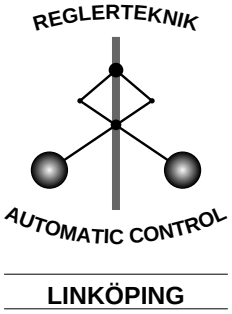
TCP Performance based on Queue Occupation

Frida Gunnarsson, Fredrik Gunnarsson, Fredrik Gustafsson

Division of Communication Systems
Department of Electrical Engineering
Linköpings universitet, SE-581 83 Linköping, Sweden
WWW: http://www.comsys.isy.liu.se
Email: frida@isy.liu.se, fred@isy.liu.se
fredrik@isy.liu.se

26th March 2001



Report No.: LiTH-ISY-R-2340
Submitted to Nordic Radio Symposium 2001

Abstract

The main protocol for flow and congestion control on the Internet is the Transmission Control Protocol, TCP. This protocol was constructed and developed based on heuristic arguments, and its main purpose is to prevent network congestion. Because of changes in the character of the Internet traffic, TCP does not work as well as when it was designed - a problem that has been addressed by researchers in different ways. In this contribution, the bad performance of TCP will be illustrated using queue occupation. Simulations have been made using a simple simulation testbed implemented in the Matlab tools Simulink and StateFlow, as well as a more complex environment in a simulator developed at Berkeley called ns-2. It is seen that a modified TCP implementation yields significant performance improvements.

Keywords: TCP, queue occupation, feedback

TCP Performance based on Queue Occupation

Frida Gunnarsson, Fredrik Gunnarsson, Fredrik Gustafsson*
Division of Control and Communications
Department of Electrical Engineering
SE-581 83 LINKÖPING, SWEDEN
{frida,fred,fredrik}@isy.liu.se

ABSTRACT

The main protocol for flow and congestion control on the Internet is the Transmission Control Protocol, TCP. This protocol was constructed and developed based on heuristic arguments, and its main purpose is to prevent network congestion. Because of changes in the character of the Internet traffic, TCP does not work as well as when it was designed - a problem that has been addressed by researchers in different ways. In this contribution, the bad performance of TCP will be illustrated using queue occupation. Simulations have been made using a simple simulation testbed implemented in the Matlab tools Simulink and StateFlow, as well as a more complex environment in a simulator developed at Berkeley called ns-2. It is seen that a modified TCP implementation yields significant performance improvements.

I INTRODUCTION

The Internet is a vast communication system, which makes the used control algorithms/protocols interesting. The main control protocol on the Internet is the transmission control protocol, TCP. Its main purpose is to control the data flow and to prevent congestion. The success of the Internet has caused the use of TCP and IP to spread. Research shows that TCP does not work as well as it did when it was invented. There are proposals for different solutions but they all affect the way that TCP is implemented. However, new applications might imply the use of a new set of routers, which allows modified TCP implementations. One such an example is the transport network between base stations in cellular phone systems, where the use of IP is considered.

We will start in Section II by a short introduction to TCP's control algorithm and also discuss a solution that adds explicit feedback to this algorithm. In Section III we will show the simulations made and discuss the outcome and finally, in Section IV, draw some conclusions and discuss further work.

*This work is supported by the graduate school ECSEL (Excellence Center in Computer Science and Engineering in Linöping), Linköpings universitet, which is acknowledged.

II TRANSMISSION CONTROL PROTOCOL

The control algorithm in TCP is window based. A control window limits the amount of data that is sent each instant. During data transmissions the control window changes in size depending on detection of packet losses and throughput. The control window is increased every time a packet gets through and decreased every time a packet is lost. This gives an adaption to the state of the network and the algorithm is designed to estimate the available bandwidth. There are two ways for packet loss detection:

- The sender expects an acknowledgment of received data from the receiver. The acknowledgment contains the number of the packet that the receiver is expecting. If three duplicates of an acknowledgment arrives, the packet with that number is assumed to be lost and is thus retransmitted. This method of detection is called *three duplicate acks*. Three duplicate acks are assumed to indicate a single packet loss and the control window is decreased.

- When a packet is sent a timer is set and if no receipt for that packet is received for a certain time, the packet is retransmitted along with the rest of the packets sent after that one. This is called a *timeout*, and it is supposed to indicate that the connection has been idle for a longer time due to a major congestion. Therefor, the control window is set to its initial value.

To prevent a fast sender from overflowing a slow receiver, the packets and receipts also contain the capacity of the sender, called the Advertised Window. The resulting window is the minimum of the control window and the advertised window subtracted by the packets that are sent but not yet acknowledged. Figure 1 illustrates this behavior when the advertised window is smaller than the congestion window. The control algorithm gives an implicit feedback when something goes wrong. For more details on TCP's control algorithm, see Stevens [6].

Explicit Feedback Control

As an alternative, Mascolo [5] gives TCP explicit feedback using a solution based on a Smith predictor, see for example Glad and Ljung [2]. The resulting algorithm is shown to be easily interpreted into existing TCP variables and to require only a few changes.

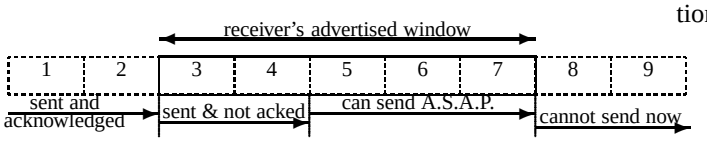


Figure 1: *TCP's sliding window. The sender has sent four packets, and received acknowledgements for the two first. The size of the advertised window allows for three more packets to be sent, until another acknowledgement is received.*

This is done by modeling the network using delays and queues. A controller is designed using a Smith predictor with proportional control. The resulting control signal is proportional to the space left in the slowest queue subtracted by the packets that are sent but not yet acknowledged. This gives a modified algorithm using the capacity of the smallest queue instead of the capacity of the receiver. This requires that each router on the path stamps the header with its momentaneous capacity if it is smaller than the existing value. This window is called the Generalized Advertised Window as opposed to the standard Advertised Window.

There are also several other proposals made, and a lot of ongoing research. Gunnarsson [3], Section 2.4 and Appendix B, gives a summary on work in different areas concerning TCP and flow control on the Internet.

III SIMULATIONS

This section describes the simulations that have been made and discusses the results.

Ordinary TCP versus Feedback TCP

To compare the existing control algorithm in TCP and the algorithm using a Smith predictor a simulation environment has been implemented in Simulink and Stateflow, [1, 4]. The simulations were made over a single TCP connection. The network was implemented as three subsequent queues in each direction. The processing times in the queues where assumed to depend on the size of the packet and also included additive Gaussian noise to take into account outer variations. The details of the simulation testbed is further described in Gunnarsson [3].

Differences between the two algorithms were observed. For example, the occupation of the shortest and slowest queue looks very different. Ideally, this queue, the bottleneck queue, should never be empty (full utilization) and never be overflowed (no packet losses). In Figure 2, the histograms over the length of this queue for the two algorithms are shown. In the simulation environment the shortest queue can hold at most six packets. For comparison, if we consider a case where the queue experienced a constant incoming rate until it was filled and then a constant outgoing rate until it was empty, the outermost position will be occupied half the time of the other positions.

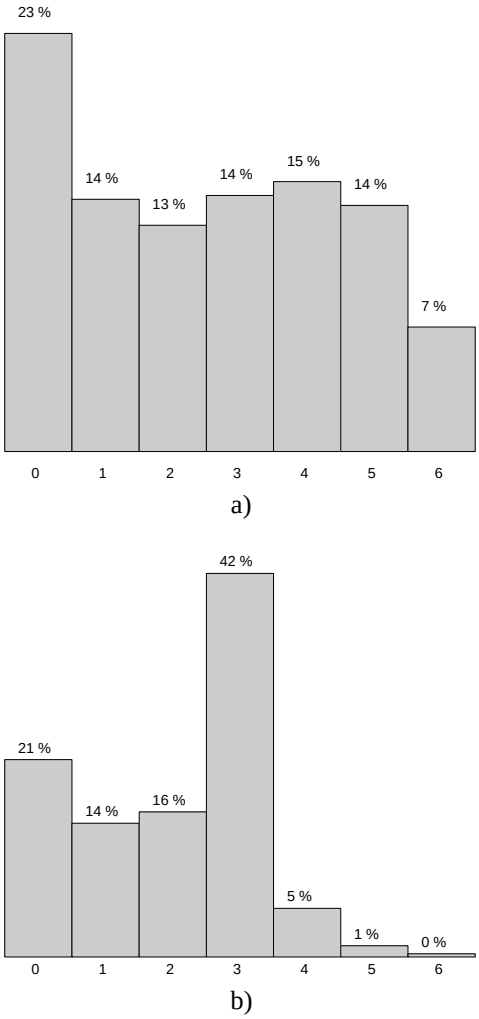


Figure 2: *The distribution of the bottleneck queue length when using* a) *ordinary TCP and* b) *TCP with a Smith predictor.*

For the existing TCP control algorithm we see that the queue is filled until a packet is lost and then no more packets are sent until it is empty again. This is a very oscillative behavior. On the other hand, for TCP with a Smith predictor, there is a specific peak in the histogram and the queue is almost never full, which would indicate much fewer packet losses. In fact, for this single connection simulations we saw no packet losses at all. This simulation study shows the benefits of the new feedback solution compared to the existing solution. It also shows some of the problems with TCP that has been discussed in other research.

TCP's behavior in a more complex environment

To investigate the reliability of the simulations in the previous section and the behavior of TCP for more complex networks the Network Simulator from Berkeley,*ns-2* [7], was used. Figure 3 shows the setup of the simulation. Flow 1 and flow 2 are carried over TCP connections through the

network. The connection for flow 1 is always up during the simulation and flow 2 is used as a disturbance. We studied the occupation of the queue in node 6, the bottleneck queue. Three different scenarios were studied:

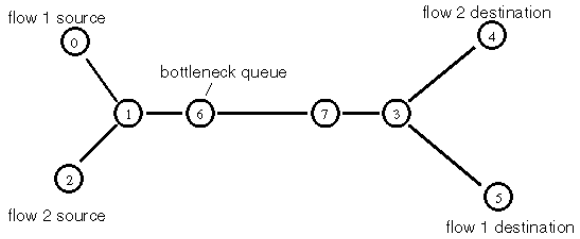


Figure 3: The network model.

- A single flow. This is the same scenario as for the Simulink case. Flow 2 is never running. The resulting histogram is shown in Figure 4. The result is not as flat as in Figure 2, and the differences depend on the simulation setup. In the Simulink model, the delay of the link was also modeled into the queues and this is not the case here. The qualitative behavior is the same. One striking thing is that the queue is empty for such a large amount of time. This seems to be a waste of capacity since there were always data to send.

- One flow with a disturbance. Flow 2 is running in small intervals and will then compete with flow 1. The result is shown in Figure 5. We see that this severely lowers the occupation of the queue, both the occupation by the main flow and the total occupation. This is a setup where a long transmission is disturbed by short ones. TCP clearly cannot handle this.

- Two competing flows. Flow 1 and 2 run parallel for a longer time. The resulting histograms are shown in Figure 6. We see that one of the flows occupy more space than the other. This can be due to the fact that flow 2 (Figure 6 b)) started after flow 1 and adjusted faster than the disturbed flow. The occupation of the total queue is similar to the one for a single flow, Figure 4.

The conclusion is that TCP handles disturbances very aggressively and that the network is not efficiently used even when no disturbances are present.

IV CONCLUSIONS AND FUTURE WORK

We have seen that TCP slowly adapts to changes in the network, the algorithm cannot handle small disturbing flows but works better when two flows compete for a longer time. Even when there were no disturbances, TCP does not use the total capacity of the network, The slowest queue is empty for more than one fourth of the time. We have also seen that a feedback solution can improve TCP's performance significantly.

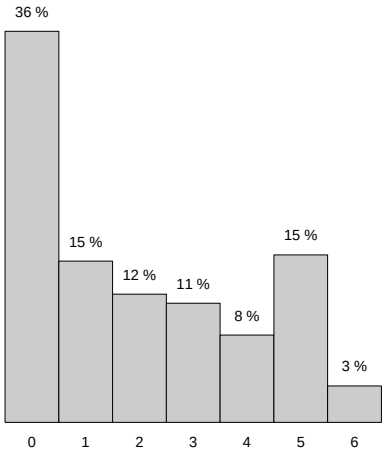


Figure 4: *The histogram for the bottleneck queue in the single flow model.*

This work will continue with implementing Mascolo's solution in ns-2 and compare the results. Investigations will be made about the fairness of the algorithm. The aim is to build a framework that allows easy changes to the implementations. This can then be used to evaluate other changes in the algorithm.

Then the aim is to develop a method for estimation of queue lengths using information about the data traffic. This method will be used and tested in the ns-2 environment and then adjusted to work in real data traffic cases. This estimate can then be used as a performance measure for existing and new networks.

REFERENCES

[1] http://www.mathworks.com.

[2] T. Glad and L. Ljung. *Reglerteknik, Grundläggande teori*. Studentlitteratur, Lund, 2nd edition, 1989.

[3] F. Gunnarsson. Problems and Improvements of Internet Traffic Congestion Control. Master's thesis, Linkopings Universitet, Oct. 2000. LiTH-ISY-EX-3098.

[4] The Mathworks Inc. StateFlow User's Guide. http://www.mathworks.com/products/stateflow/.

[5] S. Mascolo. Smith's predictor for congestion control in TCP Internet protocol. In *Proceedings of the 1999 American Control Conference, 1999.*, volume 6, pages 4441–4445, 1999.

[6] W.R. Stevens. *TCP/IP illustrated vol 1:The protocols.* Addison Wesley, 1994.

[7] University of California, Berkeley. The Network Simulator - ns-2. http://www.isi.edu/nsnam/ns/.

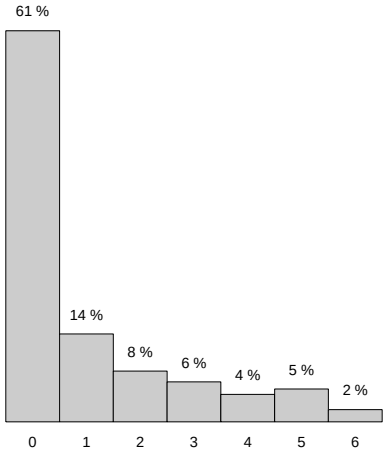


Figure 5: *The histograms for* a) *the queue occupation of the disturbed flow and* b) *the total queue occupation in the disturbance model.*

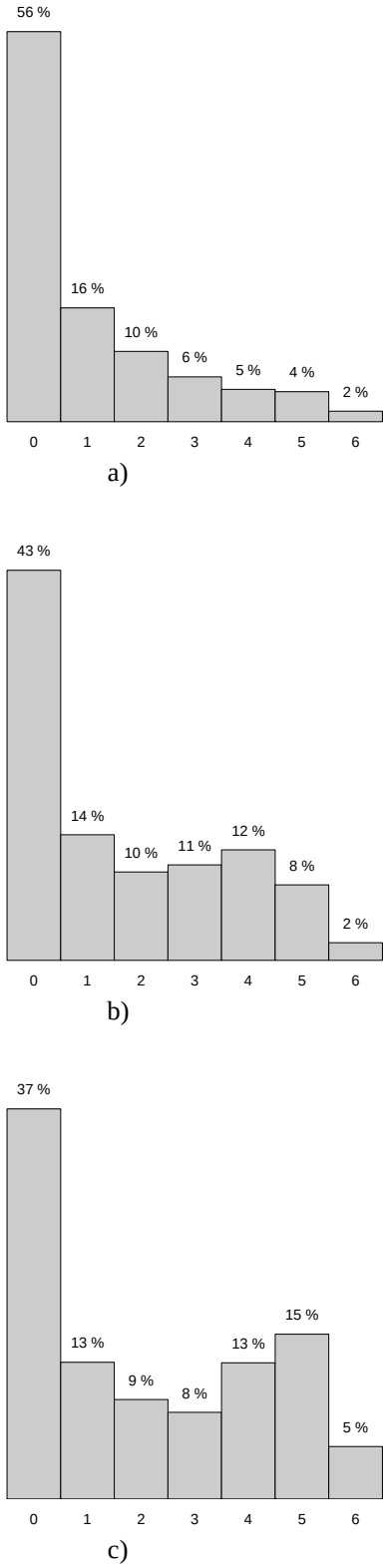


Figure 6: *The histograms for* a) *flow 1*, b) *flow 2 and* c) *the total queue occupation in the two flow model.*