Performance simulation of buffer bloat in routers

Bert Gijsbers, Deepthi Devaki Akkoorath

Faculty of Science, Universiteit van Amsterdam

Abstract

Due to the dramatic price drop of memory network devices are configured with much larger buffer sizes than before in order to prevent packet drop. This undermines the assumptions of the TCP congestion avoidance algorithm which depends on packet loss in order to detect and prevent network congestion. We investigate the effects of this buffer bloat phenomenon on throughput and latency. Our simulations show that increasing buffer sizes reduces throughput slightly, but sharply increases latency. When the link is saturated then large buffer sizes cause dramatic reduction of throughput rates for applications which transmit data only periodically. In addition we show that TCP is in fact remarkably resilient to packet drop and can tolerate very small buffer sizes very well at the benefit of optimal latency.

1. Introduction

Recently a discussion has been started on performance problems of the Internet [Getty][Cring]. The authors observe poor throughput and high latency when they use high-bandwidth applications from home to servers located on the Internet. Their reasoning is that due to the dramatic price drop of memory network equipment is routinely configured with very large memory buffers. One effect of this is that queues can now grow very long causing high latency. Another effect is that TCP algorithms may no longer work appropriately, causing or aggravating network congestion. The authors coin the term “Buffer bloat” for this problem.

TCP uses a sliding window protocol to reliably transfer data from sender to receiver [RFC793]. The receiver indicates to the sender how much data it is willing to receive by communicating the size of a receive window. The sender buffers data until it has received acknowledgment from the receiver. To avoid overloading the network data transmission is further limited by a congestion control algorithm.

TCP congestion control works as follows [RFC5681]. It has a slow start method, where the initial rate of sending data is low. It increases the speed of sending data until a packet drop is detected. TCP increases and decreases the speed of data transmission according to the packet loss to get an optimum data rate which does not cause congestion. Thus TCP relies on the timely detection of packet loss in order to prevent congestion and achieve an optimal transmission rate.

The amount of unacknowledged data TCP can send is limited by the receiver window and the congestion window. Clearly there is no justification for the sender to transmit faster than the rate of the slowest link in the connection as this will only cause congestion. The desired maximum for the amount of unacknowledged data for a TCP sender is therefore the product of the rate of the slowest link times the round-trip time. This is commonly called the bandwidth-delay product. The purpose of the congestion

avoidance algorithm can therefore be phrased as to detect the current minimum of all maximum possible transmission rates of all links along the connection path.

The transmission rate of a sender can thus be calculated as:

minimum(receive window,congestion window)/round-trip time

When an outgoing link of a router becomes congested, but nevertheless the router queues newly arriving packets for this link instead of dropping them then TCP cannot properly detect congestion. TCP may even further increase its transmission rate according to the slow start algorithm. In this situation the round-trip time has already increased. This in turn adversely affects latency sensitive applications. When finally the router buffers become exhausted the router has to drop packets. At this point TCP will detect acknowledgment timeouts and start retransmissions. When TCP senders simultaneously start retransmissions this may even add to the congestion [SS97].

When there is a sudden increase in round-trip time TCP will detect this as an acknowledgement timeout, conclude there is network congestion and reduce its transmission rate. However, it is not completely clear how TCP reacts to a situation where a queue of an outgoing link in a router is slowly increasing in length. In such a situation the TCP acknowledgement timeout will not go off. TCP may detect the slow increase in round-trip time, adjust its round-trip estimate and use longer round-trip timeouts. Effectively behaving as if there is no congestion. An interesting idea here is to let TCP tune its window size to approach a minimum round-trip time [CdLnet].

TCP segments are frequently sent in bursts: sequences of segments transmitted with little or no intermediate delay. When such a burst arrives at a bottleneck queueing will result potentially causing congestion. It is known that chances for congestion would be reduced if the sender would pace the transmission at the end-to-end throughput rate using a leaky bucket algorithm. To do this for high transmission rates would incur a lot interrupt overhead [Ant03]. However, the sender could split a large burst into several smaller ones, thus largely avoiding this extra interrupt overhead.

Until now we have only talked about the sender's role in causing or avoiding congestion, but the receiver can also play a role. The receiver advertises a receive window. If the receiver has knowledge about the round-trip time and the bandwidth then it could potentially help prevent the sender from sending at too fast a rate by limiting the size of the receive window. This is what Linux does by means of the tcp_moderate_rcvbuf configuration variable. The receiver could also help reduce the burstiness at the sender by moderating the rate at which it opens the receive window.

How much memory does a router need? The common rule-of-thumb is that routers need buffer size $B = RTT \times C$, where C is the capacity of the link [Vil94]. Appenzeller et all have investigated this question in [App04] and concluded that backbone routers with 10,000 flows need no more than $B = (RTT \times C) / \sqrt{n}$ memory where n is the number of simultaneous flows. This is a reduction of 99%. Their simulations show that using this formula even short bursty flows with 14 or less packets complete about 20-30% faster. However, various authors caution against these conclusions, e.g. [Dha06].

In this paper we analyse how bandwidth, latency and packet drop rate is affected by various buffer configurations in networking equipment.

2. Related work

The problems introduced by endlessly increasing buffer space in routers was first examined in [Nag87]. By analyzing what happens if routers were equipped with infinite memory Nagle showed that throughput approaches zero.

Home connections to the Internet frequently have an asymmetrical download/upload capacity especially with ADSL. [Bro05] analyzes the effects of this asymmetry on TCP and makes recommendations for optimal buffer sizes and traffic shaping.

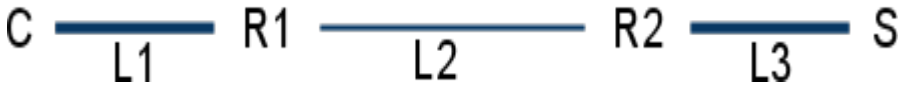
Analysis of the effects of buffer size in routers can be found in [Ant03] and [CdLnet]. These papers also consider the desirability of shaping TCP traffic in order to eliminate bursts.

3. Simulation

We performed simulations of TCP traffic in Omnet++/INET. Omnet++ is a discrete event simulation framework for queueing and network simulations. The INET module provides implementations of the TCP/IP protocols for the Omnet++ framework. Analysis of the shortcomings of the TCP module of INET can be found in [Ids04] and [Res10] which also discuss improvements which have been incorporated in the current version of INET.

4. Experiments

The network model we adopted for the experiment is the following:



C sends data to S using a TCP connection via routers R1 and R2. The links L1, L2 and L3 have bandwidth B1, B2 and B3 respectively and propagation delay D1, D2 and D3 respectively. For our experiments we assume B1 = B3 and B1 >= B2. The propagation delay D1 is assumed to be small and D2 to be a large value to simulate a longer link. Router R1 and R2 have at each outgoing link a queue with buffer size M. If B1 is equal to B2 then R1 can forward data at the incoming rate and M will remain unused. If B1 is greater than B2 then a bottleneck occurs in R1 in the direction of L2. In that case the hypothesis to be tested is that R1 will adversely affect the performance characteristics of the TCP protocol which is used by S and C as soon as M becomes larger than needed. To show this we want to measure the incoming data rate of the TCP application at S and also measure the latencies between S and C.

For the reverse direction of S to C only small packets with acknowledgements flow. We assume that these are much smaller than the data packets and that for our purposes they can be ignored.

In the following sections, we explain different experiments we did using Omnet and discuss their results.

4.1 Experiment 1 : Effect of Buffer size on Goodput

In experiment 1 we set up a network in Omnet++/INET based on the above model. The client and server are connected via two routers as shown in figure 1. The client is connected to the router through a high bandwidth channel. The router to router connection is through a low bandwidth channel to simulate the congestion bottleneck at the first router.

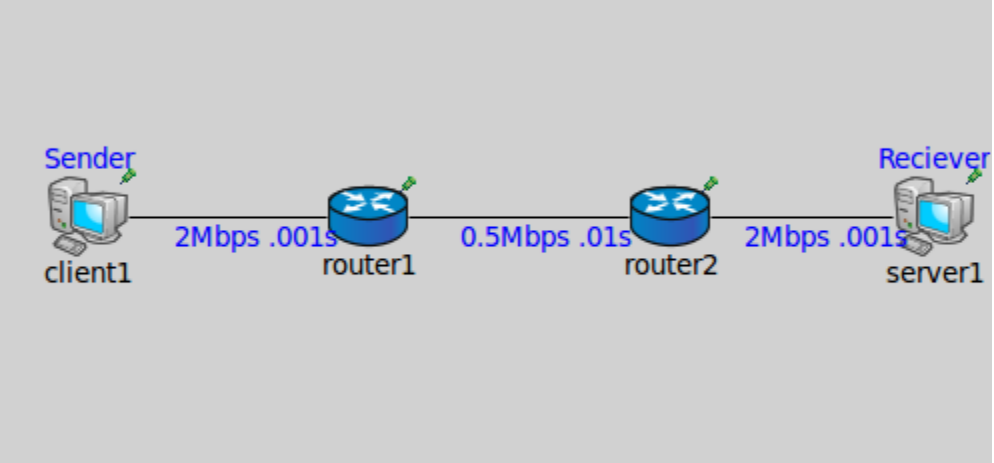


Fig.1: Network Model for simulation

The values for the different network parameters are as follows:

B1 = B3 = 2 Mbps

B2 = 0.5 Mbps

D1 = D3 = 0.001s

D2 = 0.01s

In the rest of the experiment, we have used these values unless otherwise stated. The shortest possible round-trip time of this configuration is 0.024 seconds. In all experiments we used a TCP maximum segment size of 536 bytes.

This experiment shows how TCP performance is affected by large buffers. The client runs a TCP application which sends a large amount of data. This application creates heavy traffic in the network and causes congestion at the router. It transfers 1000MB of data to the server. We measured the rate at which the server S is receiving packets from the client. The experiment was run for 1000 seconds. The following table shows the average goodput at the server.

Table 1: B2 = 0.5 Mbps

Buffer Size (No. of Frames)	Goodput at Receiver (Bytes/s)	Average RTT(seconds)	Packets dropped
10	54722	0.088	31168
50	53375	0.232	2300
100	53282	0.448	3643
500	53181	1.95	711

The results show a slight decrease of goodput with higher buffer size. Remarkable here is that a reduction of packets dropped doesn't improve the goodput, but even reduces it slightly.

We repeated this experiment with a different Router1 - Router2 channel speed. The channel bandwidth B2 was changed to 0.1 Mbps. This makes the network more congested.

Table 2: B2 = 0.1 Mbps

Buffer Size (frames)	Goodput at Receiver (Bytes/s)	Average RTT(seconds)	Packets Dropped
10	10968	0.0324	5969
50	10953	1.517	8659
100	10695	2.304	2406
500	10631	8.078	0

The results show that the receiving speed is slightly decreasing with increase in buffer size. Even though the drop in performance is not significantly large, this shows the effect of large buffers on the performance of TCP applications.

We also measured the packet drops at the congested router, for different Buffer Size. Larger Buffer size was designed to reduce packet drops. For smaller buffer sizes, even though there is larger number of packet drops, the throughput is slightly higher. From the tables, we can also see that the packet drops is not strictly decreasing with increasing buffer size. In table 1, packet drop for Buffer size 100 is greater than that for Buffer size 50. So we cannot say that, larger buffer sizes would help to reduce packet drops in all situations. In our case, the higher packet drops for buffer size 100 can be explained by TCP congestion control mechanism. Since TCP does not receive timely feedback on congestion, it continues sending data at higher rates. This might cause more congestion and hence more packet loss.

With a single TCP connection we could observe network congestion only when we enabled TCP window scaling and set the maximum advertised window size to a large value of 100MB. This is because TCP

flow control itself limits the rate of data transmission. So it is not able to generate enough traffic to fill the buffer. With large advertised windows TCP is able to send data at higher rate and fill the buffer at the router. The difference in the goodput at receiver was observed only when the simulation was run for a long time.

4.2 Experiment 2 : Effect of Buffer Size on Latency

In the next experiment, we measured how latency is influenced by different buffer sizes.To measure the latency a client-server pair was added to the network in experiment 1. Client 1 is the same as in the previous experiment which runs the same TCP application. The new client, Client 2 is connected to Router 1. A ping application is running on the client2. This application sends a ping request every 10 ms to the Server 2 which is connected to Router 2. The round-trip time for ping requests, which is the time interval between sending the ping request and receiving the ping reply at the client, has been recorded. It was observed that the RTT is increasing with increase in buffer size. The RTT observed over the period for different buffer sizes is plotted and shown in the Fig.2. The ping application can be considered as a representative of an interactive application. The RTT observed denotes the potential latency experienced by an interactive application. This shows how much a latency sensitive application is affected by increase in buffer size.

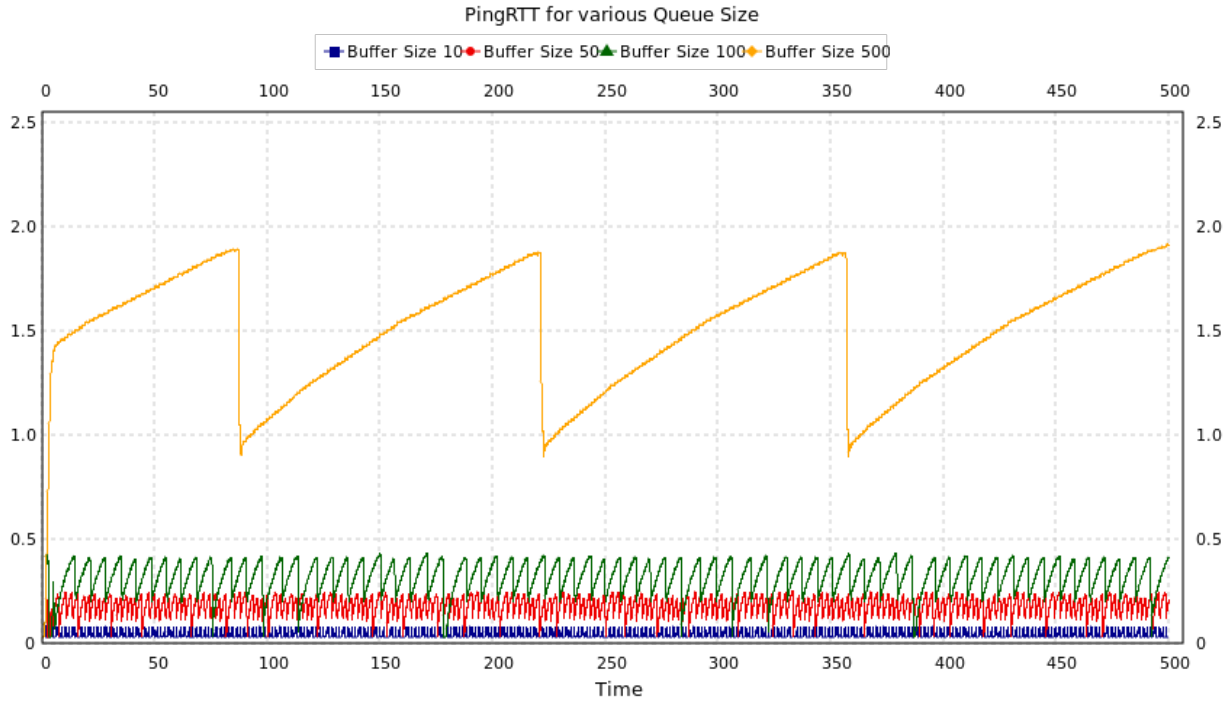


Fig.2: PingRTT for different Buffer Size

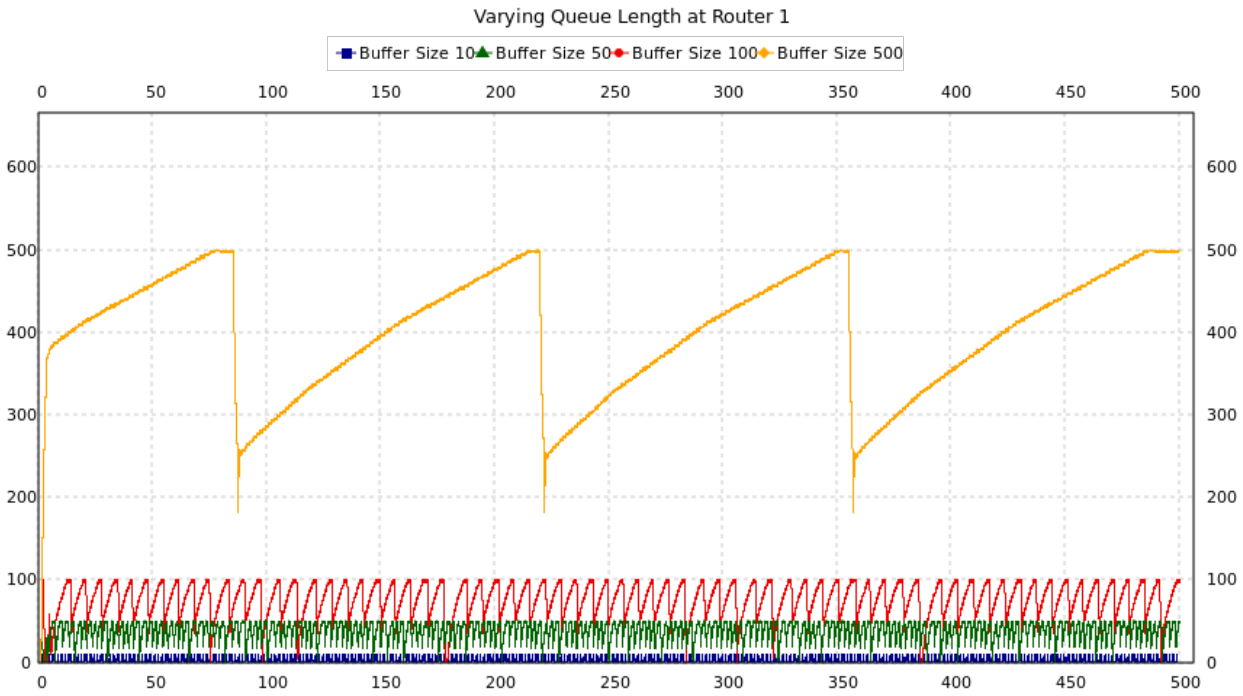


Fig.3: Variation of Queue length over time

Fig.3 shows the plot for queue length. It was observed that the maximum RTT is experienced when queue is full. Hence larger buffers near a congestion point aids to increase the latency of packets.

4.3 Experiment 3: Effect of Buffer Size on two TCP applications

In experiment 3 ,the network set up is the same as in the previous experiment. There are two client-server pairs. Client 1 runs a TCP application similar to previous exeriment. Client 2 runs a TCP application which sends data to Server 2 at a much lower rate than client 1. This application sends 200KB of data every 10 seconds. As in previous experiments, the data rate at the receiver was observed. Table 3 shows the results of this experiment.

From the results we see that the performance of the low traffic generating client worsens when the buffer size increases. When the buffer size is larger then client 1 is able to push its data to the queue since it is sending data at higher rate. This causes the packets from client 2 to be appended at the end of the large queue in the router buffer most of the times. Due to the long waiting time at the buffer queue client 2 experiences latency on all the packets it sends. Thus the effective goodput at the receiver is much lower when the router buffer is large and bloated.

Table 3: Goodput for two TCP applications running simultaneously

Buffer Size (No. of Frames)	Goodput at Server 1 (bytes/s)	Goodput at Server 2 (bytes/s)	Change in Goodput at Server 2 compared to Buffer size 10 (in %)
10	34796	17800	0
25	36214	17800	0
50	35784	17743	-0.32
100	35964	17143	-3.7
250	43933	9196	-48.3
500	47969	5139	-71.1

4.4 Experiment 4: Effect of TCP receive window

Because in previous experiments we set the advertised window size to the large value of 100 MB it is relevant to examine the effects of changing this parameter. In this experiment we observed the resultant average values for throughput, round-trip time and queue length. We used the network with one client, two routers and one server from experiment 1 with C2 configured at 1 Mbps. We fixed the buffer size in the router at 100 frames. Each measurement lasted 180 seconds.

Table. 4: Effect of receive window size on performance

advert.window	throughput Kbit/s	RTT in sec	packet drops	avg. queue length
300	90	0.028	0	0.5
1000	135	0.028	0	0.5
3000	676	0.031	0	0.5
4000	946	0.033	0	0.5
5000	987	0.043	0	1.5
7000	987	0.059	0	5.5
10000	987	0.082	0	10.5
30000	987	0.254	0	47.6
100000	984	0.381	31	76.0

We see in this table that an advertised receiver window of around 4 KB is achieving most of the throughput while keeping the latency low. Increasing this value to 5 KB maximizes the throughput, but latency increases by 30%. Increasing the advertised receiver window further only increases the latency. From this table one can deduce that the receiver can improve the latency by limiting the advertised window. If there is only one TCP flow then a large router buffer doesn't need to affect latency as long as the receiver is careful in choosing an optimal advertised receiver window.

4.5 Experiment 5: Relation of buffer size, packet drops and performance.

In experiment 5 we want to more closely analyse the effects of buffer size and packet drops on performance. We used the same network model as in experiment 4, but now we fixed the advertised receiver window at the high value of 100 KB and changed the value for the buffer size in the first router in order to observe the average queue length and the number of dropped packets. The buffer size is the maximum number of frames that can be buffered in one direction. Each measurement lasted 180 seconds and transmitted up to 38500 TCP frames from sender to receiver.

Table 5: Relation between buffer size, packet drop and performance

buffer size	throughput Kbit/s	RTT in sec	packet drops	avg.queue length
1	796	0.030	790	0.5
2	880	0.033	725	1.0
3	900	0.035	575	1.5
6	977	0.045	473	3.5
12	984	0.067	261	8
25	982	0.113	132	18
50	979	0.204	100	37
100	984	0.381	31	76

In this table we see that we can severely reduce the router buffer without losing much throughput. There is a throughput-latency optimum between 3 and 6 queue buffers. The large number of packet drops don't affect the throughput much. In the table the highest percentage of dropped sender TCP packets is 2% for a buffer size of 1 frame.

5. Conclusions

In this paper we investigate how TCP throughput and latency are affected by router buffer size. Our experiments show that by increasing buffer size throughput is slightly reduced and latency is sharply increased. When the router buffer size increases then one high-throughput application can severely reduce throughput rates for applications which only transmit periodically.

Our experiments also show that TCP is remarkably resilient to very small router buffer sizes and to packet drop by routers. By configuring small buffer sizes latency can be substantially improved at a slight cost of throughput.

As a corollary of our experiments we show that a receiver can overcome the adverse effects of large router buffers by advertising an optimal receive window. Finding such a value is in principle just as hard as detecting an optimal congestion window size, but the receiver may have additional information, like for instance the speed of its access link or a history of previous connections.

Our findings suggest that manufacturers of networking devices should carefully tune the buffer size of their products. Or at least make the maximum queue length configurable. If this is neglected then latency and goodput for interactive applications will quickly degrade. Because the use of interactive network applications has dramatically increased over the past decade it is now mandatory to sell, buy and use equipment with optimal latency characteristics.

6. Acknowledgements

We would like to thank Mihai Cristea for introducing us to network simulation in Omnet++/INET and for his help in overcoming initial pitfalls. Cees de Laat gave us a first hand introduction into high-speed networking and the effects of router memory configuration on performance. Rudolf Strijkers was always willing to enter a lively discussion on problems we were facing. We are grateful to Paola Grosso for her inspirational lectures on diverse topics in advanced networking. Last but not least we like to thank the authors of Omnet++ and INET and all their contributors for creating and maintaining such powerful open-source software.

7. References

[Ant03] A. Antony e.a., "Microscopic Examination of TCP flows over transatlantic Links", Future Generation Computer Systems, Volume 19, Issue 6, August 2003, Pages 1017-1029.

[App04] G. Appenzeller e.a., “Sizing router buffers”, Comput. Commun. Rev., vol. 34, no. 4, pp. 281-292, 2004.

[Brou05] J.D. Brouer, “Optimization of TCP/IP Traffic Across Shared ADSL”, master thesis, available from http://www.adsl-optimizer.dk/thesis/; Internet; accessed 23 feb 2011.

[CdLnet] C. de Laat, “Networks & Buffers”, available from http://ext.delaat.net/netbuf/; Internet; accessed 15 feb 2011.

[Cring] R. Cringeley, “2011 predictions: One word - bufferbloat. Or is that two words?”, available from

http://www.cringely.com/2011/01/2011-predictions-one-word-bufferbloat-or-is-that-two-words/; Internet; accessed 21 feb 2011.

[Dha06] A. Dhamdhere e.a., “Open issues in router buffer sizing”, ACM Sigcomm Computer Communication Review, 36(1):87-92, 2006.

[Getty] J. Gettys, “The criminal mastermind: bufferbloat!”, available from

http://gettys.wordpress.com/2010/12/03/introducing-the-criminal-mastermind-bufferbloat/ ; Internet; accessed 21 feb 2011.

[Ids04] J. Idserda, “TCP/IP modelling in Omnet++”, available from http://www.utwente.nl/ewi/dacs/assignments/completed/bachelor/reports/B-assignment_Idserda.pdf; Internet; accessed 24 feb 2011.

[Nag87] J. Nagle, “On Packet Switches with Infinite Storage”, IEEE Transactions on Communications, vol. COM-35, pp. 435-438, April 1987.

[RFC793] J. Postel, "Transmission Control Protocol", STD 7, RFC793, September 1981.

[RFC5681] M. Allman e.a., “TCP Congestion Control”, available from http://tools.ietf.org/html/rfc5681; Internet; accessed 21 feb 2011.

[Res10] T. Reschka e.a., “Enhancement of the TCP module in the OMNeT++/INET framework”, Proceedings of the 3rd International ICST Conference on Simulation Tools and Techniques, ICST, Brussels, Belgium, 2010.

[SS97] H. Sawashima e.a., “Characteristics of UDP packet loss: Effect of tcp traffic”, Proceedings of INET ’97: The Seventh Annual Conference of the Internet Society, Kuala Lumpur, Malaysia, June 1997.

[Vil94] C.Villamizar and C. Song, “High performance tcp in ansnet”, ACM Comp. Comm. Rev., 24(5):45-60, 1994.