On Network Measurement and Monitoring of end-to-end paths used for e-VLBI.

Julianne Sansa
Makerere University
Faculty of Computing and IT
P.O.Box 7062 Kampala
Uganda
sansa@cit.mak.ac.ug

Arpad Szomoru
Joint Institute for VLBI in Europe
Oude Hoogeveensedijk 4
7991 PD Dwingeloo
The Netherlands
szomoru@jive.nl

J.M. van der Hulst
Kapteyn Astronomical Institute
Postbus 800
9700 AV Groningen
The Netherlands
vdhulst@astro.rug.nl

February 25, 2006

Abstract

We report on the results of data transfer tests that we conducted with the aim of understanding and removing the bottlenecks currently limiting the transfer speed of astronomical data from radio telescopes across the world to the central processing centre in the Netherlands.

Keywords: e-VLBI, radio astronomy, network performance, transport protocols

1 Introduction

Very long baseline interferometry (VLBI) is a technique that enables astronomers to observe cosmic objects at radio wavelengths with an extremely high angular resolution. This is done by combining the signals of widely separated radio telescopes using a correlator, a custom-built supercomputer. In this way a telescope with the size of a continent, and even of the earth, can be simulated.

The telescopes used in the European VLBI Network (EVN) produce data at rates of up to 1Gbps each. Traditionally, these data streams were recorded on tapes (nowadays hard disk drives) and shipped to the correlator located at JIVE, the Joint Institute for VLBI in Europe, in Dwingeloo, the Netherlands. During the last few years JIVE, in collaboration with the European National Research Networks and the pan-European Research Network GEANT, have worked on a proof-of-concept project to connect several telescopes across Europe in real-time to the correlator via the Internet (electronic VLBI or e-VLBI). This project has led to an EC sponsored project called EXPReS, which over the next few years will transform the EVN to a fully functional real-time e-VLBI network.

During the PoC project it became clear that in spite of the vast capacity of the connecting networks, the actual transport of large amounts of data poses quite a challenge especially for real time correlation and in utilizing all the physically available bandwidth. In order to investigate the throughput and examine what limits the data flow we carried out a number of network transfer tests under varying conditions. In this paper we report how tests were set up and evaluate the results in terms of what limits the network performance. The Mark5 [1] application that handles e-VLBI data uses the Transport Control Protocol (TCP). By the nature of e-VLBI, huge amounts of data have to be transported over long distances from geographically dispersed telescopes to one central correlator. We conducted network performance tests to investigate the behaviour of the network connections. Internet measurement tools are classified into two classes, passive and active [2]. Active tools inject traffic into the network and use it to gather network statistics, while passive tools use already existing traffic to compute the status of the network. We monitored the network paths used for e-VLBI in the Netherlands with web100 [3, 4] (a mix of active and passive tool) and tcpdump [5] (a passive tool) to gather statistics on the congestion window, Round Trip Time (RTT), packet loss and the

resulting throughput. The aim was to clarify the relation between these variables and the achieved throughput.

1.1 Background

When a TCP connection is established between sender and receiver, a congestion window (CWND) is negotiated. This is the amount of data that a sender can send before receiving any feedback from the receiver. A TCP receiver maintains a receive window (RWND), for the purpose of informing the sender how much data it is willing to accept in one go without needing to generate acknowledgements. It is thus in the best interest of the TCP connection that CWND and RWND are close to one another, otherwise the lesser value will be the effective value for both. TCP senders update the CWND in response to acknowledgments of received packets and the detection of congestion. Standard TCP interprets packet loss as congestion. This packet loss is detected through 1) the timeout of an unacknowledged packet, 2) the receipt of several duplicate acknowledgments, which emphasize that a previously acknowledged packet is being re-acknowledged implying to the sender that the next packet whose turn it was to be acknowledged has been lost, or 3) through selective acknowledgments (SACK), which indicate specific packets received among a block that was sent implying that the rest were lost.
A TCP sender can be in two states, the slow-start or the congestion avoidance state. A sender which has just established a TCP connection will be in the slow-start state where the initial value of CWND is very small (one packet for standard TCP) and increases exponentially as it senses the network congestion status with the aim of using the unused bandwidth as quickly as possible. As the sender's CWND continues to grow it will cross the so-called slow-start threshold and the connection will move into the congestion avoidance state. In this state, the increased chance of congestion causes the increase of the CWND to slow down.
In these two states the sender will adjust the size of the CWND according to the following equations (1) - (4) depending on whether the previous packet was acknowledged or lost:

Slow-Start:

On acknowledgement:

$$CWND_{new} = CWND_{old} + c \tag{1}$$

On packet loss:

$$CWND_{new} = CWND_{initial} \tag{2}$$

Congestion Avoidance:

On acknowledgement:

$$CWND_{new} = CWND_{old} + a/CWND_{old} \tag{3}$$

On packet loss:

$$CWND_{new} = CWND_{old} - b \times CWND_{old} \tag{4}$$

where $a = 1$, $b = 0.5$, $c = 1$

On high speed links the TCP connection quickly passes through the slow-start state into the congestion avoidance state and as a result most of the connection's lifetime is spent in the congestion avoidance state.
From Floyd [6] the following relationships are taken:

$$CWND_{optimal} = Bandwidth \times RTT \tag{5}$$

$$Throughput = \frac{CWND_{average}}{RTT} \tag{6}$$

$$CWND_{average} = \frac{1.2}{\sqrt{p}} \tag{7}$$

where RTT is the Round Trip Time, and p is the packet loss rate.

From equations (6) and (7), we note that the TCP throughput is directly proportional to the CWND and inversely proportional to the RTT and that the CWND in turn is inversely proportional to the square root of packet loss rate. Making long distance connections over high speed links as is the case for e-VLBI, implies that the optimal CWND required to ensure high throughput has to be large, as suggested by equation (5). The TCP congestion avoidance algorithm, in equations (3) and (4), however exhibits a weakness in maintaining a large CWND in the presence of packet loss.

1.2 Questions to address

The main question we want to address is what information we need to know to predict network performance. We can specify this further with the following questions:

i) What is the available bandwidth for a connection that TCP is about to make.

ii) If the available bandwidth is less than the theoretically available bandwidth, what is the limiting bottleneck (sender, network or receiver).

iii) How much packet loss and RTT occurs in the end-to-end path of the established TCP connection.

iv) What is the stability of the TCP connection established.

v) How can we minimize the impact of the identified bottleneck(s).

In this paper we will address questions i) ii) iii) iv) and part of v).

2 Methodology

In this section we describe the network topology over which we conducted the network tests as well the hardware configuration and the tools we used to gather network statistics.

2.1 Setup

Tests were conducted between Mark5 units located at JIVE interconnected via Amsterdam through Netherlight [7] as shown in Figure 1.

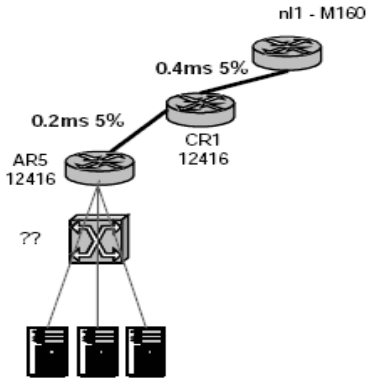

Figure 1: *Network topology for which tests were conducted*

Each of the links from the router AR5 to each of the Mark5's is 1 Gbps. Statistics were gathered for iperf [8] flows or Mark5 disk2net-net2disk transfers from one Mark5 through the router AR5 to another Mark5. Some tests were conducted through Amsterdam to Manchester, UK, via UKlight [9].

2.2 Hardware configuration

The hardware configuration of the Mark5 units is as follows:

- Motherboard: Supermicro P3TDLE, FSB 133 MHz
- Processor: Pentium III 1.26 GHz, FSB 133 MHz, 512K L2
- Chipset: ServerWorks Serverset III LE
- Memory: 256M PC133 SDRAM
- Operating System: RedHat Release 9.0, linux kernel 2.4.20-6

2.3 Kernel Tuning

In order to ensure maximum throughput, the following well known high performance networking options [10] were tuned. Maximum Transmission Unit (MTU) of 8192 bytes was supported for all tests in the Netherlands, while that of 4470 bytes was supported for tests involving Manchester, compared to the default 1500 bytes. TCP buffers were set to 4 Mbytes compared to the 64 Kbytes default while txqueuelen was set from a default of 100 to 20000, which has been proved to offer good performance [11]. The default values are too small to support high speed data transfers.

2.4 Measurements with Web100 & TCPdump

We use Web100 [3, 4], a set of tools that together provide an advanced management interface for TCP. It is a passive tool in that it uses ordinary TCP traffic to calculate TCP event statistics per connnection from the received acknowledgments. However, since statistics are computed for TCP connections from TCP acknowlegments, web100 is also an active tool. The diagnostics provided by Web100 are extremely useful to evaluate the link performance.
During e-VLBI transfers we gather statistics similar to those reported with Web100 using TCPdump [5]. TCPdump generates no traffic, it merely keeps track of all the traffic going through a particular network interface, making it a passive tool.

3 Tests

In this section we present the results of our tests in four subsections: observed CWND & RWND, packet loss, RTT and TCP throughput.

3.1 Observed CWND & RWND

As mentioned in Section 1.1, CWND and RWND should be nearly equal and large enough to allow full bandwidth utilisation. The links have a bandwidth of 1 Gbps, the RTT for tests in the Netherlands is ~ 4 ms, while the RTT for tests involving Manchester is ~ 16 ms. From equation (5) in Section 1.1 the optimal CWND and RWND for tests within the Netherlands is 0.5 MB, while for tests to Manchester is 2 MB.
In Figure 2A we show the result of a single data flow between JIVE and Manchester. Both CWND and RWND are in close proximity of each other with an average of 1.58×10^6 bytes, which is 79% of the optimal CWND. When producing two parallel competing data flows as shown in Figure 2B we begin to see unfair allocation of the CWND for each flow, one with average 1.48×10^6 bytes (which is 74% of the optimal CWND) and the other with 1.19×10^6 bytes (which is 59.5% of the optimal CWND). The same situation is seen in

Figure 2C for five parallel flows where we see that all the five values of the CWND ranging from 5.29×10^5 bytes to 5.76×10^5 bytes (which is $\sim 25\%$ of the optimal CWND). For multiple flows Figures (2B and 2C), we do not show the corresponding RWND plots because RWND values are always higher than the CWND values, making CWND the effectively used variable. In both Figures 2B and 2C the effective combined CWND of the multiple flows is greater than 100% of the optimal CWND, thus the implied throughput that could be achieved is close to the peak value of 1 Gbps.

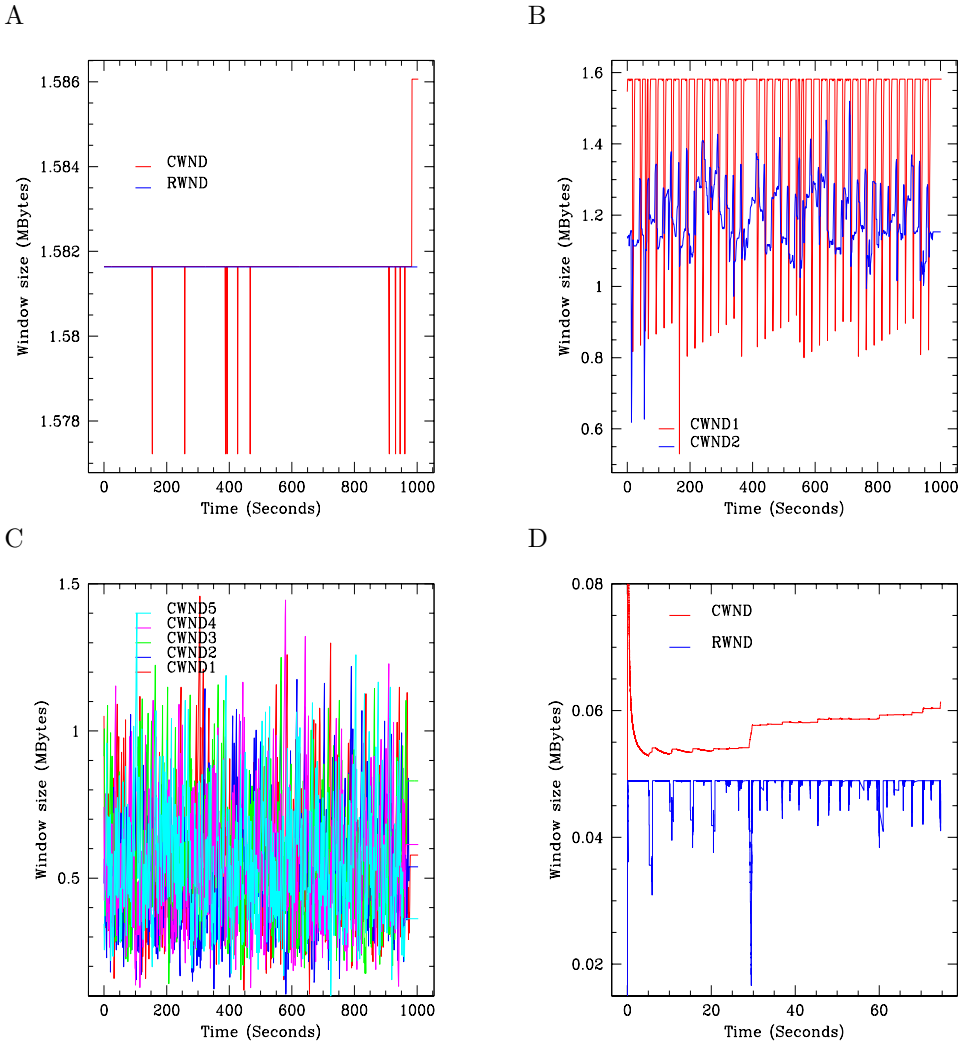

Figure 2: Congestion Window and Receive Window for A: a single memory-to-memory data flow (CWND and RWND are nearly equal), B: two parallel memory-to-memory data flows (RWND is not plotted as it is much larger than CWND, making CWND the effective limiting value), C: five parallel memory-to-memory data flows (here too RWND is not plotted as it is much larger than CWND) and D: disk2net-net2disk e-VLBI data transfer (RWND smaller than CWND).

3.2 Packet loss

From our measured maximum CWND we calculate a steady state packet loss of 7.49×10^{-6} [6]. A rate larger than this steady state rate will cause a decrease of the CWND, while a smaller rate will cause an increase. We observed zero packet loss rate for single flows of both iperf and the Mark5 application and variable average packet loss of 1.18×10^{-9} and 2.81×10^{-8} for two parallel flows, while for five parallel flows it ranged from 3.03×10^{-8} to 5.02×10^{-8} as shown in Figure 3A and Figure 3B for 2 and 5 flows repectively. As the observed packet loss rates are much smaller than the steady state packet loss rate, the CWND should be increasing. This however is not the case. We will return to this in Section 4.

3.3 RTT

In iperf tests from Dwingeloo to Manchester we observed an alternating RTT between two points 10 ms and 20 ms for a single flow, alternating RTTs between between three points 10 ms, 20 ms and 30ms for two parallel flows and five parallel flows. During e-VLBI data transfers between two hosts both located at JIVE in the Netherlands, we noted an average RTT of 3.8

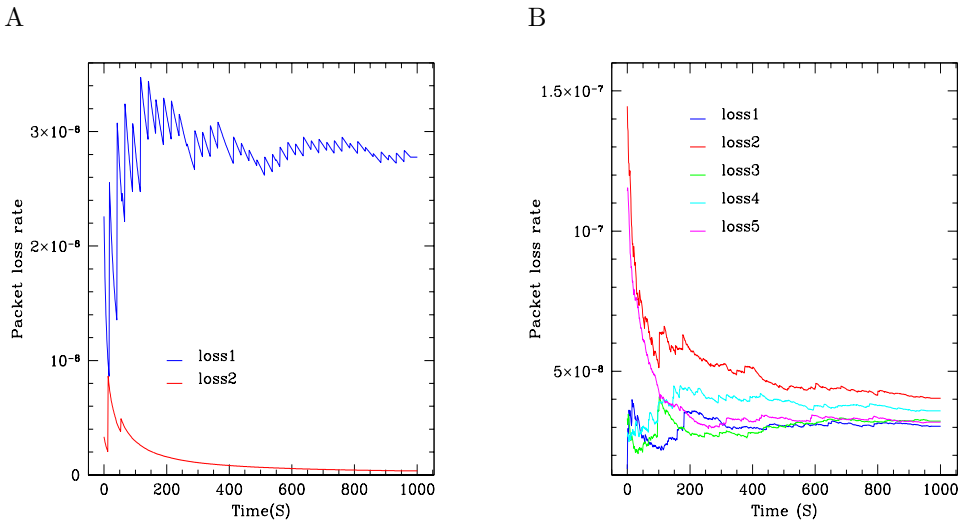

Figure 3: Packet loss rate for A: two parallel memory-to-memory data flows (loss1 is variable, loss2 tends towards zero) and B: five parallel memory-to-memory data flows (values for each flow are more variable than in the two parallel case (3A) and tend towards a unified stable value)

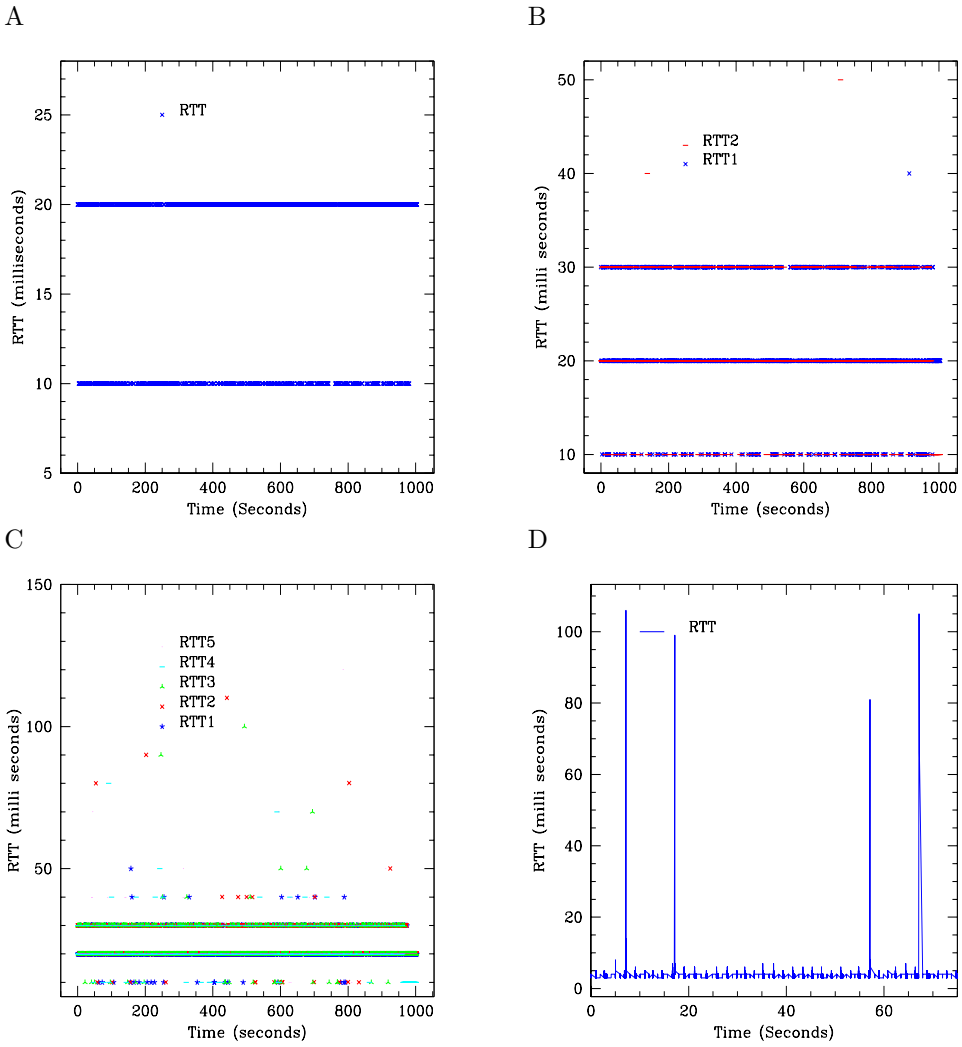

Figure 4: Round trip time for A: a single memory-to-memory data flow, B: two parallel memory-to-memory data flows, C: five parallel memory-to-memory data flows between a Mark5 in the Netherlands and a machine in Manchester and D: a disk2net-net2disk e-VLBI data flow between two Mark5's both in the Netherlands

ms with a few spikes to approximately 100 ms. These results are shown in Figure 4. The general observation is that RTT values are quite stable even when multiple flows are generated, making RTT ideal to use as a sign of congestion.

3.4 TCP throughput

An average TCP throughput of 810.7 Mbps was achieved for a single TCP flow as shown in Figure 5A. For two parallel flows the aggregate TCP throughput was 973.4 Mbps, which is more than what was achieved by a single flow. There was however unfair sharing of this throughput with one flow having average 578.6 Mbps and the other having 394.8 Mbps. This is illustrated in Figure 5B.
For five parallel flows the aggregate TCP throughput is 977.8 Mbps, which is closest to linespeed, with the various flows having 203.8 Mbps, 174.7 Mbps, 204.2 Mbps, 192.8 Mbps and 202.3 Mbps. The unfair sharing of the available bandwidth is shown in Figure 5C. For the disk2net-net2disk e-VLBI transfer we observed an average of 366.9 Mbps, although performance was affected by the tcpdump running at the same time by a factor of between 20% and 40%, which implies this average throughput would have been a value between 440.3 Mbps and 513.7 Mbps without TCPdump.

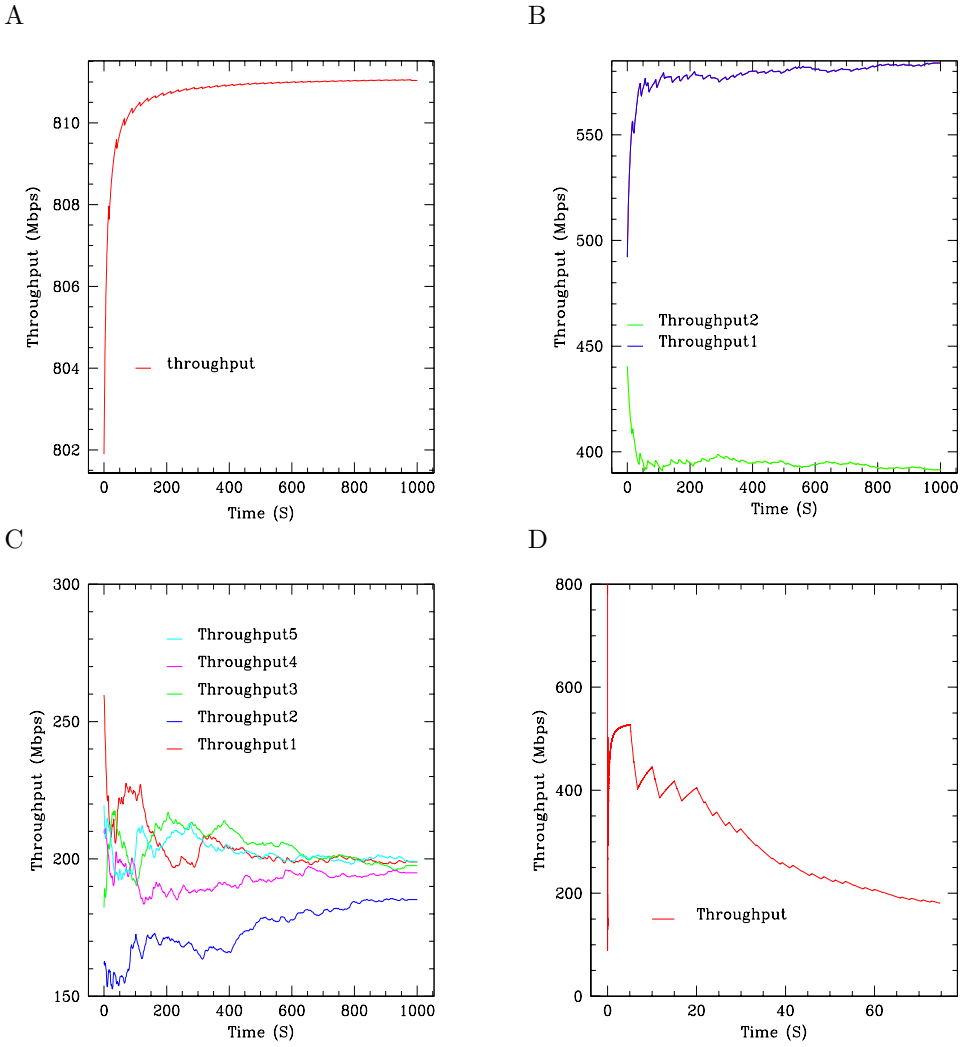

Figure 5: Achieved throughput for: A - a single memory-to-memory data flow (a high stable throughput is achieved), B - two parallel memory-to-memory data flows (unequal throughput is attained by each flow, which slightly fluctuates in each case), C - five parallel memory-to-memory data flows (unequal throughput, which fluctuates significantly and tends towards stable value for each flow) and D - disk2net-net2disk e-VLBI data flow (throughput fluctuates at the beginning of the flow and stabilises a short while later).

Table 1 shows a summary of the average CWND, RWND, packet loss rate, RTT and throughput observed for each data flow under the different scenarios.

4 Discussion of Results

In this section we discuss the results obtained from the tests presented in the Section 3 as well as the bottle

Data Flow category	CWND (MBytes)	RWND (MBytes)	Packet loss rate	RTT (ms)	Throughput (Mbps)
Single memory-to-memory data flow	1.58	1.58	0	16.24	810.73
Two parallel memmory-to-memory data flows	1.47 1.19	1.58 1.02	2.81×10^{-8} 1.17×10^{-9}	20.52 21.65	578.62 394.81
Five parallel memmory-to-memory data flows	0.57 0.529 0.576 0.557 0.562	1.58 1.58 1.58 1.58 1.58	3.04×10^{-8} 5.02×10^{-8} 3.03×10^{-8} 3.79×10^{-8} 3.64×10^{-8}	23.35 22.50 23.31 23.29 23.15	203.76 174.70 204.23 192.74 202.27
Disk2net-net2disk e-VLBI data flow	0.0562	0.0483	0	3.8	366.90

Table 1: Average CWND, RWND, packet loss rate, RTT and throughput for each data flow under the different scenarios

necks we found to be present.

4.1 Single Flows

In the case of one TCP flow (both iperf and e-VLBI) we see the same CWND sustained for a period of time and yet with both slow-start and congestion avoidance algorithms we should be seeing CWND either increase or decrease. This can be explained as idle connection window validation [17], in which the same constant CWND is maintained during an application limited period, which means the CWND is not increased merely by reception of ACKs, as long as during the previous RTT the flow did not fully use the available CWND. This seems to imply that the single flows experience severe application limitation on these high speed links. Application limitation happens when the application does not produce data fast enough [19] for two reasons. Either the application is transfering small amounts of data at a relatively constant rate to the TCP layer or the application is producing data in bursts separated from each other by idle periods. Based on this explanation we conclude that the iperf flow is application limited due to the former while the e-VLBI data flow is application limited due to the latter. The focal interest being the e-VLBI data flow, we therefore need to eliminate or shorten the idle period between the data bursts during the lifetime of the e-VLBI data flow.
It seems then that single flow transport of e-VLBI data is limited by two distinct effects. Firstly, idle periods between data bursts, and secondly, packet losses now caused by network congestion (as the observed zero packet loss rate does not result in an increase of the CWND). These packet losses must be hardware related (e.g. PCI bus conflicts, NIC performance) [16, 18].

4.2 Parallel Flows

Our tests for parallel iperf flows are similar to those done by Hacker [13] but differ in that we list the CWND values in addition to the packet loss and throughput. The network which we use in our tests also differs in terms of RTT and the supported MTU. Finally, we not only measure memory-to-memory transfers, but also compare the results with the transfer of real e-VLBI data, involving the performance of more hardware components in the endpoint systems. With parallel flows the aggregate throughput is much higher than what single flows achieve. The parallel flows are able to transfer data without application limitation surfacing because each flow does not need as much data to be fully utilized compared to a single flow, however the TCP congestion avoidance algorithm limits the further increase of the CWND since packet loss is experienced. It is under such circumstances that the use of modified congestion avoidance algorithms [6, 20, 21, 22, 23] may improve network performance.

4.3 Bottlenecks

For the single memory-to-memory data flow, the hardware in both the sending and receiving hosts limited the throughput, because both CWND and RWND were stable throughout the connection's lifetime.
For the multiple memory-to-memory data flows the TCP congestion avoidance algorithm was the bottleneck because the CWND kept fluctuating implying the detection of packet loss.
For the disk2net-net2disk e-VLBI data flow the receiver's hardware limited the throughput because the RWND continuously fluctuated and was lower than the CWND, indicating that the receiving host was overwhelmed. Apart from the limitations discussed above, a number of other bottlenecks are possible:

- Undue retransmissions. TCP's fast recovery mechanism may yield unnecessary packet retransmissions, thus reducing the effective throughput.

- Interface stalls. These happen whenever the network interface halts as it waits to receive data from upper layers. These stalls are interpreted

as a sign of congestion, reducing the CWND and consequently the throughput.

- Vendor specific TCP implementations. Some TCP implementations have incorporated a mechanism of validating the CWND in cases where its value has been constant for a given period of time and this mechanism also contributes to reducing or maintaining a constant CWND depending on the situation.

5 Conclusions & Further Work

In this paper we show that application limitation can be a significant factor in single flows on high speed links. When parallel flows are used, the TCP congestion avoidance algorithm takes over as the limiting factor. We also note that packet losses other than those caused by network congestion can be quite important. Future work will include eliminating the idle time between data bursts during the lifetime of an e-VLBI data flow. We intend to do an analysis of the proposed aggressive congestion avoidance algorithms. We also intend to model TCP e-VLBI data flows and relate flow analysis with correlation to determine how correlation is impacted by packet loss and delay.

References

[1] The Haystack observatory website, "Mark5 VLBI Data System." *http://web.haystack.mit.edu/mark5/*

[2] The Cooperative Association for Internet Data Analysis website, "Internet Tools Taxonomy (2003)." *http://www.caida.org/tools/taxonomy*

[3] Mathis, M., Heffner, J. and Reddy, R. , Web100: Extended TCP Instrumentation for Research, Education and Diagnosis. *ACM Computer Communications Review, Vol 33. Num 3, July 2003*

[4] The Web100 Project, "Project Documents." *www.web100.org/docs/*

[5] The TCPDUMP public repositiry. *www.tcpdump.org*

[6] Floyd, S., "HighSpeed TCP for large Congestion Windows." *RFC 3649, Dec 2003.*

[7] The open optical Internet exchange in Amsterdam. *www.netherlight.net*

[8] Iperf: The TCP/UDP Bandwidth Measurement Tool. *www.dast.nlanr.net/Projects/Iperf*

[9] Point of Access in London to the Global Lamba Infrustructure Facility. *www.uklight.ac.uk*

[10] Mahdavi, J., Matt, M. and Reddy, R., "Enabling High Performace Data Transfers (System Specific Notes for System Administrators and Privileged Users)." *www.psc.edu/networking/projects/tcptune*

[11] Yee-Ting Li "Effect of TxqueueLen on High Bandwidth Delay Product Network (DataTAG)." *www.hep.ucl.ac.uk/ ytl/tcpip/txqueuelen/datatag-tcp/*

[12] Jacobson, V. Braden, R. and Borman, D., TCP Extensions for High Performance. *RFC 1323, May 1992*

[13] Hacker T. et. al., "The End-to-End Performance Effects of Parallel TCP Sockets on a Lossy Wide-Area-Network," *In Proceedings of 16th IEEE-CS/ACM International Parallel and Distributed Processing Symposium(IPDPS), 2002.*

[14] Floyd & Fall, Promoting the use of end-to-end congestion control in the Internet, *IEEE/ ACM Trans. on Networking, August 1999.*

[15] Padhya et.al Modeling TCP throughput: A Simple model and its empirical validation, *In Proc ACM SigCOMM 1998*

[16] Antony et.al Exploring Practical Limitations of TCP over Transatlantic Networks, *Submitted to Elsevier Science(2004)*

[17] Handley, M., Padhye, J., & Floyd, S., "TCP Congestion Window Validation," *RFC 2861, June 2000*

[18] Laksham, T.V., & Madhow, U., "The Performance of TCP/IP for Networks with High Bandwidth-Delay Products and Random Loss." *IFIP Transactions C-26, High Performance Networking, pages 135-150, 1994*

[19] Siekkinen, M. Urvoy-Keller, G. , Biersack, E. W. & En-Najjary, T., "Root Cause Analysis for Long-Lived TCP Connections." *Proceedings of the 2005 ACM Conference on Emerging Networking Experiments and technologies, October 2005*

[20] Kelly, T., "Scalable TCP: Improving Performance in Highspeed Wide Area Networks." *ACM SIGCOMM Computer Communications Review, Vol 33,Num 2.*

[21] Jin, C., Wei, D. X., & Low, S. H., "FAST TCP: Motivation, Architecture, Algorithms, Performance." *Proceedings of IEEE Infocom, Hong Kong, March 2004.*

[22] Mascolo,S., Grieco, L. A. Ferorelli, R., Carmada, P. & Piscitelli, G., Performace evaluation of Westwood+ TCP Congestion Control *Performance Evaluation 55 (2004) 93-111.*

[23] Katabi, D. , Handley, M. , & Rohrs, C., "Congestion Control for High Bandwidth-Delay Product Networks." *Proceedings ACM Sigcomm, August 2002.*