

# Evaluating the behaviour of Peer-to-Peer IP traffic in wireless access networks

D. Astuti and M. Kojo

Department of Computer Science, University of Helsinki, Finland

e-mail: {davide.astuti,markku.kojo}@cs.helsinki.fi

## Abstract

During the last few years, the use of Peer-to-Peer (P2P) applications for file exchange has increasingly gained popularity in the Internet. P2P file sharing applications generate an intensive amount of traffic both in downstream and upstream direction as they tend to exploit the available capacity of the underlying network. The coexistence of these greedy applications with delay sensitive applications such as real-time audio or video is very challenging as they can significantly reduce the available capacity for other applications and increase the congestion level and delay in the network. In this paper, we study the coexistence of P2P traffic with audio and video applications and study the dynamics of P2P flows under various network configurations. We analyze two ways for controlling the P2P bandwidth consumption in order to limit the impact on time-critical applications: network-controlled and user-controlled. Results show that both approaches are beneficial. The former improves performance of multimedia applications. The latter is beneficial to P2P traffic.

## Keywords

Peer-to-Peer file sharing, wireless access networks, TCP, UDP, DiffServ

## 1. Introduction

During the last years, the use of Peer-to-Peer (P2P) applications for file exchange has increasingly gained popularity. P2P traffic component of the global Internet traffic has been growing. P2P file sharing applications build up a virtual network of hosts (peers) able to communicate with each other without following the classical client-server paradigm. Instead, each peer has the same functionalities as the other peers so that the traffic load is distributed among the connected users. In some P2P networks, a special role is assigned to a set of hosts (super-peers), which coordinate the operation.

P2P file sharing applications generate an intensive amount of traffic as they tend to exploit all available link capacity in the underlying network. Traffic is generated both in downstream and upstream direction. This can have a significant impact on increasing network congestion and reducing the capacity available for other applications, such as interactive (web) or multimedia applications. In particular, interactive applications are adversely affected as the high network congestion may significantly increase the network delay and hence the application response time. The performance of multimedia applications (audio conversation video broadcasting and conferencing) may become unacceptable as they are strongly sensitive to network delay variations. In the near future we may expect that the P2P applications will be increasingly used

also over wireless links. When wireless networks are involved, arranging peaceful coexistence of P2P and delay sensitive applications becomes even harder.

In this paper, we study the coexistence of TCP-based P2P traffic with delay sensitive audio and video applications in an asymmetric wireless access networks. We study the nature of problems caused by P2P traffic both in downlink (from the network to the user) and uplink (from the user towards the network) direction and analyse how the P2P traffic affects UDP-based audio and video traffic under various wireless access network configurations. The wireless access network often presents bandwidth asymmetry, so that the user is provided with different amount of link bandwidth in downlink and uplink direction.

We analyze two approaches for controlling the P2P bandwidth consumption to limit its impact on time-critical applications: network-controlled and user-controlled. In the former approach, Differentiated Services (DS) (Blake et al, 1998) architecture is applied in the wireless access network so that audio and video flows have higher priority and are scheduled before the P2P flows. In the latter approach, the user poses a limit on the maximum amount of bandwidth P2P applications can exploit in the uplink direction. Finally, we also combine the approaches.

The simulation results show that both approaches improve performance, although neither approach alone is enough in yielding acceptable performance for both traffic types. Using DS in conjunction with bandwidth limitation gives the best performance. The rest of the paper is organized as follows. Section 2 gives an overview on P2P file sharing. Section 3 describes the methodology and the test arrangements for the experiments. Section 4 presents the simulation results. Finally, in Section 5 we conclude the work and discuss the future work.

## **2. Peer-to-Peer file sharing applications**

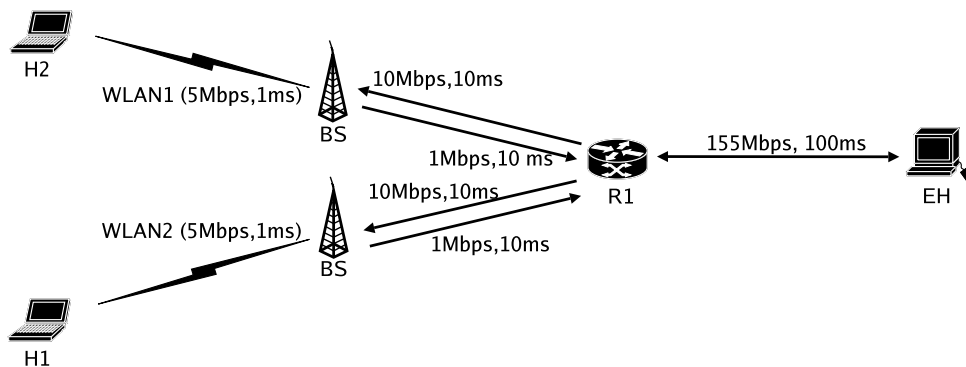
Peer-to-Peer (P2P) file sharing applications allow users to share files among the community which forms the P2P network. Such a network is composed by a set of hosts (peers) able to communicate with each other without following the classical client-server paradigm. Instead, each peer has the same functionalities as the other peers so that the traffic load is distributed among the connected users. However, a special role may be assigned to a set of hosts (super-peers), which coordinate the network functioning. Various approaches are used to search file resources in the network: centralized, for example Napster (Napster, 2005), or decentralized, for example KaZaA (KaZaA, 2005), BitTorrent (BitTorrent, 2005).

Users running P2P applications attempt to utilize the available network capacity by downloading the requested files from multiple sources using several simultaneous TCP connections. This generates an intensive amount of traffic in the downlink direction (from the network towards the user) for a long period of time as the leading content typically shared in P2P networks is characterized by audio and video files. At the same time, users may choose to make several files available for other peers. This creates a significant amount of traffic also in the uplink direction (from the user towards the network). Some P2P applications (i.e. BitTorrent) allow users to set a limit on the maximum amount of upstream bandwidth the P2P application may share between the simultaneous uploading TCP connections.

### 3. Performance Experiments

We run experiments using ns simulator (Network Simulator 2, 2005) to study the coexistence of P2P file sharing applications with audio and video traffic in an asymmetric wireless access network. Figure 1 shows the network topology. Mobile hosts H1 and H2 are connected to a Base Station (BS), which acts as a last-hop router. The two BSs are connected to router R1 through an asymmetric transmission network (1Mbps in uplink, 10Mbps in downlink). The router R1 is connected to the End Host (EH) using a 155 Mbps link. Link latencies are shown in Figure 1, no link error models are applied. TCP NewReno version is used.

With peer to peer applications the mobile host does not act only as a client but also as a server for the other end. In our tests, the peer-to-peer traffic is modelled with 4 unidirectional TCP connections in both directions between H1 and EH and between H2 and EH simultaneously. The TCP transfers run for 100 seconds. As many P2P applications allow the user to control the amount of link capacity used for uploading, we run tests without any upload bandwidth limitation and with a maximum upload bandwidth of 30% the available capacity of 1Mbps. Such a bandwidth limitation is application-based as data are injected into the TCP layer and further to the network at a rate which is not higher than the maximum allowed rate. The Maximum Transmission Unit (MTU) is 1500 bytes on each link.



**Figure 1 - Network topology**

Two different kinds of multimedia traffic are simulated: audio and video. The audio traffic is modelled with UDP flows between the mobile hosts H1 and H2. A crude model is used to simulate an audio conversation where H1 and H2 talk to each other in turns, sending packets at 64 kbps for time duration of 10s. The audio conversation starts at  $t = 20s$  and ends at  $t = 80s$ . The video traffic consists of one unidirectional UDP flow between EH and H1 and another one between EH and H2 at 600 Kbps. The two flows are transmitted simultaneously. They start at  $t = 20s$  and ends at  $t = 80s$ . The packet size is 576 bytes for both audio and video.

In order to study the impact of Differentiated Services (DS) based scheduling on improving performance of the UDP traffic, tests without and with DS are run. In case of DS is used, a basic priority queue scheme (Huston, 2000) for UDP packets is applied on both BSs and on R1 (not in the mobile hosts as the user is not expected to install and configure any QoS mechanism) so that whenever an UDP packet reaches a DS-enabled node (BS, R1), it is forwarded immediately to the next hop before any TCP packets. We run tests with different router buffer sizes on the BS nodes in order to have a broader view on the phenomena

involved in the modelled network environment and not focusing only to a specific network configuration. Router buffers sizes of the two BSs are set to 8, 32,128 packets for both uplink (from BS to R1) and downlink (from BS to mobile hosts) directions. The buffer sizes in the mobile hosts and in router R1 are fixed to 64 packets, which is a value large enough so that these links are not the bottlenecks of the path.

#### 4. Results

Figure 2 shows the aggregate throughput of all four TCP flows running between H1 and EH (uplink) and from EH to H1 (downlink) for the various uplink and downlink buffer size combinations on the BS. TCP performance with an audio conversation and video broadcast is shown in Figure 2 a) and Figure 2 b), respectively. The TCP performance with H2 is not reported as the results are similar to the results with H1. For each transfer direction (downlink and uplink), the results for four test cases are shown: Baseline with no DiffServ nor uplink bandwidth limitation, DS with DiffServ, BW 30% with a maximum upload bandwidth limit of  $0.3 \times 1\text{Mbps}$ , BW 30% and DS with both bandwidth limitation and DiffServ.

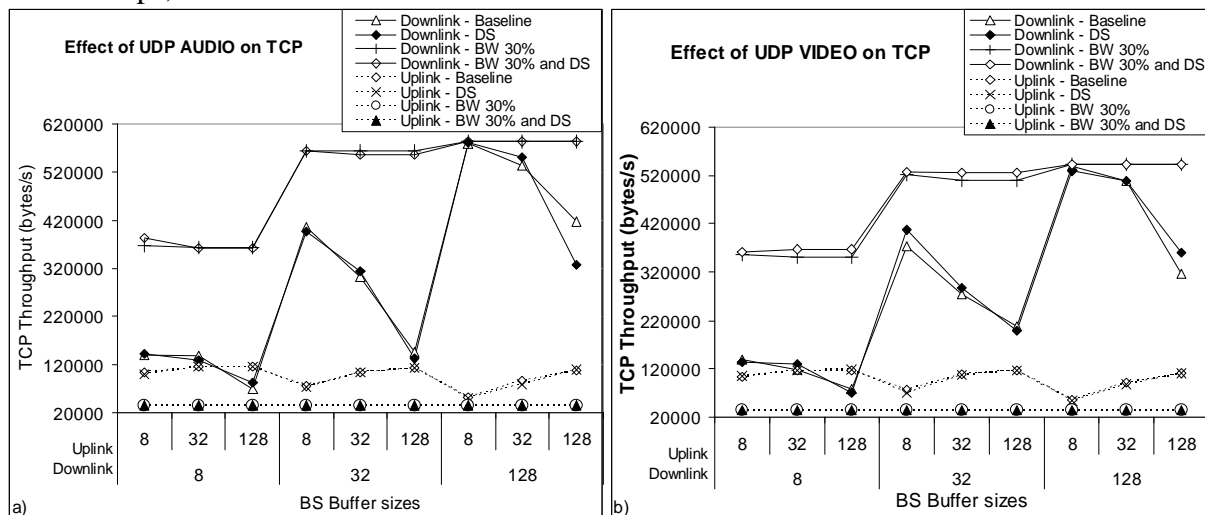


Figure 2 – TCP Performance in uplink and downlink for audio (a) and video (b)

We first analyze the results with the audio traffic for the Baseline case (see Figure 2a). The TCP throughput in the uplink direction (label: Uplink – Baseline) increases with larger uplink buffer sizes. In the case with the downlink buffer size set to 128 and the uplink buffer size enlarged from 8 to 128 packets the throughput increases from 51 Kbytes/sec to 105 Kbytes/sec. The reason is that TCP needs a certain amount of buffering capacity in order to inflate its congestion window and take fully advantage of the available link capacity. The BS buffering capacity with 8 packets is clearly less than the end-to-end bandwidth-delay product (27 Kbytes). Therefore, when the BS buffer is exhausted resulting in a congestion-related packet drop, the TCP sender halves its sending rate and continues at rate lower than the available bottleneck link capacity would allow.

TCP throughput in the downlink direction (label: Downlink – Baseline) increases with larger downlink buffer size as expected. However, the throughput is adversely affected by the uplink router buffer size on BS: the throughput decreases as the uplink buffer size increases. The reason for this is outlined as follows. At  $t = 0$ , all TCP transfers start both in downlink and

uplink direction. Uplink TCP flows inflate their congestion window exponentially during the initial slow start in order to probe the available link capacity (until the uplink buffer becomes full). Downlink TCP flows behave in the same way. The TCP flows in the uplink direction cause congestion in the BS uplink buffer, affecting delivery of the TCP Acks for the downlink transfers. Such congestion causes 1) Delays on arrival of Acks to the TCP sender in the downlink direction and 2) Ack compression (Zhang et al, 1991) (Mogul, 1992).

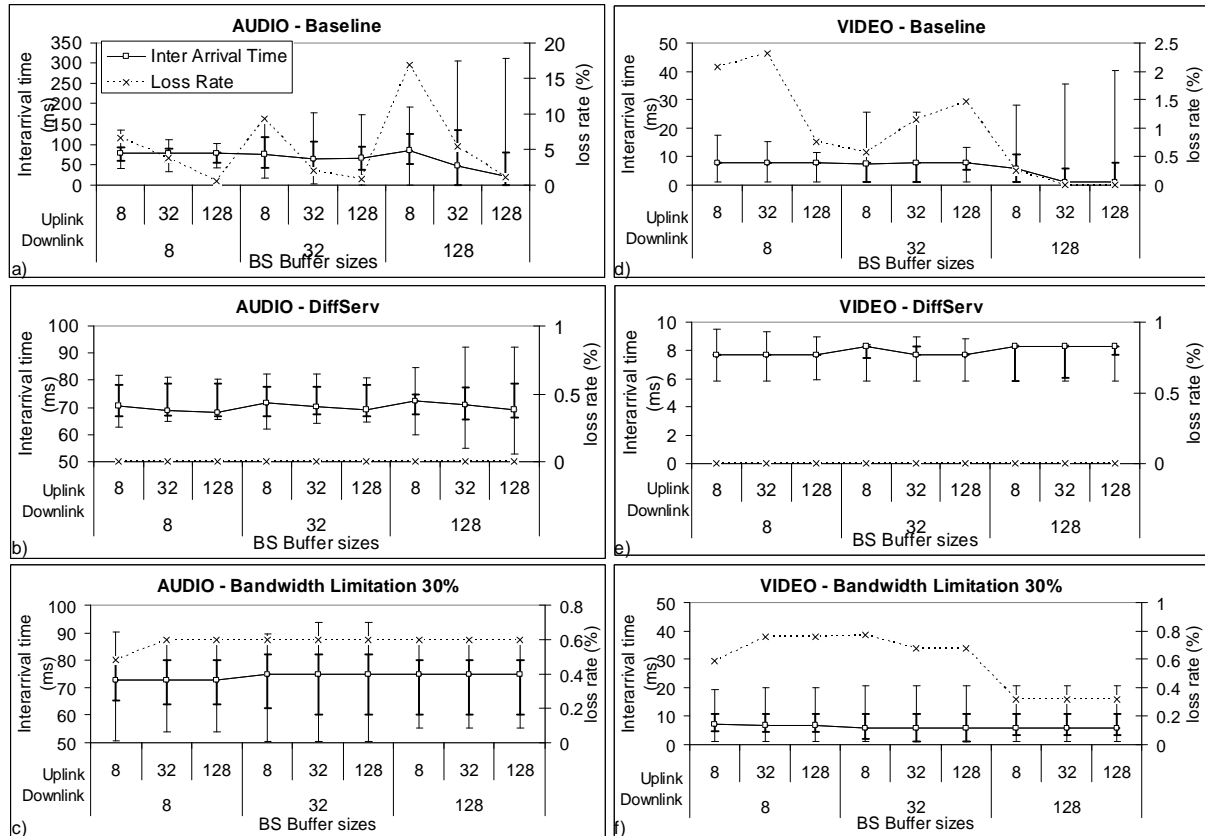
The first phenomenon affects TCP recovery capabilities as it increases the queuing delay on the Ack path and thereby the round-trip time (RTT). Therefore, detecting a packet loss is delayed and the recovery from the loss as well as resuming the previous transmission rate takes much longer with the increased RTT since TCP NewReno is able to recover only one packet per RTT during the recovery phase. With a BS downlink buffer size of 128 packets, increasing the BS uplink buffer from 8 to 128 packets increases the average RTT by 60%. Same situation occurs with the other BS downlink buffer sizes (8, 32), but there TCP performance is already limited due to too small downlink router buffer. In addition, an excessively delayed Ack may cause a spurious Retransmission Timeout (RTO). A spurious RTO occurs when the RTT suddenly increases, to the extent that it exceeds the retransmission timer that had been determined a priori (Ludwig and Katz, 2000). This results in unnecessary packet retransmissions and congestion window reduction (Sarolahti et al, 2003). The TCP traces show that several spurious RTOs per flow occur, especially with smaller downlink buffers. Most of the observed spurious RTOs occur during the fast recovery phase, where a segment is sent again although it was correctly received, enforcing the TCP sender to enter slow-start. In wireless networks, a spurious RTO may also occur during normal transmission of packets due to a sudden delay in the data or Ack path. Algorithms such as F-RTO (Sarolahti et al, 2003) and Eifel (Ludwig and Katz, 2000) are suggested to mitigate the problem.

The second phenomenon, Ack compression, occurs as the delayed Acks allow later Acks to catch up with the slower ones, resulting in reception of a burst of Acks back-to-back at the TCP sender. This is harmful to TCP performance as it increases the burstiness of TCP. In normal conditions (no recovery phase), each incoming Ack triggers a new packet transmission and allows inflating the congestion window. If many Acks are received back-to-back, a burst of packets is injected into the network (some TCP versions are able to mitigate this problem); such a burst may easily fill up router buffers along the path and cause congestion losses and, consequently, a dramatic decrease of the congestion window, resulting in reduced TCP throughput. We observed this when the BS uplink buffer is increased. With a BS downlink buffer of 128 packets, enlarging the BS uplink buffer from 8 to 128 packets doubles the downlink TCP data loss rate (from 0.3 % to 0,6 %). With smaller downlink buffer, this phenomenon is more severe and further reduces performance as it causes a higher packet loss rate: with a BS downlink buffer of 32 packets, enlarging the BS uplink buffer from 8 to 128 increases the downlink TCP data loss rate from 0.5 % to 1 %.

Decreasing the uplink buffer size tends to increase Ack losses. This may cause reduced TCP recovery capabilities and TCP throughput. Lost Acks typically result in stretch Acks (Acks that cover more than 2 segments of previously unacknowledged data (Allman et al, 1999) and may also increase the TCP sender burstiness (Balakrishnan et al, 2002). In case of a BS downlink buffer of 128 packets, we observed that the Ack loss rate is 15.7 % with a BS uplink buffer of 8 and decreases down to 0.9% with the BS uplink buffer set to 128. The higher Ack loss rate

with the small uplink buffer sizes does not, however, cause significant problems for the downlink TCP transfers in our test cases.

Reducing the amount of TCP traffic in the uplink direction significantly improves TCP performance for downlink TCP transfers. This can be seen in Figure 2a) by comparing the throughput in downlink baseline case and the case with limited P2P upload rate.



**Figure 3 - Inter Arrival Time and Loss Rate for audio (a, b, c) and video flows (d, e, f).**

Next we analyze the impact of coexisting TCP traffic on delay sensitive audio and video traffic. Figure 3 shows the Inter Packet Arrival Time and Loss Rate for audio and video flows for various configurations. Figure 3 a), b), and c) are related to audio traffic; Figure 3 d), e), and f) to video traffic. Five indices on the inter-arrival time are shown: median, 25th and 75th percentile, 5th and 95th percentile. Bars depict the inter-quartile range (where 50% of samples are included) and the range 5-95 percentile (where 90% of samples are included). Figure 3a) (baseline case) shows that the performance of audio flows is strongly affected by network congestion. The inter-arrival time dramatically oscillates from the expected value (72 ms), being more than 4 times the expected value in the worst case. The loss rate strongly depends on the BS buffer size and is high especially in case of small buffer size as the presence of TCP background traffic causes network congestion. Reducing the TCP traffic and thereby congestion in the uplink direction improves UDP flow performance (reduced jitter and loss rate) as Figure 3c) shows, but leaves it at quite unacceptable level.

In order to further reduce inter-arrival time variations and loss rate, the DiffServ approach with prioritized UDP traffic significantly reduces loss rate and jitter. Figure 3b) shows that in case

of DS-based prioritization the inter-arrival times are much more stable and closer to the expected value than in the other cases. Remaining variations are due to the fact that DS is not applied in the mobile hosts H1 and H2. The loss rate is zero for all buffer sizes. Figure 2a) shows that using DS with audio traffic does not have notable impact on TCP performance as the amount of UDP traffic (64kbps) is not significant compared to the available link capacity. Furthermore, combining bandwidth limitation and DS approaches yields best overall performance, improving both TCP (Figure 2a), label “BW 30% and DS”) and UDP audio performance (statistics for the combination are similar to the case where DS only is used).

Figure 2b) shows TCP performance when video traffic is present in the network. Similar considerations as reported in case of audio traffic (Figure 2a) can be done. However, the influence of video traffic on TCP throughput is more visible compared to the corresponding case with audio traffic: TCP throughput is lower than in case of audio. As expected, the UDP flow, which is non-congestion-responsive, steals bandwidth from TCP and makes TCP to adjust its sending rate to the remaining link capacity (Floyd and Fall, 1997).

Performance of video traffic is depicted in Figure 3d), e), and f). With baseline configuration (Figure 3d), the inter-arrival time is sensitive to the downlink buffer size. In general, the larger the downlink buffer is the higher the inter-arrival time variation as the queuing delay increases due to the TCP background traffic. Using DS reduces jitter and reduces the loss rate down to zero with all buffer sizes (see Figure 3e). Using bandwidth limitation causes less jitter and reduces UDP loss rate compared to the baseline case as the lower congestion level in the uplink buffers makes downlink TCP senders less bursty (see Figure 3f). As for the audio case, using bandwidth limitation in conjunction with DS gives the best overall performance.

## 5. Concluding Remarks and Future Work

We have studied the coexistence of TCP-based P2P file sharing applications with UDP-based audio and video applications. We carried out experiments in an asymmetric wireless access network where two hosts communicating with each other in a voice conversation or receiving a video stream from the network are also downloading and uploading data using a P2P application. In general, TCP throughput increases as the bottleneck router buffer size in data direction increases as TCP needs a certain amount of buffering (capacity) to inflate its congestion window and fully utilize the available link capacity.

P2P applications try to utilize all available link capacity. This may significantly increase the level of network congestion. We examined the nature of problems caused by P2P traffic in downlink and uplink direction and showed that congestion in router buffers in upstream direction causes several problems for both TCP and UDP traffic. In particular, TCP performance in downlink direction is adversely affected by large router buffers in the uplink direction as this causes delays on arrival of TCP Acks and Ack compression.

Congestion in the uplink reduces performance of audio flows. The packet inter-arrival time may strongly fluctuate, being 4 times longer than the expected value in the worst case. The UDP loss rate strongly affects the UDP throughput. We showed that the loss rate is affected by the BS buffer size and is high especially in case of small buffer sizes. In general, performance of video traffic is sensitive to buffer sizes in the transfer direction. As expected,

large buffer size causes higher inter-arrival time variation as the queuing delay increases due to background TCP traffic. Results show that having lower congestion in the uplink is indirectly beneficial to downlink video flows as this makes downlink TCP senders less bursty.

In order to control the P2P bandwidth consumption and limit its impact on other applications, we evaluated the use of Differentiated Services (DS) in the access network and uplink bandwidth limitation for P2P. Using DS improves audio and video performance, but does not significantly affect TCP performance. As opposite, using bandwidth limitation mitigates the TCP problems due to congestion on the Ack path. It also slightly improves audio and video performance. For best performance, it is important to combine DS and bandwidth limitation.

In the future, we intend to use alternative wireless network models where link errors and possibly link-layer-based retransmissions are present in order to have a better understanding of the impact of the typical wireless link characteristics. The observed problems may exacerbate in presence of error-prone links and related link-layer retransmissions.

## 6. References

Allman, M., Paxson, V. and Stevens, W. (1999), "TCP Congestion Control", RFC 2581.

Balakrishnan, H., Padmanabhan, V.N., Fairhurst, G. and Sooriyabandara, M. (2002), "TCP Performance Implications of Network Path Asymmetry", RFC 3449.

BitTorrent Web Site (2005), <http://bitconjurer.org/BitTorrent/>, (Accessed 21 March 2005).

Blake, S., Black, D., Carlson, M., Davies, E., Wang, Z. and Weiss, W. (1998), "An Architecture for Differentiated Services", RFC 2475.

Floyd, S. and Fall, K. (1997), "Router Mechanisms to Support End-to-End Congestion Control", LBL Technical Report.

Huston, G. (2000), "Internet Performance Survival Guide: QoS strategies for Multiservice networks". John Wiley & Sons.

KaZaA Web Site (2005), <http://www.kazaa.com>, (Accessed 21 March 2005).

Ludwig, R. and Katz, R. H. (2000), "The Eifel Algorithm: Making TCP Robust Against Spurious Retransmissions". ACM Computer Communication Review, 30(1).

Mogul, C.J. (1992), "Observing TCP Dynamics in Real Networks", Proc. SIGCOMM '92 Symposium on Communications Architectures and Protocols.

Napster Web Site (2005), <http://www.napster.com>, (Accessed 21 March 2005).

Network Simulator 2 Web Site (2005), <http://www.isi.edu/nsnam/ns/> (Accessed 21 March 2005).

Sarolahti, P., Kojo, M. and Raatikainen, K. (2003), "F-RTO: An Enhanced Recovery Algorithm for TCP Retransmission Timeouts", ACM Computer Communication Review, 33(2).

Zhang, L., Shenker, S. and Clark, D.D. (1991), "Observations on the Dynamics of a Congestion Control Algorithm", Proc. SIGCOMM '91 Symposium on Communications Architectures and Protocols.