A Simulation Study of Packet Path Diversity for TCP File Transfer and Media Transport on the Internet

Eckehard G. Steinbach *[1], Yi J. Liang [2], and Bernd Girod [2]

[1] Institute of Communication Networks
Munich University of Technology
Steinbach@ei.tum.de

[2] Information Systems Laboratory
Department of Electrical Engineering
Stanford University

ABSTRACT

We study the performance of TCP-based media transport when combined with packet path diversity (PPD) in the Internet. We present a simple PPD-TCP transport scheme where the acknowledgments (ACKs) from the receiver are duplicated and sent across independent network paths. We also investigate a PPD-TCP scheme where in addition to path-diversified ACKs on the backward path, packet path diversity is also used on the forward path. In order to keep the additional network load small, only small probe packets that carry the same TCP header information but not the payload are sent across the alternative path. From the time difference between the arrival of the probe packet and the corresponding data packet at the receiver we derive the superior path and, if advantageous, switch the path for the data packets. For comparison purposes we also implement a PPD-TCP transmission scheme where both, acknowledgment and data packets, are duplicated and sent over different network paths. This scheme provides us with an upper bound of the performance improvements that we can obtain compared to a corresponding single-path TCP transmission. The different PPD-TCP schemes are compared to standard single-path TCP transmission using ns *network simulations. It is shown that all investigated PPD-TCP schemes greatly outperform standard single-path TCP transmission for time-critical file transfer and media delivery.*

1 INTRODUCTION

Media content delivery such as video streaming over the Internet either uses a guaranteed-delivery transport protocol such as TCP or a partially reliable protocol based on UDP and RTP combined with proprietary ARQ, flow control, and congestion control mechanisms [12],[13]. While UDP-based transport makes the transmission process more continuous, one advantage of TCP-based transport is that the receiver is guaranteed to successfully receive all media packets which is beneficial if the media file is either to be stored after playout or to be replayed several times locally. Modern video encoding schemes like MPEG-2, MPEG-4, or H.263 introduce strong dependencies between successive packets. Hence, continuous video playout can only be guaranteed if all packets are available for decoding. Another advantage of TCP-based media delivery systems is the ease of implementation since no additional flow control, congestion control, or ARQ mechanisms have to be implemented. A simple HTTP-based web server can therefore be used to deliver the encoded media content as for instance demonstrated by the early streaming media program VivoActive 1.0 [16]. Most importantly, however, TCP-based media delivery naturally leads to transmission friendliness with respect to other flows that are competing for the same network resources.

One reason why TCP is not often used for real-time or near real-time media delivery is the potentially substantial fluctuation in the delivery times for the fragments of the encoded data, which is partly due to the lack of control over the rate at which the server pushes data to the client. Another reason is that TCP congestion control actively reacts to network load variations with the intention to share resources in a fair manner with other flows. This can also lead to significantly varying instantaneous throughput.

In this work, we investigate the combination of the TCP transport protocol with path-diversified packet transmission in the Internet. We show that this approach overcomes some of the aforementioned limitations of traditional single-path TCP-based media transport. For transmission with packet path diversity we assume that packets can be sent along more than one network path between server and client. Without loss of generality, we assume in the following path diversity with two network paths. The diversity gain that can be observed depends on the correlation of congestion, delay, and loss events on the two paths. For uncorrelated paths, excessively delayed or dropped packets should occur independently.

In order to illustrate the benefit of path-diversified TCP-based media transport, let us consider, for instance, temporary congestion on the path from the client to the

*This work has been supported, in part, by a gift from Hewlett Packard, and, in part, by the Stanford Networking Research Center while the first author was at Stanford University.

server which may lead to dropped acknowledgment packets. If an acknowledgment is not received, TCP automatically retransmits the data segment after the retransmission timer expires and increases the time until the next retransmission attempt is scheduled. This reaction unnecessarily reduces throughput if the forward path is not affected by the congestion. Sending the acknowledgment information along two independent network paths increases the probability that the acknowledgment information is received by the server and therefore reduces undesired throughput variations.

This paper is organized as follows. The next section briefly discusses prior work on multi-path media transport. In Section 3 we describe three different implementations of TCP transport with packet path diversity. In Section 4 we discuss the establishment of independent network paths in the Internet. Next, in Section 5 we present ns simulations where we compare the performance of the proposed PPD-TCP variants to traditional single-path TCP transport for various network topologies and network loads.

2 RELATED WORK

Almost 30 years ago, transmission with path diversity was proposed for load balancing, the reduction of transmission times, and fault tolerance purposes in store and forward networks by Maxemchuck [11]. More recent examples of multi-path transmission include [5],[9],[15]. A literature survey on multi-path transmission can be found in [8].

For media transport, multi-path transmission of complementary descriptions of a video signal has been considered in [2],[7]. In [3] the concept of transmission with path-diversity is considered in the context of media content delivery networks where a client can stream simultaneously from more than one content server. Significantly reduced infrastructure requirements for multi-path content delivery in comparison to single-path delivery are reported in [3]. In [10] packet path diversity augmented UDP-transport has been used for real-time voice transmission over the Internet and significant reductions in end-to-end latency and effective packet loss are observed.

The combination of TCP transport with path diversity for media delivery has received relatively little attention. One reason for this might be that most real-time or near real-time media transport in today's Internet is UDP-based. Nevertheless, the transmission of media content using TCP as a transport protocol has various advantages as discussed in the previous section which makes it worthwhile to study this combination. Prior work on path-diversified TCP-transport includes [6] where non real-time data transport over multiple paths in the context of ad-hoc networks has been proposed. Two different approaches for path-diversified TCP-transport are discussed in [6]. In the first approach, the data file transfer is split into multiple independent TCP-connections at a so called Meta-TCP layer. Since individual TCP-windows are used for each TCP-connection, the receiver has to resequence the corresponding subflows in a resequencing buffer. The second approach assumes path-diversity on the IP-layer which means that individual IP packets can be routed along different paths and one TCP-window is used.

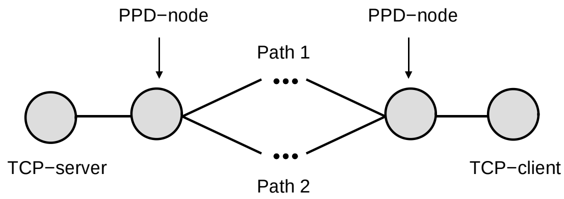

Figure 1: Packet path diversity with two PPD-nodes and two independent network paths.

3 TCP WITH PACKET PATH DIVERSITY

In this section we first discuss various general issues involved when combining TCP transport with packet path diversity. We then describe the three PPD-TCP variants investigated in this work.

Fig. 1 shows the general architecture for two-path PPD-TCP. From one PPD-node to the other, two network paths exist. We assume in the following that both PPD-nodes can control the path along which a packet has to travel. The reason for separating the TCP hosts and the PPD hosts in Fig. 1 is to illustrate that the TCP server and client implementations remain unmodified in our study. The brute force way of exploiting packet path diversity would be to duplicate all packets at the PPD-nodes and to send one copy across each path. The corresponding receiving PPD-node would then simply use the earlier copy and discard the second copy. This is, of course, a significant waste of network resources but will serve as an upper performance bound for PPD-TCP in our study. If the PPD-node is not the final destination of the TCP connection, i.e., the TCP host and the PPD host differ, the earlier copy would be forwarded to the corresponding TCP host. This ensures that the transmission looks like a normal TCP connection from the perspective of the TCP server and client. In this scenario we can think of the PPD-node as a path diversity enabled router that can duplicate packets and send those packets along two different network paths. Obviously, this functionality could also be implemented on the TCP hosts themselves. The PPD implementation would then interface the TCP protocol. In this case at least the last link would typically be shared by both paths. The advantage of moving the PPD-nodes away from the edge of the network is that the two paths can be established without sharing last-mile links. If the number of shared links is minimized, we expect the congestion and loss characteristics of the two paths to be less correlated. We now discuss the three PPD-TCP variants implemented and compared in this work.

3.1 TCP WITH PPD ACKS

In this version of PPD-TCP only the acknowledgments (ACKs) from the client to the server are duplicated and sent across two different network paths. The data packets from the server to the client travel on the default path determined by the routing tables. If both copies of the ACK arrive at the PPD-node that is close to the server, the copy that arrives first is forwarded to the server while the second copy is dropped. Discarding the second copy of the ACK is necessary to avoid the arrival of duplicate ACKs which would trigger the fast retransmission mode of TCP [14].

In single-path or standard TCP transport it can easily happen during congestion that acknowledgments are dropped by a router with a full buffer on their way to the server. The server has to assume that the corresponding data portion has not been successfully received by the client until a later ACK acknowledges this data fragment. With path diversity the probability that one of the two identical ACK packets will make it to the sender increases. The effect on the current throughput can be quite dramatic since TCP reacts to the lost ACKs by reducing its window size although the forward path might be of good quality. By path-diversifying the ACKs, TCP only has to handle congestion if it actually occurs on the forward path.

The overhead introduced by duplicating the acknowledgment packets is small. If we assume that for every 1500 Byte data packet two instead of one 40 Byte ACK packets are returned, we encounter an overhead of $(40)/(1500+40) = 2.6\%$.

3.2 PPD ACKs and PPD data packets

As mentioned at the beginning of Section 3, the maximum path diversity gain can be expected if copies of both, data and acknowledgment packets, are sent along two independent network paths. In Fig. 1 this would mean that the PPD-nodes duplicate the data packets from the TCP server to the client as well as the ACKs from the TCP client to the server. As before, the receiving PPD-nodes forward the copy that arrives earlier while discarding the later one. From the perspective of one single TCP flow this scheme will provide the maximum path diversity gain at the expense of penalizing all other flows competing for the same network resources. Obviously, this scheme results in sending twice as much data across the network (100% overhead). Another issue is that TCP congestion control will not react properly to congestion on the data path since transmission with one congested and one uncongested path will look like a single good path to TCP. As mentioned before we investigate this version of PPD-TCP only for the sake of comparison and use it as an upper performance bound. Please note that this form of TCP-PPD could be used to prioritize single TCP flows. This, however, is not the focus of this paper.

3.3 PPD ACKs and PPD probe packets

A more practical version of PPD-TCP that exploits path diversity in both directions generates a small probe packet for each data packet and sends data and probe packets along different network paths. In the following we refer to the path currently being used for data packets as the *active path*, while the other path used for probe packets is refered to as the *inactive path*. The probe packets are used to test the current transmission quality of the inactive path. Probe packets are small packets that carry just enough information for the PPD-node to uniquely identify a TCP data packet and its corresponding probe packet. Hence, the overhead introduced by packet path diversity remains small. At the receiving PPD-node we record the arrival time of both, data and corresponding probe packet, and use the time difference between the two to infer which path is currently better. In case the active data path is inferior to the inactive path we signal the other PPD-node to switch the paths and to send the data packets from now on along the inactive path which then becomes the active path. Assuming that the size of a probe or ACK packet is 40 Byte, the overhead we encounter for this version of PPD-TCP is $(40 + 40)/(1500 + 40) = 5.2\%$.

An interesting issue that arises in this context is how to decide when to switch the two paths. We have implemented a simple scheme that compares the quality of the two paths based on the arrival times of the probe and the corresponding data packet at the PPD node. In case a data packet arrives before the probe packet we assume the active path to be superior. If a probe packet arrives before we receive the corresponding data packet, we start a timer of duration ΔT. After this timer expires we check if meanwhile the corresponding data packet has been received. If yes, again the active path is assumed to be better. If no, we assume that the path that carries the probe packet is better and signal the PPD-node to switch paths. The probe packets are discarded as soon as they have served their purpose. The TCP data packets are always forwarded by the PPD-node to the TCP client.

ΔT is the additional delay that we would observe for a data packet if it were sent along the inactive path instead of a probe packet. This *virtual delay* for a data packet on the inactive path has to be inferred from the measured delay of the probe packet. The delay difference ΔT is mainly caused by the different transmission times of packets of different sizes.

In order to estimate ΔT, we send a packet pair along the alternative path at the start of a transmission session. The packet pair consists of a probe packet immediately followed by a data packet. The difference of the arrival times of the two packets is measured, which is the estimate of ΔT. During normal operation, only probe packets are sent over the inactive channel and the virtual arrival time of the data packets is computed by adding our most recently measured value of ΔT to the arrival time of the probe packet. This virtual delay can now be compared to the measured delay of the data packet that is sent along the active channel.

The switching between the paths so far is based on only one packet pair. This is obviously the most simple scheme. In future work we plan to regularly estimate ΔT during the media file transfer and to keep a short history of these measurements and base our decision to switch paths for instance on a majority vote of these recorded values. In this way, path changes caused by single packets that are extremely delayed or dropped and do not reflect the actual current load level on the paths can be avoided.

4 Internet Packet Path Diversity

In order to maximize the benefits of path diversity we have to select paths that exhibit largely uncorrelated

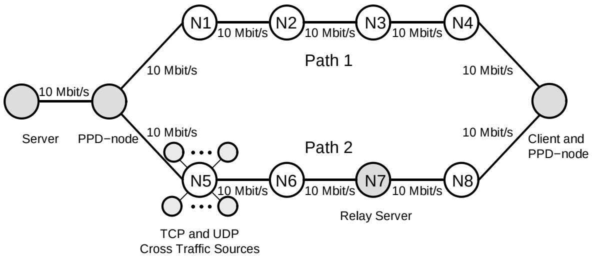

Figure 2: Multi-hop topology for *ns* network simulations. Each of the intermediate nodes N1 through N8 has a number of TCP data sources and UDP traffic sources attached (only shown for node N5).

transmission characteristics. With today's Internet protocols, the path a packet takes across the Internet is a function of its source and destination IP addresses as well as the entries of the routing tables involved. Selecting a specific path for a packet is largely unsupported in today's infrastructure.

As discussed in [2], IPv4 source routing is usually turned off within the Internet for security reasons. It would be more promising to implement path diversity by means of an overlay network that consists of relay nodes [2],[1]. Here, packets can be sent along different routes by encapsulating them into IP packets that have the addresses of different relay nodes as their destination. At the relay nodes, packets are forwarded to other relay nodes such that the packets travel along as few common links as possible.

With the next-generation IP protocol IPv6, the source node has a great amount of control over each packet's route. IPv6's loose source routing (LSR) allows packets to be sent via specified intermediate nodes. If widely implemented, this source routing feature of IPv6 will make future implementation of transmission with packet path diversity even simpler.

5 SIMULATION RESULTS

In order to investigate the performance of TCP with packet path diversity we have implemented the proposed PPD-TCP schemes as part of the *ns-2* network simulator[1]. The first path that is used in our simulations is the default path determined by the routing algorithms available in *ns*. The second path is explicitly specified by means of a relay node. The relay node works bidirectionally and forwards TCP packets from the server to the client and vice versa. The PPD-sender node that wishes to send a packet along the second path simply addresses the relay server as the destination. The relay server receives and forwards the packet without any delay to the PPD receiver node.

Fig. 2 shows the first network topology under investigation. The first path follows the nodes N1 to N4 which is the default path, while the second path is along nodes N5 to N8 with N7 being the relay server. Each of these nodes has 20 data sources attached, with a large amount of TCP and UDP traffic heading for different destinations and competing for the network resources. We use the log-normal model proposed in [4] for the file size of each TCP session. UDP traffic is randomly switched on and off at each source. In our simulations we control the path load by varying the idle time between successive TCP and UDP transmissions. We measure the network path load as the ratio of average data throughput to the rate of the link. In Fig. 2 all links have 10Mbit/s transmission rate and introduce 20ms propagation delay.

5.1 COMPARISON OF PPD-TCP VARIANTS

Fig. 3 shows the average time required to transmit a file of size 100Kbyte from the TCP server to the TCP client node in Fig. 2 as a function of network path load. The TCP data packet size is 1000Byte. The cross traffic is adjusted such that on average both paths have the same load. We record the time when a session starts and the time when the entire file has been successfully transmitted. The transmission times shown in Fig. 3 are averaged times over 1500 file transfers. In the following we name the different transport schemes as follows:

TCP	standard single-path TCP transport
PPD-TCP1	two-path TCP transport with duplicate ACKs (Section 3.1)
PPD-TCP2	two-path TCP transport with duplicate ACKs and duplicate data packets (Section 3.2)
PPD-TCP3	two-path TCP transport with duplicate ACKs and probe packets on the forward channel (Section 3.3)

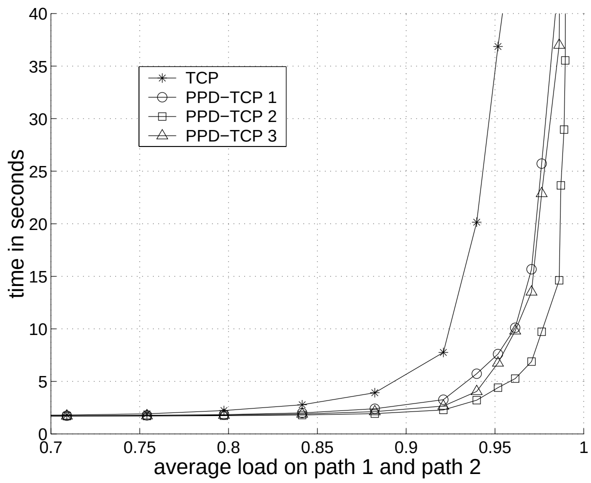

Figure 3: Average transmission time of a 100Kbyte media file versus network path load for the network topology in Fig. 2.

As the average network load increases, the time to transmit the file increases significantly for single-path TCP transport. As can be seen from Fig. 3 all three PPD-TCP variants significantly outperform single-path TCP transport for high path load. The largest gain, as expected, is obtained for the PPD-TCP scheme that duplicates both, data and acknowledgment packets. The performance of PPD-TCP1 and PPD-TCP3 for this simulation setup is very similar. This can be explained by the fact that both schemes exploit the advantage of having duplicate ACKs on the path from the client to the server. Congestion-dependent switching between the two

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

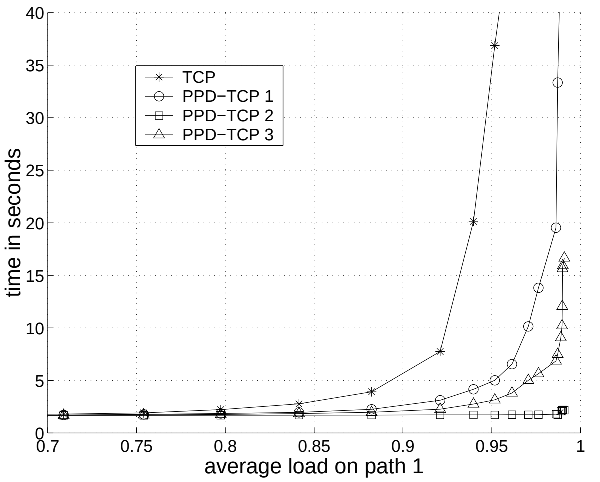

Figure 4: Transmission time versus network path load. Path 2 exhibits 20% lower average load than path 1.

forward paths, however, gives only a small advantage since both paths are equally loaded. The transmission time for our file at an average path load of 95% is reduced from around 35s for single-path TCP transport to about 7s for duplicated ACKs (PPD-TCP1), around 6s for duplicated acknowledgments in combination with probe packets (PPD-TCP3), and 4.5s for our upper bound scheme PPD-TCP2.

We expect even better performance of the PPD-TCP schemes if the two paths exhibit different average load. Fig. 4 shows the same simulation as in Fig 3 with the difference that the average load on Path 2 in Fig. 2 is 20% lower compared to Path 1. If we assume that the default path for single-path TCP transport corresponds to Path 1, we obtain the same curve as in Fig. 3. For all three PPD-TCP schemes, however, we see different results than in Fig. 3. For PPD-TCP 2, where data packets are duplicated, we observe a dramatic decrease in transmission time for the file at high loads on path 1. This is expected since the TCP server sees a much better connection (Path 2) to the client compared to the standard single-path TCP case. It is interesting to note in Fig. 4 that the use of probe packets (PPD-TCP3) turns out to be very beneficial. While the curves for PPD-TCP1 and PPD-TCP3 in Fig. 3 were almost identical, the PPD-TCP scheme with probe packets shows significant advantages here. This can be explained as follows. Even if we start sending the data packets along Path 1, the better transmission characteristics of Path 2 lead to an earlier arrival of the probe packets which then triggers switching of the active and the inactive data path. Therefore, most of the time, the data packets are transmitted along the path with the lower average load (Path 2) which leads to a performance improvement for our media file transmission.

For media delivery, the total transmission time of a media file is only one indicator of the improvements that are obtainable when combining TCP transport with packet path diversity. For continuous playout of media content at the receiver, uniform throughput is equally important. In the next experiment we investigate the throughput variation observed for the various PPD-TCP schemes in comparison to single-path TCP transmission. For this, we measure the size of the continuously playable media file at the receiver as a function of time. Only those packet numbers up to the first missing packed are counted since missing packets lead to interruption of playout due to the interdependency of successive packets.

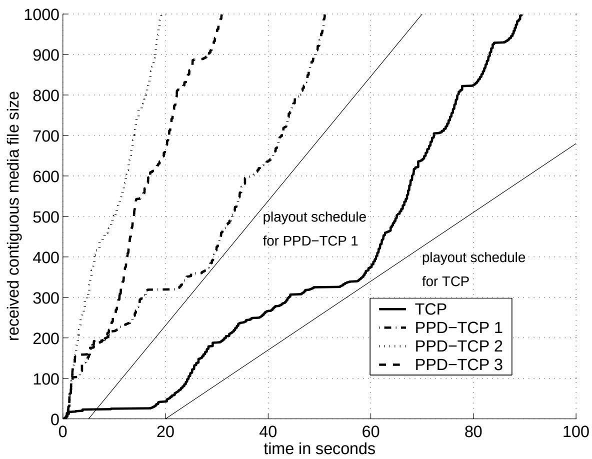

Figure 5: Playable media file size in packets at the receiver for average loads of 0.94 on Path1 and 0.94 on Path 2.

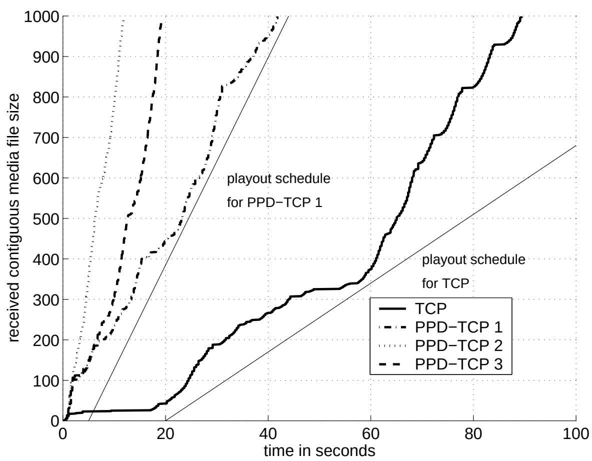

Figure 6: Playable media file size in packets at the receiver for average loads of 0.94 on path1 and 0.85 on Path 2.

Fig. 5 shows the playable file size as a function of time for identical load (94%) on Path 1 and Path 2. The total file size corresponds to 1000 TCP data packets each of size 1000 Byte. Single-path TCP performs very badly at this load level while the PPD-TCP variants lead to a significant improvement of the situation. Especially the probe packet based approach PPD-TCP3 comes very close to the duplicate data packet approach PPD-TCP2 which is very encouraging since the overhead introduced for this scheme is small. From the perspective of a media playout algorithm at the receiver, single-path TCP transmission requires a significant amount of pre-buffering in order to guarantee continuous playout during media file transfer. For the PPD-TCP schemes, the required pre-buffering is much smaller and therefore playout can start earlier and will be more robust with respect to receiver buffer underflow. We show possible playout schedules that avoid playout interruption at the receiver for single-path TCP and PPD-TCP1 in Fig. 5 as thin solid lines. The received file size has to stay above that line in order to guarantee continuous playout. The point where this line intersects the x-axis represents the time we have to pre-buffer before playout can start and the slope of these lines represent the maximum consumption rate our playout algorithm uses at the receiver. The steeper the line the higher this rate which translates into potentially higher quality of the media content. While 20 seconds pre-buffering is required for the single-path transmission only 5s pre-buffering is required for PPD-TCP1. Fig. 6 shows the same experiment as in Fig. 5 but for an average load level of 94% for Path 1 and 85% for Path 2. For this unbalanced case we observe a further significant improvement in throughput as can be seen for instance by the steeper playout deadline for PPD-TCP1.

So far we have considered packet path diversity examples where the two paths were completely independent. Now we investigate the performance of our proposed multi-path TCP transport schemes when the two paths share links. Naturally, shared links occur if we have to implement the PPD-functionality on the TCP host and if only a single link is available to access the network. Shared links will also be present if the two paths have to share a link due to a missing alternative (e.g., a cross-Atlantic link). In this case, however, the shared link can be assumed to be of high speed. It is therefore unlikely that this shared link represents the bottleneck link of the two paths that introduces significant correlation. Fig. 7 shows the topology that is used for the following experiments. The PPD-functionality is implemented on the TCP server host which has a single connection to the network. The two paths share a 20Mbit/s link between nodes N1 and N2 which introduces a small correlation between the two paths. The correlation is small since the bottleneck for each path are the 10Mbit/s links that are not shared. As before, we vary the TCP and UDP cross traffic to simulate different levels of network load. The number of TCP and UDP cross traffic sources attached to nodes N1 and N2 is now twice as large as before with half the cross traffic heading to and coming from Path 1. The other half heads towards and comes from Path 2. This way we can still run our simulations with equal average network load on both paths.

Fig. 8 shows the average transmission time of the media file as a function of network load for the shared link topology in Fig. 7. It can be seen from Fig. 8 that for the shared path example with low correlation we again observe a significant improvement for path diversified TCP transport.

When we increase the correlation between the two paths, we expect the gains for PPD-TCP to become smaller. This is illustrated in Fig. 9 where we repeat the previous experiment with the difference that the shared link now only has a link speed of 15Mbit/s. The number of TCP and UDP sources that produce cross traffic and their scheduling remain fixed. The link between nodes N1 and N2 now becomes the common bottleneck link

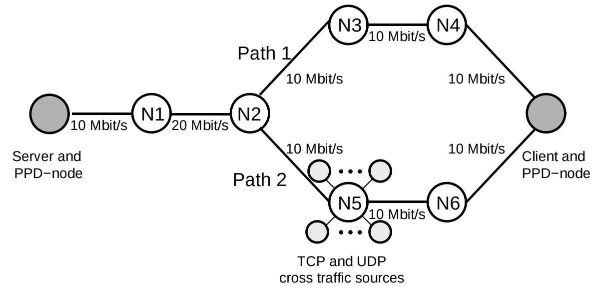

Figure 7: Multi-hop topology with shared link for ns network simulations. Each of the nodes N1 through N6 has a number of TCP data sources and UDP traffic sources attached (only shown for node N5).

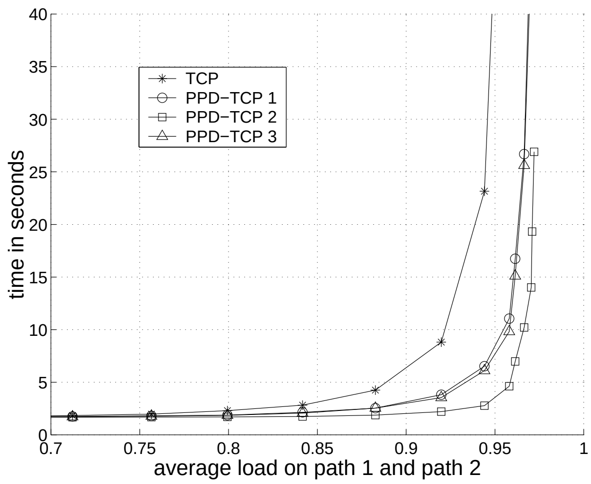

Figure 8: Average duration of a 100Kbyte media file transfer versus network path load. Both paths exhibit identical average network load. The load for Path 1 is measured between nodes N2 and N3 while the load for Path 2 is measured between nodes N2 and N5.

for the two paths. Fig. 9 shows the average transmission time of the 100Kbyte media file as a function of network load for this case. It can be seen from Fig. 9 that for the shared path example with high correlation we observe much smaller improvements for our PPD-TCP schemes.

5.2 Estimation of ΔT

In this section we provide simulation results for the measurement of the time difference ΔT that is used in Section 3.3 for determining the virtual arrival time of a data packet given that the corresponding probe packet has been received. In the first experiment we consider the symmetric network topology in Fig. 2 with all nodes being connected with 10Mbit/s links. We send a packet pair (probe packet first and data packet second) along the inactive path and measure ΔT for different network loads. Table 1 lists the average ΔT for different loads.

Since ΔT is mainly the difference of transmission times at zero network load, we can compute the time dif-

Table 1: Delay difference between the data and probe packet for the path with 10Mbit/s links.

Load	0	0.67	0.75	0.84	0.95	0.99
ΔT (ms)	4.6	3.6	3.5	3.4	3.3	3.1

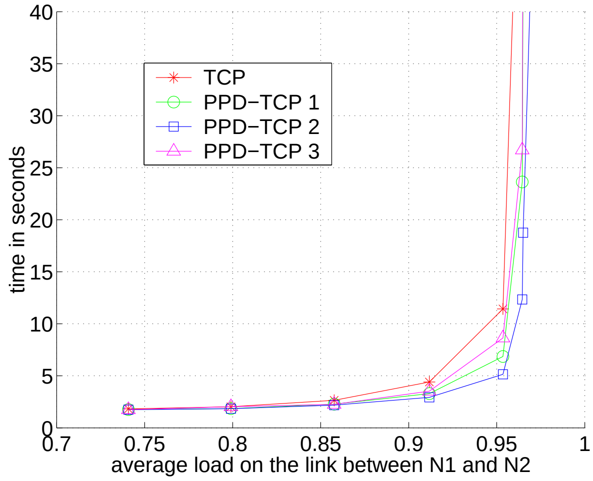

Figure 9: Average transmission time of a 100Kbyte media file versus network path load for the shared link topology in Fig. 7 with a 15Mbit/s link between nodes N1 and N2. The load is measured between nodes N1 and N2.

ference also analytically if we know the link transmission rates and the number of hops as

$$\Delta T = \frac{S_{data} - S_{probe}}{R_{link}} \times N_{hops} \quad (1)$$
$$= \frac{(1000-40) \times 8bits}{10^7 bits/s} \times 6 = 4.6ms.$$

where S_{data} and S_{probe} are the packet sizes of the data and the probe packet, respectively. N_{hops} is the number of hops for the path (6 in Fig. 2) and R_{link} is the transmission rate on the links.

In practice we typically do not know the number of hops and the individual link rates and therefore have to measure ΔT regularly. At higher load, ΔT decreases since the first packet in a packet pair is more likely to be slowed down due to the traffic in front of it. However, this decrease is less than 2ms even at very high load.

Although the delay difference is very small for the path with 10Mbit/s links, it can be much larger for links with lower bandwidth where packets take longer to transmit. We also measure ΔT as a function of network load for a network topology where the second path in Fig. 2 has links of 1Mbit/s instead of 10Mbit/s. The measured results are listed in Table 2.

The analytical value for ΔT in (2) now becomes 46ms. Since ΔT is large in this simulation setup, the virtual delay of a data packet is much greater than the actual delay of the probe packet, which means we cannot directly compare the delay of the probe packet on the inactive path to that of the data packet on the active path. For example, if the end-to-end latency of the probe packets on the inactive path is much smaller due to its lower propagation delay, it does not necessarily mean that the inactive path

Table 2: Delay difference between the data and probe packet for the path with 1Mbit/s links.

Load	0	0.68	0.77	0.85	0.94	0.99
ΔT (ms)	46.1	35.5	35.0	33.3	33.7	32.6

is superior. If the transmission rate is very limited on that path, the delay a larger data packet would experience can be significantly higher.

In practice, we use a threshold of twice ΔT in determining whether to switch paths or not. This alleviates the problem that ΔT is smaller than it should be at higher load. Therefore we avoid frequent switching between paths unless the monitored channel condition for the inactive path is significantly superior.

In the next experiment, we study how effective the estimation of the virtual delay of the data packet is using the packet pair technique. We compare the estimated virtual delay of the data packet to actually measured values. The latter are obtained by repeating the simulation under exactly the same conditions, but data packets are sent over the inactive path instead of the probe packets. The estimated and the actual value are plotted in Fig. 10 for the two different network topologies used before. The estimated and the actually measured end-to-end values agree closely.

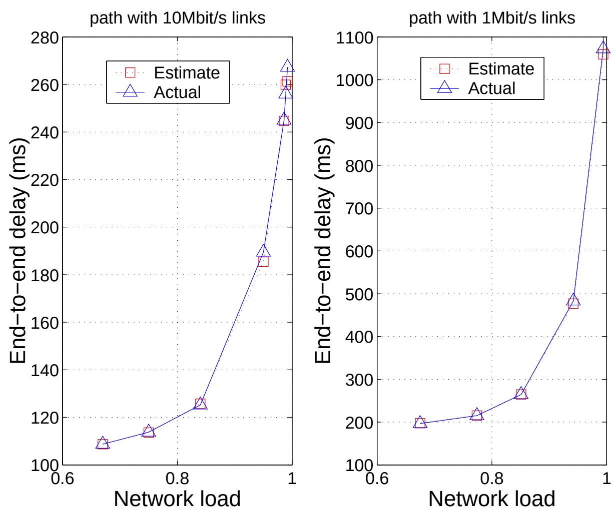

Figure 10: Estimated and actual end-to-end delays of data packets.

6 Conclusions

We study TCP-based file transfer and media transport in the Internet in combination with packet path diversity. The TCP protocol implementation remains unmodified in our study. We propose two different versions of packet path diversified TCP transport where in the first scheme only the acknowledgment packets from the TCP client to the sender are duplicated and sent along different paths. In the second scheme, probe packets are sent on the alternative path from the TCP server to the client in addition to the duplicated acknowledgment packets sent from the TCP client to the server. For this scheme we propose a simple strategy that is used to decide when to switch between the active and the inactive data path. This strategy is based on regularly measuring the arrival times of corresponding data and probe packets. Since the acknowledgment packets and probe packets are much smaller than the data packets, only a small overhead is introduced.

For comparison, we have implemented a third scheme where both, data and acknowledgment packets are path-diversified.

Our simulations show that the proposed PPD-TCP schemes lead to greatly improved time-critical file transfer and delivery of media content at high network load if the correlation between the two network paths is low. For the case of shared links that introduce correlation between the path congestion we still observe noticeable improvements. Only for a shared bottleneck link the diversity gains become small. For the maximum diversity gain it is important to select the two network paths such that the correlation between the two paths is minimized. This not necessarily means that the two paths should not have links in common but it is important that the bottleneck links are not shared by the two paths.

References

[1] D. G. Andersen, H. Balakrishnan, M. F. Kaashoek, and R. Morris, "The case for resilient overlay networks," Proc. of the 8th Annual Workshop on Hot Topics in Operating Systems (HotOS-VIII), May 2001.

[2] J. G. Apostolopoulos, "Reliable video communication over lossy packet networks using multiple state encoding and path diversity," SPIE Visual Communication and Image Processing, pp. 392-409, Jan. 2001.

[3] J. G. Apostolopoulos, T. Wong, W. Tan, and S. Wee, "On Multiple Description Streaming with Content Delivery Networks," IEEE INFOCOM '02, Jun. 2002.

[4] M. Arlitt and T. Jin, "Workload characterization study of the 1998 World Cup web site," IEEE Network, vol. 14, no. 3, pp. 30-37, May 2000.

[5] A. Banerjea, "Simulation study of the capacity effects of dispersity routing for fault-tolerant real-time channels," ACM SIGCOMM '96, vol. 26, no. 4, pp. 194-205, Oct. 1996.

[6] N. Gogate and S.S. Panwar, "Supporting video/image applications in a mobile multihop radio environment using route diversity," Proc. Int. Conf. Communications, pp. 833-844, San Jose, California, Jan., 1999.

[7] N. Gogate, D. Chung, S.S. Panwar, and Y. Wang, "Supporting image/video applications in a mobile multihop radio environment using route diversity and multiple description coding," IEEE Trans. on CSVT, to appear.

[8] E. Gustafsson and G. Karlsson, "A Literature Survey of Traffic Dispersion," IEEE Network Magazine, vol. 11, no. 2, pp. 28-36, March/April 1997.

[9] R. Krishnan and J.A. Silvester, "Choice of Allocation Granularity in Multipath Source Routing Schemes," IEEE INFOCOM '93, pp. 322-329, March 1993.

[10] Y. Liang, E. Steinbach, and B. Girod, "Real-time Voice Communication over the Internet Using Packet Path Diversity," ACM Multimedia '01, Ottawa, Canada, Oct. 2001.

[11] N.F. Maxemchuk, "Dispersity Routing in Store and Forward Networks," Ph.D. thesis, University of Pennsylvania, May 1975.

[12] Microsoft, Windows Media Player software and documentation

`http://www.windowsmedia.com`

[13] Real Networks, RealPlayer software and documentation

`http://www.realnetworks.com`

[14] W. R. Stevens, "TCP/IP Illustrated, Volume 1: The Protocols," Addison-Wesley, 1994, ISBN 0-201-63346-9.

[15] H. Suzuki and F.A. Tobagi, "Fast bandwidth reservation scheme with multi-link and multi-path routing in ATM networks", IEEE INFOCOM '92. pp. 2233-2240, 1992.

[16] Vivo Software, VivoActive software and documentation

`http://www.vivo.com`