NetPIPE: A Network Protocol Independent Performance Evaluator

Quinn O. Snell, Armin R. Mikler and John L. Gustafson
Ames Laboratory/Scalable Computing Lab, Ames, Iowa 50011, USA
`snell|mikler|gus@scl.ameslab.gov`

Abstract

This paper presents the design of NetPIPE, a new Network Protocol Independent Performance Evaluator. NetPIPE maps the performance of a network across a wide range and presents the data in a new manner. Its protocol independence allows for visualization of the overhead associated with a protocol layer. Using NetPIPE has led to the discovery of a deep plunge in ATM performance for certain transfer block sizes, which is examined in detail. The performance drop is also shown to exist at certain block sizes for FDDI. Both aberrations are not detected by other benchmarks.
Keywords Performance Analysis, Network, ATM, FDDI, Ethernet

1 Introduction

In recent years, much research has been directed towards evaluating the performance of high speed networks. [2, 3, 4, 5] The design of NetPIPE, a network protocol independent performance evaluator, has been motivated by the need to assess the performance of communication bound applications. NetPIPE helps answer questions that surround network communications inherent to these applications. These applications include file transfer and graphical simulations for display in a virtual reality environment, such as CAVE [13] applications, which require frame transfers from a compute server. While file transfer applications allow streaming of data, a graphical simulation requires blocks of data transmitted at regular intervals to maintain full-motion video. The size of each block and the number of frames per second are enough to specify a minimum network throughput required to maintain realistic animation.

With the applications in mind, several questions can be asked in reference to the network communication. For instance, how soon will a given data block of size k arrive at its destination? Which network and protocol will transmit size k blocks the fastest? What is a given network's effective maximum throughput and saturation level? Does there exist a block size k for which the throughput is maximized? How much communication overhead is due to the network communication protocol layer(s)? How quickly will a small (< 1 kbyte) control message arrive, and which network and protocol are best for this purpose?

The answers to such questions are not always straightforward and easy to obtain with current network performance tools. The two most popular tools, `ttcp` [3] and `netperf` [2], are based on the TCP/IP [7, 8, 9] communications protocol. While netperf has the ability to map network performance, comparing network protocols with these tools is difficult if not impossible. Finding the effective maximum bandwidth using ttcp is an exercise in delving into protocol internals. Knowledge of the appropriate buffer size, alignment address, and protocol settings is required to achieve data transfer at the effective maximum bandwidth.

With the various network types available (ATM, FDDI, HIPPI, Ethernet, etc.), it is difficult to select a network infrastructure which best satisfies an application's bandwidth requirement. The design of NetPIPE has been motivated by the need to select a network infrastructure for various types of applications and communication with a CAVE virtual reality environment. In addition NetPIPE provides for visualization of network performance and the information necessary to answer the above questions.

This paper presents NetPIPE and some of the results obtained through its use. In the next section, we present the NetPIPE driver and its underlying principles. Sections 3 and 4 consist of results obtained using NetPIPE in a variety of network infrastructures. A summary and conclusion with answers to the questions posed above can be found in Section 5.

2 NetPIPE Design

NetPIPE consists of two parts: a protocol independent driver, and a protocol specific communication section. The communication section contains the necessary functions to establish a connection, send and receive data, and close a connection. This part is different for each protocol. However, the interface between the driver and

[1]This work was supported by the Applied Mathematical Sciences Program of the Ames Laboratory, U.S. Department of Energy under contract number W-7405-ENG-82

protocol module remains the same. Therefore, the driver does not have to be altered in order to change communication protocols.

The driver is based on the principles presented by the HINT [1] computer performance metric(See Appendix A). Just as a computer's performance cannot be accurately described using a single sized computation, neither can the performance of a network be described using a single sized communication transfer. NetPIPE increases the transfer block size k from a single byte until transmission time exceeds 1 second. Hence, NetPIPE is a variable time benchmark and will scale to all network speeds. Unlike fixed size benchmark tests, NetPIPE will not become outdated and inaccurate as technology advances (see Gustafson [6]). To increase the universality of NetPIPE, information is measured in *bits* rather than *bytes*. The definition of *byte* varies more than one might think.

For each block size c, three measurements are taken: $c-p$ bytes, c bytes, and $c+p$ bytes, where p is a perturbation factor with a default value of 3. This perturbation allows analysis of block sizes that are possibly slightly smaller or larger than an internal network buffer. For each measurement, NetPIPE uses the following algorithm:

```
/* First set T to a very large time. */
T = MAXTIME
For i = 1 to NTRIALS
    t0 = Time()
    For j = 1 to nrepeat
        if I am transmitter
            Send data block of size c
            Recv data block of size c
        else
            Recv data block of size c
            Send data block of size c
        endif
    endFor
    t1 = Time()
/* Insure we keep the shortest trial time. */
    T = MIN(T, t1 - t0)
endFor
T = T/(2 * nrepeat)
```

The variable *nrepeat* is calculated based on the time of the last data transfer. The intent is to repeat the experiment enough times such that the total time for the experiment is far greater than timer resolution. The default target time is 0.5 seconds. For most modern computers, this provides a sufficiently precise data transfer time. Given that the last transfer time was *tlast* seconds for a block size $bsz1$, the value of *nrepeat* for block size $bsz2$ is approximated as:

$$nrepeat = TARGET/((bsz2/bsz1) * tlast)$$

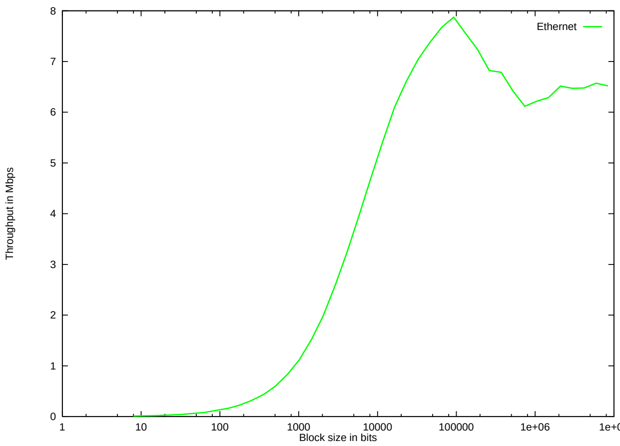

Figure 1: Ethernet Throughput

NetPIPE uses a ping-pong transfer like Hockney [11] uses for each block size. This forces the network to transmit just the data block without streaming other data blocks in with the message. The result is the transfer time of a single block, thus providing the information necessary to answer which block size is best, or what is the throughput given a block of size k.

NetPIPE produces a file that contains the transfer time, throughput, block size, and transfer time variance for each data point and is easily plotted by any graphing package. For instance, Figure 1 presents the throughput versus the transfer block size for a typical Ethernet link. This graph is referred to as the *throughput graph*. From this graph, it is easy to see the maximum throughput for this network is approximately 7.8 Mbps. However, it is difficult to analyze the latency, an equally important statistic.

A graph that is easier to read and analyze is the *network signature graph*. One such graph is shown in Figure 2. It depicts the transfer speed versus the elapsed time; hence it represents a network "acceleration" graph. This graph is a new and unique way of viewing network performance data; the key is to use a logarithmic time scale horizontally instead of the transfer block size. In this graph, as in all graphs presented, time is plotted in seconds. It is very similar to the way computer performance is presented by the HINT performance metric. Although unconventional, this graph represents perhaps a better approach to visualizing network performance. All the necessary data are clearly visible and easy to extrapolate. The network latency coincides with the time of the first data point on the graph. The maximum attainable throughput is clearly shown as the maximum point on the graph.

Plotting the block size versus the transfer time on a logarithmic scale for both the x and y axis, as in Figure 3, reveals what we define as the *saturation point*. This is the point after which an increase in block size results

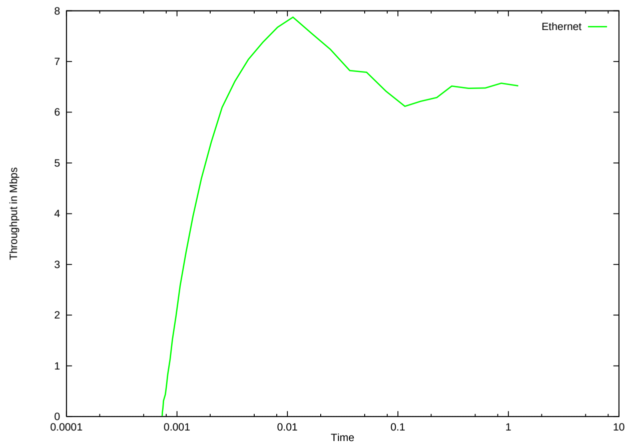

Figure 2: Ethernet Signature Graph

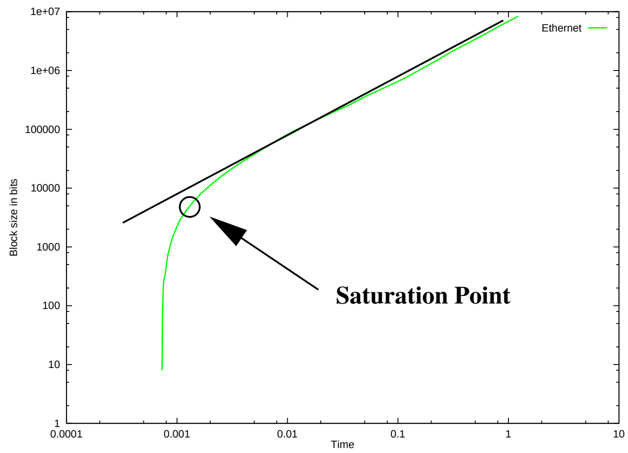

Figure 3: Ethernet Saturation Graph

in a near-linear increase in transfer time, effectively the knee of the curve. The time interval between the saturation point and the end of the recorded data is referred to as the *saturation interval*. In this interval, the graph monotonically increases at a constant rate i.e., the network throughput cannot be improved upon by increasing the block size.

Using maximum effective bandwidth to compare networks (or even worse, nominal bandwidth) is much like using peak megahertz ratings to compare computers. While it may be correct for ranking certain applications, in general, its accuracy leaves much to be desired. A given network may have a high maximum effective bandwidth but also have a high latency. So a network with a lower latency would possibly be better for small messages even though it has a much lower maximum effective bandwidth. This effect can be readily observed when comparing ATM with Ethernet, as shown below. Any ranking based on a single number does not provide sufficient insight for accurate network comparison. For network tuning and comparison, we recommend taking the entire NetPIPE signature graph together with appli-

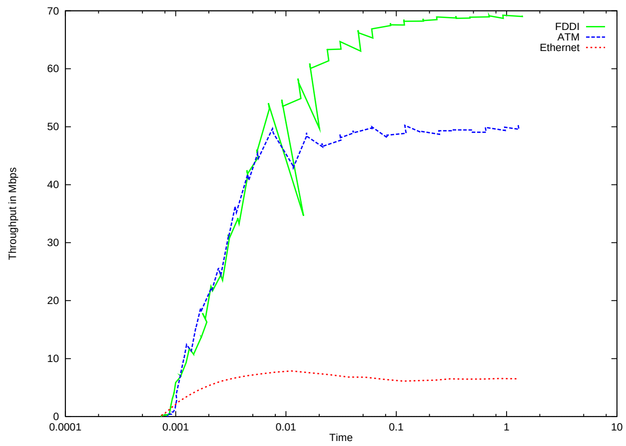

Figure 4: Signature Graphs for FDDI, ATM, and Ethernet

cation specific information into consideration.

3 Results

Figure 4 shows the signature graph for Ethernet, ATM, and FDDI networks using the TCP/IP communication protocol. All the data were collected by executing NetPIPE on two identical SGI Indy workstations. The network in each case consisted of a dedicated, noise free link between the two machines. ATM communication was performed via FORE [12] ATM interface cards using the FORE IP communication interface. Communication via the FDDI network yields the highest attainable throughput followed by ATM and Ethernet. However, notice that Ethernet has a lower latency, implying that Ethernet can outperform ATM for small messages. Ethernet latency is on the order of 0.7 ms followed by ATM at near 0.9 ms.

The reader may be alarmed to see that the signature graph is not univalued a function of time. This is not an anomaly, but an indication that a *larger* message can indeed take *less* time to transfer because of system buffer sizes and the interaction with the operating system. The phenomenon is repeatable. One suspects that it indicates the need for improvement in system and messaging software, since a superset of a task should always take longer than the task by itself.

In order to examine this further, Figure 5 presents the saturation graph. It verifies the latency order and also shows that for messages up to approximately 8 K bits, Ethernet has the shortest transmission time. It should be emphasized that all the experiments were executed on dedicated network connections.

The results presented in Figure 5 were significant enough to attempt verification by an application that

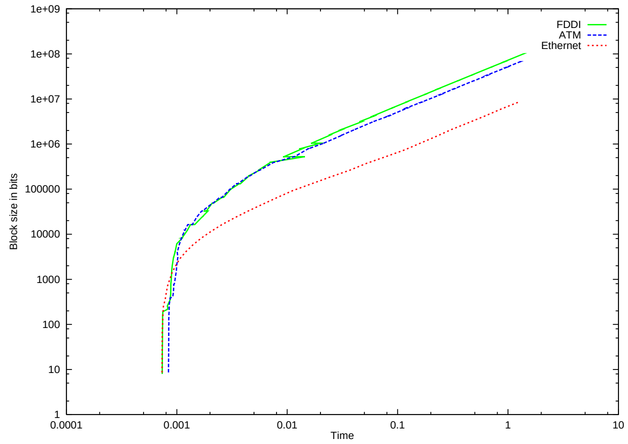

Figure 5: Saturation Graphs for FDDI, ATM, and Ethernet

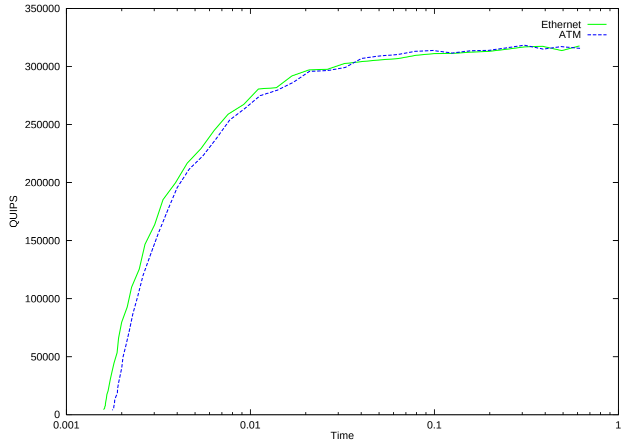

Figure 6: HINT Graphs for Ethernet and ATM based communication

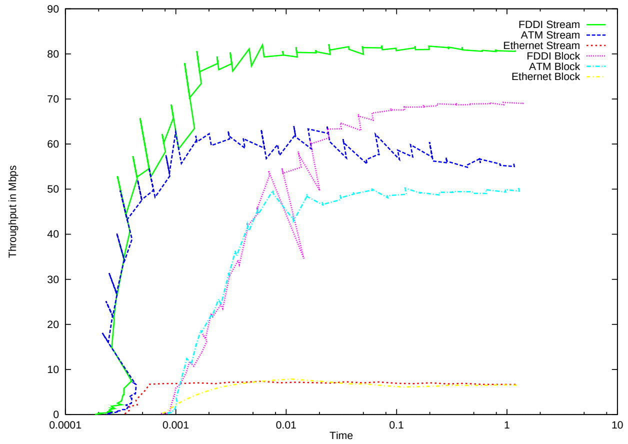

Figure 7: Block Transfer vs. Streaming Transfer

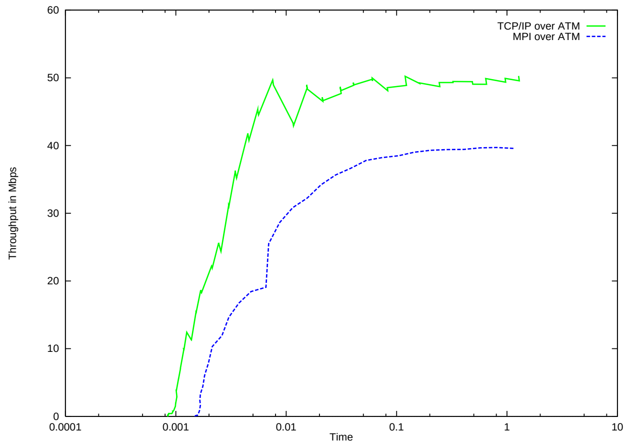

Figure 8: Protocol Layer Overhead

uses small messages. For such an application, one would expect better performance using a dedicated Ethernet connection than using a dedicated ATM connection. The ideal application for this purpose is the HINT benchmark. The communication in HINT is a global sum collapse of two double precision floating point numbers. Using the same pair of SGI INDY workstations, HINT was run using the Ethernet link and the ATM link. In each case, the links were dedicated and the configuration was identical to that used for the NetPIPE tests. The HINT QUIPS graphs for each configuration are shown in Figure 6. The Ethernet configuration is able to come up to speed sooner that the ATM configuration, and as a result, the Ethernet configuration produces better HINT performance.

The graph shown in Figure 7 depicts the differences in network throughput for block and stream transfer modes. NetPIPE simulates streaming data transfer by executing a series of sends in rapid succession without acknowledgment at the application level. In block transfer, each block is sent to the receiver, which returns the message. Figure 7 presents the signature graphs for Ethernet, FDDI, and ATM, for both streaming and block transfer modes. In streaming mode, FDDI provides the largest throughput for all block sizes. We surmise that this is due to the large network cells used by FDDI. This is important information for application programmers looking for a network solution. If the application involves streaming data across the network, FDDI presents the best solution for transferring data via a dedicated link.

4 Discoveries using NetPIPE

A driving force behind the development of NetPIPE has been protocol independence and the ability to accurately compare different protocols. The resulting bandwidth graphs for MPI [10], the message passing interface, and TCP are presented in Figure 8. All data were obtained using the same machines and all communication was over a dedicated ATM fiber pair. This graph demonstrates the effectiveness of NetPIPE to compare totally different protocols.

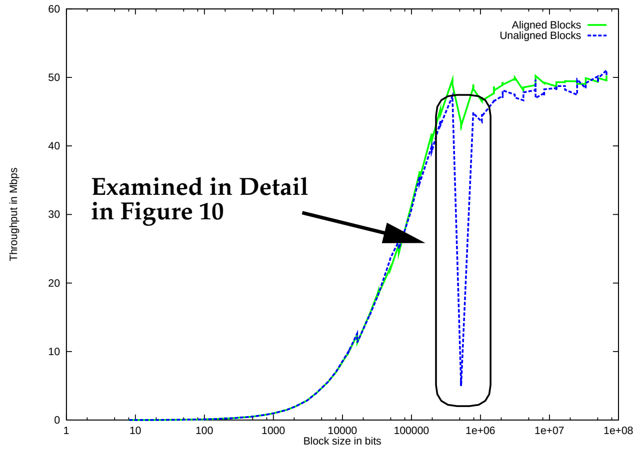

Figure 9: Page Aligned vs. Unaligned Transfer Block Throughput

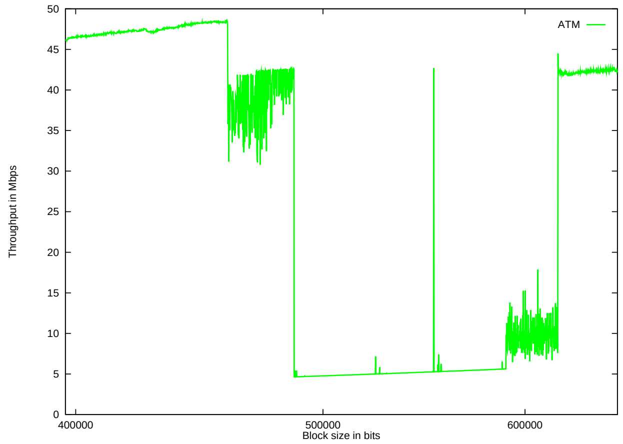

Figure 10: A Detailed Examination of the ATM Performance Dip

Often a programmer uses a communication package to avoid working with the details of setting up connections. While ease of use is clearly gained, naive use of these extra protocol layers adds communication overhead, thus reducing the network throughput. This protocol layer overhead is clearly evident in the signature graphs. The MPI library used was based on TCP, but clearly an application program pays for its ease of use by sacrificing latency and bandwidth. This sacrifice drops the aggregate bandwidth as well. The tradeoff of ease of use and throughput is currently being investigated for TCP and ATM's AAL5 application programmers interface. Nevertheless, the overhead associated with a protocol layer is now easy to visualize.

The design and use of NetPIPE has revealed interesting network anomalies and tendencies. In particular, NetPIPE demonstrated the significance of data block alignment to page boundaries. This data is shown in the signature graphs for ATM using aligned and unaligned data in Figure 9 Page aligned data blocks yield a maximum throughput that is only slightly in creased. However, note the large plunge in performance using unaligned data.

NetPIPE has the option of specifying a starting and ending transfer block size and the increment value. This option allows for a closer examination of the dip in performance due to unaligned data. Figure 10 shows throughput plotted versus transfer block size. There are three distinct regions in the graph. On either side of the chasm, the block transfer is at normal speed. For block sizes of approximately 59 K bytes to 72 K bytes, the throughput is a dismal 5 Mbps. Also note the chaotic transition regions between the two performance levels. The single data point of high throughput inside the chasm is at a block size of 67.4 bytes. The reason for an increase in throughput for that single measurement is not known, and the cause of the performance drop has not been fully investigated at this time. However, the performance plunge does appear to be linked to the TCP socket buffer size. Changing the socket buffer size moves the dip to a different portion of the graph, and aligning the data to page boundaries effectively removes it. Other studies [4, 5] have missed the performance chasm by not evaluating enough data points or always using page aligned data.

Another graph of interest is the comparison of FDDI block transfer on different architectures. Figure 11 shows the signature graphs for transfer between two identical DEC 3000 workstations in comparison to the SGI data previously shown. In both cases, the transfer blocks were aligned to page boundaries. There are three differences that are important to observe: 1) The DEC FDDI has a performance dip similar to the ATM data, 2) The latency for the DEC workstations is smaller, and 3) Regardless of the lower latency, the maximum throughput for the DEC machines is much less than that attained by the SGI workstations. Vendor defaults were used throughout the experiments. There may be some internal parameters that can be adjusted for the DEC machines to improve their overall performance.

5 Conclusions

NetPIPE readily provides the information necessary to answer the questions posed at the beginning of this paper. Also, there are various other questions concerning network performance which can be answered by careful examination and interpretation of the signature and saturation graphs generated by NetPIPE.

NetPIPE encapsulates the best of `ttcp` and `netperf` and gives a visualization of the network performance. Most importantly NetPIPE is clearly a protocol independent performance tool. It is valuable when compar-

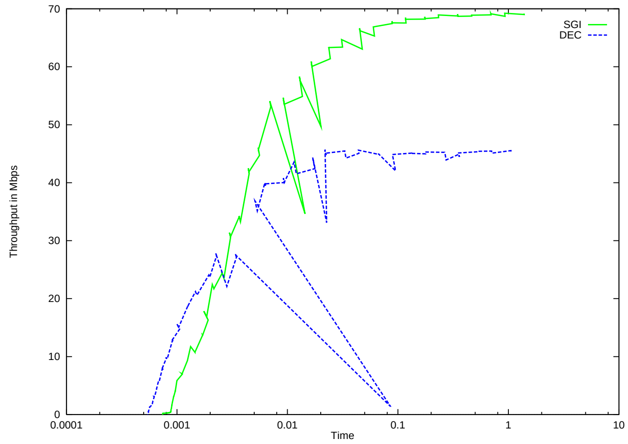

Figure 11: FDDI Block Transfer Comparison of SGI and DEC

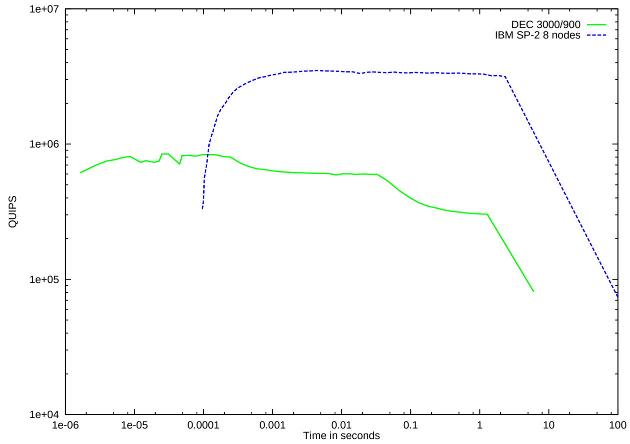

Figure 12: An Example of a HINT QUIPS Graph

ing different networks and protocols. Using NetPIPE, we have clearly shown the overhead associated with different protocol layers. While ease of use is gained by uniform protocols, network bandwidth and latency are measurably sacrificed. We also foresee being able to visualize the difference in performance for other network protocols as well: token ring, HiPPI, etc.

References

[1] Gustafson, J. and Snell, Q. "HINT: A New Way to Measure Computer Performance", *Proceedings of the 28th Annual Hawaii International Conference on Systems Sciences*, IEEE Computer Society Press, Vol. 2, pages 392-401.

[2] Netperf, http://www.cup.hp.com.

[3] ttcp, http://www.epm.ornl.gov/~batsell/NB.html.

[4] Krivda, C. *Analyzing ATM Adapter Performance The Real-World Meaning of Benchmarks.* http://www.efficient.com/dox/EM.html.

[5] Huang, C. and McKinley, P. *Communication Issues in Parallel Computing across ATM Networks.* Technical Report, Michigan State University, MSU-CPS-94-34, June 1994.

[6] Gustafson, J. "The Consequences of Fixed Time Performance Measurement", *Proceedings of the 25th Annual Hawaii International Conference on Systems Sciences*, IEEE Computer Society Press, Vol 3, pages 113-124.

[7] Stevens, W. *TCP/IP Illustrated, Volume 1.* Addison-Wesley, Reading, Massachusetts, (1994).

[8] Stevens, W. *Unix Network Programming.* Prentice-Hall, Englewood Cliffs, NJ. (1990).

[9] Comer, D. *Internetworking with TCP/IP; Principles, Protocols, and Architectures.* Prentice-Hall, Englewood Cliffs, NJ. (1988).

[10] Gropp, W. Lusk, E. Skjellum, A. *Using MPI.* MIT Press, Cambridge, Massachusetts, (1994).

[11] Hockney, R. "Performance Parameters and Benchmarking of Supercomputers". *Parallel Computing*, Volume 17, 1991, pages 1111-1130.

[12] FORE Systems, http://www.fore.com.

[13] C. Cruz-Neira, D.J. Sandin, T.A. DeFanti, "Surround-Screen Projection-Based Virtual Reality: The Design and Implementation of the CAVE," *Proceedings of SIGGRAPH '93*, ACM SIGGRAPH, August 1993, pages 135-142.

6 Appendix A

The HINT performance metric was developed at Ames Laboratory to gauge the overall performance of a given machine. It fixes neither the problem size nor the execution time of the problem to be solved; it measures the performance of a computer at all levels of memory. Figure 12 shows a HINT graph for a typical workstation and a small parallel supercomputer. The graph plots the QUality Improvement Per Second (QUIPS) versus the log of the time it took to obtain a answer of given quality. The use of the *log* of time weights smaller times more heavily. A workstation starts quickly and thus has a higher initial QUIPS. The supercomputer, on the other hand, does not reach its peak QUIPS value until much later due to communication overhead. In general, the area under the QUIPS graph is the net performance and is summarized in a single number called the Net QUIPS. A more complete discussion of HINT can be found in [1] or on the HINT homepage at `http://www.scl.ameslab.gov/HINT`.