WPI-CS-TR-02-17 May 2002

FairPlayer or FoulPlayer? - Head to Head Performance of RealPlayer Streaming Video Over UDP versus TCP

by

Jae Chung
Yali Zhu and Mark Claypool

Computer Science Technical Report Series

WORCESTER POLYTECHNIC INSTITUTE

Computer Science Department
100 Institute Road, Worcester, Massachusetts 01609-2280

FairPlayer or FoulPlayer? - Head to Head Performance of RealPlayer Streaming Video Over UDP versus TCP

Jae Chung, Yali Zhu and Mark Claypool

Computer Science Department
Worcester Polytechnic Institute
100 Institute Road
Worcester, MA 01609

{goos|yaliz|claypool}@cs.wpi.edu

Abstract— **The growth in power and connectivity of today's PCs promises a continued increase in streaming media over the Internet. Hand-in-hand with the increase in streaming media comes the impending threat of unresponsive traffic, often cited as the major threat to the stability of the Internet. The responsiveness of commercial streaming media applications will play an important role in the network impact of streaming media. Unfortunately, there are few empirical studies that analyze the responsiveness, or lack of it, of current streaming media products. In this work, we measure the responsiveness of RealVideo over UDP compared with RealVideo over TCP by simultaneously playing video clips selected from numerous RealServers on the Internet to two distinct video clients along the same network path. By varying the bottleneck bandwidth to the clients, we are able to analyze the "head-to-head" performance of RealVideo over UDP as compared to RealVideo over TCP, and correlate the results with network and application layer statistics. We find that most streaming RealVideo clips are not bandwidth constrained for typical broadband connections, resulting in a fair share of link bandwidth used by both RealVideo over TCP and RealVideo over UDP. In times of congestion, most RealVideo over UDP does respond to Internet congestion by reducing the application layer encoding rate, often achieving a TCP-Friendly rate. In times of severe congestion, RealVideo over UDP gets a proportionately larger share of available bandwidth than does the same video over TCP.**

Keywords— **RealPlayer, Video Streaming, TCP, UDP, Fairness**

I. Introduction

The growth in power and connectivity of today's computers has enabled streaming video across the Internet to the desktop. Increasingly, users can access online video clips through a Web browser by simply clicking on a link and having the Web browser start up an associated video player. Web sites today offer streaming videos of news broadcasts, music television, live sporting events and more. For example, in 2001 an estimated of 350,000 hours of online entertainment was broadcast each week over the Internet [1], with countless more hours downloaded on-demand.

While voice quality audio typically operates over a narrow range of bandwidth (32-64 Kbps), video operates over a much wider range of bandwidths. Video conferences and Internet videos stream at about 0.1 Mbps[1], VCR quality video at about 1.2 Mbps[2], broadcast quality video at about 2-4 Mbps[3], studio quality video at about 3-6 Mbps[3], and HDTV at about 25-34 Mbps[3]. Uncompressed video can require hundreds and even thousands of Mbps. Thus, video applications have the potential to reduce their data rates when bandwidth is constrained but also have a potential to demand enormous amounts of bandwidths, often greater than the available network capacity.

While TCP is the de facto standard transport protocol for typical Internet applications, there are as of yet no widely accepted rate-based transport protocols for multimedia. Unlike typical Internet traffic, streaming video is sensitive to delay and jitter, but can tolerate some data loss. In addition, streaming video typically prefers a steady data rate rather than the bursty data rate often associated with window-based network protocols. Recent research has proposed rate-based TCP-Friendly protocols in the hope that streaming media applications will use them [2], [3], [4], but such protocols are not yet widely part of any operating system distribution. For these reasons, streaming video applications often use UDP as a transport protocol rather than TCP. Moreover, with the use of repair

[1] H.261 and MPEG-4
[2] MPEG-1
[3] MPEG-2

techniques [5], [6], [7], [8], packet losses can be partially or fully concealed, enabling streaming video to operate over a wide range of packet loss rates. Repair techniques can reduce the impact of loss on the quality of the video by the user, thus reducing the incentive for multimedia applications to reduce their bandwidth in the presence of packet loss during congestion. Potentially high-bandwidth video over UDP using repair techniques suggests that video flows may not be *TCP-friendly* or, even worse, that video flows may be unresponsive to network congestion.

In the absence of end-to-end congestion control, TCP flows competing with UDP flows reduce their sending rates in response to congestion, leaving the unresponsive UDP flows to expand to use the vacant bandwidth, or, worse, contribute to congestion collapse of the Internet [9]. In light of this, recent research has explored router queue management approaches to identify and police unresponsive flows [10], [11], [12], [13], [14], [15], [16]. Such research often models unresponsive flows as transmitting data at a constant packet size and constant packet rate (CBR) or, as "firehose" applications, transmitting at an unyielding, maximum rate. However, commercial media products have been shown to not be strictly CBR [17], and, although using UDP, may respond to congestion at the application layer. A better understanding of the traffic rates and responsiveness of current streaming media applications may help create more effective network techniques to handle unresponsive traffic.

The responsiveness of commercial streaming media products will play an important role in the impact of streaming media on the Internet. The use of commercial streaming products, such as the Microsoft Windows Media Player and RealNetworks' RealPlayer, has increased dramatically [18]. Unfortunately, there are few empirical studies that analyze the responsiveness, or lack of it, of current streaming media products.

This study measures the responsiveness of RealVideo over UDP compared with RealVideo over TCP by simultaneous playing video clips selected from numerous distributed Web servers to two clients along the same network path. We set up a network testbed where two clients streamed video through a network router we control, connected to the Internet via a broadband connection. We varied the bottleneck bandwidth to the clients by limiting the bandwidth of the router's outgoing connection, allowing us to explore a range of congestion situations. The two clients then simultaneously streamed hundreds of selected videos with a variety of content and encoding formats from a diverse set of servers, while measuring packet loss rates and round-trip times of the connections as well as application level statistics such as encoding bandwidth and frame rate. By analyzing the "head-to-head" performance of the two video streams, we are able to assess the responsiveness of the video stream over UDP as compared to the video stream over TCP, and correlate the results with network and application statistics.

In analyzing our data, we make several contributions to better understanding the characteristics of potentially unresponsive streaming video on the Internet. We find that overall, most streaming RealVideo clips are not constrained under bandwidth from a typical broadband connection, resulting in a fair share of link bandwidth for both RealVideo over TCP and RealVideo over UDP. In times of congestion, most streaming RealVideo does respond to Internet congestion by reducing the application layer encoding rate. In times of severe congestion accompanied by high loss rates and/or high round-trip times, RealVideo over UDP gets a proportionately larger share of available bandwidth than does the same video over TCP. We also find numerous incentives for video streams to use UDP rather than TCP, which suggests that potentially unresponsive streaming media over UDP will likely persist for some time.

The rest of this paper is organized as follows: Section II presents background on RealPlayer needed to understanding our results; Section III describes our approach to obtain a wide-range of Internet measurements; Section IV and V present and analyze the measurement data obtained; Section VI discusses our findings; Section VII summarizes our conclusions and Section VIII presents possible future work.

II. RealVideo Background

RealPlayer, provided by RealNetworks[4], is the most popular streaming media player on the US Internet, with over 47% of the commercial market share [18]. RealVideo content providers create streaming videos using a number of possible video codecs, convert it to RealNetworks' proprietary format and place it on an Internet host running RealServer. During creation, content providers select target bandwidths appropriate for their target audience and specify other encoding parameters, such as frame size and frame rate, appropriate for their content. A RealServer then streams the video to a user's RealPlayer client upon connecting.

RealServer and players primarily use Real Time Streaming Protocol[5] (RTSP) for the session layer protocol. Occasionally, RealServer will use HTTP for metafiles or HTML

[4] http://www.real.com
[5] http://www.rtsp.org/

pages, and it may also be used to deliver clips to RealPlayer clients that are located behind firewalls. For this measurement study, all the video clips selected used RTSP, as described in Section III-A.

At the transport layer, RealServer uses both TCP and UDP for sending data. The initial connection is often in UDP, with control information then being sent along a two-way TCP connection. The video data itself is sent using either TCP or UDP. By default, the actual choice of transport protocol used is determined automatically by the RealPlayer and RealServer, resulting in UDP about 1/2 the time and TCP the other half [19]. The choice of UDP or TCP can also be specified by the user [20]. For our study, we specifically set RealPlayer to use UDP in some cases and TCP in others, as described in Section III-B.

RealSystem supports an application level media scaling technology called *SureStream* in which a RealVideo clip is encoded for multiple bandwidths [21]. When streaming a SureStream RealVideo clip, RealServer determines which encoded stream to use based on feedback from RealPlayer regarding the current network conditions. The actual video stream served can be varied in mid-playout, with the server switching to a lower bandwidth stream during network congestion and then back to a higher bandwidth stream when congestion clears. We study the flexibility of SureStream scaling in Section V-F.

For each video clip, RealPlayer keeps a buffer to smooth out the video stream because of changes in bandwidth, lost packets or jitter. Data enters the buffer as it streams to RealPlayer, and leaves the buffer as RealPlayer plays the video clip. If network congestion reduces bandwidth for a few seconds, for example, RealPlayer can keep the clip playing with the buffered data. If the buffer empties completely, RealPlayer halts the clip playback for up to 20 seconds while the buffer is filled again. We measure the rate at which RealPlayer fills the buffer in Section V-D.

III. Approach

In order to empirically compare and contrast the performance of RealVideo over UDP with RealVideo over TCP, we employed the following methodology:

- Select RealVideo URLs that use the Real Time Streaming Protocol (RTSP) using well known Web search engines (see Section III-A).
- Construct a "head-to-head" environment for comparing network layer performance of two competing RealVideo streams, UDP and TCP, that simultaneously playout a RealVideo clip (see Section III-B).
- Construct a "media scaling" environment for comparing the application layer behavior of non-competing UDP or TCP RealVideo streams (see Section III-C).

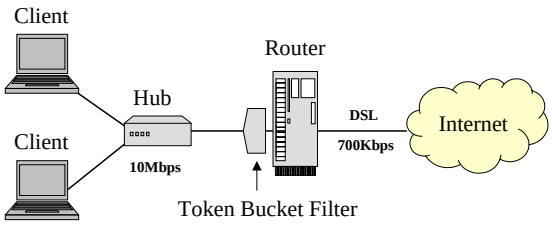

Fig. 1. Testbed Network Setup: Head-to-Head Environment

- Iteratively play the selected RealVideo URLs in both environments with different configurations and analyze the results (see Section IV and Section V).

A. RealVideo Clips

To form a playlist, we searched for RealVideo clips (URLs) accessible through Web pages using well-known search engines, such as Yahoo and Google, and randomly selected 100 RTSP RealVideo URLs from the search results. Of the selected URLs, 76 are from the United States, 9 from Canada, 8 from the UK, 6 from Italy, and 1 from Germany. While our selection method of using US/English based commercial search engines likely influenced the predominance of North American URLs, our RealPlayer clients ran from North America and it is likely that there is typically strong locality of access for most streaming players. Other statistics on the selected RealVideo URLs (or clips) are available in Section IV and Section V-F.

B. Head-to-Head Comparison Environment

To compare the network layer performance of RealVideo over UDP and RealVideo over TCP, we had two RealPlayers, one using UDP and the other using TCP, simultaneously stream video from the same URL, along the same network path, while we captured network statistics. As depicted in Figure 1 the two RealPlayers ran on separate PCs, attached to the same 10 Mbps hub. Each PC was equipped with Pentium III 700MHz processor, 128 MB RAM and a UDMA-66 15 GB hard disk, and was running Linux kernel version 2.4. Both PCs ran RealPlayer version 8.0.3, with one RealPlayer configured to use UDP and the other RealPlayer configured to use TCP.

While the testbed network setup did not guarantee that two streams with same source and destination address always traveled the same network path, the relatively persistent characteristics of Internet routing [22] suggest they shared the same route in most cases. Also, any routing changes made during streaming applied to both TCP and UDP streams.

The hub facilitated capturing network layer performance since packets destined to either PC were broadcasted to both PCs. We ran `tcpdump`[6], a well-known network packet sniffer, on one PC to filter and log the video stream packets. During pilot studies, we verified that the packet filtering and logging did not induce much CPU nor disk load and did not interfere with the video playout. At the end of each RealVideo stream, information such as the IP packet size and arrival time were extracted from the tcptrace log using `ethereal`[7] and processed to obtain network layer statistics, such as throughput.

We wrote an *expect* script to automate the process of starting RealPlayer on each client and measuring network layer performance. For each RealVideo URL in our playlist (from Section III-A), the script contacted the RealServer to see if it was alive. If so, the script then started a *tcpdump* and a *ping* (at 1 second intervals) ping) to the server to obtain samples of the round-trip time and packet loss rate. Next, the script ran one RealPlayer on each of two client PCs simultaneously, one player using TCP to play the clip and the other player using UDP to play the clip. If more than a minute passed without tcpdump receiving a packet, the script stopped the players and the logging, and produced the desired network layer performance statistics.

Initially, our testbed network was connected to our campus LAN which is connected to the Internet via a 15 Mbps link[8], and 10 "head-to-head" sets of experiments were carried out, but that setup showed few signs of network contention, making it difficult to measure the responsiveness of RealVideos. The UDP and TCP video streams shared bandwidth fairly as bandwidth was not constrained, a result supported by an earlier study [23]. In order to reduce the bottleneck bandwidth, the testbed was connected to a commercial DSL network that offers a maximum bandwidth of 700 Kbps, as shown in Figure 1 (without the token bucket filter), and 10 "head-to-head" sets of experiments were carried out. Unfortunately, 700 Kbps was still not constrained enough to create a bottleneck for two typical RealVideo streams.

In order to more effectively control the incoming bandwidth, we set up a private router connected to the DSL configuration to enable us to created constrained bandwidth situations. We attached a software implementation of a Token Bucket Filter (TBF)[9] to the Ethernet card at the internal network of the router, a Linux PC. The TBF queue size was set to 10 Kbytes and the *burst allowed* (the maximum number of tokens available during idle times) was set to 1600 Bytes, slightly larger than a typical 1500 Byte MTU. The *token rate* (available bandwidth) was set to 600 Kbps, 300 Kbps, 150 Kbps and 75 Kbps. Note, since we have two streaming flows, one TCP and one UDP, competing, their fair share is approximately half of each bottleneck bandwidth. With a fixed TBF queue size, the maximum queuing delay increased as the available bandwidth decreased, as is typical of relatively narrow-band network connections. The DSL-TBF-075 configuration moded a typical narrow-band modem connection with a moderate queue that often results in a high queuing delay. Thus, we had DSL-TBF configurations as follows:

DSL-TBF-600: BW=600Kbps, MAX_Q_DELAY= 133ms
DSL-TBF-300: BW=300Kbps, MAX_Q_DELAY= 267ms
DSL-TBF-150: BW=150Kbps, MAX_Q_DELAY= 533ms
DSL-TBF-075: BW= 75Kbps, MAX_Q_DELAY=1067ms

For each DSL-TBF configuration, we carried out two sets of "head-to-head" measurements, where each set played all video clips in the playlist. For consistency, this paper primarily analyzes the results from the DSL-TBF experiments (except for some LAN results in Section V-D), but the LAN and the original DSL measurement results were very similar to that of DSL-TBF-600.

C. Media Scaling Measurement Environment

Streaming video can adjust to the available bandwidth during congestion by *media scaling* [24] where video encoding is switched to a lower rate. As mentioned in Section II, RealSystem uses a media scaling technology called *SureStream* in which a RealVideo clip is encoded for multiple bandwidths [21]. The actual video stream served can be varied in mid-playout, with the server switching to a lower bandwidth stream during network congestion and then back to a higher bandwidth stream when congestion clears.

We sought to assess the ability of the RealPlayer application layer to respond to different levels of congestion. To do this, we needed to measure the number of scale levels typically used in RealVideos and the actual encoded bandwidth associated with each scale level. The metrics for application level responsiveness, then, are the number of media scales changes seen in a playout and the time taken for the application layer encoded data rate to drop to the available bandwidth during times of congestion.

To study media scaling in RealPlayer we used *RealTracer*, a tool developed for a previous study [19], which plays RealVideo streams and records application level statistics, including encoding rate. Unfortunately, Real-

[6] http://www.tcpdump.org/
[7] http://www.ethereal.com/
[8] http://www.wpi.edu/Admin/Netops/MRTG/
[9] http://www.linuxpowered.com/archive/howto/Adv-Routing-HOWTO-7.html#ss7.3

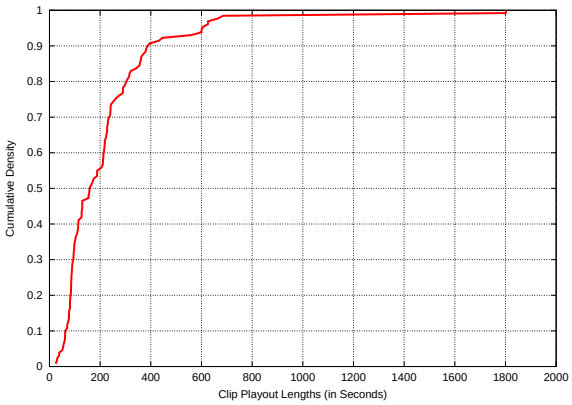

Fig. 2. CDF of Video Clip Playout Lengths (All Runs)

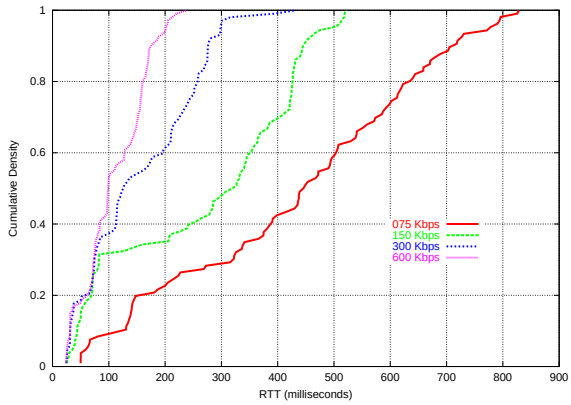

Fig. 3. CDF of Ping Round Trip Time (All Runs)

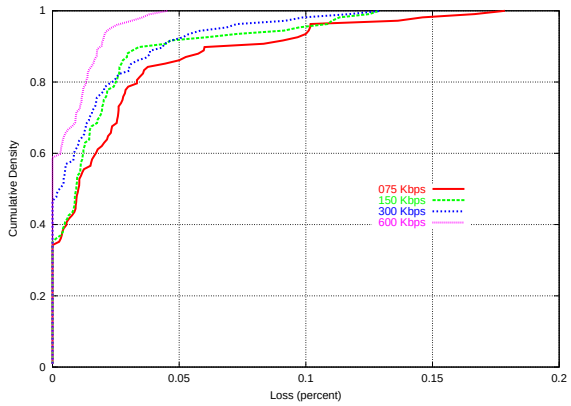

Fig. 4. CDF of Ping Loss (All Runs)

Tracer only runs on Microsoft Windows operating systems, so we were not able to capture application layer performance statistics when measuring "head-to-head" network layer performance.

Instead, we set up an alternate environment in which one of the client machines in the DSL-TBF environment was booted with Microsoft Windows ME and equipped with RealPlayer 8 Basic version 6.0.9 and RealTracer version 1.0. We then ran a non-competing, single UDP or TCP stream for each URL in the playlist, while limiting the TBF incoming bandwidth to 35 Kbps [10]. We tried other TBF rates such as 25 Kbps, 150 Kbps and 300 Kbps to ensure we measured all possible scale levels (or encoded bandwidths) used for clip playouts. However, only 2 sets of measurements, TCP for the entire playlist and UDP for the entire playlist, on the 35 Kbps DSL-TBF configuration is used to characterize the responsiveness of the RealVideo media scaling (see Section V-F).

IV. Results

Over the course of 2 months, we streamed over a total of 200 hours of video from a cumulative total of over 4000 video clips. Figure 2 depicts a Cumulative Density Function (CDF) of the video playout lengths observed over all runs. The end time is recorded by subtracting the last time the end-host received packets from the start time of the clip. The median clip was about 3 minutes, while the shortest and longest clip played out at 20 seconds and 30 minutes, respectively.

Of the original set of 100 video clips, 1 clip could not be served by UDP, perhaps because of server firewall restrictions. Also, 21 clips including the UDP restricted clip became completely unavailable after the initial selection. We removed these clips from further analysis.

Of the remaining 79 clips in the playlist, about 30% of their servers did not respond to `ping` packets, making them unavailable for loss and round trip time (RTT) analysis. For all RTT and loss analysis in this Section and in Section V-B, Section V-C and Section V-E, we removed the data from these clips. Note, however, that we did use the other data recorded for other analysis.

Figure 3 depicts a CDF of the average RTTs obtained via `ping` probes for each bottleneck bandwidth. The 75 Kbps connection had the highest round-trip times. For the 150-600 Kbps connections, about 33% of the clips had the same RTT regardless of the bottleneck bandwidth since these clips stream at less than 150 Kbps, and therefore do not suffer additional queuing delays at the router. For the remaining 70% of the clips, the lower the bottleneck bandwidth the higher the queuing delays, caused primarily by the 10 Kbytes buffer at the bottleneck router.

Figure 4 depicts a CDF of the loss rates obtained via `ping` probes for each bottleneck bandwidth. About 37% of the clips played with low bottleneck bandwidths had no loss, while about 50% of the clips played at higher bottleneck bandwidths had no loss. Overall loss rates increase about 0.1% as bottleneck bandwidths decrease from 600 Kbps to 300 Kbps to 150 Kbps to 75 Kbps.

Figure 5 depicts a CDF of the packet size for all the

[10] The queue was set to 5 Kbytes for the 35 Kbps DSL-TBF configuration.

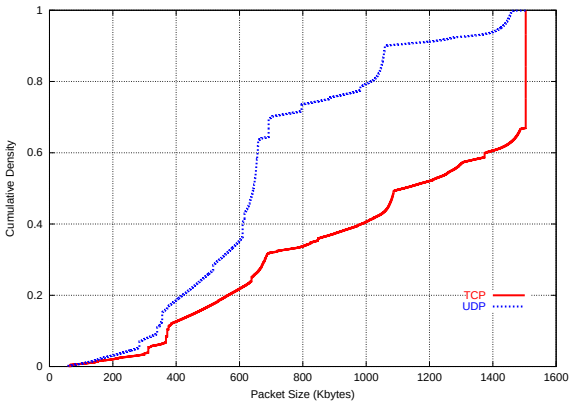

Fig. 5. CDF of Packet Size (All Runs)

RealVideo clips over TCP and the RealVideo clips over UDP. In general, the TCP streams used larger packets than the UDP streams with a median UDP packet size of about 640 Kbytes, and a median TCP packet size of about 1100 Kbytes. Moreover, more than 30% of the TCP packets were equal to the typical network MTU, 1500 Bytes. A possible reason for the larger packet sizes over TCP is that while RealServers can control the application frame sizes to send, with TCP, those frames are often grouped and sent based on the current TCP window sizes. Although not shown in the graph, for the same bandwidth (in Kbps), the larger packets meant TCP streams sent fewer network packets than the UDP streams.

In order to provide background for the forthcoming analysis in Section V, we present two examples showing throughput versus time for a RealVideo stream over UDP going "head-to-head" with a RealVideo stream over TCP.

Figure 6 (top), shows an unconstrained network which allowed both UDP and TCP to stream as desired. During the first part of each clip, both streams filled a delay buffer (as mentioned in Section V-D) at a higher data rate than the data playout rate. The UDP stream took an aggressive buffering approach, then slowly adapted to the network conditions. In contrast, the TCP stream more conservatively increased the transmission rate during buffering. Detailed analysis of RealVideo buffering over TCP and UDP is provided in Section V-D.

After the buffering phase, both UDP and TCP streamed at a steady, comparable playout rate since there was no congestion. The average bandwidth over the entire clip playout was relatively fair. Detailed analysis of RealVideo average bandwidth over TCP and UDP is provided in Section V-A.

During the playout phase, the bandwidth taken over 500 ms intervals had about the same variance (was equally "smooth") for both UDP and TCP. Detailed analysis of RealVideo smoothness over TCP and UDP is provided in Section V-G.

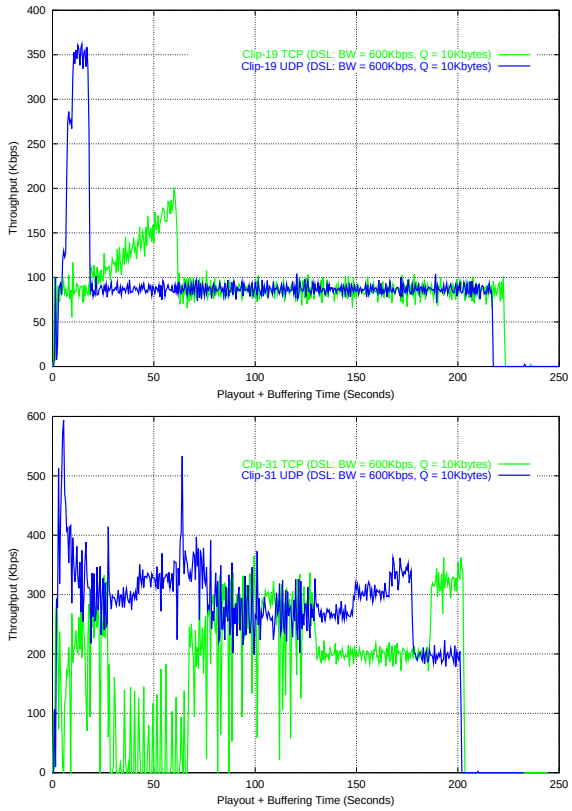

Fig. 6. Head-to-Head UDP versus TCP Streaming: FairPlayer (top) and FoulPlayer (bottom)

Figure 6 (bottom), shows two video streams contending for limited network bandwidth. During the first part of each clip, UDP aggressively grabbed the available bandwidth, then slowly reduced its data rate as it encountered congestion. TCP, on the other hand, was severely limited in bandwidth received until it slowly took up the bandwidth the UDP stream released. This unfairness in bandwidth persisted for nearly the entire clip, resulting in a lower average bandwidth for TCP than UDP. Detailed measurements of unfairness in average bandwidth for RealVideo over TCP versus UDP are provided in Section V-A, with analysis of TCP-Friendliness in Section V-E. Possible reasons for bandwidth unfairness are provide in Section V-B and Section V-C.

At 30 seconds, the TCP stream scaled its application layer data rate down to the lowest quality level (audio only), which allowed the UDP stream enough bandwidth room to scale up its quality level, creating an greater unfairness. At 70 seconds, the TCP stream increased its application data rate, causing the UDP stream to encounter congestion and decrease its application data rate. From 70 to 130 seconds, the average bandwidth for both streams was relatively fair. At 130 seconds, the TCP stream again decreased its application data rate allowing the UDP

stream to twice increase its application data rate. However, for the final 20 seconds, the UDP stream scaled its data rate down to match the TCP data rate, whereupon the TCP stream scaled its data rate up to surpass that of UDP. Detailed analysis of the behavior and role of application level data rate scaling is provided in Section V-F.

V. Analysis

In analyzing the responsiveness of RealVideo over UDP compared with RealVideo over TCP, we first analyze bandwidth aggregated over all clips and then compare head-to-head bandwidth use (Section V-A). Next, we explore correlations of bandwidth disparity with round-trip times (Section V-B) and loss rates (Section V-C). We then measure the rate of initial buffering compared with the steady playout rate for both RealVideo over TCP and RealVideo over UDP (Section V-D). After that, we compare the bandwidths of the TCP and UDP clips with a TCP-Friendly rate (Section V-E). We then move to the application layer where we analyze the application scaling capabilities (Section V-F). Lastly, we end with analysis of the smoothness of playout for RealVideo over UDP as compared with the smoothness of playout for RealVideo TCP (Section V-G).

A. Bandwidth

Figure 7 depicts CDFs of the average bandwidth used by TCP and UDP for each clip for bottleneck bandwidths of 600, 300, 150 and 75 Kbps. The TCP and UDP distributions are nearly the same for the 600 Kbps bottleneck bandwidths. However, as bandwidth becomes more constrained, the distributions separate, with UDP having a consistently higher distribution of bandwidths than TCP.

We next analyze the head-to-head bandwidth for each pair of (TCP, UDP) clips. For each clip pair, in Figure 8 we plot an (x,y) point where x is the average bandwidth used by the TCP stream and y is the average bandwidth used by the UDP stream. The points for each bottleneck bandwidth are depicted by a different point style. The dashed 45 degree line provides a reference for bandwidth equally shared by TCP and UDP. Points above the line indicate UDP received more average bandwidth while points below the line indicate TCP received more average bandwidth. The distance from the line indicates the magnitude of the average bandwidth difference.

From Figure 8, while there are some points that lie along the equal bandwidth line, there are many cases of bandwidth disparity. The highest bandwidth playouts for the 600 Kbps bottleneck bandwidths had the greatest bandwidth disparities. For the 600 Kbps bottleneck bandwidths, there are visually as many points below the equal bandwidth line where TCP received more bandwidth as

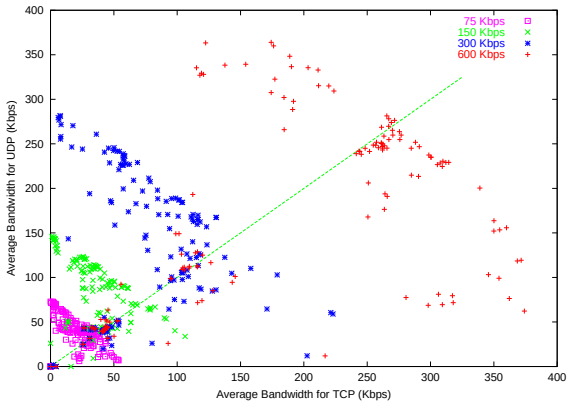

Fig. 8. Head-to-Head Average Bandwidth (All Runs)

there are above the equal bandwidth line where UDP received more bandwidth. For the lower bottleneck bandwidths, there are visually considerably more points above the equal bandwidth line, indicating UDP received more bandwidth.

We next analyze the bandwidth disparity relative to the bottleneck bandwidth available. For each clip pair, we subtract the UDP average bandwidth from the TCP average bandwidth and divide the difference by the bottleneck bandwidth. Thus, equal sharing of bandwidth has a value of zero, a value of -1 indicates UDP got the entire bottleneck bandwidth, and a value of +1 indicates TCP got the entire bottleneck bandwidth. Figure 9 depicts CDFs of the normalized bandwidth differences for each bottleneck bandwidth.

For the 600 Kbps bottleneck bandwidth, about 40% of the clips shared the bandwidth equally. As indicated by the region in the top right, about 30% of the TCP clips got more bandwidth than their counterpart UDP clips while about 20% of the UDP clips got more bandwidth than their counterpart TCP clips, as indicated by the region in the bottom left. The greatest bandwidth disparity was approximately half the bottleneck bandwidth.

For the lower bottleneck bandwidths, there were increasingly fewer clips with equal bandwidth. The UDP clips got substantially more bandwidth than did their TCP counterparts, as indicated by the large areas under the distributions on the bottom left. For the 300 Kbps bottleneck bandwidth, about 60% of the UDP clips got more bandwidth than their TCP counterparts, and for the 150 Kbps and 75 Kbps bottleneck bandwidths, about 70% of the UDP clips got more bandwidth than their TCP counterparts. For 300, 150 and 75 Kbps bottleneck bandwidths, about 20% of the UDP clips got twice the normalize bandwidth of their TCP counterparts. For 150 and 75 Kbps bottleneck bandwidths, about 20% of the UDP clips received more than 80% more of the bottleneck bandwidth

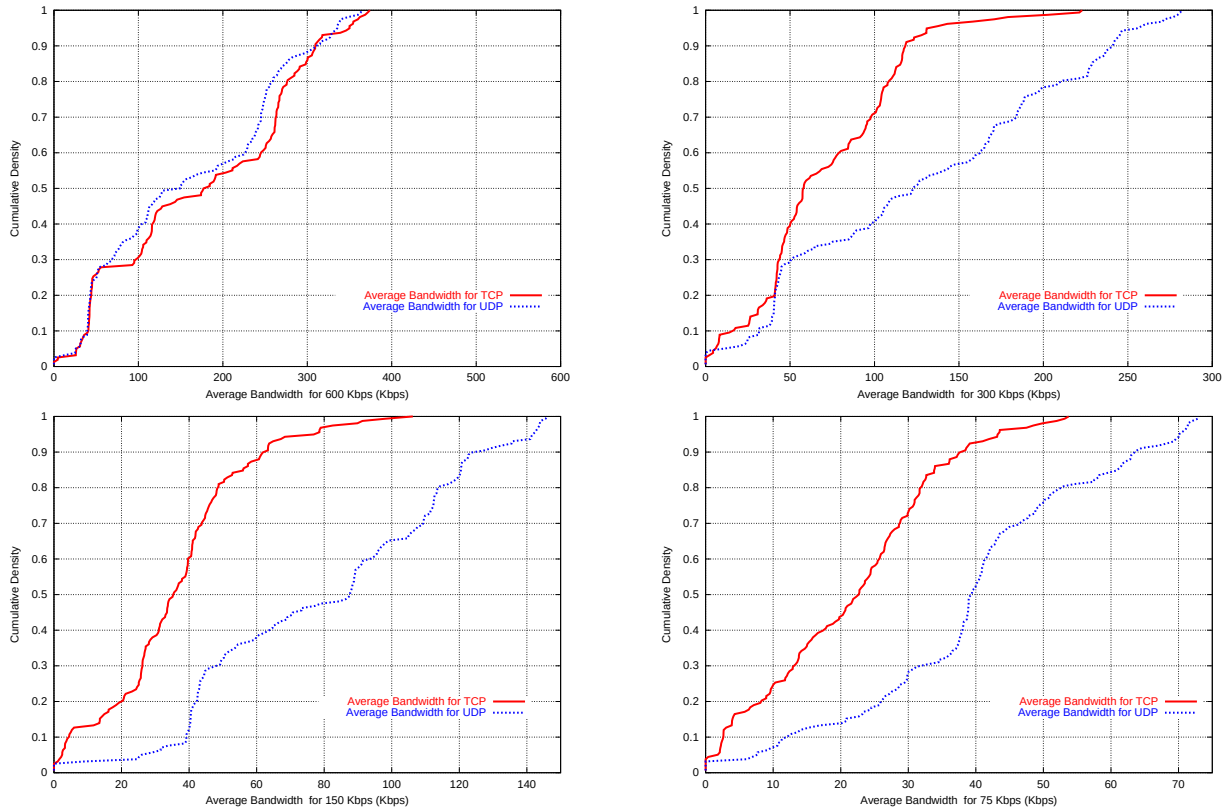

Fig. 7. CDFs of Average Bandwidth for Bottleneck Bandwidths of 600, 300, 150, and 75 Kbps

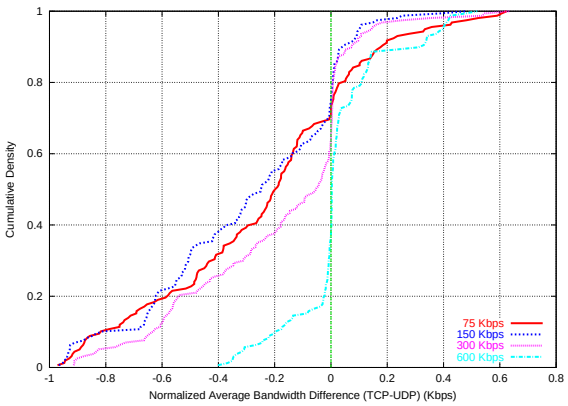

Fig. 9. CDF of the Difference (TCP - UDP) in the Head-to-Head Average Bandwidth, Normalized by the Bottleneck Bandwidth (All Runs)

than their TCP counterparts. However, even for the lowest bottleneck bandwidths, there were still cases where the TCP clips got more bandwidth than their UDP counterparts, as depicted by the areas above the distributions in the upper right.

In general, as bandwidth becomes more constrained, streaming RealVideo clips over UDP receive relatively more bandwidth than do streaming streaming RealVideo clips over TCP.

B. Round-Trip Time

We next analyze if bandwidth disparity increases with round-trip time. The data rate of TCP is paced by acknowledgments, so a higher round-trip time directly results in a lower maximum throughput. The data rate of UDP, however, is not similarly constrained, suggesting that higher round-trip times may make the end-hosts slower to respond to congestion, thereby increasing the overall bandwidth used by UDP.

For each clip pair, we subtract the UDP average bandwidth from the TCP average bandwidth and graph the difference versus the average RTT for that run. Figure 10 depicts the difference versus RTT for all clips. The points below the horizontal zero line are runs in which the UDP clips got more bandwidth than their TCP counterparts, while the points above the horizontal zero line are runs in which the TCP clips got more bandwidth than their UDP counterparts. Visually, the UDP clips appear to have gotten slightly more bandwidth as the RTTs get larger. The correlation coefficient is -0.13. For reference, we also graph a least-squares line of the difference versus the RTT.

We continue the round-trip time analysis in Figure 11 which depicts the differences versus RTT for each bottleneck bandwidth. For each bottleneck bandwidth, we graph a least-squares error line and compute the correlation co-

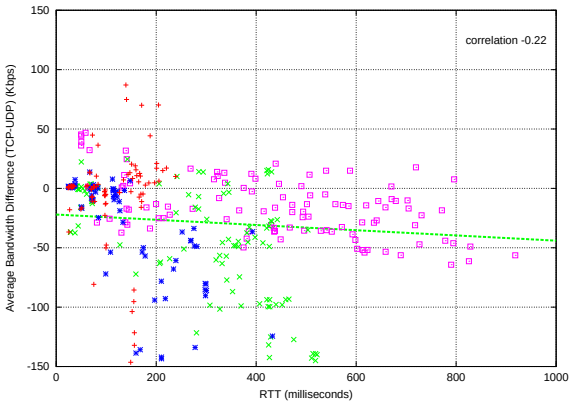

Fig. 10. Head-to-Head Difference (TCP - UDP) in Average Bandwidth vs. Round-Trip Time for All Bottleneck Bandwidths

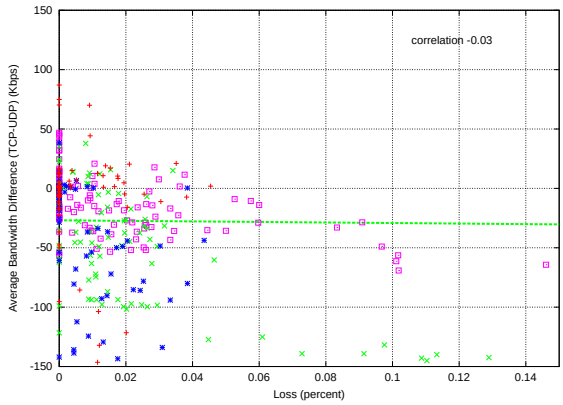

Fig. 12. Head-to-Head Difference (TCP - UDP) in Average Bandwidth vs. Loss for All Bottleneck Bandwidths

efficient. For 600 Kbps, the correlation with bandwidth difference and RTT is very slight. However, when bandwidth becomes constrained, as in the cases of 300, 150 and 75 Kbps bottleneck bandwidths, there was a modest correlation between bandwidth difference and RTT, with UDP having gotten increasingly more bandwidth than TCP as the round-trip times increased.

In general, as round-trip time increases, streaming RealVideo clips over UDP receive relatively more bandwidth than do streaming RealVideo clips over TCP for bandwidth constrained conditions.

C. Loss

We next analyze if bandwidth disparity increases with loss rate. The data rate of TCP is typically halved with each loss event, so a higher loss rate directly results in a lower maximum throughput. The data rate of UDP, however, is not similarly constrained, suggesting that higher loss rates may be ignored or downplayed, thereby increasing the overall bandwidth used by UDP.

For each clip pair, we subtract the UDP average bandwidth from the TCP average bandwidth and graph the difference versus the loss rate for that run. Figure 12 depicts the difference versus loss rate for all clips. Visually, the UDP clips appear to get only slightly more bandwidth as the loss rates increase. The correlation coefficient is -0.03. For reference, we also graph a least-squares error line of the difference versus the loss rate.

We continue the round-trip time analysis in Figure 13 which depicts the differences versus loss rate for each bottleneck bandwidth. For each bottleneck bandwidth, we graph a least-squares error line and compute the correlation coefficient. For 600 Kbps, there is no correlation with bandwidth difference and loss rate. However, for all other bandwidth constrained conditions, the correlations between bandwidth difference and loss rate were moderate, with UDP having gotten increasingly more bandwidth than TCP as the loss rates increased.

In general, as loss rate increases, streaming RealVideo clips over UDP receive relatively more bandwidth than do streaming RealVideo clips over TCP for bandwidth constrained conditions.

D. Buffering Data Rate

As shown in the example at the end of Section IV, RealPlayer buffers data at an accelerated rate for the first part of a clip. Analyzing the rate of this buffering rate versus steady playout rate may help to characterize the bursty nature of RealVideo streams.

The buffering rate was difficult to determine by looking at bandwidth during fixed time periods, especially when TCP was contending for bandwidth. An aggressively buffering UDP stream often reduced the playout rate for the TCP stream for tens of seconds. After this time, the UDP stream might have finished buffering, allowing the TCP stream to increase its bandwidth to its normal buffering rate. So, for each clip, we compute the maximum bandwidth averaged over 10 second intervals taken over the first 80 seconds (calling this the *buffering data rate*) and compared this to the average bandwidth over the time from 100 seconds until the clip ends (calling this the *steady playout rate*).

Figure 14 depicts the ratio of the average buffering data rate to the average steady playout rate for different steady playout rates. For reference, a ratio of 1 indicates that the buffering data rate was equivalent to the steady playout rate. Low bandwidth clips buffered at up to 6 times their average playout rate. Higher bandwidth clips buffered at relatively lower rates, possibly because total bandwidth restrictions limited them from buffering at a higher rate.

In order to determine if bandwidth restrictions limit

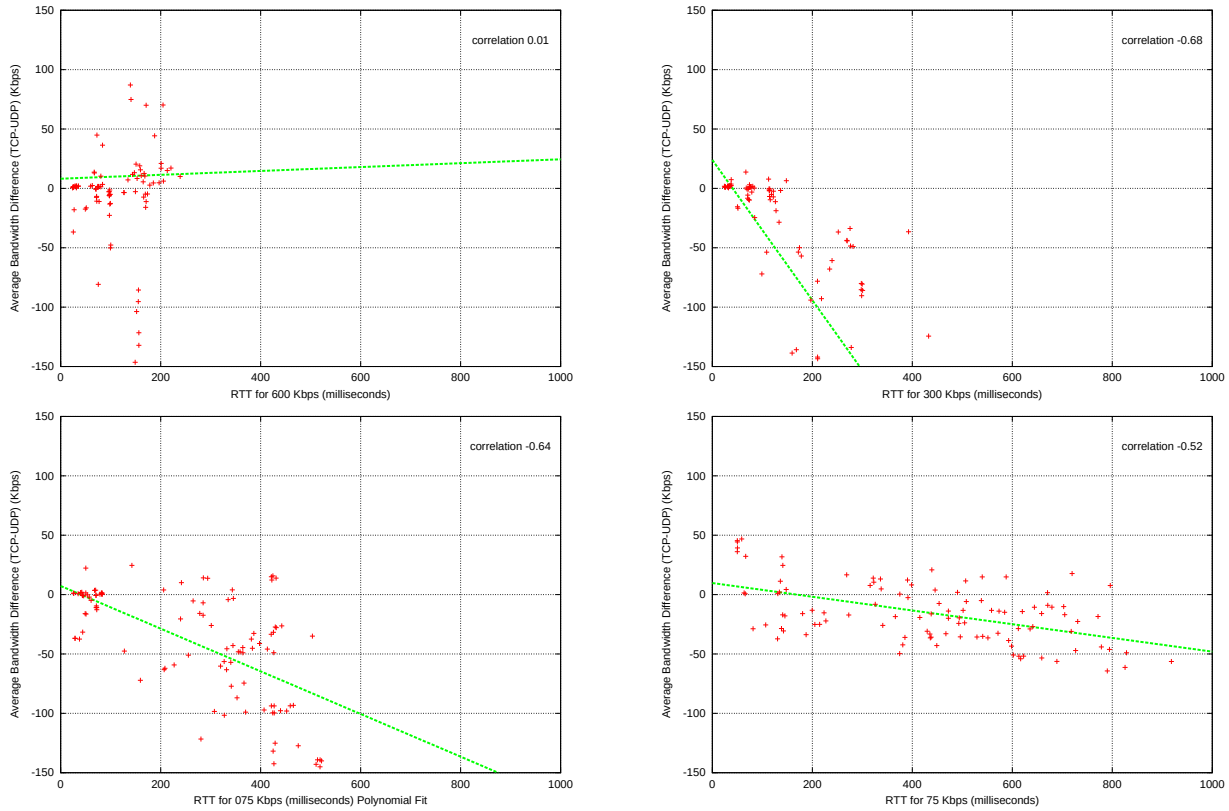

Fig. 11. Head-to-Head Difference (TCP - UDP) in Average Bandwidth vs. Round-Trip Time for Bottleneck Bandwidths of 600, 300, 150, and 75 Kbps

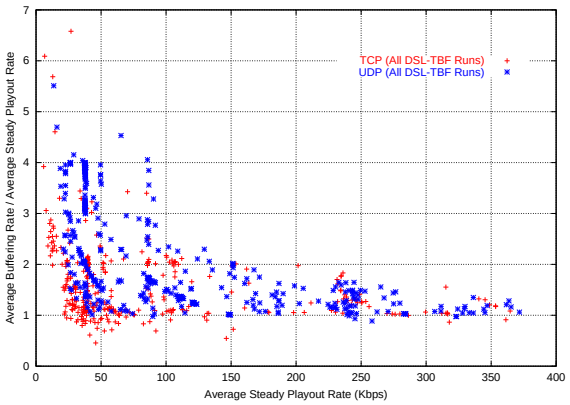

Fig. 14. Ratio of Average Buffering Rate to Average Steady Playout Rate versus Average Steady Playout Rate (All DSL-TBF Runs)

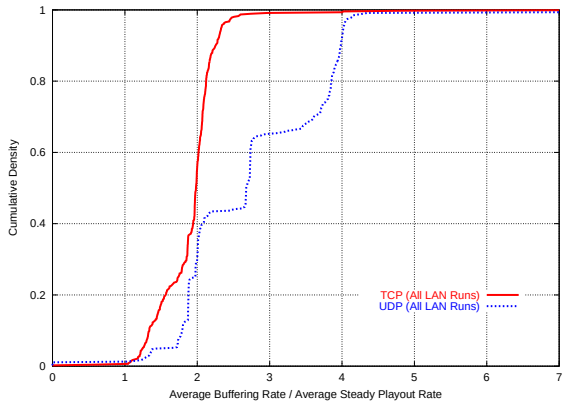

Fig. 15. CDF of Ratio of Average Buffering Rate to Average Steady Playout Rate for LAN

buffering rates, we ran a set of experiments with the bottleneck bandwidth being the campus LAN attached to the Internet via a 15 Mbps link[11]. In this setup, the LAN environment was relatively unconstrained, having a bottleneck bandwidth which was typically at least three times that of our 600 Kbps bottleneck bandwidth.

Figure 15 depicts a CDF of the ratio of the average buffering data rate to the average steady playout rate. The buffering rate to steady rate for UDP was nearly the same as that of TCP for 40% of the clips. For 60% of the clips, however, the ratio of buffering rate to steady for UDP was significantly higher than that of TCP. For UDP, the "steps" in the CDF are at typical bandwidth encoding rates, where the buffering rate was a fixed multiple of these rates. For TCP, the steep slope in the CDF at around 2 suggests TCP streams typically buffered at a rate twice that of the steady playout rate.

In general, both RealVideo clips over UDP and Re-

[11] http://www.wpi.edu/Admin/Netops/MRTG/

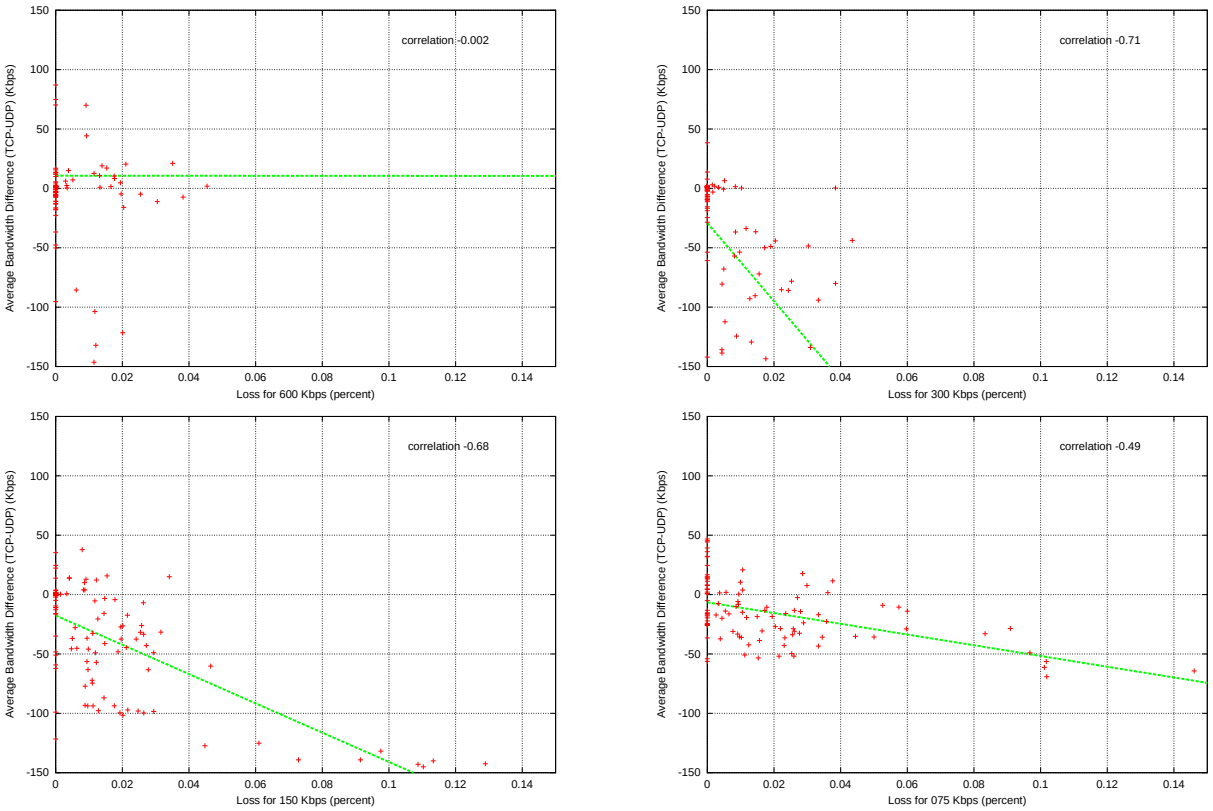

Fig. 13. Head-to-Head Difference (TCP - UDP) in Average Bandwidth vs. Loss for Bottleneck Bandwidths of 600, 300, 150, and 75 Kbps

alVideo clips over TCP buffer data at a significantly higher rate than the steady playout rate, suggesting that overall RealVideo traffic is bursty.

E. TCP-(Un)Friendly

Although RealVideo over UDP may receive a disproportionate share of bandwidth versus their TCP counter parts, this may be because RealVideo TCP clips transmit at less than their maximum rate. A more serious test of unfairness is whether RealVideo over UDP is *TCP-Friendly* in that its data rate does not exceed the maximum arrival of a conformant TCP connection in the same circumstances. The TCP-Friendly rate, T Bps, for a connection is given by [9]:

$$T \leq \frac{1.5\sqrt{2/3} \times s}{R \times \sqrt{p}} \quad (1)$$

with packet size s, round-trip time R and packet drop rate p. From section V-C, approximately 40% of the clips that had no measured loss, and so we remove them from futher TCP-friendly analysis. For the remaining clips for each run, we compute the TCP-Friendly rate (T) (equation V-E), using a packet size (s) of 1500 bytes[12] and the loss rate (p) and RTT (R) obtained from the corresponding `ping` samples. We then compare T to the average bandwidth used by first the UDP clip and then the TCP clip. For each bottleneck bandwidth, we record the total number of times each protocol type was not TCP-Friendly. The results are shown in Table I.

Overall, about 11% of the UDP clips were not TCP-Friendly, with the lower bottleneck bandwidths having a slightly higher percentage. For the 600 Kbps bottleneck bandwidth, some TCP clips showed up as not being TCP-Friendly, about the same number of UDP clips that were not TCP-Friendly. This anomaly was most likely not caused by an incorrect TCP implementation, but rather was because the loss rates were sampled and the 600 Kbps bottleneck bandwidth clips had very low loss rates, making them susceptible to unlucky sampling.

The TCP-Friendly formula in equation V-E is conservative in that it computes the maximum bandwidth an aggressive TCP connection would receive. Thus, connections that achieve more bandwidth than computed in equation V-E are clearly not TCP-Friendly. In general, while there are many cases where streaming RealVideo over UDP is, in

[12] The maximum packet size recorded. See Figure 5

Bottleneck Bandwidth	Un-Friendly UDP	Un-Friendly TCP
75 Kbps	8 (12%)	0 (0%)
150 Kbps	7 (11%)	0 (0%)
300 Kbps	9 (14%)	0 (0%)
600 Kbps	4 (6%)	5 (8%)
Total	28 (11%)	5 (2%)

TABLE I
NUMBER (AND PERCENT) OF NON TCP-FRIENDLY FLOWS

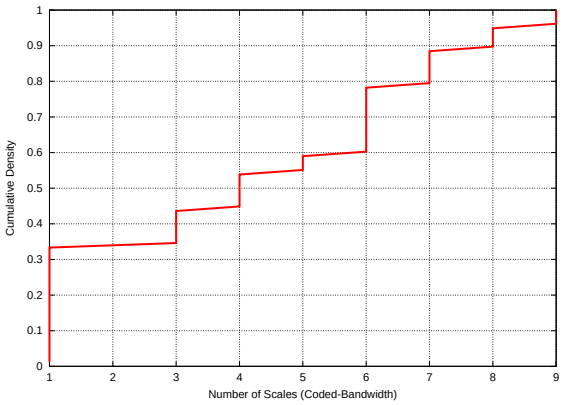

Fig. 16. CDF of Media Scales (All Runs)

principle, TCP-Friendly, evidence suggests that streaming RealVideo clips over UDP can be non TCP-Friendly, particularly for bandwidth constrained conditions.

F. Media Scaling

Media scaling technologies adapt media encoding to the available bandwidth in an effort to provide acceptable media quality over a range of available bandwidths [25], [26]. In times of congestion, media scaling benefits both the network, by reducing offered load, and also the user, by providing graceful degradation in perceived quality [24]. As mentioned in Section II, RealSystems provide SureStream media scaling at the application level that can select an adequate quality version of a video for the current network conditions.

In the previous section, we showed that even if using media scaling, RealVideo streaming (over UDP) can be still non TCP-Friendly. This section analyzes data from the media scaling measurement experiments, as described in Section III-C, in an effort to determine why.

Figure 16 shows CDF of the number of distinct encoded-bandwidth levels seen in each clip for all runs. About 35% of the clips were not using media scaling at all, and therefore, over UDP, these clips were unresponsive to network congestion. Less than 50% of the clips

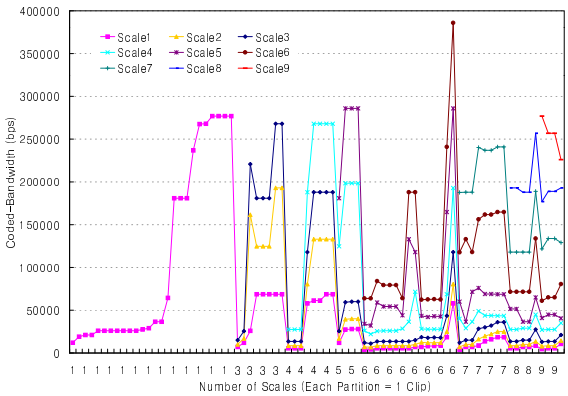

Fig. 17. Media Scales and Encoded-Bandwidth (All Runs)

were using more than 4 levels of scaling and so could only adjust to the available bandwidth coarsely.

Figure 17 shows the scale levels and corresponding bandwidths for each clip, sorted first by number of levels, and second by the lowest encoded bandwidth. For the unresponsive clips (those with only 1 scale level), 40% were high-quality video clips that required more than 150 Kbps of bandwidth. Also, over 1/2 of the clips with 3 to 5 scale levels were targeted primarily for broadband connections and could not adapt to bandwidths below 50 Kbps. Streaming these clips on bandwidth constrained links using UDP would cause unfairness to any competing TCP flows. RealVideo clips with more than 5 scale levels were designed to adapt more readily to low bandwidth conditions, evidenced by the number of scale levels with low bandwidth, but may still have been unfair at higher bandwidths.

When bandwidth reduces during congestion, in order to preserve timing the real-time servers must employ media scaling whether streaming over UDP or TCP. Figure 18 shows the media scaling behavior of two sample RealVideo clips streaming over UDP and TCP, where the inbound bandwidth available was 35 Kbps. For both clips and both streams, the initial encoded bandwidth was significantly higher than the available bandwidth, depicted by the horizontal line at 35 Kbps. Each step represents an application layer scaling of bandwidth. In the top graph of Figure 18, both TCP and UDP scaled their application data rate 6 times before the encoded rate settled at a proper application rate below the available bandwidth. However, UDP was able to obtain this application level rate much more quickly than did TCP. In the bottom graph of Figure 18, UDP quickly used 7 scale levels to adjust the application's data rate to the available bandwidth, while TCP, on the other hand, took more than 20 seconds to adjust the rate, and then it did so in one, large encoding rate change.

We believe the difficulty RealPlayer over TCP has in ad-

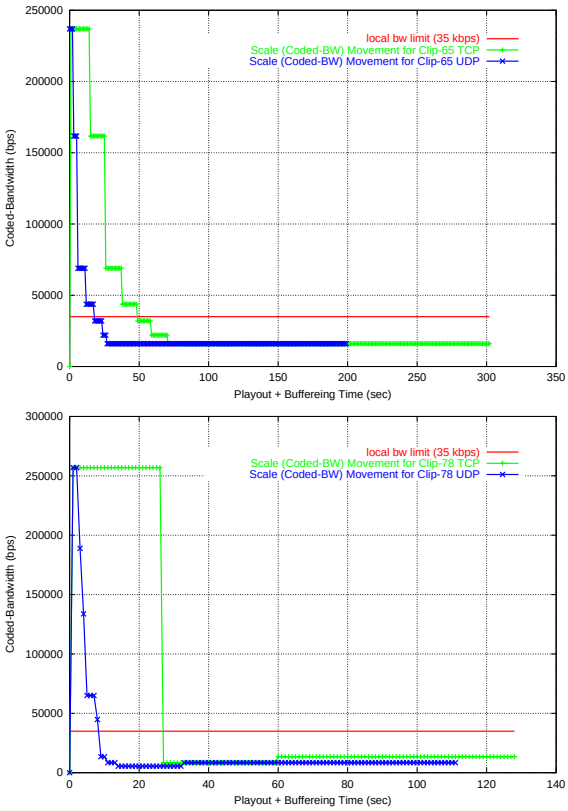

Fig. 18. Media Scaling Dynamics: Clip-65 (top) and Clip-78 (bottom) (DSL: BW=35 Kbps, Q=5 Kbytes)

justing the application data rate to the network data rate is because TCP hides network information. RealPlayer over TCP can only measure application level goodput and not information on packet drop rates or network packet round trip times. RealPlayer over UDP, on the other hand, can more easily detect packet losses and measure round-trip times, allowing it to more quickly adjust the application data rate to the network rate.

Moreover, for high-quality videos not able to detect network congestion is critical. As evidenced by the TCP stream in the bottom graph of Figure 18, the server fills available TCP buffers with high quality video frames that must be delivered by the transport layer before it is able to scale down. For the user, this results in a large delay before frame playout begins as the high-quality frames are buffered over a low-bandwidth connection. Quantitatively, by looking at the end-time of transmission, the top graph of Figure 18 shows that to play 3 minutes of video, streaming over UDP took about 200 seconds while streaming over TCP took more than 300 seconds. In other words, streaming over UDP required 20 seconds of buffering to play 3 minutes of a video clip, while streaming over TCP required more than 2 minutes of buffering to play the same clip.

In Figure 19, the CDFs depict the number of media scale changes seen for each video clip, and summarize the relative responsiveness of RealVideo to scale the application data rate to below the network bandwidth. Overall, UDP streams had more scale changes than did TCP streams. Also, Figure 19 shows that about 20% (55% - 35%) of the streams that scaled when streamed over UDP did not scale at all when streamed over TCP.

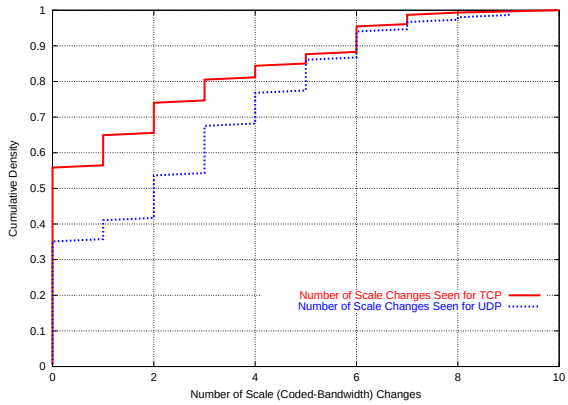

Fig. 19. CDF of Media Scale Changes (DSL: BW=35 Kbps, Q=5 Kbytes)

Figure 20 summarizes the responsiveness of RealVideo media scaling based on how quickly the video stream adapted to the available bandwidth after streaming started. Specifically, we measure the time taken for the coded-bandwidth to drop under the inbound bandwidth limit, depicted as the first point under the 35 Kbps limit for each stream in Figure 18. Figure 20 shows that about 15% of video clips were low-quality and always required less than 35 Kbps. Also, 25% (40% - 15%) of the video clips were able to adapt to the available bandwidth within a couple of seconds, independent of the transport protocol used. However, for the remaining 60% of the clips, the TCP video streams took significantly more time to adapt their scales to the available bandwidth. For example, 80% of the UDP video streams adapted to the available bandwidth within 10 seconds, while it took more than 25 seconds for the same percentage of the TCP video streams to adapt.

In general, a significant fraction of RealVideo clips are unable to adapt their application data rates to the available network bandwidth, causing UDP streaming to be unfair under bandwidth constrained conditions. However, most RealVideo clips can, and do, scale their application data rates to the available network bandwidth. RealVideo streams over UDP can adjust their application data rate to the available bandwidth more efficiently than can RealVideo over TCP.

G. *Smoothness*

Streaming media has more strict timing constraints than does traditional media. Streaming video requires not only

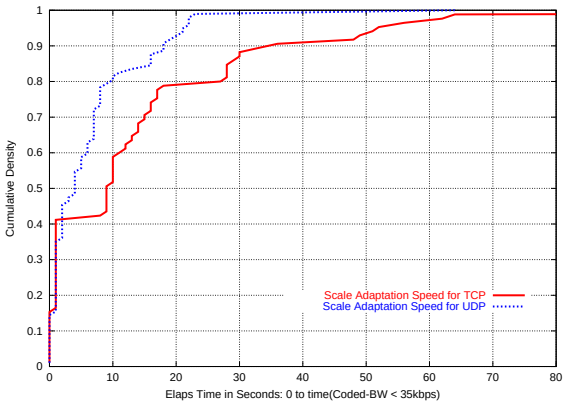

Fig. 20. CDF of Media Scale Adaptation Speed (DSL: BW=35 Kbps, Q=5 Kbytes)

a moderate to high bandwidth but also a smooth data rate. TCP's acknowledgment based window advancement can result in a bursty data rate, especially for high round-trip times. Streaming media applications often cite these reasons in choosing UDP as a transport protocol.

For each clip, we calculate the "smoothness" of the network data rate by taking the ratio of consecutive bandwidths measured over 500 ms intervals. For example, if the data rate is 200 Kbps for one time interval and 400 Kbps the next time interval, the smoothness would be 2. If the data rate then dropped by half back to 200 Kbps, it would be 0.5. Figure 21 depicts CDFs of smoothness for each network bottleneck bandwidth. Both TCP and UDP were smooth for a bottleneck of 600 Kbps. With bottleneck bandwidths of 300, 150 and 75 Kbps, both TCP and UDP became noticeably less smooth, with TCP often far less smooth than UDP.

In general, streaming RealVideo clips over UDP receive a smoother playout rate than do streaming streaming RealVideo clips over TCP for bandwidth constrained conditions.

VI. Discussion of Results

In the current Internet, there are no concrete incentives for applications that use UDP to initiate end-to-end congestion control. In fact, at the network level, unresponsive applications may be "rewarded" by receiving more than their fair share of link bandwidth. As seen in Section V, streaming media over UDP can result in a higher average bandwidth rate than streaming media over TCP, primarily because competing TCP sources are forced to transmit at a reduced rate. Plus, as seen in Section V-F, it is more difficult for the application layer to adjust the encoding rate to the available bandwidth when using TCP (because there is no API that gives you available bandwidth, for example). Lastly, as seen in Section V-G UDP often provides a smoother media playout rate than TCP. Thus, there are strong application-oriented reasons for streaming media to use UDP rather than TCP, suggesting potentially high-bandwidth video over UDP may contribute to congestion collapse.

However, given the current climate where end-to-end congestion control, and even TCP-Friendly congestion control, is fundamentally important to the well-being of the Internet, there are likely social pressures for video software designers not to release products without some form of end-to-end congestion control. Moreover, an unresponsive "fire-hose" application, such as high quality video over a congested link, is ineffective from the application standpoint primarily because having a congested router randomly drop packets can cause more important data packets to be dropped. Instead, applications can significantly benefit by using media scaling, as illustrated by RealPlayer in Section V-F to make intelligent decisions about which packets not to send beforehand, making low quality video over the same congested link quite effective. Anecdotally, in our pilot tests with severe congestion, older versions of RealPlayer would continue to attempt to stream video, inducing even more congestion, while newer versions of RealPlayer would terminate the connection under the same conditions. Moreover, as shown in Section V-F, RealVideo over UDP clearly scales the application data rate to meet the available bandwidth. Thus, while it is not clear as to exactly what degree practical or social incentives are effective, they may be having a significant impact.

The higher buffering rate seen in Section V-D is beneficial for users, but possibly harmful to the network. A higher buffering rate either allows the player to build up a larger buffer before beginning frame playback and thus better avoiding any unsmoothness caused by network jitter or transient congestion, or allows the frame playback to begin earlier. However, the increased buffering rate makes the streaming traffic more bursty and, with UDP, it can cause even more unfairness versus other TCP flows. Overall, from the network point of view, the buffering rate should be limited to the playout rate.

VII. Conclusions

The decreasing cost of powerful PCs and the increase in video content on the Web is fueling the growth of streaming video over the Internet. Unlike traditional applications, streaming video often uses UDP as a transport protocol rather than TCP, suggesting that streaming video may not be TCP-friendly or that streaming video may be unresponsive to network congestion. Since congestion control is fundamentally important to the health of the Internet, a better understanding of streaming video using UDP ver-

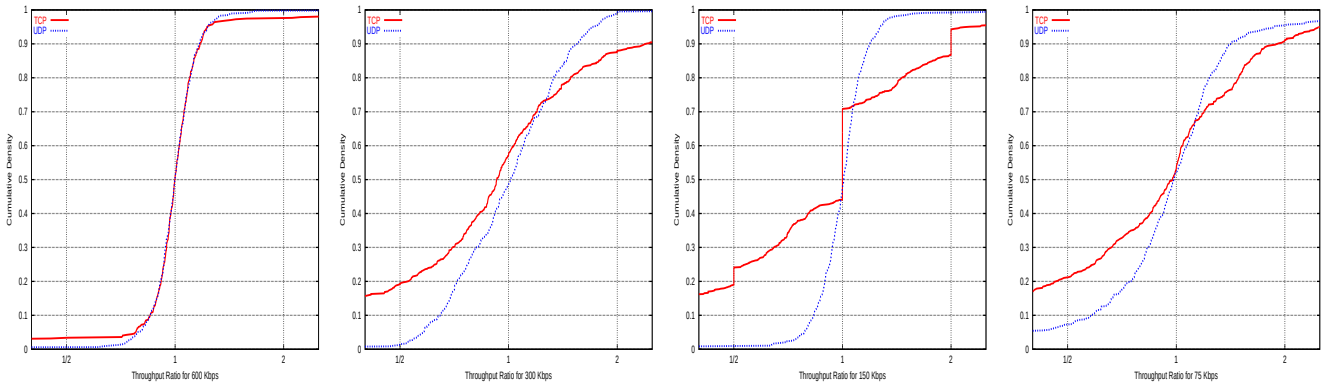

Fig. 21. Smoothness Ratio for Bottleneck Bandwidths of 600, 300, 150, and 75 Kbps

sus TCP can help focus network layer research that detects and polices unresponsive flows, or transport layer research that develops better streaming protocols.

Commercial streaming video players, such as RealNetworks' RealPlayer, promise to have a large influence on the impact of streaming video on the Internet. While previous empirical studies have focused on Internet traffic in general or have concentrated on overall measurements of streaming applications, to the best of our knowledge, there have been no detailed studies on the responsiveness to congestion of commercial players streaming over UDP relative to TCP.

In this work, we do "head-to-head" network performance comparisons of RealVideo streaming over UDP with RealVideo streaming over TCP. We set up a testbed that allows us to simultaneously stream two RealVideo clips, one over TCP and one over UDP, along the same network path. Our testbed also lets us control the network bottleneck bandwidth, thus allowing us to compare the responsiveness to congestion of the UDP and the TCP streams. Using our testbed, we stream over 1000 video clips with a variety of content and encoding bandwidths selected from across the Internet.

Overall, we find there are many incentives for RealVideo streams to use UDP versus TCP, including higher average bandwidth during congestion, smoother playout rate during congestion and more efficient application layer media scaling.

RealVideo over UDP typically receives the same bandwidth as that of RealVideo over TCP. Even during periods of packet loss, most RealVideo over UDP is TCP-Friendly. However, under constrained bandwidth conditions, RealVideo over UDP can get substantially more bandwidth than RealVideo over TCP. The bandwidth use gets increasingly unfair with an increase in packet loss rate and round-trip time.

Most RealServers can, and often do, scale the application layer data rate in an attempt to match the network data rate. Application scaling tends to be coarser at higher levels of bandwidth but is often fine grained at lower levels of bandwidth. While application scaling can be an effective means of responding to congestion, about 35% of RealVideos cannot do application scaling at all. Adjusting the application data rate to the network bandwidth is more difficult when streaming over TCP versus UDP, most likely because TCP streams do not have as much information about the current network state as do the UDP streams.

RealPlayers typically buffer video data for up to 40 seconds at a much higher rate than the average playout rate. While beneficial to the user, this initial burst of traffic can cause considerable congestion and probably makes RealVideo network traffic more difficult to manage.

When bandwidth is constrained, RealVideo streams over UDP typically playout at a smoother rate than RealVideo streams over TCP. However, both TCP and UDP streams are equally smooth when there is no contention for bandwidth.

VIII. Future Work

This work is only another step in the analysis of streaming multimedia traffic on the Internet, leaving many areas for future work.

The major commercial competitor to RealNetworks' RealPlayer is Microsoft's Windows Media Player. A "head-to-head" performance comparison of MediaPlayer streaming over UDP might be beneficial to understand the differences in responsiveness across commercial players.

We intentionally selected pre-recorded video clips to help ensure consistency in the videos played out during each set of experiments. Live content, captured and served directly from a video camera or television, typically has different characteristics than does pre-recorded content. Future work could be to measure the performance of live RealVideo content on the Internet and compare it to that of

the pre-recorded RealVideo content in our study.

The work in this paper did not explore the relationship between perceptual quality of the video, influenced by application level metrics such as frame rate and jitter, and network metrics. A better understanding of the impact on perceptual quality on video streaming over UDP versus TCP might further aid in developing more effective ways to use a TCP-Friendly share of bandwidth.

REFERENCES

[1] Real Networks Incorporated, "RealNetworks Facts," 2001, URL: http://www.reanetworks.com/gcompany/index.html.

[2] Rezza Rejaie, Mark Handley, and D. Estrin, "RAP: An End-to-end Rate-based Congestion Control Mechanism for Realtime Streams in the Internet," in *Proceedings of IEEE Infocom*, 1999.

[3] Sally Floyd, Mark Handley, Jitendra Padhye, and Jorg Widmer, "Equation-Based Congestion Control for Unicast Applications," in *Proceedings of ACM SIGCOMM Conference*, 2000, pp. 45 – 58.

[4] Joerg Widmer, Martin Mauve, and Jan Peter Damm, "Probabilistic Congestion Control for Non-Adaptable Flows," in *Proceedings of NOSSDAV*, May 2002.

[5] J-C. Bolot, S. Fosse-Parisis, and D. Towsley, "Adaptive FEC-Based Error Control for Internet Telephony," in *Proceedings of IEEE INFOCOM*, Mar. 1999.

[6] Yanlin Liu and Mark Claypool, "Using Redundancy to Repair Video Damaged by Network Data Loss," in *Proceedings of IS&T/SPIE/ACM Multimedia Computing and Networking (MMCN)*, Jan.25-27 2000.

[7] C. Padhye, K. Christensen, and W. Moreno, "A New Adaptive FEC Loss Control Algorithm for Voice Over IP Applications," in *Proceedings of IEEE International Performance, Computing and Communication Conference*, Feb. 2000.

[8] K. Park and W. Wang, "QoS-Sensitive Transport of Real-Time MPEG Video Using Adaptive Forward Error Correction," in *Proceedings of IEEE Multimedia Systems*, June 1999, pp. 426 – 432.

[9] Sally Floyd and Kevin Fall, "Promoting the Use of End-to-End Congestion Control in the Internet," *IEEE/ACM Transactions on Networking*, Feb. 1999.

[10] R. Mahajan, S. Floyd, and D. Wetherall, "Controlling High-Bandwidth Flows at the Congested Routers," in *In Proceedings of the 9th International Conference on Network Protocols (ICNP)*, Nov. 2001.

[11] Ion Stoica, Scott Shenker, and Hui Zhang, "*Core*-Stateless Fair Queueing: Achieving Approximately Fair Bandwidth Allocations in High Speed Networks," in *Proceedings of ACM SIGCOMM Conference*, Sept. 1998.

[12] W. Feng, D. Kandlur, D. Saha, and K. Shin, "Stochastic Fair Blue: A Queue Management Algorithm for Enforcing Fairness," in *Proceedings of IEEE INFOCOM*, Apr. 2001.

[13] D. Lin and R. Morris, "Dynamics of Random Early Detection," in *Proceedings of ACM SIGCOMM Conference*, Sept. 1997.

[14] Debasis Mitra, Keith Stanley, Rong Pan, Balaji Prabhakar, and Konstantinos Psounis, "CHOKE, A Stateless Active Queue Management Scheme for Approximating Fair Bandwidth Allocation," in *Proceedings of IEEE INFOCOMM*, Mar. 2000.

[15] A. Clerget and W. Dabbous, "Tag-based Unified Fairness," in *Proceedings of IEEE INFOCOM*, Apr. 2001.

[16] Z. Cao, Z. Wang, and E. Zegura, "Rainbow Fair Queuing: Fair Bandwidth Sharing Without Per-Flow State," in *Proceedings of IEEE INFOCOMM*, Mar. 2000.

[17] Art Mena and John Heidemann, "An Empirical Study of Real Audio Traffic," in *In Proceedings of the IEEE Infocom*, Mar. 2000, pp. 101 – 110.

[18] Jupiter Media Metrix, "Users of Media Player Applications Increased 33 Percent Since Last Year," Apr. 2001, Press Release. http://www.jup.com/company/pressrelease-.jsp?doc=pr01040.

[19] Yubing Wang, Mark Claypool, and Zheng Zuo, "An Empirical Study of RealVideo Performance Across the Internet," in *Proceedings of the ACM SIGCOMM Internet Measurement Workshop*, Nov. 2001.

[20] Real Networks Incorporated, "RealPlayer 8 User Manual," copyright 2000, URL: http://service.real.com/help/player-/plus_manual.8/rppmanual.htm.

[21] Real Networks Incorporated, "RealProducer User's Guide," copyright 2000, URL: http://www.service.real.com/help/library-/guides/producerplus85/producer.htm.

[22] Vern Paxson, "End-to-End Routing Behavior in the Internet," *IEEE/ACM Transactions on Networking*, vol. 5, no. 5, pp. 601–615, 1997.

[23] Yubing Wang, Mark Claypool, and Zheng Zuo, "An Empirical Study of RealVideo Performance Across the Internet," Tech. Rep. WPI-CS-TR-01-16, Worcester Polytechnic Institute, July 2001.

[24] Avanish Tripathi and Mark Claypool, "Improving Multimedia Streaming with Content-Aware Video Scaling," in *Workshop on Intelligent Multimedia Computing and Networking (IMMCN)*, March 2002.

[25] Paul Bocheck, Andrew Campbell, Shih-Fu Chang, and Raymond Lio, "Utility-based Network Adaptation for MPEG-4 Systems," in *Proceedings of International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV)*, June 1999.

[26] Jonathan Walpole, Rainer Koster, Shanwei Cen, Crispin Cowan, David Maier, Dylan McNamee, Calton Pu, David Steere, and Liujin Yu, "A Player for Adaptive MPEG Video Streaming Over The Internet," in *Proceedings of the SPIE Applied Imagery Pattern Recognition Workshop*, Oct. 1997.