End-to-end Congestion Management for Real-Time Streaming Video over the Internet

Panagiotis Papadimitriou and Vassilis Tsaoussidis

Demokritos University of Thrace, Electrical & Computer Engineering Department
Xanthi, 67100 GREECE
E-mail: {ppapadim, vtsaousi}@ee.duth.gr

***Abstract* – In this paper, we propose a new transport protocol, namely *Scalable Streaming Video Protocol* (*SSVP*), which employs an AIMD-oriented congestion control mechanism. SSVP, in a complementary role, operates on top of UDP and is specifically designed to support unicast video streaming applications. The transmission rate is controlled by properly adjusting the inter-packet-gap, spacing outgoing packets evenly to produce a smoothed flow. SSVP attempts to optimize the performance of streaming video delivery with concern to friendliness with interfering traffic. Quantifying SSVP's performance, we identify that the protocol utilizes a higher fraction of the available bandwidth, and maintains a regular transmission rate with oscillations of a smaller magnitude in comparison with existing congestion control schemes.**

I. INTRODUCTION

Time-sensitive applications, such as streaming media, gain popularity and real-time data is expected to compose a considerable portion of the overall data traffic traversing the Internet. These applications generally prefer timeliness to reliability. Real-time video streaming, in particular, calls for strict requirements on end-to-end latency and delay variation. More precisely, end-to-end delays exceeding 250 ms affect the timely delivery of video-data causing data unavailability and unintelligible real-time interaction with frustrating consequences to the end-user. Furthermore, reliability parameters, such as packet loss and bit errors, usually compose an impairment factor, since they cause a perceptible degradation in video quality. Unlike bulk data transfers, video streaming seeks to achieve smooth playback quality rather than simply transmit at the highest attainable bandwidth.

Real-time applications commonly "rely" on the unreliable transport services offered by *User Datagram Protocol* (*UDP*). UDP is a fast, lightweight protocol without any transmission or retransmission control. The protocol does not have functionality to override application characteristics, such as its transmission rates. It simply transmits at application rate and pattern. However, the lack of congestion control poses a threat to the network: had such applications dominated the Internet, it would have faced risk of congestion collapse. In this context, Internetworking functionality evolves towards punishing free-transmitting protocols.

On the other hand, *Transmission Control Protocol* (*TCP*), based on the principles of congestion management [5], *Slow-Start* [10], and *Additive Increase Multiplicative Decrease* (*AIMD*) [2], provides a reliable data delivery service to Internet applications, and is in large part responsible for the remarkable stability of the Internet. More precisely, the process of probing for bandwidth and reacting to observed congestion causes oscillations to the achievable transmission rate. With TCP's increase-by-one and decrease-by-half control strategy, even an adaptive and scalable source coding scheme is not able to conceal the flow throughput variation. Furthermore, TCP occasionally introduces arbitrary delays, since it enforces reliability and in-order delivery. In response to standard TCP limitations, several TCP protocol extensions [1, 4] have emerged providing more efficient bandwidth utilization and sophisticated mechanisms for congestion control. *TCP-Friendly* protocols, proposed in [4, 12] achieve smooth window adjustments, while they manage to compete fairly with TCP flows. In order to achieve smoothness, they use gentle backward adjustments upon congestion. However, this modification has a negative impact on protocol responsiveness [11].

Overcoming the oscillatory nature of AIMD congestion control, as well as the risks arising by the extensive usage of UDP, we need sophisticated congestion control that interacts efficiently with other flows on the Internet. An overview of Internet's current congestion control paradigm reveals that routers play a relatively passive role: they merely indicate congestion through packet drops or *Explicit Congestion Notification* (*ECN*). It is the end-systems that perform the crucial role of responding appropriately to these congestion signals. Numerous video-streaming applications have implemented their own congestion control mechanisms, usually on a case-by-case basis on top of UDP. However, implementing application-level congestion control is difficult and not part of most applications' core needs. We believe that a new transport protocol is needed, which would combine unreliable datagram delivery with built-in congestion control. This protocol would act as an enabling technology: new and existing applications could use it to timely transmit data without destabilizing the Internet.

In this context, we have been working on a congestion control mechanism for adapting outgoing video streams to the characteristics of the end-to-end network path. We had the option to rely on the unreliable UDP datagrams or modify TCP

to provide unreliable semantics. However, the latter seems particularly inappropriate considering the TCP semantics and its reliance on cumulative acknowledgments. Consequently, we considered UDP as a better choice, due to its unreliable and out-of-order delivery. Along these lines, we developed a new transport protocol, namely *Scalable Streaming Video Protocol* (*SSVP*), operating on top of UDP. SSVP is well suited for unicast streaming video applications. The protocol incorporates end-to-end congestion control and does not rely on *Quality of Service* (*QoS*) functionality in routers, such as *Random Early Drop* (*RED*), *ECN* or other *Active Queue Management* (*AQM*) mechanisms. Our objective is to provide efficient and smooth rate control while maintaining fairness and friendliness with corporate flows. SSVP adopts the generic AIMD approach by adjusting the inter-packet-gap (*IPG*) additively or multiplicatively, depending on whether or not congestion signal is captured. Our mechanism composes a viable alternative to existing congestion control schemes, alleviating most of the impairments induced by limited bandwidth and transient errors.

The rest of the paper is organized as follows. The following section briefly reviews related work. In Section III we discuss the design and implementation details of the proposed congestion control scheme. In Section IV we evaluate our mechanisms through extensive simulations. Finally, Section V concludes the paper.

II. Related Work

The literature includes several studies and proposals towards efficient video streaming over the Internet. *Rate Adaptation Protocol* (*RAP*) [8] is a rate-based protocol which employs an AIMD-oriented algorithm for the transmission of real-time streams. The sending rate is continuously adjusted by RAP in a TCP-friendly fashion using feedback from the receiver. Authors in [3] study the impact of selected congestion control algorithms on the performance of streaming media delivery. They also propose a *Real-time Transport* (*RTP*) [9] compatible protocol, namely *SR-RTP*, which employs SQRT binomial congestion control. *Datagram Congestion Control Protocol* (*DCCP*) [6] is a transport protocol that provides a congestion-controlled flow of unreliable datagrams, and constitutes a generalized framework for delay-sensitive data transport. DCCP aims to add to a UDP-like foundation the minimum mechanisms necessary to support TCP-like or TFRC congestion control.

TCP-friendly protocols [4, 12] constitute an elegant solution for time-sensitive applications. *TCP-friendly Rate Control* (*TFRC*) [4] is a representative TCP-friendly protocol, where its transmission rate is adjusted in response to the level of congestion, as indicated by the loss rate. TFRC eventually achieves the smoothing of the transmission gaps; however, the protocol becomes less responsive to bandwidth availability [11]. *TCP Westwood* [7] is a TCP-friendly protocol that emerged as a sender-side-only modification of TCP Reno congestion control. TCP Westwood exploits end-to-end bandwidth estimation to properly set the values of slow-start threshold and congestion window after a congestion episode. TCP-Real [13] is a high-throughput transport protocol that incorporates a congestion avoidance mechanism in order to minimize transmission-rate gaps. TCP-Real employs a receiver-oriented and measurement based congestion control mechanism that significantly improves real-time performance over heterogeneous networks and asymmetric paths.

III. SSVP Design and Implementation

A. Sender and Receiver Interaction

SSVP, in a complementary role, operates on top of UDP and supports end-to-end congestion control relying on sender and receiver interaction. SSVP acknowledges each datagram received by transmitting a control packet (containing no data). In accordance with the relaxed packet loss requirements of streaming video and considering the delays induced by retransmitted packets, we did not integrate reliability into UDP datagrams. Hence, control packets do not trigger retransmissions. However, they are effectively used in order to determine bandwidth and *Round Trip Time* (*RTT*) estimates, and properly negotiate and adjust the rate of the transmitted video stream.

We have encapsulated additional header information to UDP datagrams (Fig. 1), including packet type, packet sequence number, frame type, frame number and timestamp. Packet type field denotes whether a segment with video-data or a control packet is transmitted. Frame type and number can be exploited in order to augment a prioritized transmission (where I frames will be prioritized). Timestamp field is used to handle RTT computation. More precisely, when the sender transmits a video-packet, it updates the specific field with current time. Upon the receipt of the corresponding control packet, the sender subtracts the included timestamp from current time in order to estimate the RTT sample.

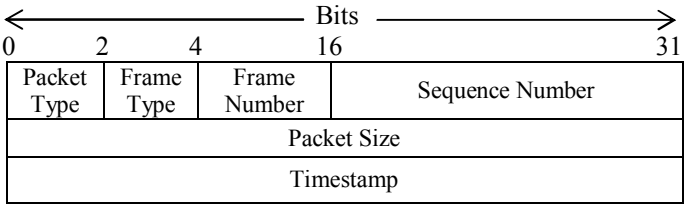

Figure 1. SSVP Header

Since UDP is an unreliable protocol, some datagrams may be lost due to congestion or inability of the receiving host from reading the packets rapidly enough. The receiver uses packet drops or re-ordering as congestion indicator. Consequently, congestion control is triggered, when a packet is received carrying a sequence number greater than the expected sequence number or the receiver does not acquire any packets within a timeout interval. Along these lines, the proper adjustment of the timeout interval is critical. A timeout interval that is set too short will claim false packet drops resulting in a wasteful reduction of the transmission rate. On the other hand, a long and consequently conservative timeout interval will inevitably impact the protocol responsiveness. The timeout interval for SSVP (*STO*) is properly

estimated based on current RTT measurements (*SampleRTT*), as follows:

$$STO = \gamma \times STO + (1 - \gamma) \times SampleRTT$$

where γ is the smoothing factor adjusted at 0.9.

B. Rate Adjustment

SSVP adjusts the sending rate in a TCP-friendly fashion, exploiting the feedback of reception statistics (control packets). Both *binomial* [1] and AIMD congestion control are implied to achieve TCP-friendliness. Although binomial schemes, such as IIAD or SQRT [3], are quite attractive to multimedia applications for their smooth rate variations, they are not able to achieve TCP-friendliness independent of link capacity. Apart from link capacity, the selection of increase rate and decrease ratio composes another influencing parameter. Along these lines, in order to attain TCP-friendliness, SSVP incorporates AIMD congestion control. Let α, β the specific values of additive increase and multiplicative decrease rate, respectively. The choice of α and β has a direct impact on protocol responsiveness to conditions of increasing contention or bandwidth availability.

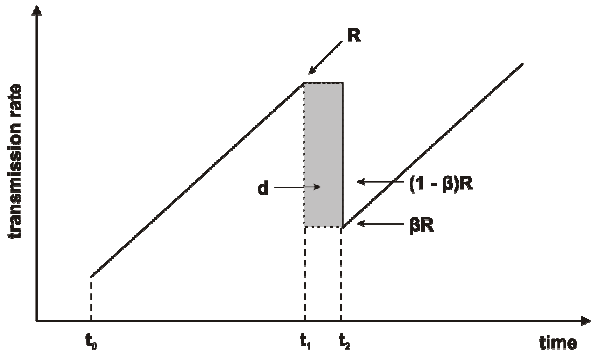

Figure 2. SSVP transmission rate evolution

Fig. 2 illustrates an SSVP flow under AIMD congestion control with instantaneous transmission rate *R*. Transmission initiates at time t_0 and R evolves, as follows:

$$R(t + \Delta t) = R(t) + \alpha \, \Delta t$$

Assuming a packet drop at time t_1, the transmission rate is reduced from R to βR and immediately the video coder is notified to reduce the video coding rate. This process (i.e. coding rate reduction) inevitably incurs a delay d and eventually transmission resumes at time $t_2 = t_1 + d$. If we assume that the server sends video data by dividing each frame into fixed packets of size *S*, we investigate the maximum end-to-end packet delay with respect to the stringent streaming video latency requirements. Let a packet *P* generated at time t_2. P will be enqueued after a number of packets that were generated during the time period: $t_1 \leq t_s < t_2$. Consequently, such a packet will suffer the longest delay. During t_s, the queue inside the sender is increased by d (1 - β) R / S packets, approximately. Hence, the sending delay D that denotes the amount of time that packet P will rest inside the server is derived by:

$$D \approx \frac{\frac{d\,(1-\beta)\,R}{S}}{\beta R} = \frac{d}{S}\left(\frac{1}{\beta} - 1\right) \quad (1)$$

The sending delay D depends on the decrease parameter β. With respect to equation (1), we may alleviate the impact of delayed packets on video quality by choosing an appropriate β. Apparently, a large value of β reduces packet sending delays smoothing transmission gaps. Furthermore, the video server is enforced to gracefully degrade video quality in response to congestion signals. Based on equation (1), if we calculate the sending delays D_1, D_2 which correspond to $\beta_1 = 0.5$ and $\beta_2 = 0.875$ respectively, we obtain an 85% gain for β_2, which significantly improves video delivery in the situation of frequent packet drops. However, a large β enforces the selection of a small α, according to the TCP-friendly condition [4].

Based on our analysis and with respect to user perception of video quality, we employ SSVP under AIMD congestion control with a final selection of $\alpha = 0.2$ and $\beta = 0.875$. Transmission rate is controlled by properly adjusting the inter-packet-gap (*IPG*). If no congestion is sensed, IPG is reduced additively; otherwise, it is increased multiplicatively. As a rate-based control, SSVP spaces outgoing packets evenly to produce a smoothed flow. The selected parameters result in oscillations of a smaller magnitude than standard TCP (1, 0.5), while per-RTT rate adjustments enforce a relatively responsive behavior.

IV. Performance Evaluation

A. Experimental Environment

The evaluation plan was implemented on the *NS-2* network simulator. Simulations were initially conducted on the typical single-bottleneck *dumbbell* topology with a bottleneck capacity of 1 Mbps and a round-trip link delay of 30 ms. We also enabled simulations on a complex network topology (Fig. 3), which addresses the heterogeneity of the Internet. The specific topology includes multiple bottlenecks, cross traffic, wireless links and diverse RTTs. The propagation delays of the access links from all the source nodes, as well as the links to the peripheral sink nodes range from 5 ms to 15 ms, while the corresponding bandwidth capacities range from 2 Mbps to 10 Mbps. Cross traffic includes diverse FTP flows over TCP Reno. NS-2 error models were inserted into the access links to the sink nodes with packet error rate (*PER*) adjusted at 0.01. In both topologies we used drop-tail routers with buffer size adjusted in accordance with the bandwidth-delay product. Furthermore, we set the packet size to 1000 bytes and the maximum congestion window to 64 KB for all TCP connections. The duration of each simulation is 60 seconds.

In order to simulate real-time traffic, we developed an *MPEG-4 Traffic Generator*. The traffic generated closely matches the statistical characteristics of an original MPEG-4 video trace. We used three separate *Transform Expand Sample*

(*TES*) models for I, P and B frames, respectively. The resulting video stream is generated by interleaving data obtained by the three models.

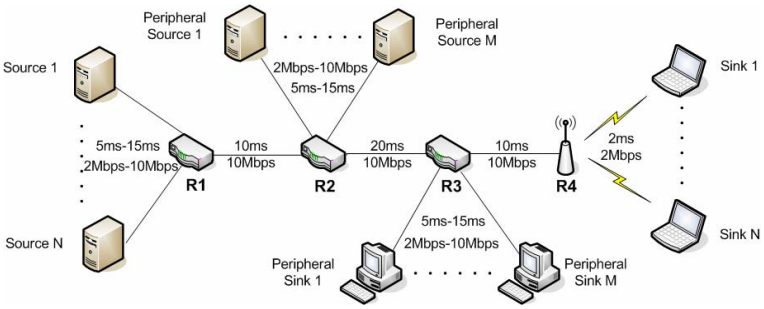

Figure 3. Simulation topology

We hereby refer to the performance metrics supported by our simulation model. Since both topologies include MPEG flows competing with corporate FTP flows, the performance metrics are applied separately to the MPEG and FTP traffic. Goodput was used to measure the overall system efficiency in bandwidth utilization. Inter-protocol fairness measurements were conducted based on normalized throughput, which is the ratio of the average throughput received by each flow over the bandwidth fair-share on each case. In order to quantify the performance on video delivery, we demonstrate packet jitter and delayed packets statistics. The proportion of delayed packets is reflected in *Delayed Packets Rate*, where we monitor packet inter-arrival times and eventually distinguish the packets that can be effectively used by the client application from delayed packets (according to a configurable packet inter-arrival threshold). In accordance with video streaming requirements, we adjusted the packet inter-arrival threshold at 100 ms.

B. Results and Discussion

In the sequel, we demonstrate and briefly analyze the most prominent results from the experiments we performed. Fig. 4 illustrates an excerpt from an MPEG transfer over SSVP. The simulation was conducted on the dumbbell topology, where an SSVP flow competes with a single FTP flow (over TCP Reno). SSVP is able to sustain a regular transmission rate inducing oscillations of relatively small magnitude. More precisely, the integrated AIMD (0.2, 0.875) congestion control results in gentle rate reductions in response to packet drops.

The performance of video delivery is additionally depicted in Fig. 5. Delay variation scarcely exceeds the frustrating limit of 100 ms, since SSVP effectively smoothes transmission gaps validating our choice to apply a multiplicative decrease factor of 0.875, as derived in Section III-B. Furthermore, such a performance does not necessitate the use of deep playback buffers in order to ameliorate the effect of jitter.

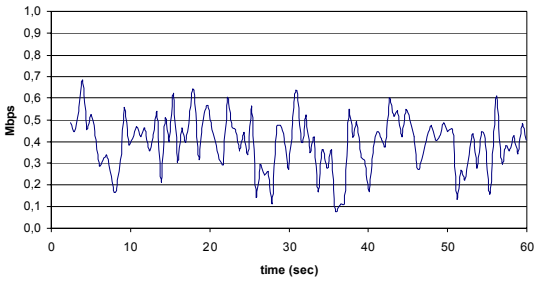

Figure 4. SSVP Receiving Rate

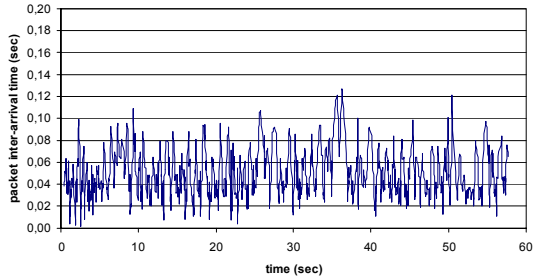

Figure 5. SSVP Packet Jitter

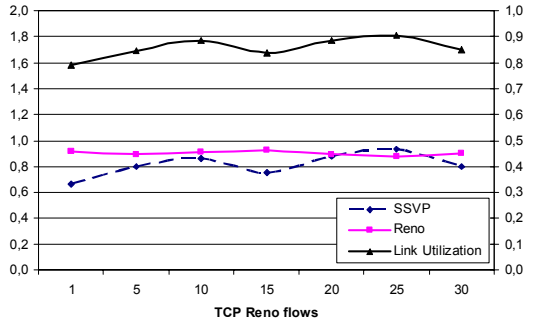

Figure 6. Normalized Throughput

Based on the same network topology (i.e. dumbbell), we investigate the impact of SSVP on corporate traffic. We simulated a single SSVP flow competing with a diverse number of FTP flows (1-30) successively. Fig. 6 illustrates the associated normalized throughput measurements. The target sending rate for SSVP is adjusted at 380 Kbps in order to enforce strong contention with interfering TCP flows. Despite the limited link resources (1 Mbps) and the high SSVP sending rate, TCP flows are allowed to obtain a fair share of the link (in each case they score a normalized throughput of nearly 1). On the other hand, SSVP manages to allocate the remaining resources, since bottleneck link utilization is always more than 80%. Consequently, SSVP co-exists fairly with TCP.

We also carried out a series of simulations in order to assess the performance of our approach versus TCP-friendly and UDP traffic (Figs. 7, 8). TCP-friendly contenders include the rate-based TFRC and the measurement-based TCP-Real, both implied to yield remarkable efficiency on video delivery over a

wide range of network and session dynamics. The associated experiments were conducted on the complex network topology, where we simulated diverse MPEG flows (1-50) competing with cross FTP traffic (10 flows).

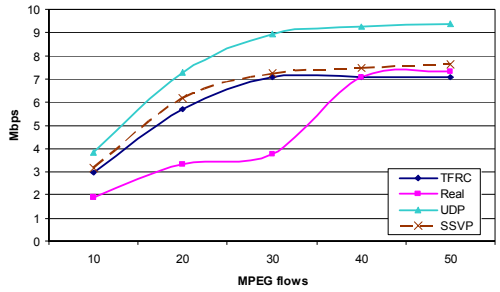

Figure 7. Goodput of MPEG flows

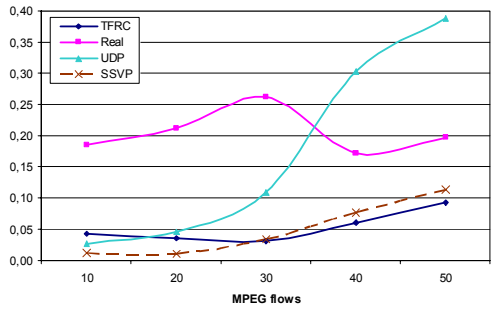

Figure 8. Delayed Packets Rate

UDP achieves the highest goodput rates (Fig. 7), since it steadily transmits at application rate regardless of the prevailing network conditions. SSVP also exhibits high link utilization, outperforming both TFRC and TCP-Real, regardless of link multiplexing (Fig. 7). Inline with the single SSVP flow results, the properly selected decrease rate (0.875), as well as the per-RTT rate adjustments effectively contribute to the remarkable performance of SSVP. Since goodput gains do not necessitate an improved performance on video delivery, we also demonstrate statistics from delayed packets in order to quantify protocol efficiency. Fig. 8 illustrates that SSVP, as well as TFRC achieve the timely delivery of most packets inducing minimal impairments on perceived video quality.

An overview of Figs. 7, 8 reveals that UDP causes long and variable delays that degrade video quality, since it results in rapidly growing queues and bottleneck buffer overflows. TCP-Real fails to control the tradeoff between responsiveness and smoothness, exhibiting an inherent weakness when bandwidth becomes available rapidly. As a result, the protocol yields a limited performance (relatively to SSVP and TFRC) in the situation of bandwidth availability (i.e. low link multiplexing). However, in the case of scarce bandwidth (high contention), the advantage of SSVP and TFRC (i.e. efficient bandwidth utilization) is diminished and consequently, all three protocols achieve equivalent goodput performance (Fig. 7). On the other hand, in heterogeneous environments with random transient errors, TFRC occasionally fails to obtain accurate estimates of the loss event rate, invoking an inappropriate equation-based recovery that impacts bandwidth utilization. Finally, SSVP incorporates a simple and yet efficient end-to-end congestion management scheme on top of the light-weight UDP. A careful selection of protocol parameters enables SSVP to effectively adapt to the vagaries of the network, enabling the delivery of smooth video in a wide range of network dynamics.

V. Conclusions

We have proposed a congestion control scheme that provides efficient support and QoS provisioning for streaming video applications over the Internet. Through simulations, we validated the robust behavior of our approach and we also demonstrated its feasibility in terms of wide range deployment. Furthermore, we showed that our implementation compares very favorably with congestion control mechanisms dedicated for time-sensitive traffic, such as TFRC.

References

[1] D. Bansal and H. Balakrishnan, "Binomial Congestion Control Algorithms", In Proc. of *IEEE INFOCOM 2001*, Anchorage, Alaska, USA, April 2001

[2] D. Chiu, R. Jain, "Analysis of the increase/decrease algorithms for congestion avoidance in computer networks", *Journal of Computer Networks*, 17(1), pp. 1-14, June 1989

[3] N. Feamster, D. Bansal and H. Balakrishnan, "On the Interactions Between Layered Quality Adaptation and Congestion Control for Streaming Video", In Proc. of *11th Int/nal Packet Video Workshop*, Kyongju, Korea, April 2001

[4] S. Floyd, M. Handley, J. Padhye, and J. Widmer, "Equation-Based Congestion Control for Unicast Applications", In Proc. of *ACM SIGCOMM 2000*, Stockholm, Sweden, August 2000

[5] V. Jacobson, "Congestion avoidance and control", In Proc. of *ACM SIGCOMM '88*, Stanford, USA, August 1988

[6] E. Kohler, M. Handley, and S. Floyd, "Designing DCCP: Congestion control without reliability", In Proc. of *ACM SIGCOMM 2006*, Piza, Italy, September 2006

[7] S. Mascolo, C. Casetti, M. Gerla, M. Sanadidi, and R. Wang, "TCP Westwood: Bandwidth Estimation for Enhanced Transport over Wireless Links", In Proc. of *ACM MobiCom '01*, Rome, Italy, July 2001

[8] R. Rejaie, M. Handley and D. Estrin, "RAP: An End-to-end Rate-based Congestion Control Mechanism for Realtime Streams in the Internet", In Proc. of *IEEE INFOCOM*, New York, USA, March 1999

[9] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: A transport protocol for real-time applications", *RFC 1889*, January 1996

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

[11] V. Tsaoussidis and C. Zhang, "The Dynamics of Responsiveness and Smoothness in Heterogeneous Networks", *IEEE Journal on Selected Areas in Communications* (*JSAC*), 23(6), pp. 1178-1189, June 2005

[12] Y. R. Yang and S. S. Lam, "General AIMD Congestion Control", In Proc. of *8th IEEE ICNP*, Osaka, Japan, November 2000

[13] C. Zhang and V. Tsaoussidis, "TCP Real: Improving Real-time Capabilities of TCP over Heterogeneous Networks", In Proc. of *11th IEEE/ACM NOSSDAV*, Port Jefferson, New York, USA, June 2001