Very High Performance Networking for Supercomputing

Marke Clinger
Solbourne Computer
1900 Pike Road
Longmont, CO 80501
marke@solbourne.com

Abstract

NASA Ames has installed a very high bandwidth, 1 gigabit/second, local area network provided by Ultra Network Technologies to study the feasibility and performance of networking supercomputers with minisupercomputers and workstations at the effective bandwidth needed for computing environments of the future. A network display device, which provides interactive graphics over the network, was also installed to study the impact of this new bandwidth on visualization and the impact of interactive graphics on network performance.

Three classes of computers were defined for study: workstations, minisupercomputers, and supercomputers. Interactions between these classes in a typical operational environment were also studied. This paper presents the results of investigations of network performance in a production facility through distributed processing.

1.0 Introduction

The 1980s brought the computer community enormous increases in CPU speed, physical memory size, and disk storage capacity. However, one component of the computer system was left behind in this quest for speed and capacity, the network. With the cost and power of today's computers, networking has become a necessity. Accessing a supercomputer at 10 Mbits/second does not allow the user to fully utilize the power of the computing resource. Low-cost workstations can perform well as a display medium for visualization tasks, using a supercomputer to perform many of the CPU-intensive mathematical calculations. This is not very feasible when the bandwidth and throughput of the network cannot support the amount of data that needs to be downloaded to the workstation.

Most supercomputer centers around the world currently use three types of computers to perform the necessary computing tasks: supercomputers, minisupercompuers, and workstations. All but workstations are shared on a large scale, creating a need for a large, high-speed data path into these computers. Visualization has made its way into most aspects of research and many portions of industry. This type of computing environment has brought on the need for a super network.

The NASA Ames Research Center Numerical Aerodynamic Simulation (NAS) Facility has installed such a network and commissioned a study of its characteristics. This network consisted of three computer classes: supercomputer-class (Cray Y/MP and a Cray-2), minisupercomputers-class (Alliant FX-8 and a Convex C1), and workstation-class (two Sun Workstations). Also installed on the network is a network graphics display device, which is a network based graphics frame buffer.

The study done for NAS consisted of four segments: Subtask 2.1 - *Measurement of Effective Transfer Rates*, Subtask 2.2 - *Performance Analysis of Network Architectures*, Subtask 2.3 - *Demonstration of Remote Disk Operation and Functionality*, and a *Final Report*. Subtask 2.1 included memory-to-memory, disk-to-disk, and disk-to-memory data transfer performance analysis between the three computer classes. This subtask also looked into the utility of a network-based graphics display device and its impact on the network. Subtask 2.1 also measured the effect of network load on individual high-speed data transfers. Subtask 2.2, network architecture analysis, looked into the performance difference between host-based and network-based protocol processing and the impact of each on the host system. An explanation for the difference was also sought out. Subtask 2.3 studied the usefulness of a gigabit network for remote file access. The final report draws conclusions from the data in the three subtasks.

2.0 Test Methodology

This section briefly describes the network, computer hardware, and software used in all three phases of the study.

2.1 Network Configuration

A prototype very high speed network was installed at NAS to support this study. The configuration of this network is shown in Figure 1. Two UltraNet 1000 Hubs where used with a Cray Y/MP and the UltraNet network graphics display device on one hub and the Cray-2, Alliant, Convex, and two Sun Workstations on the second hub. Ultra HSX channel adapters were used to connect the two Cray computers to the network. The two minisupercomputers and workstations were connected to the network via the Ultra VMEbus host adapter. The hubs were interconnected with a four-way striped optical fiber cable and all VMEbus-based hosts were connected with coaxial cable. The optical fiber length was 1 kilometer.

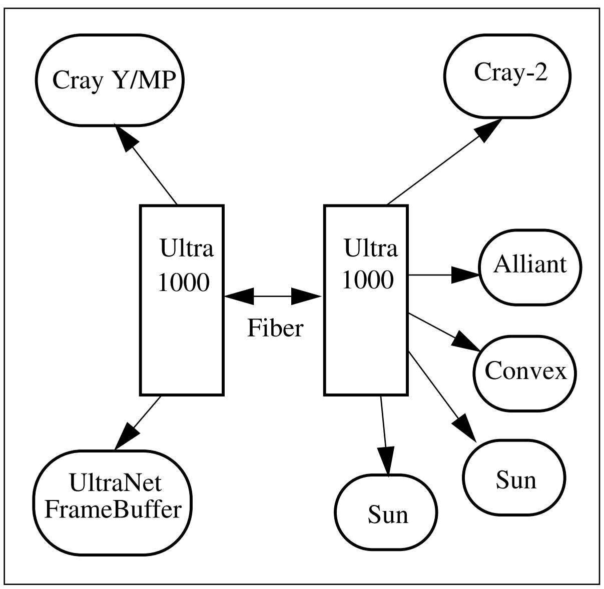

Figure 1 - NAS Gigabit Network Configuration

The first computer class, workstations, consisted of two Sun 3/260s each with 8 megabytes of main memory and SMD disks. An Alliant FX-80 and a Convex C-120 formed the minisupercomputer computer class. The Alliant was equipped with 128 megabytes of main memory and four Multibus disks striped two ways. The Convex C-120 had 128 megabytes of main memory and four VMEbus disk drives striped four ways. The host used in the supercomputer class was a Cray-2 with four CPUs, a 4.1 nanosecond clock, and one path to memory per CPU that consisted of 2 gigabytes.

This network configuration was used for Subtask 2.1 and Subtask 2.3. Subtask 2.2 used an UltraNet 250 Hub with one Sun 4/260 and one Sun 3/160 workstation.

2.2 Test Suite

The software programs *tsock, ufb, mem2mem_rcv, mem2mem_tns, dsk2dsk_tns, dsk2dsk_rcv,* and *vmstat* were used to collect the data. *tsock* is a sockets-testing program that transfers a specified amount of data over a network and reports statistics about the data transfer rate. *ufb* is a test program for the network graphics display device. It can be used to send a variety of data to the frame buffer. *mem2mem_rcv*, *mem2mem_tns, dsk2dsk_tns, and dsk2dsk_rcv* are shell scripts that control the *tsock* program. These scripts generate data points by varying several parameters, most notably the user buffer size and number of processes to start simultaneously. The two mem2mem programs performed memory-to-memory data transfers while *dsk2dsk_tns* and *dsk2dsk_rcv* performed similar data transfers while also reading and writing to disk. These programs were written by Ultra Network Technologies. *vmstat* is a standard UNIX utility that generates system resource usage statistics. This command was embedded into the *mem2mem_rcv* and *mem2mem_tns* to keep track of the system impact during network data transfers.

2.3 Effective Performance

Effective performance is measured at the user application level in the protocol stack. All transfers include the establishment of the connection to the remote host, completion of the data transfer, including packing and unpacking of protocols, and disconnection. This method provides performance numbers that can be realistically seen by users in a fully operational environment.

2.4 Channel-Oriented vs. Bus-Oriented Data Transfers

Two communication architectures are used in this gigabit network, channel-oriented and bus-oriented, to match those of the computer systems to which the network attaches.

Channel-oriented interfaces are used in systems with a master/slave I/O philosophy: I/O devices are typically not intelligent, all data transfers can be initiated only by the CPU. The Cray HSX is a channel-oriented interface, providing separate input and output channels.

Bus-oriented interfaces are used in systems with a peer-to-peer I/O philosophy: I/O devices are relatively intelligent, they have direct access to host memory, and data transfer is typically initiated by the device itself. The bus is a shared resource with input and output data travelling over the same path. The VMEbus systems (Alliant, Convex, and Sun) are of this interface type.

2.5 Loopback Data Transfers

Data transfers are normally performed between two different computers. In this test there was one Cray-2, one Cray Y/MP, one Convex, and one Alliant. In order to gather data that reflected the data transfer between like machines loopback mode was used. In this mode the sending and receiving processes where executed on the same machine. On the bus-oriented machines, the results from each data transfer were added for the final throughput measurement. This was done because a bus is a shared resource. On channel-oriented machines, a mostly non-shared resource, the results of only the receiver were used. Although neither one of these methods is entirely accurate, they are useful in predicting the data transfers if two of each computer were available.

2.6 Data Transfer Architecture

In Subtask 2.1 the overall throughput from the user memory on one system to the user memory on another system was studied. Subtask 2.2's goal is to understand the impact of network architecture on the data transfer throughput and host resources. There are two data paths possible on the gigabit network, which are illustrated in Figure 2.

The protocol stack modifications needed to integrate an ISO TP4-based gigabit network are reflected in Figure 2. Two components, which are not part of the standard protocol stack, are shown in this figure: the sockets compatibility library (SCL) and the network driver software. The SCL allows a user with a standard sockets program to relink and use the gigabit network without knowing anything about the adapter driver or ISO TP4. This provides two new data paths to the user direct data path and indirect data path, as well as the standard Ethernet data path.

When the network driver is built into the system kernel, the parameter *minphys*, the maximum transfer size supported by the VMEbus adapter, is set to 128 Kbytes. When a data transfer is initiated using the direct path, the adapter driver sends a request block (RB) to the adapter and locks down the amount of memory equal to *minphys*. An RB, which contains no user data and asks for various services from the adapter, tells the adapter that a buffer is ready for transfer. For example, when a 512-Kbyte user buffer needs to be transferred, the RB points to the 128-Kbyte locked down portion. the adapter then uses direct memory access (DMA) to transfer this buffer from memory in Transport Protocol Data Unit (TPDU) size chunks, packetizes the data, performs checksums, and sends it across the network. When the transfer of this 128-Kbyte buffer has completed, the adapter sends an RB back to the kernel, interrupting the CPU. Transferring a 512-Kbyte user buffer over the direct data path causes four interrupts (512/128 = 4) on a Sun workstation. This data transfer model is the same for most bus-oriented systems. Notice that for efficiency, the data movement occurs directly between the user buffers and the adapter without any intermediate copying.

The main variance in this scheme for the indirect path is that the *minphys* size is the same as the maximum transmission unit (MTU) size, 8.4 Kbytes (1 disk cache buffer plus one NFS header). Now, instead of dividing the 512 Kbytes into 128-Kbyte chunks, it is divided into 8.4-Kbyte chunks, causing a total of 60 interrupts to transfer the same user buffer. The Ethernet network has an MTU of 1.5 Kbytes, or 341 interrupts, to transfer the 512-Kbyte buffer.

Data transfers over the direct data path on channel-oriented systems follow a slightly different model. The adapter transfers 32 Kbytes (one TPDU) of information from memory. Each TPDU creates an interrupt to the CPU, plus the interrupt from the RB telling the network driver that the transfer has completed. Therefor, transferring the 512-Kbyte buffer caused 17 interrupts to the CPU (buffer size/TPDU + RB or 512/32 + 1 = 17).

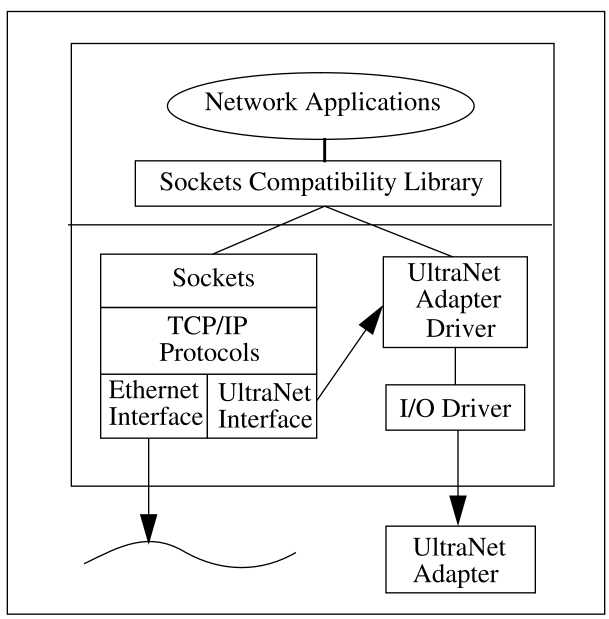

Figure 2 - Data Paths: Host Memory to Network

The data flow of each path is:

Ethernet Data Path

An application makes a socket call such as write which causes data to be copied into the kernel socket managed buffers (MBUFS). Protocol headers, trailers, and checksums are performed by kernel-resident TCP/IP software. Then the TCP/IP software contacts the Ethernet driver, which performs the data transfer over Ethernet.

Direct Data Path

Data movement for the direct data path is different from the host-resident networking software, thus avoiding the bottlenecks imposed by data copying and kernel-buffer management. Therefore, when an application does a write on a socket, the network adapter driver is told where in memory the data resides. The adapter then DMAs the data from memory. The protocol processing task are done by the adapter as the data is being transmitted (Hathaway, 1990).

Indirect Data Path

The indirect data path is similar to the Ethernet data path with the exception of who performs the actual data transfer and the transmission unit size. In the gigabit network the TCP/IP software hands the datagram to the adapter driver to be transferred using an MTU of 8.4 Kbytes.

3.0 Synopsis of Results

With the background information presented in Section 2.0, the performance of this gigabit network can now be characterized.

Subtask 2.1:
Measurement of Effective Transfer Rates

Subtask 2.1-*Measurement of Effective Transfer Rates*, studied the effective performance and feasibility of networking supercomputers, minisupercomputers, and workstations. Visualization, along with its impact on the network and the impact of the network on visualization, was studied using the network graphics display device. The overall goal of the project was to "demonstrate effective performance at least one order of magnitude faster than networks currently in use at supercomputer centers and to measure these rates in an operational environment."

The tests consisted of memory-to-memory and disk-to-disk data transfers. These tests were performed with and without network traffic. Three methods were used to generate a constant load on the network: a continuous, full-speed transfer to the network graphics display device from a supercomputer, a transfer between two hosts on the Link Adapter, and a transfer between a supercomputer and a host system on a Link Adapter. While the network was under load, the test suites, which had been executed previously without traffic, were again performed to collect a new set of data points.

Effective network performance was defined to be the throughput that the user can realize or the data rate at the application layer. This was the main focus of Subtask 2.1. Measuring effective performance was accomplished using the *tsock* program, which transfers data from one host to another over the desired network data path. This data can be read from local memory and sent to remote memory or read from local disk and sent to remote disk. Single and double buffering was used to measure the data transfers. Currently, single buffering is the most commonly used method; however double buffering is optimum for the network adapters (Clinger, 1989, page 18). The utility *ufb* was used to measure the data rate to the network graphics display device.

The two goals of Subtask 2.1 were to show effective performance between host pairs and to prove the theory of "non-impact." Theoretically, this gigabit network can transfer data on a shared data path between several hosts without impacting the throughput of any of the individual communications, provided the total bandwidth of 1 gigabit/second is not exceeded.

Workstation Results

A memory-to-memory transfer over Ethernet was measured at 415 Kbytes/second is compared to 4000 Kbytes/second (4 Mbytes/second) over the gigabit network. The single-buffer transfer came in at 3500 Kbytes/second, 500 Kbytes/second below the double-buffered case. When Sun-to-Sun memory transfers were executed on a loaded gigabit network no performance degradation was seen. This is true when either the Link Adapter or the UltraBus backplane is under a full load. In a full-duplex measurement, all processes divide the maximum throughput of 4000 Kbytes/second equally, for an aggregate bandwidth of 4000 Kbytes/second. When a Sun transfers to a minisupercomputer, it kept the same data rate characteristics seen during a Sun-to-Sun transfers. The 500 Kbytes/second data spread between single- and double-buffer memory transfers disappeared when the Sun received data from the Cray-2. In the Sun-to-Cray-2 case the Sun received data into its memory 500 Kbytes/second faster than the maximum Sun-to-Sun memory transfer rate of 4500 Kbytes/second.

Local disk reads occurred at 833 Kbytes/second for a 10-Mbyte file and 625 Kbytes/second, 75% of local disk speed, over the direct data path. With the exception of a few minor variances in data rates between the Sun and the minisupercomputer and between the Sun and the supercomputer, the disk transfer rates were similar to those seen during the Sun-to-Sun tests. This shows that the disk access speed is the limiting factor in the data transfer.

Minisupercomputer Results

The Alliant and Convex both had a single-buffered Ethernet memory-to-memory transfer rate of 69.5 Kbytes/second over Ethernet. The Ethernet double-buffer case averaged 67 Kbytes/second for both minisupercomputers. On the gigabit network, in loopback, the Alliant performed single-buffered memory-to-memory transfers at 6000 Kbytes/second. It performed double-buffered data transfers at 12000 Kbytes/second. On the gigabit network, in loopback, the Convex performed a single-buffered memory-to-memory data transfer at 6000 Kbytes/second and the double-buffered transfers at 8000 Kbytes/second. When transferring between these two machines, single-buffered transfers occurred at a rate of 4000 to 5000 Kbytes/second, while double-buffered transfers occurred at 8000 Kbytes/second. Data rates between these machines did not fluctuate during the network load testing. When the Cray-2 transmitted to the Alliant, the data rate followed the Alliant loopback rate very closely. When the Alliant transmitted to the Cray-2, both the single- and double-buffer cases overlapped at the data rate that was previously only obtained during a loopback transfer with double buffers. The Convex received data from the Cray-2 500 Kbytes/second faster for both single and double buffers than the loopback case. When the Convex transmitted to the Cray-2, the data rates were the same as the Convex loopback data rates.

Ethernet disk transfer rates between the Alliant and Convex varied from 66.5 to 105 Kbytes/second. The Alliant local disk transfer rate was measured at 2000 Kbytes/second for files over 1 Mbyte. The Convex transferred a 10-Mbyte file at 5000 Kbytes/second; it transferred a 1-Mbyte file at 2000 Kbytes/second. Alliant disk-to-memory transfers, at 1500 Kbytes/second, preserved 75% of the local file transfer rate. Although the Convex preserved only 50% of the disk data rate during a disk-to-memory transfer, it provided the user with 2500 Kbytes/second. When these two machines were put up against a faster computer, each reacted differently. The Alliant maintained its 1500 to 2000 Kbyte/second transfer rate, while the Convex jumped 1500 Kbytes/second in each direction. This means that, if the Convex is the receiver or transmitter, it has a maximum disk data transfer rate over the network, with a 10 Mbyte file and a user buffer size of 512 Kbytes, of 4000 Kbytes/second. The Alliant-to-Cray-2 peak disk data rate, 2000 Kbytes/second, occurred with a 10-Mbyte file and a user buffer size of 1 Mbyte.

Supercomputer Results

Initial Testing between two Cray-2s running UNICOS 4.0 yielded a maximum throughput in memory-to-memory transfers of 52,000 Kbytes/second. NAS upgraded the software on the Cray-2 and traded out the second Cray-2 for a Cray Y/MP before official testing began. Under this new operating system memory data transfer measurements between the Cray Y/MP and the Cray-2, with both computers under a very heavy load, had an average data rate of 30,000 Kbytes/second. Individual loopback numbers for the Cray-2 and the Cray Y/MP were 40,000 Kbytes/second and 25,000 Kbytes/second, respectively. Because two separate resources are used for input and output on a channel-based system, the input and output numbers were not added as was done in the VMEbus tests. Based on these results, it is apparent that the loopback data rate is lower than the actual overall data rate obtainable by that computer architecture.

Disk transfer rates are more difficult to characterize because of the impact of system load on disk access performance. Local file copies of 100-Kbyte, 1-Mbyte, and 10-Mbyte file produced data rates of 1177 Kbytes/second, 4572 Kbytes/second, and 6687 Kbytes/second respectively. During the Cray-2 disk-to-memory transfer of these files over the gigabit network, the performance was the same for the 100-Kbyte and 1-Mbyte file. for the 10-Mbyte file, a data rate of 15,000 Kbytes/second was measured. This was measured on several different days during the test period, so it is believed that it is more accurate than the local copy. The local copy may have been adversely affected by the system architecture.

Network Load Results

Placing a load on shared data paths on the gigabit network did not affect the performance of individual data transfers. All the hosts used in this test were able to transmit simultaneously at their peak data rates for a total network throughput of 106,000 Kbytes/second. This is not the maximum of the gigabit network, rather all that the hosts in this configuration could provide. The network graphics display device was able to display data from the Cray-2 at 96,500 Kbytes/second.

Subtask 2.2: Performance Analysis of Network Architectures

This subtask explored the direct, indirect, and Ethernet data paths. The direct data path bypasses host protocol processing. All protocol processing is done on the network. The indirect and Ethernet data paths traverse the entire TCP/IP protocol stack before performing the network data transfers. This places the burden of protocol processing on the host.

The first part of Subtask 2.2 examined the performance difference between host-resident and network-resident protocol processing. On the Sun 4/260 workstation, the direct data path had a throughput of 4000 Kbytes/second, while the indirect data path had a throughput of less than 500 Kbytes/second. This is attributed to two factors: host-based protocol processing and Maximum Transfer Unit (MTU) size. Performing protocol processing on the host system requires CPU cycles to process the TCP/IP functions. Host-resident protocol processing also causes more interrupts because of the smaller MTU.

The second part of Subtask 2.2 analyzed the effects of network architecture on the host system. In this subtask, *vmstat* was used to gather statistics about the resource usage on the host system, a Sun 4/260. It was found that the direct data path had the lowest impact on the host system, while the indirect data path had slightly greater impact. The Ethernet data path had the most impact on the host system.

Of the four categories of host system resources studied in this subtask, faults and CPU usage presented the most interesting results. Tables 2.1 and 2.2 summarized the statistics collected in this subtask.

The important statistics are as follows: interrupts per second, system calls per second, and context switches per second. During a data transfer using a 128-Kbyte user buffer, the direct data path cased 34 /second, indirect data path cased 290 interrupts/second (8 times the direct data path), while Ethernet caused 640 interrupts/second (19 times direct data path). The indirect and Ethernet data paths, using a user buffer size of 128 Kbytes or less, caused 150 system calls/second, while the direct data path caused one-fifth as many (30 systems calls/second). Context switches per second decreased from 140 to 48 when 128-Kbyte or larger user buffer sizes were used on the direct data path. The indirect data path performed consistently at 77 system calls/second, while Ethernet caused 110 system calls/second for all user buffer sizes.

CPU usage was the same for the two data paths that performed host-resident protocol processing, 45% system time usage with 4% user, leaving 51% idle time. The direct data path left 70% idle time, with 30% being used by the system to perform the data transfer.

	Summary of Host Impact			
Data Path	Interrupts/second	Cxt Switches/second	System Call/second	CPU Utilization
Direct Data Path	64	140	30	30%
Indirect Data Path	290	77	150	51%
Ethernet Data Path	640	110	150	51%

Table 2.1 - Summary of Network Architecture Host Impact (buffer size < 128 Kbytes)

	Summary of Host Impact			
Data Path	Interrupts/second	Cxt Switches/second	System Call/second	CPU Utilization
Direct Data Path	34	48	15	30%
Indirect Data Path	290	77	150	51%
Ethernet Data Path	640	110	150	51%

Table 2.2 - Summary of Network Architecture Host Impact (buffer size > 128 Kbytes)

Subtask 2.3: Demonstration of Remote Disk Operation and Functionality

In this subtask it was shown that the direct data path can serve as a transport medium for remote record reads at data rates approaching the speed of the disk. In the three computer classes studied, all data transferred over the gigabit network disk-to-memory obtained 50% to 75% of the local disk transfer rate. Remote access to striped disks benefited from the gigabit network also. In comparison, Ethernet delivered less than 10% of the local disk transfer rate and less than 5% for striped disk systems.

4.0 Recommendations

Based on the results from all three subtasks, several ideas that may increase the performance of user applications on very high speed networks are presented.

Performance Increases Hardware and Software

Based on disk-to-disk data transfers it is apparent that disk access is much slower than the ability of the network to transport the data. A significant increase in performance can be obtained by increasing the speed in which data can be read and written from the disk. This can be done either in software or hardware. Disk striping showed a significant performance increase in this area. FTP striping research has been done at NAS for other media and improved overall throughput. All three computer classes showed an increase in throughput when double buffers were used. Therefore, striping FTP over a gigabit network may increase the throughput obtained. Obviously, increasing the local disk read and write speeds will also provide users with faster disk access over the network.

Distributed Networking Applications

With the speed and bandwidth offered by gigabit networks, distributed computing applications can be written to utilize the best computer system architecture for a particular portion of a problem. Network services that give a user the information needed to utilize the resources on the network need to be developed to facilitate this. Methods for distributed processing, distributed disk access, distributed memory sharing (shared memory on the scope of a network), and other network resource utilization techniques need to be created to benefit from the speed and bandwidth of gigabit networks. These types of methods will provide user programs with an entire network of computing resources as if they were on the same host computer.

High-Speed, VMEbus-Based Memory

Preliminary studies have been done to measure the performance of high-speed, VMEbus-based memory. They showed nearly double the performance over transfers to system memory. Further research into applications using this type of memory needs to be performed to fully understand its utility for providing faster network data transfers. This type of research applies mainly to workstations and other computer systems without dual-ported memory.

NFS Performance Enhancements

The gigabit network delivers up to 75% of the local disk rate. Most of this performance is lost when a distributed file system program, such as NFS, is used to provide this service. This is partly due to NFS's entry point into the host protocol stack. A transport-layer-independent NFS is needed. This is currently in specification (Sun Microsystems, Inc., February 1990) and will allow users to implement large block sizes for reads and writes, which will help to increase the performance of NFS. This study has shown that user buffer sizes of 128 Kbytes or larger provide nearly the fastest throughput over this gigabit network.

These larger buffer sizes also impact the system less than smaller buffer sizes. FTP over the gigabit network experienced a 55% increase in speed when the block size was changed from 8 Kbytes to 128 Kbytes. Further research needs to be done into the performance and system implications of making an NFS read 128 Kbytes. This could be used to build a read-ahead cache, enabling users to go to local memory more often, therefore gaining an overall NFS performance increase. This increase in buffer size might also enhance NFS write speeds.

A scheme similar to this has been implemented by one vendor. This vendor provides a high-speed, nonvolatile memory board that intercepts incoming NFS writes. It responds once the information is placed in nonvolatile memory. It then organizes the data for writing to disk. This has significantly increased the speed of NFS writes and has also increased the number of NFS clients a server can support (Lyon, June 1989). Combining this technology and a gigabit network has proven to increase the performance of NFS writes.

If this write caching concept is then applied to reads, using the read-ahead strategy discussed earlier, the user will be doing the majority of his NFS reads from memory. Combining this with the NFS write cache, the user now has memory-to-memory read and write over a very high speed network. This will greatly increase the speed of NFS. If VMEbus memory is used as the storage place for the read and write caching, the memory-to-memory speed will increase, adding another level of performance increase to NFS. Finally, if the buffer size is increased to 128 Kbytes, the knee of the performance curve on this gigabit network, another increase in throughput rates will be observed. Many issues are involved in implementing any one of these schemes, little along all of them simultaneously. Each of these schemes will need to be addressed before the feasibility of this architecture can be quantified.

5.0 Conclusions

A good characterization of the performance of this gigabit network is available from these results. The study started by collecting data on memory-to-memory transfers at the application layer. This gave a baseline characterization of performance on the network. Next a load to the network was added to understand how a gigabit network holds up when burdened. The theory of "non-impact" was tested and found to be true. Then disk-to-disk transfers were added to get a better understanding of the maximum performance of a gigabit network in a more true to life environment. Disk testing was also performed with a load on the network. This data was then broken into two categories for analysis Measurement of Effective Transfer Rates and Performance Analysis of Network Architectures. Results were analyzed within the computer classes and between classes. The impact of a network graphics display device on the network and the impact of the network on interactive network graphics both were studied.

Subtask 2.1
Measurement of Effective Transfer Rates

As shown in Subtask 2.1, gigabit networks have bridged the gap between CPU and disk performance increases, providing users with high bandwidth and throughput network for their production, development, and research computers. This performance increase opens up new areas of research for supercomputer scientists. With high network data rates, the network can be used for things previously not possible.

Network distributed processing can be realized by using the performance increases provided by this gigabit network. The high bandwidth provided by the network would allow computer servers of varying architectures to easily share data and synchronize communications. This would be helpful in designing systems to solve complex computational fluid dynamics problems. This would enable a scientist to take advantage of the best possible compute system for each portion of the problem. Development of a network based solid-state network memory couple with a lock management function would allow fast and effective implementation of common memory across a network of distributed processors.

Visualization over a gigabit network was also proven effective, bringing interactive graphics to the supercomputer. Real time animations can be created using the computers by network distributed processing, with the data being passed through the high bandwidth of a gigabit network. This was demonstrated by the network graphics display device. Visualization could also be done between supercomputers and graphics workstations. In this scenario the supercomputer is the compute engine and the workstation is the graphics engine.

Subtask 2.2
Performance Analysis of Network Architectures

The two results from this section are throughput increases by not performing host-resident protocol processing and host system impact decrease by using the direct data path. This gives the user two ways to increase the performance of a network application, first, by providing greater network throughput capabilities to the application and second by freeing the CPU to perform local tasks instead of protocol processing.

A user can also improve a network application by employing VMEbus-accessible memory. Sun workstations system memory, like memory from other workstation manufactures, resides on a private bus and can be accessed only by DVMA through the CPU. This creates a bottleneck that can be bypassed with VMEbus-based memory. Using this technology, users can tailor their applications to read and write into this data space instead of the system memory. New architectures and ways of thinking about network software are needed to take full advantage of the performance increase that gigabit networks provide.

One of the reasons distributed file systems are relatively slow is because they are designed around the idea of sending small amounts of data. To obtain the maximum speed that gigabit networks can provide, the protocol processing and connection overhead needs to be amortized over a larger data packet. A new concept can be created by combining the ideas of header prediction and file system caching. This concept could be called application caching. In this type of architecture, when a request for a file from a remote file server is made, instead of transmitting just the first block of the file to the request, many blocks are transferred. The extra blocks of the file are cached on the local system, so when the user requests the next block, it is locally available. This could be done either in the users application or, preferably, in the distributed file system. Implementing application caching on per-application basis is fairly trivial. However, adding this functionality to a distributed file system is complex and requires more research as to the viability of this approach.

Subtask 2.3
Demonstration of Remote Disk Operation and Functionality

NFS has become an integral part of many computer centers. It allows users to transparently share files. Diskless workstations have thrived on this technology. With the increased use of NFS comes congestion on the network and in the host computer. The results from Subtask 2.3 have shown that up to 75% of local disk speed can be delivered to users over the network. This enhances the 'transparency' of a distributed file system service to the users. It also allows for more users to benefit from remote disk access.

With the deployment of gigabit networks, the architecture of current distributed file systems need to be changed to provide users with better performance. To obtain high speeds over a network the setup costs of a connection need to be amortized over a large amount of data. Current distributed file system utilize very small packet sizes (8 Kbytes plus headers) for data transfers. If this packet size is increased to 128 Kbytes, the full speed of gigabit networks, like UltraNet, can be realized. The extra data will be stored in the local system memory until requested, and would then be transmitted at local memory speeds. This would also decrease the number of times the user needs to go to the network, which would help cut down on congestion. The implications of this type of change need to be explored before the viability of this solution to NFS performance can be determined.

6.0 References

1. Clark, David D., Van Jacobson, John Romkey, Howard Salwen, "An Analysis of TCP Processing Overhead," *IEEE Communications*, pp 23-29, June 1989.
2. Clinger, Marke, "Subtask 2.1: Measurement of Effective Transfer Rates, Very High Speed Network Prototype Development," Ultra Network Technologies, 1989.
3. Clinger, Marke, "Final Report including Subtask 2.2: Performance Analysis of Network Architectures and Subtask 2.3: Demonstration of Remote Disk Operation and Functionality," Ultra Network Technologies, 1990.
4. Hathaway, Wayne, "Overcoming Network Bottlenecks," *UNIX REVIEW*, Vol 8, No. 4, pp 63-68 (April 1990).
5. Leffler, Samuel J., Marshall Kirk McKusick, Michael J. Karels, and John S. Quarterman, *The Design and Implementation of the 4.3BSD UNIX Operating System*, Addison-Wesley Publishing Company, Inc., 1989.
6. Lyon, Bob and Russell Sandberg, *Breaking Through the NFS Performance Barrier*, Legato Systems, Inc., June 1989
7. Sun Microsystems Inc., *Network Extensible File System Protocol Specification Draft*, February 2, 1990