements can include max-min fairness, proportional fairness, Jain's fairness index, and the product measure, a variant of network power. In this work we use the Jain's fairness index that is defined in [18].

1) TCP-fairnes: We simulate the transmission of a video file to a number of multicast groups in an environment with multiple bottleneck links (figure 8). In this topology the bottleneck links are shared between ASSP and TCP flows.

We set up three multicast groups with forty five ASSP receivers per group that are behind bottleneck links which differ in capacity and delay. We use Drop Tail queues in the routers and set the same packet size for TCP and ASSP to obtain a fair comparison. At start time the sender starts the transmission of three multicast streams with different quality and all multicast ASSP receivers join the low quality stream. The subscription in a higher or lower multicast stream is based on the functionality of ASSP. Figure 9 depicts the throughput of the background TCP flows and a representative receiver of each multicast group (high, medium ad low quality streams). The measured Jain's fairness index is 0.962 indicating that all flows almost equally share the bandwidth of the bottleneck

links. The frame loss ratios of the low, medium and high quality streams were measured to be 0.013, 0.016 and 0.009, respectively. With a first visual observation we can see also the steady rates not only of the ASSP but also of the TCP flows. By plotting the CoV of all flows we observe that after the 20th simulation second all flows are in equilibrium with steady rates that do not change for the rest of the simulation lifetime (figure 10).

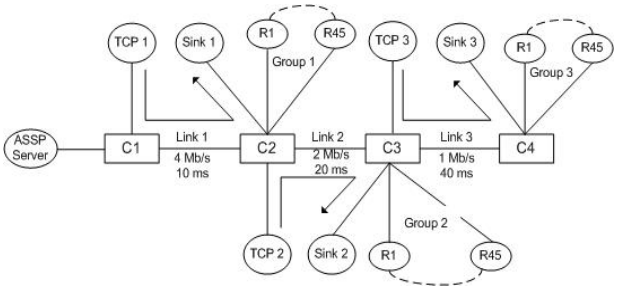

Figure 8. Network topology for TCP-fairness simulation

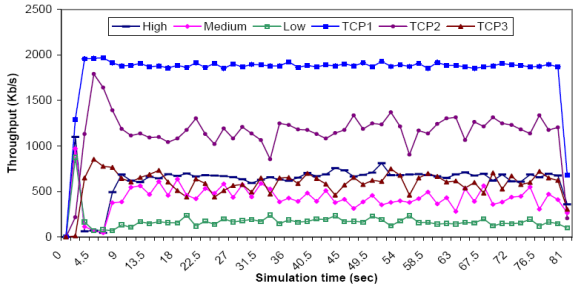

Figure 9. Achieved throughput of all flows (TCP and ASSP)

Cumulative jitter measurements (figure 11) indicate low values for the high and medium quality streams, and higher values (above 1 second) for the low quality stream. A playback buffer with duration of 1.5 seconds can accommodate the negative effects due to congestion. PSNR measurements (figure 12) indicate that all the received video files have on average a fair quality even for ASSP receivers that are below a highly congested low capacity link. At this point we need to point out that we have succeeded in transferring a video file to different multicast groups with a minimum number of streams, as the underlying congestion control protocol is rate adaptive.

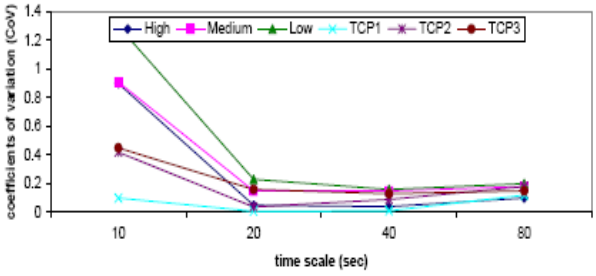

Figure 10. CoV of all flows (TCP and ASSP)

If we locate an ASSP agent at the edge of the network we would probably be able to serve a fairly large number of users that are located behind the same bottleneck links with a very small number of multicast streams. Our assessment from this simulation is that ASSP is indeed a TCP-friendly protocol as it fairly shares network resources with TCP. The smoothing functions of ASSP make it less aggressive in network conditions with competing TCP traffic. This is an important attribute of ASSP as most of today's user applications in the global web are TCP-based.

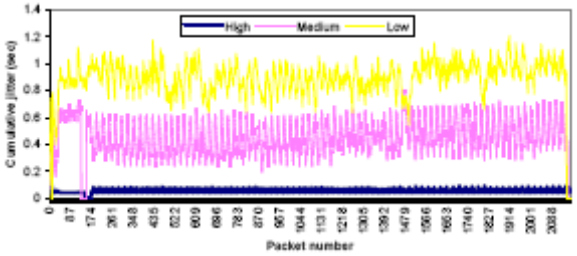

Figure 11. Cumulative jitter of the three ASSP flows

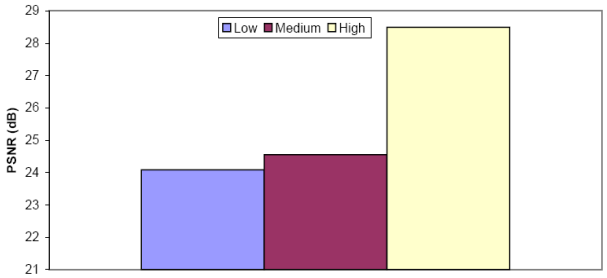

Figure 12. PSNR values of the received MPEG-4 video (Average PSNR: Low=24.09dB, Medium=24.56 dB, High=28.05 dB)

2) Intra-protocol fairness: In this simulation we use the same network topology in figure 8 and connect two ASSP sources to C1. There are not any other competing sources other than the two ASSP sources. In each bottleneck link we connect two groups of forty five receivers and each group receives video traffic from one of the two ASSP sources. Figures 13 and 14 depict the achieved throughput of a representative receiver from the high, medium and low capacity multicast groups. We observe that the results are identical for the case of the two competing ASSP sources. The Jain's fairness index in the three bottleneck links is 0.99, indicating that network resources are equally shared by the two ASSP sources. The frame loss ratio for the low, medium and high quality streams was measured to be 0.0045, 0.0005 and 0.0015, respectively. The multicast receivers lost only a few frames due to buffer overflow in the routers. The coefficients of variation (figure 15) are very low in all streams and are bounded between 0.2 and 0.005. This is a clear indication of the stability of ASSP. PSNR values (figure 16) are higher when compared with the previous simulation results against TCP traffic. This is also a result of a very low frame loss ratio. Simulation results indicate a high level of fairness when ASSP sources share network resources. Video quality was also measured to take the highest grading when

taking into account the low capacity of the three bottleneck links.

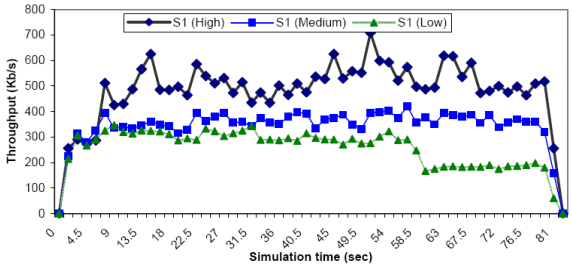

Figure 13. Achieved throughput of ASSP receivers that join the first ASSP source

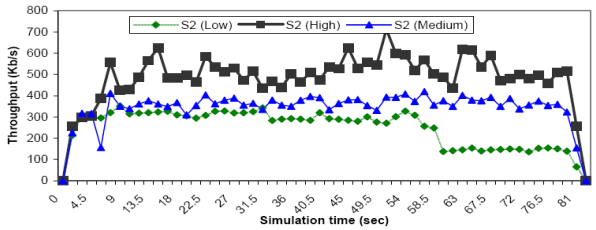

Figure 14. Achieved throughput of ASSP receivers that join the second ASSP source

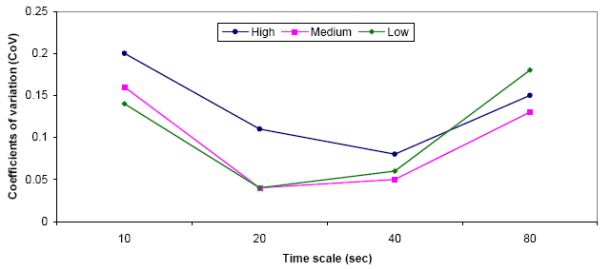

Figure 15. CoV of three representative receivers.

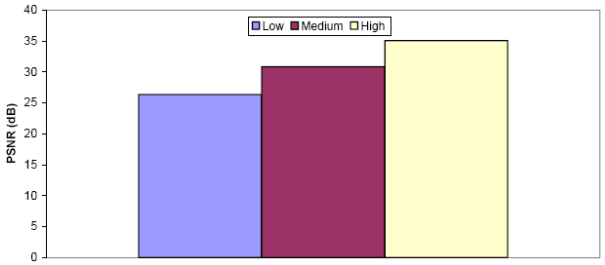

Figure 16. PSNR values of the received MPEG-4 video (Average PSNR: Low=26.36dB, Medium=30.83 dB, High=35.06 dB)

C. *Performance with ON-OFF UDP flows as background traffic entify the Headings*

In this simulation scenario, we replace the TCP sources in the network topology of figure 8, with UDP flows. Figure 18 presents results with background traffic provided by ON/OFF UDP flows. For a better presentation we plot the results of one ASSP receiver of each multicast group. The duration of ON and OFF time is set to 10 seconds in such way that at any given time there is at least one active UDP source sending at 500 Kb/s during the ON time. By plotting the CoV (figure 18) we observe that in the beginning of the simulation the ASSP receivers present a slight variable behavior. After the 20th simulation second, ASSP presents a stable behavior without rate variations for the rest of the simulation lifetime. This is an important attribute for video transmission applications.

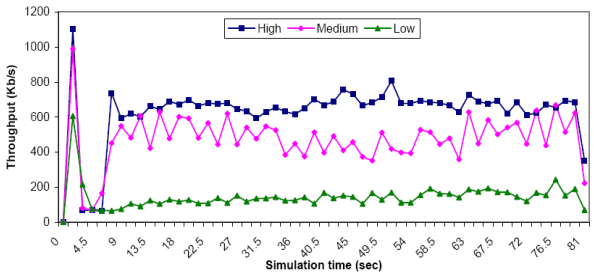

Figure 17. Achieved throughput of ASSP flows

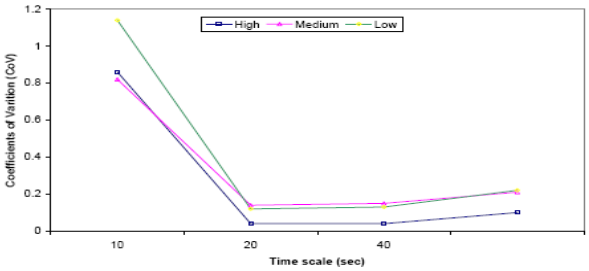

Figure 18. CoV of all flows (TCP and ASSP)

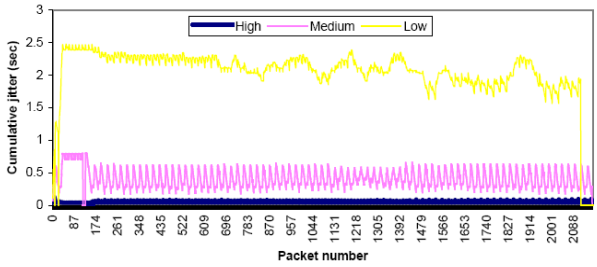

Figure 19. Cumulative jitter of the three ASSP flows

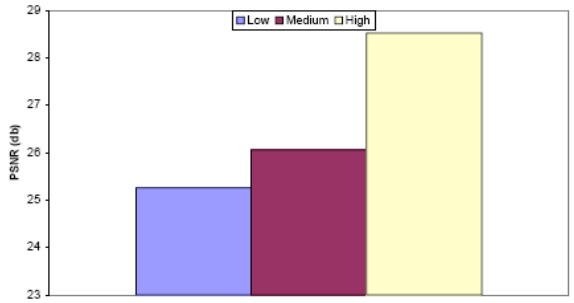

Figure 20. PSNR values of the received MPEG-4 video (Average PSNR: Low=25.26 dB, Medium=26.06 dB, High=28.52 dB)

The frame loss ratio of the three flows (high, medium and low) is 0.007%, 0.006%, and 0.02%, respectively, which is rather low for this simulation environment with competing

UDP flows. Cumulative jitter measurements (figure 19) indicate low values for the high and medium quality streams and higher values (above 2 seconds) for the low quality stream. This is an indication of mild network congestion due to the presence of uncontrolled UDP traffic. We have also observed that the frame loss ratio measurements were higher than those from the previous simulations. However, our general assessment is that ASSP retains its functionality when competing for network resources with uncontrolled UDP traffic.

V. Conclusions-Future Work

We presented in this work a solution for simulcast video transmission over heterogeneous networks. With the use of simulcast transmission we avoided the extra overhead of the more sophisticated scalable encoding.

We used an integrated simulation environment for our experiments and implemented the full reference methods for the evaluation of our proposal. The performance evaluation metrics were a combination of "classic" networks metrics with objective video quality metrics. This joint evaluation method provided a better understanding on the correlation and the dependencies between network metrics and the effects on the perceived video quality by the end user.

ASSP presented high stability with minimum oscillations due to delay and bandwidth variances in the network. The internal functions of ASSP minimized join and leave requests. The low frame loss ratio in all simulation scenarios indicated that the underlying congestion control algorithm in ASSP was able to early detect and handle upcoming congestion due to competing traffic.

Fairness is an important aspect of any transport protocol so that traffic is equally shared and distributed between applications in a network. ASSP proved to be TCP-friendly as TCP traffic enjoyed a high level of fairness when competing with ASSP traffic. This is an important attribute of ASSP as most popular user applications today are TCP-based. 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.

Uncontrolled transmission leads to network congestion and engages the retransmission of TCP packets due to packet losses. This is an undesirable situation as the retransmission of TCP packets wastes network resources. Intra-fairness of ASSP was very high, almost reaching the absolute fairness of one in accordance with the Jain's fairness index. A direct effect of packet losses for a video application is PSNR degradation. We observed the difference in the PSNR values in the same network topology when ASSP competed against UDP uncontrolled traffic. The higher the packet loss ratio in a video transmission was the lower the obtained PSNR values were and hence the lower the level of satisfaction at the end user was.

However, congestion control for video transmission is not the panacea to solve all current issues related to video transmission. Higher capacity links with Forward Error Correction (FEC) when combined with congestion control mechanisms will increase network performance and also the level of satisfaction at the end users. In our future work we will test ASSP against known proposals in the area of multi-rate congestion control schemes for multimedia data transmission. It would also be interesting to evaluate the performance of ASSP in wireless scenarios in which the cause of packet losses is not always a congested link. Finally, sources, simulation scripts and results are available in [19].

References

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

[2] S. McCanne, V. Jacobson, M. Vetterli, "Receiver-driven layered multicast", in Proceedings of ACM SIGCOMM, 1996.

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

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

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

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

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

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

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

[10] 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, 10 - 11 October 2002, pp. 105 – 122.

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

[12] 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.

[13] 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.

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

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

[17] Lie A., Klaue J., "Evalvid-RA: Trace Driven Simulation of Rate Adaptive MPEG-4 VBR Video" ACM/Springer Multimedia Systems Journal autumn 2007.

[18] R. Jain, D. Chiu, and W. Hawe, "A Quantitative Measure of Fairness and Discrimination for Resource Allocation in Shared Systems," DEC Research Report TR-301, Tech. Rep., Sept 1984.

[19] http://ru6.cti.gr/ru6/ns_rtp_home.php