

# Validation of the WAND simulator through comparison with laboratory tests

A. J. McGregor  
 National Laboratory for Applied Network  
 Research  
 (NLANR), San Diego Super Computer Center,  
 10100 Hopkins Drive, San Diego, CA 92186-0505,  
 USA  
 tonym@cs.waikato.ac.nz

M. W. Pearson  
 The University of Waikato,  
 Private Bag 3107, Hamilton,  
 New Zealand  
 mpearson@cs.waikato.ac.nz

D. H. T. Lawson  
 The University of Waikato,  
 Private Bag 3107, Hamilton,  
 New Zealand  
 dhtrl1@cs.waikato.ac.nz

## I. INTRODUCTION

A common approach to studying the Internet is to collect traffic measurements and then explore a range of network conditions by using the measurements as input to a simulator. Studies of this type have been presented in many workshops and conferences, including previous PAM workshops.[1][2]. While much can be learned using this approach it is important to validate the results. Simulations, by their nature, are simplifications of the real world system that they model. Details that are thought to make the model simpler without having a significant impact on the results are omitted. Network systems have an element of chaos[3] in their behaviour. Because of this it is not always possible to be sure of the impact of these simplifications. To increase confidence in the results of simulations validation of the simulation process should be performed.

There are a number of approaches to validation. These include: running simulations for situations where the results are known before the simulation, perhaps because they are simple cases or match measured cases; sensitivity analysis where the effect of changes in the inputs to the are studied to see if they match expected changes in the outputs; and creating a physical test network and comparing the results of simulations against measurements taken on the test network. In all approaches validation does not cover the full range simulation results. If that were possible the simulation would probably not be required. Instead a range of validations are undertaken and, given satisfactory results confidence in the other results of the simulations is increased.

Over recent years the WAND group has been developing a simulation environment to study Internet behaviour. A

number studies using this simulator have been undertaken, mostly studying the behaviour of a fictionally international Internet connection such as might connect a distant country (like New Zealand) to the US Internet.

Previous comparisons with a measured case and a sensitivity study established some confidence in the validity of the simulator and also highlighted some weaknesses[1]. In this paper we report on the results of a comparison between the simulator and measurements taken from a laboratory test network.

The paper is organised as follows: Section 2 describes the simulator. Section 3 describes the laboratory network and the measurements taken from it. Section 4 describes two scenarios that we simulated and constructed in the laboratory. Section 5 gives some analysis of the results of the tests. The paper concludes with reflections on the similarities and differences between the results and outlines out intentions for further work in this area.

## II. SIMULATOR

Over the last couple of years we have been studying the performance of TCP/IP traffic on high-performance links; in particular international links between New Zealand and the US as shown in Figure 1. The simulator developed for these studies is based on the ATM-TN simulator[4]. The simulator contains a TCP model that includes the actual TCP code from 4.4 BSD Lite, modified to suit the simulation environment. Each of the TCP connections are simulated on a packet by packet basis and include the modelling of slow start, congestion control, fast retransmit, and fast recovery algorithms.[5]

The simulated network can be considered as a number of connected components. These are the HTTP requesters (the Web clients in the New Zealand Internet), the NZ proxy, the international link, including the routers that

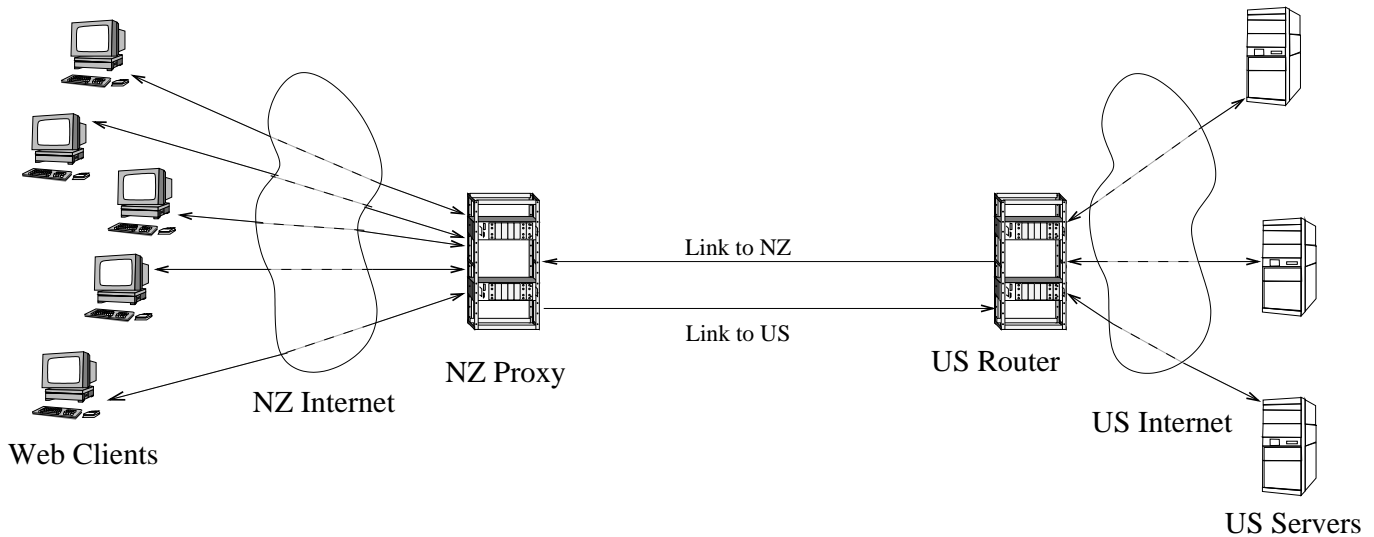


Fig. 1. Network Architecture for HTTP traffic

feed it (which is the key component under study), the US network and the HTTP servers.

An HTTP traffic model is responsible for creating TCP connections, sending HTTP GET requests, receiving the request at the destination and returning the results, and for recording the time required to complete the HTTP requests. The HTTP model makes use of information about hosts and a URL trace collected from a real network using passive measurement.

The delays in the US part of the network are simulated by the HTTP traffic model which releases the packets that make up the HTTP response at a regulated rate so that the complete response arrives at the US proxy at the same mean rate as applied when the page was fetched over the real network.

When developing the simulation a number of assumptions about aspects of the network structure that could be omitted without adversely affecting the simulation results were made.

To make the descriptions in the paper simpler the components of the simulated network are described in terms of an international connection between New Zealand and the United States. The results are, of course, more widely applicable.

#### A. Simulated Workload

Most of the information required to generate the simulation input files (described in the next section) was gathered from HTTP log-files collected from the New Zealand Internet exchange (NZIX). The trace files used were collected from 3:00pm to 3:10pm in July 1997.

There were, on average, 421 requests<sup>1</sup> per interval.

To generate higher loads than that experienced when the trace files were collected, traces for the same time on

<sup>1</sup>The actual trace includes more requests. This number is the number of successful international requests that were not satisfied by the cache hierarchy

successive days in July were integrated into a single trace. When higher still loads were required more than one copy of each trace was integrated into the log-file. Each copy was offset in time to minimise the effect of the artificial self correlation of the trace generated in this way.

#### B. Simulator Design

The simulation process is shown in figure 2. It can be considered as three interlinked processes, pre-processing, simulation and post-processing.

##### Pre-Processing

In the preprocessing stage the input files for the simulator are prepared. These are:

- A “hostfile” which contains an entry for each server accessed during the simulation. The entries contain: a unique ID for each host, the DNS name of the host, the maximum window size for the host and the TCP maximum segment size (MSS) for connections to the host.
- A “log-file” which contains an entry for each HTTP request. The entry contains the host ID for the server the request is fetched from, the size of the HTTP GET request, the size of the HTTP response and the time taken in the US component of the network.
- The simulation parameters including: the buffer sizes used by the proxies and the international link speed and delay in each direction.

##### Post-Processing

Post processing is mostly a matter of collecting the results of interest from many simulation runs into a single set of plots. This was done with an array of perl scripts. GNU plot was used to draw the plots.

The simulator used in this study was based on the ATM-TN simulator[4] with modifications for this problem. The changes include replacing the ATM infrastructure with a simpler and more general bit serial interface.

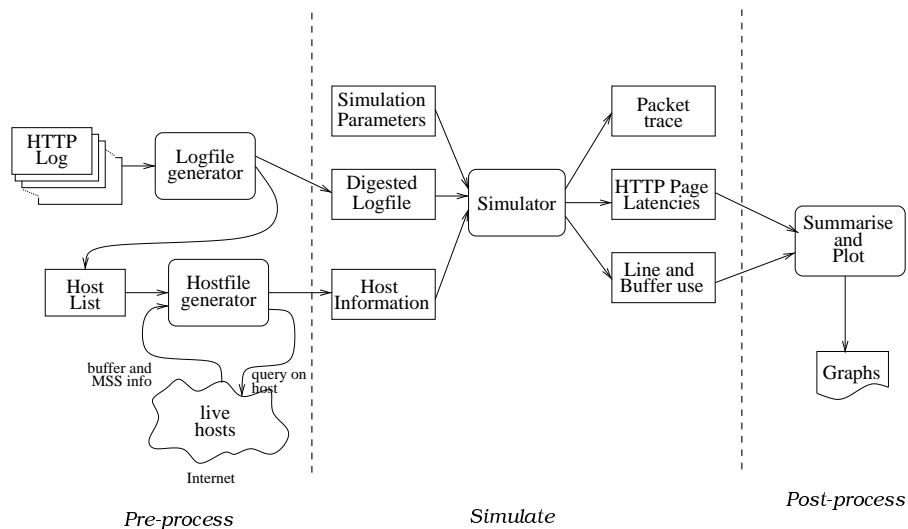


Fig. 2. Simulator Design

The main two components used from ATM-TN are the conservative (as opposed to parallel) simulation engine and the TCP model. ATM-TN's TCP model includes the actual TCP code from 4.4 BSD Lite, modified to suit the simulation environment. Connections are simulated on a packet by packet basis and include slow start, congestion control, fast retransmit, and fast recovery algorithms.[5] The simulator design is shown in figure 3. The simulator simulates the connections between the NZ proxy and the servers in the US. It does not include the web client to NZ proxy to component of the network because this has not been significant to the studies carried out in the past. Additional delays that are dependent on the type of connection (e.g. modem or direct connect) will be incurred in the NZ component of the real network.

#### HTTP traffic model

The HTTP traffic model is responsible for creating TCP connections, sending HTTP GET requests, receiving the request at the destination and returning and results and for recording the time required to complete the HTTP requests. The HTTP model makes use of the hostfile and the log-file to control the simulation. The delays in the US part of the network are simulated by the HTTP traffic model which releases the packets that make up the HTTP response at a regulated rate so that the complete response arrives at the US proxy at the same mean rate as it did when the page was fetched on the real network.

#### TCP Connection

The TCP connection model simulates an end-to-end TCP connection and are based on a pair of TCP stacks, one for each end of the connection. It is assumed that the effect of errors is negligible.

The simulation assumes that the proxy has sufficient CPU and memory to manage the workload and that the delay imposed by processing on the proxy not due to TCP queuing and transmission is negligible.

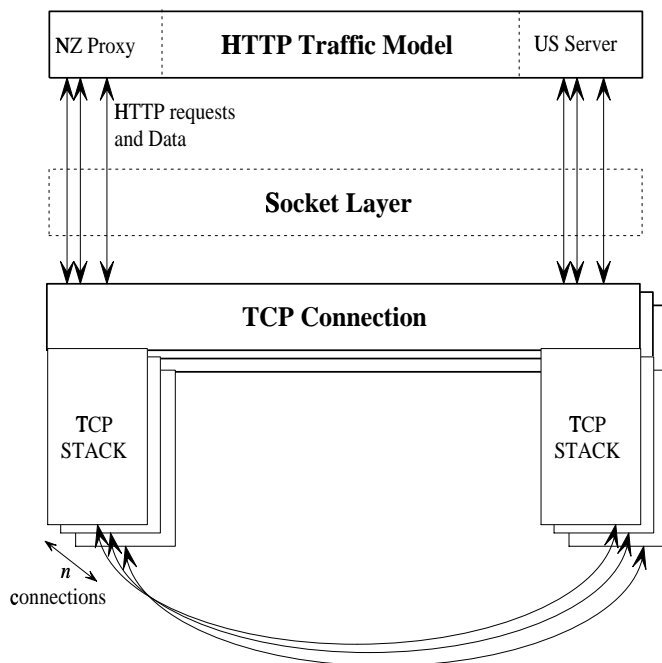


Fig. 3. Simulator Design

### III. LABORATORY TEST NETWORK

The laboratory test network is based around CISCO routers. We have 5 2500 series routers and one 4000 series router available. The basic test network is shown in figure 4.

The client and server computers are P166, 32MB, Linux based systems with two DEC Tulip Ethernet NICs. The delay machines are P166, 72MB, FreeBSD based systems with two DEC Tulip Ethernet NICs<sup>2</sup>.

<sup>2</sup>Because of hardware limitations we are forced to use half-duplex 10Base2 (thinnet) ethernet. Utilising two NICs in each computer enable us to create a full-duplex network.

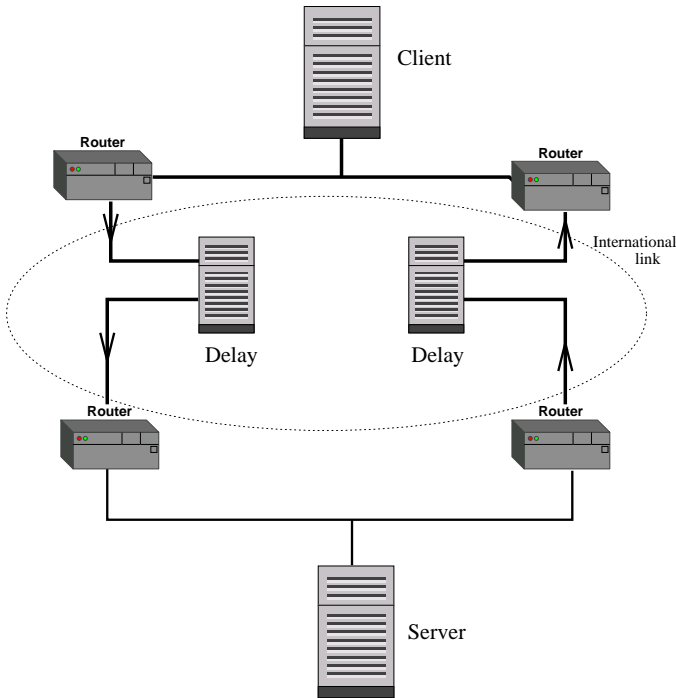


Fig. 4. Laboratory Test Network

The client reads the same log file as the simulator (produced by the preprocessing described in section II-B above). For clarity we call this the *constructed logfile* in this section, to distinguish it from the original HTTP traces that it is constructed from and from the logs that are produced from the simulation and lab network tests. The word *original* is used to highlight a term related to the original HTTP traces collected from the NZIX proxy cache. For example the original HTTP request or the original server.

For each entry in the constructed logfile the client constructs a request message that has the same size as the original request in the HTTP log. At the time indicated in the constructed logfile the client opens a TCP connection (using the standard Linux TCP stack and associated mechanisms) to the server. Once the connection is open the client sends the request message which contains enough information for the server to respond with a data stream that is similar to the HTTP response of the original server. Padding is used to make the message the correct size. The server responds by send a number of reply messages over the TCP connection. The sum of the sizes of the sent messages matches the size of the original HTTP response. With the exception of the last message the individual sizes match the original MTU (less the TCP and IP overhead). The messages are spaced so that the total reply takes approximately the same time as the original reply. We can only approximate this time because we do not know in advance how long the last message will take to be delivered.

The client collects the total fetch time for each file and generates an output file similar to that of the simulator.

Transmission times in the laboratory are short but many of the most interesting scenarios involve much longer laten-

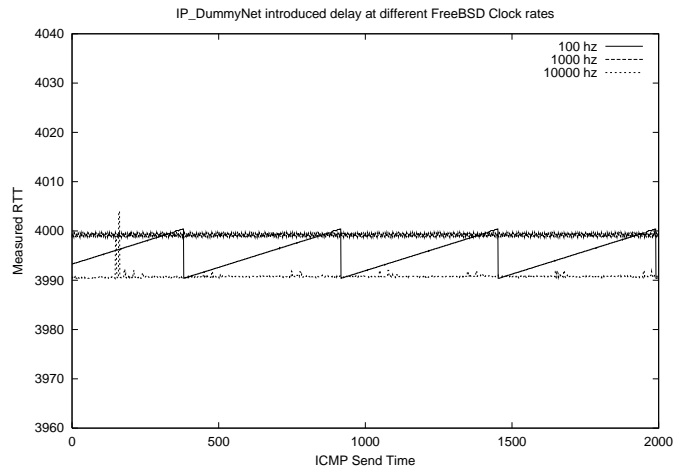


Fig. 5.

cies. To allow testing that matches these we introduce an artificial delay using the delay systems shown in figure 4.

#### A. Behaviour of Delay Software

Delays are introduced into the network using a package called *IP\_DummyNet*, part of the *FreeBSD* firewall code. Two network interfaces are configured to operate as a bridge, and rules are created to allow forwarding of packets over the bridge. Using *IP\_DummyNet*, we are able to insert a specified delay for a given queue.

Early in the process of setting up the lab network we measured the delay introduced by *IP\_DummyNet* when it was configured to introduce a 4s delay. We observed the 100Hz line shown in figure 5. Note the variation from 10ms less than the target time to about 1ms greater than the target time. Other tests performed indicated that the variance was not relative to the total delay. The delays we wanted to introduce matched those of a trans-pacific link, I.E. about 60ms one-way delay. 10ms represents a 17% error which was greater percentage error than we were happy with.

After some experimentation we found that the variation could be reduced by altering the FreeBSD clock rate. Increasing the clock by one order of magnitude produced the result we were looking for (1-2ms variation). Increasing the clock by another order of magnitude, further reduced the typical variance, at the cost of an overall offset from the desired value. This increased clock rate also caused a much higher, negative, dependence, upon other activities on the computer, such as performing a directory search. The kernel was configured to operate with a clock rate frequency of 1000Hz for the rest of the tests.

## IV. TEST SCENARIOS

We studied two situations. The first was a balanced example where all links were 10Mbps 10base2 Ethernets. This configuration was chosen for pragmatic reasons. Our routers (which were donated) came with those interfaces and it seemed a reasonable starting point.

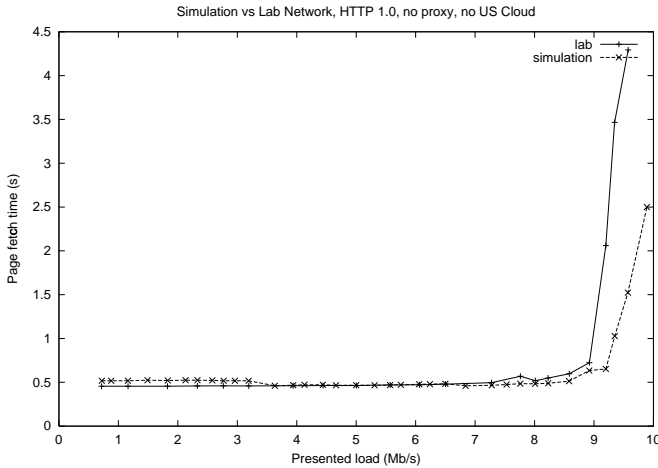


Fig. 6.

After gaining some experience with this configuration and some early promising results we extended our test configuration to include a congested link, as shown in figure 7. This involved utilising the HSSI (High Speed Serial Interface) ports on the Cisco routers.

#### A. Balanced Capacity

The comparison undertaken was using the balanced configuration shown in figure 4. HTTP 1.0 was used. The main implication of this is that a separate connection was used for every HTTP request. No US proxy was included, that is there is a direct connection from the US routers to the server. The existence of the US cloud was not modelled. That is, no delays were inserted into the reply data stream to model the delays imposed by the presence of Internet components between the server in the US and the routers connected to the international link.

The international link is modelled using two separate 10Mbps 10base2 Ethernets to match a full duplex circuit.

Simulations with the same parameters (10Mbps full duplex, international link, 60ms delay, HTTP1.0, no US cloud) and a range of workloads were run. The same workload were delivered to the lab network. The results are presented in figure 6. The most striking characteristic of this graph is the close correlation of the measured and simulated page latencies, at low and medium loads. The second most obvious characteristic is that the latencies vary at high loads, with the lab networks showing higher latencies than the simulator.

#### B. Restricted Link

International links are often congested. Most of the simulation studies we have done have included a simulations in the congested region. The laboratory network design shown in figure 4 does not allow the international link to be congested. Instead congestion will occur at the server and client access links. To remedy this we added a 2Mbps full duplex PPP serial link as part of the lab based ‘international’ link as shown in figure 7. This introduced congestion

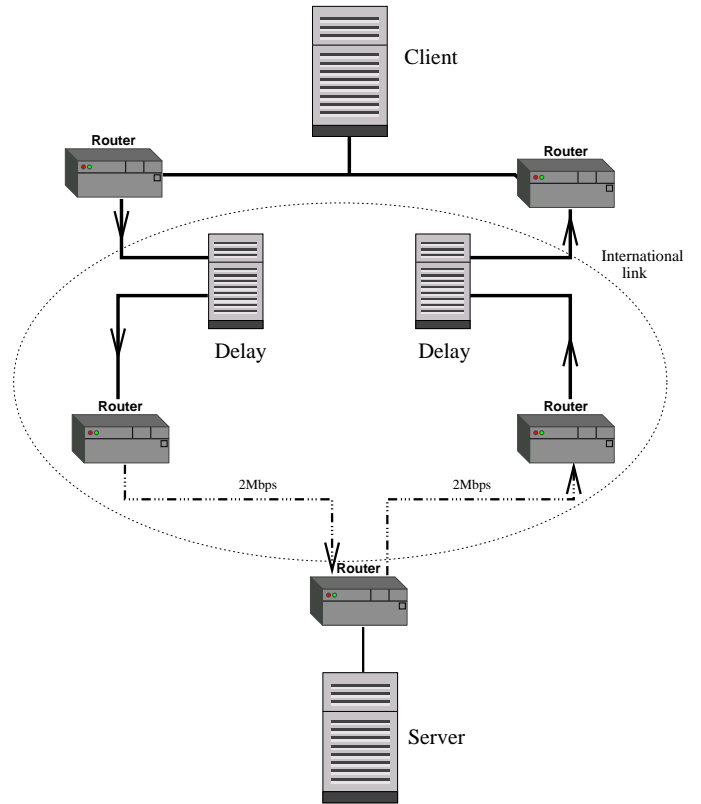


Fig. 7. Laboratory test network with a congested link

and queuing within the access routers.

The results for the congested path, as shown in figure 8 show a different behaviour however. The two plots follow approximately the same trend, although in this case the simulation is slightly pessimistic about the page request latency — it is consistently higher than the lab network for low and medium loads. Interestingly, the same difference is noticeable in figure 6 for presented loads lower than about 3.5Mbps.

The main point of interest in these congested plots however is at the upper end of the load. In both cases (simulation and test network) the link was configured with a 2Mbps return path<sup>3</sup>. The average latency rises slowly as the presented load increases, up to the point where it reaches 100% of the link capacity, or 2Mbps, after which it increases rapidly. This is rather different from the non-congested link in figure 6, where the request latency starts to increase dramatically at about 90% link capacity, or 9Mbps.

<sup>3</sup>In the case of the simulation this was as simple as setting a configuration option: this is normal operational procedure for the simulation. For the Cisco routers, the PPP connection was configured with a clock rate of 2000000, giving a data serialisation rate of 2Mbps.

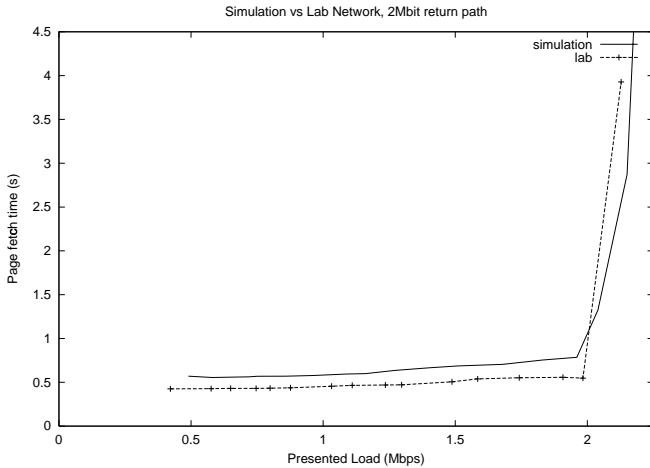


Fig. 8.

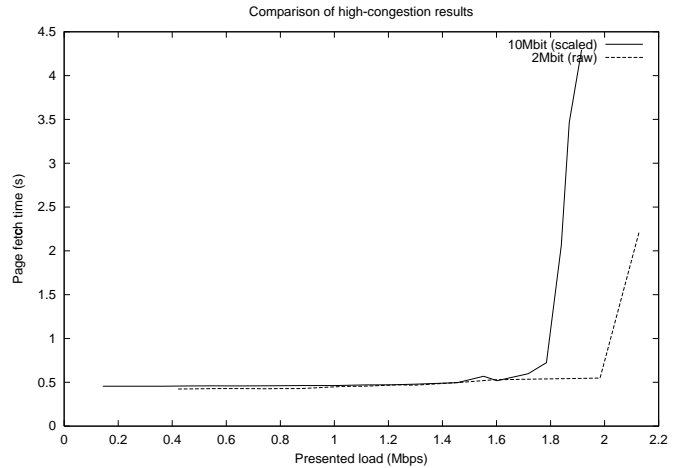


Fig. 9.

## V. ANALYSIS OF RESULTS

### A. Comparison of Simulated and Measured Results

#### A.1 Balanced Capacity

For the purposes of this research, it is very reassuring to note the high correlation between the simulator and lab network experiments. The differences in behaviour at high congestion in the balanced link (figure 6) are interesting however, and the cause of the difference has not yet been determined as of the time of writing this paper.

There are some obvious possibilities however, which could cause the higher latency in the lab network than noticed in the simulator. One possibility is that of protocol overhead; the presented load figures only account for the application layer workload, and don't take into account TCP and IP headers, ethernet frame overhead, and other TCP 'features' such as ACK packets. However, the presented load is calculated identically in both the simulator and the lab network, so this does nothing *per se* to solve the higher latency in the lab network.

One aspect of the simulator is that it is very accurate when defining bandwidth limits. Setting a bandwidth limit of 10,000,000 bytes per second gives exactly that. The "10Mbit" connection available to the lab network may not have the same level of accuracy however. There are a number of places where error could creep into the lab networks available bandwidth, from poor tolerances on physical equipment to backplane speed in the routers or processing load.

#### A.2 Congested Link

The congested validations at 2Mbit (figure 8) also show strong correlations, although the behaviour is different under high congestion, as can be seen in figure 9. Why the behaviour is so different is not instantly obvious.

If the change in behaviour only exhibited itself in the validation system, it could perhaps be explained by different behaviour in the router when sending data over the HSSI link on the Cisco router, for example router queue-

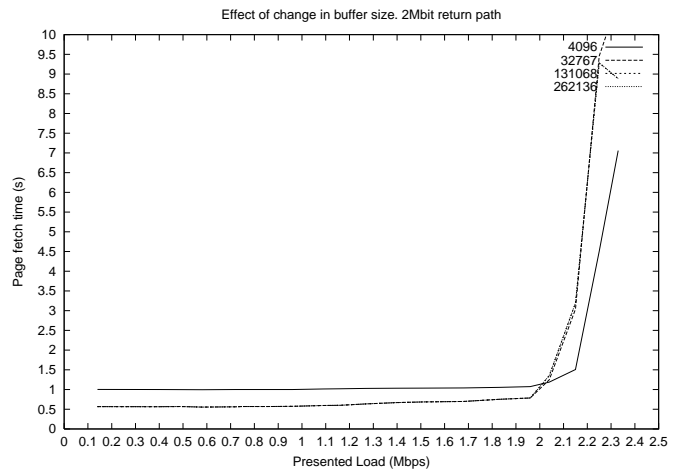


Fig. 10.

ing. However, the same shape is seen in both the simulation runs and the lab network.

One reason for this change in behaviour could be the relative protocol overheads in use. PPP has a slightly lower protocol overhead than Ethernet, therefore gives better performance. There is however no reason for the similar behaviour in the simulation runs, as the simulator in its current use does not actually make any assumptions about the underlying network layers, therefore cannot show any benefits of lower protocol overhead.

#### A.3 Buffer Size Analysis

We speculated that other factors such as available buffer size may have made an impact. Both links (simulation and the lab network) were originally configured for 10Mbit, but were reduced to 2Mbit on the return path. Further tests were run, varying the buffer size in the simulator (see figure 10), and the same behaviour for the 2Mbit return path was seen in each case, varying only when the buffer was too small for effective use.

This indicates that buffering within the simulator is not

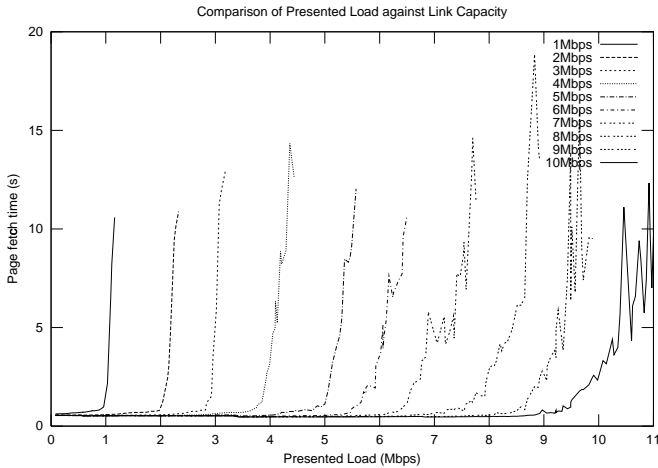


Fig. 11.

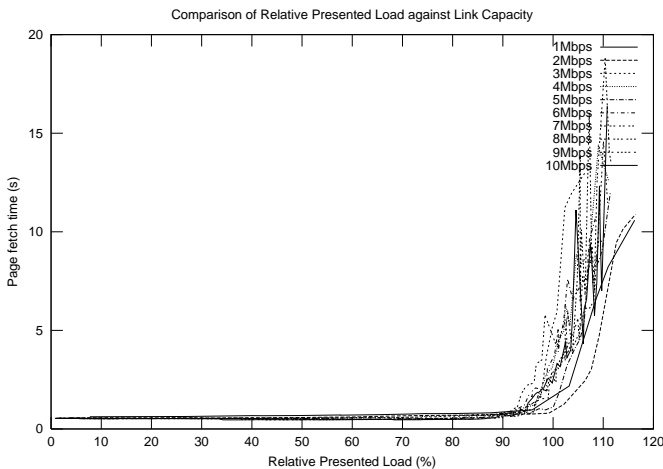


Fig. 12.

causing the discrepancy between the congested and non-congested test cases.

### B. Variable Loads

In an attempt to establish a pattern behind the change in high-congestion behaviour within the simulator, it was decided to run a series of simulations with varying link capacities, fully loading the link each time. This should show a “threshold” at which the aberrant behaviour started occurring, if at all.

Simulations were run at varying presented loads and link capacities, from 1 to 10Mbit/second. The raw results are shown in figure 11 and the proportional results are shown in figure 12. However, these showed no obvious trend in behaviour above the 90% link capacity mark (where presented load == 90% of the link capacity). In fact, the 2 and 5Mbit runs show approximately the same behaviour, a low rise in latency up until 100% saturation, whereas the intervening runs (3 and 4 Mbit) showed a marked rise in latency between 90% and 95%.

With no obvious pattern in the behaviour that we could

attribute to the simulator, we are tentatively concluding that the aberrant behaviour is possibly an artifact of the traffic model used within both the simulator and the lab network.

## VI. CONCLUSIONS

### A. Performance of the simulator

The initial goal for the research behind this paper was to investigate the performance of the simulator with respect to its network model. The results from the validation tests run on the lab networks (figures 6, 8) indicate that the simulator’s network model is close to a real-life scenario. That is, the results from the simulator and the lab network agree with a reasonable level of accuracy, for low and medium congestion levels.

The different behaviour between the simulator and the lab network at high congestion levels has not yet been fully explained at the time of writing, however it is possible that it is an artifact of the current lab network, such as an unknown side-effect occurring within the routers or as a property of the physical medium. If this is the case then it is worth noting, as it is an area in which the simulator fails to adequately take into account some aspects of a “real network”. Note that the simulator does not fail completely in this area, it is just perhaps slightly optimistic about the results.

It is worth considering however, that the major differences between the simulation and lab network in the balanced test case (figure 6) only appear at very high loads — above 90% link utilisation. We do not expect to have “good” performance from a network that is constantly utilised above 90% of its link capacity. Moreover, if a network link is constantly sitting at 90% or higher utilisation, it is probably time to upgrade the link.

Some of the results indicate that the traffic generation model or the data analysis method may need some investigation. The correlation between the simulator and the test network for the congested case (figure 8) indicates that, in this case, the behaviour at high congestion is independent of any physical or otherwise attributes of the test network. This is reinforced by the erratic behaviour of the simulator when varying the presented load and link capacity simultaneously (section V-B, figures 11, 12). These plots show that the odd behaviour occurs at seemingly random load levels (1Mbps, 2Mbps and 5Mbps are very similar patterns). While we cannot easily verify the 5Mbps behaviour on the lab network, the more or less identical results for the 2Mbit plot imply that we don’t really need to. This behaviour (a slow increase in latency until after link saturation occurs, see section V-A.2), is not intuitive. We expect poor network performance at congestion levels greater than about 90%.

Overall, the validation network has been a success. We did not expect as close a correlation in the balanced case as we have seen (figure 6). The research has both given us confidence in the simulations current results, and given us direction for possibilities to improve the simulator.

## B. Future Work

The results of this research has given us some insight into areas for future work on the simulator and lab network. Further investigation into the affect the actual lab network has on the results can be conducted by using different components. The methodology behind the traffic generation and data sampling models can also be inspected to verify that they are sound.

We plan to follow two directions after this study. The first is to extend the study into a few more characteristics of the simulation. Modelling of the US cloud and the use of a US based proxy are two of high interest to us. The first because it is a source of significant variation between the real network and out simulations and the second because our results in this area were particularly interesting (significant performance gains can be made by breaking the TCP between the client and the server into two by using a non-caching proxy, see [6][7]).

The other direction for future work is to continue our simulation programme, with the additional confidence that this validation has provided.

## REFERENCES

- [1] M.W. Pearson and A.J. McGregor, "Sensitivity analysis of event driven simulation results," in *PAM2001 A Passive and Active Measurement Workshop, Amsterdam, The Netherlands*, May 2001, pp. 132-144.
- [2] J. Curtis, J. Cleary, A. McGregor, and M. Pearson, "Measurement of voice over ip traffic," in *PAM2000 Passive and Active Measurement Workshop*, Apr. 2000, pp. 43-59.
- [3] Prigogine Ilya, *The End of Certainty: Time, Chaos, and the New Laws of Nature*, The Free Press, N.Y., 1998.
- [4] M. Arlitt, Y. Chen, R. Gurski, and C.L. Williamson, "Traffic modeling in the ATM-TN TeleSim project: Design, implementation, and performance evaluation," in *Proceedings of the 1995 Summer Computer Simulation Conference*, Ottawa, Ontario, July 1995.
- [5] W. Stevens, "TCP slow start, congestion avoidance, fast retransmit, and fast recovery algorithms," Tech. Rep. RFC2001, IETF, Jan. 1997.
- [6] M. Pearson and A. McGregor, *A simulation study of network architectures to support HTTP Traffic on Symetric high-bandwidth\*delay circuits*, chapter 3, pp. 19-25, C.S.R.E.A. Press, 2000.
- [7] A.J. McGregor, M.W. Pearson, and J.G. Cleary, "Improving the performance of HTTP over high bandwidth-delay product circuits," in *CNDS '01: Communication Networks and Distributed Systems Modeling and Simulation Conference Proceedings. Phoenix, Arizona*, Znati T. and Simon R.P., Eds., Jan. 2001, pp. 97-102.