Computing Systems in Engineering Vol. 3, Nos 1-4, pp. 521-524, 1992
Printed in Great Britain.

0956-0521/92 $5.00 + 0.00
Pergamon Press Ltd

HIGH-SPEED NETWORKS, VISUALIZATION, AND MASSIVE PARALLELISM IN THE ADVANCED COMPUTING LABORATORY

D. W. FORSLUND, C. HANSEN, P. HINKER, W. ST. JOHN, S. TENBRINK and J. BREWTON
Los Alamos National Laboratory, Los Alamos, NM 87545, U.S.A.

Abstract—Very high performance massively parallel machines are now available to solve significant problems for the nation that have been heretofore inaccessible. However, these machines, with their large memories are of little use for these problems without existing in an environment in which the scientist can perceive what is happening in the calculation and analyze the results effectively. This requires being able to transmit enormous amounts of data from these machines through very high performance networks in a way which is easily comprehended and manipulated by the scientist. In order to provide such an environment for these Grand Challenge applications, researchers in the Advanced Computing Laboratory have been developing a very high-speed network (100 Mbytes/s) based on the standard HIPPI protocol developed at Los Alamos. This is used to send data over a multiple cross bar network between supercomputers, high performance graphics machines, and high-speed framebuffers. Examples of scientific problems being explored in this way will be given, including approaches to high-speed archival data storage and interactive data exploration over these networks.

NOMENCLATURE

CM-2	Connection Machine 2
CBI	cross bar interface
HIPPI	high performance parallel interface
JPEG/MPEG	compression standards
NTSC	standard for American television
PVM	parallel virtual machine
SGI VGX	silicon graphics workstation
TCP/IP	a standard networking protocol
VME	a standard computer bus architecture

INTRODUCTION

Because of the rapid increase in computer power in microprocessors there is a corresponding increase in performance and capacity of massively parallel computers. This change has caused a radical shift in the system requirements to support large scale applications and has resulted in the Grand Challenge program in the Federal High Performance Computing Initiative. In order to support this change in the way large scale applications are being solved and analyzed, we are developing a high performance distributed computing environment based on the multiple cross bar network using the HIPPI ANSI standard. We are designing the tools to provide the storage of the enormous amounts of data and rapid visualization and data analysis of these data. In this paper, we report some of the progress in distributed computing over high-speed networks and visualization in this high performance environment.

DATA MOVEMENT

To date we have demonstrated the capability of moving files over the network between various supercomputers located both nearby and over longer distances using Broadband Communications Products fiber-repeaters. Speeds of 20–80 Mbytes/s have been observed with multiple megabyte packet sizes. We also have demonstrated the usability of the network for doing distributed applications between a CM-2 and more than one CRAY Y/MP. In particular, for the Global Climate Grand Challenge, we have implemented a distributed application where a general circulation model for the ocean is computed on the CM-2 while the atmospheric general circulation model is simultaneously run on the CRAY. Information at the ocean surface involving wind velocities, salinity and temperatures is exchanged between the code every cycle or every few cycles to couple the applications together. While we await higher level protocols to function between these two disparate machines, we are using raw HIPPI to exchange the information in single block transfers. We have used PVM[1] to control and synchronize the data communications between the two machines. Because the CM-2 HIPPI channel is actually managed by a separate computer on the CM-IO bus, the control can be somewhat complicated, and as a result, the latencies are quite high.

To start the process, we initialize the PVM daemons on the CM-2 frontend, on the CRAY and on the CM-2 HIPPI computer. The CM-2 HIPPI computer is attached to the CM-2 I/O bus and can send or receive data over that bus from either the CM-2 memory or the CM-2 Datavault. The distributed application is controlled from the CM-2 front end, which initializes tasks on the CRAY and on the CM-2 HIPPI computer. We have used two different techniques to transfer the data to try to hide latency. In the first case, the CM-2 writes its ocean surface data to the CM-2 Datavault, asynchronously notifies the CRAY and then the CRAY reads the datavault through the CM-2 HIPPI computer when it is ready to do so. The process then reverses itself to send data back to the CM-2. The data vault is used to simulate asynchronous I/O on the CM-2 so that both computers do not have to synchronize in order to send the data. The price to be paid, however, is the additional transfer of data over the CM-2 I/O bus. Although the two systems need to be synchronized, we see much faster overall performance when we send the data directly between memory on the two machines. A significant portion of the time, in either case, is spent in transposing the data on the CM-2 to make it understandable by the CRAY. We see no degradation of performance over the fiber-extenders we have and see much enhanced performance when sending data between two Y/MPs than between the CM-2 and Y/MP. Sometime in FY93 we hope to demonstrate this capability using the CNRI sponsored CASA Gigabit network between Los Alamos, San Diego Supercomputer Center, JPL, and CalTech.

The development of the HIPPI network in the ACL involves both hardware and software to support a standard protocol. The initial choice for a protocol is TCP/IP with implied FTP, Telnet and socket support. To accomplish this, an out-board processor is required for the CM-2 because of its separate HIPPI computer configuration. A TCP/IP implementation on the CM-2 HIPPI computer (Sun 4), without an out-board processor, would give unacceptable transfer rates—around 5 Mbytes/s or less. The out-board protocol processor, called the Cross Bar Interface (CBI) is designed to accept block transfers of data from the CM-2 HIPPI, assemble and disassemble IP packets and establish TCP sessions through the network at rates around 30–50 Mbytes/s. The CBI also will provide buffering (4 Mbytes), security, and network management. To complete the support of the CM-2, work is being done on a block transfer protocol for HIPPI called Memory Interface (MI). This MI effort, along with the CBI, will also help with the high performance disk systems that will be needed to store the large amounts of data produced with gigabit networks.

The CBI will also be able to support multiple HIPPI Frame Buffers for visualization. These frame buffers were developed at Los Alamos last year and provide a high resolution display of 1280 by 1024 by 24 bits. Display rates of 15 frames per second are possible. On the current raw HIPPI network in the ACL, a frame buffer connected through the network to a HIPPI disk array has received data at around 60 Mbytes/s. From memory on the CRAY Y/MP, it can receive data in excess of 80 Mbytes/s.

VISUALIZATION

Because of the cost of setting up very high performance visualization facilities to analyze large volumes of data, resources are frequently concentrated in a visualization laboratory. However, there is a great reluctance to utilize a visualization laboratory: the papers/journals/notebooks etc. relating to the science are in their office not the visualization lab, one loses their train of thought when having to walk down the hall (possibly into another building), the spontaneity of quickly reviewing or studying an animation is not possible, etc.[2] Framebuffers directly connected into a particular machine are typically in a very limited number of fixed locations. The capability of having a networked framebuffer overcomes this problem. In the ACL, we utilize HIPPI for the play-back of animation sequences to a HIPPI framebuffer in the scientist's office. Because of this distributed requirement, we have developed fairly low cost HIPPI framebuffers to make this possible.

Scientists want to view image sequences representing visualizations of their data with a VCR type interface. These image sequences can be generated using post-processing visualization techniques or from graphics which are directly coupled to the model running on a supercomputer (monitoring running models). An obvious option is to stream the frames to the local workstation and play them back locally. This is often unfeasible due to the huge size of the aggregate frames (1 Mbyte/frame) and the small memory size of workstations. A typical interactive session consists of a limited number of small packets controlling the animation (from the scientist to the supercomputer) and a large number of very big packets arriving which contain the frames to be viewed.

Data compression can be very useful for the post-processing scenario but for simulation tracking or simulation steering, it poses problems. The time required to perform compression on one side and decompression on the other is the limiting factor. Recently, silicon implementations of compression algorithms have been brought to market (i.e. JPEG). However, these are directed at single frames, not sequences. MPEG addresses image sequences by using both spatial and temporal compression. However, when dealing with image compression, one must consider whether the compression technique is lossy or lossless. Lossless compression retains the same quantitative information in the decompressed image, whereas with lossy techniques, information content is traded for file size. Both JPEG and MPEG are lossy compression techniques. While these are acceptable for video teleconferencing, NTSC images or digital video, techniques which modify the quantitative information are unacceptable for many applications. The scientist must not be distracted from examining phenomena by artifacts introduced by a compression/ decompression technique. Worst still is the introduction of artifacts which might be misconstrued as phenomena within the data.

We are in the process of developing a general capability where the scientist can review high resolution frames of images (animation) via the HIPPI framebuffer through a VCR type interface. Los Alamos has developed a 1024 by 1024 by 24-bit image HIPPI frame buffer. This 24-bit device can run in two modes: a resolution of 1280 by 1024 at 15 frames/s and a resolution of 640 by 512 at 60 frames/s. Currently, this device is driven in the production environment directly off of CRAY Y/MPs. When driven by the Y/MP, the framebuffer user contends with other concurrent users (time-slicing) as well as I/O subsystem contention. However, we have found the device to give quite acceptable results. It is easy to raise the priority of the framebuffer job to receive a more generous timeslice. However, disk contention is much more difficult to schedule. Rather than build a 24-bit movie in memory or on disk, the interface provides for on the fly decoding of eight-bit color-mapped image sequences. This reduces the I/O subsystem requirements. Additionally, we have interfaced, via HIPPI, a RAID disk for caching of animation loops for later and smoother playback at a later time. This work provides the best results for post-processing of data produced by models. For interactive work, model monitoring or simulation steering, the simulation must be paused while the visualization is produced and the image sequence is loaded onto the RAID disk for smooth playback. We have found the attached framebuffer provides better results in this scenario. Since the framebuffer is a component of our HIPPI network, the framebuffer is an addressable network device, thereby providing HIPPI animation capability into a large number of locations. In addition to the 24 bits of image information per pixel, the framebuffer requires 3 bits of control information, 1 bit of audio, and 4 bits are reserved for future use (multimedia or other). With 1 bit per pixel reserved for audio at 1280 × 1024, each frame has the capacity of 1 Mbyte of accompanying audio information. At 15 frames/s, this provides 22 channels of CD quality sound (assuming 16 bits/sample and 44,100 samples/s). This is not currently being used in the production environment but research projects are investigating both auditorialization and multimedia.[3,4]

We are also studying distributed visualization via HIPPI. The typical visualization process consists of moving the raw data computed on the supercomputer to a graphics workstation. The data are then culled, filtered, mapped and rendered on the graphics workstation. This can be thought of as a visualization pipeline. In the high-speed network distributed visualization model, parts of the pipeline are migrated to the appropriate hardware within the network. The most obvious is to cull and filter the data on the supercomputer, then transport to the graphics workstation where mapping and rendering take place. Although the bandwidth of most workstations' backplanes is lower than HIPPI, HIPPI to VME cards are commercially available. This still remains a bottleneck on the workstation side but can still be very effective with clever partitioning of the visualization pipeline.

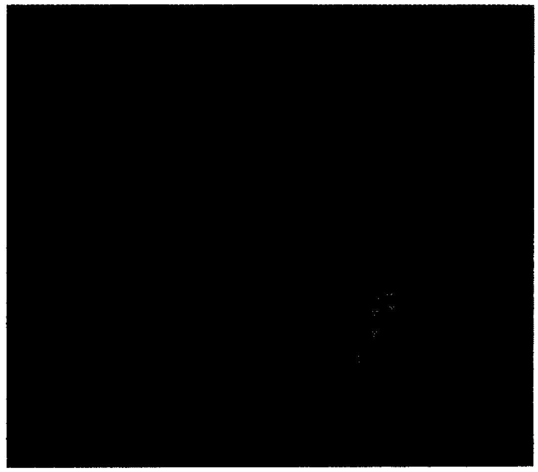

Fig. 1. Three-dimensional rendering of a fluid dynamics simulation of a hypervelocity micrometeorite penetrating a space station shield. The simulation and iso-surface extraction were done on the CM-2 and rendered on a SGI VGX system.

Another migration is to perform the mapping on the massively parallel computer and transport geometry via the high-speed network for rendering on the graphics workstation. We have implemented a massively parallel isosurface extraction algorithm, based on Marching Cubes, on the CM-2.[5] In this environment, the scientist's model or simulation executes the isosurface extraction algorithm and the resulting geometry is transferred, via a high-speed network, to their workstation, in our case an SGI VGX, for rendering. Figure 1 shows one of a sequence of rendered images produced with this distributed environment. For a 256 × 256 × 256 volume of floating point data, the raw data require 530 Mbits per time step. Considering that dynamic simulations contain hundreds of time steps, this is obviously too much raw data to transport in the typical visualization process. If 50 K polygons (triangles) are extracted, the data shipped over the network are reduced to 14 Mbits. This represents a compression factor of almost 37 times! As previously mentioned, due to the VME restrictions on current graphics platforms the network remains the bottleneck for this problem. To help overcome this problem, we have implemented a temporal lossless compression algorithm for transmitting only changed geometry between time steps. We continue to investigate other mappings of the visualization pipeline onto the high-speed networked environment.

CONCLUSIONS

Proper utilization of the large massively parallel machines that are coming on line in the 90s will require robust distributed computing tools over high-speed networks combined with advanced visualization techniques. These tools are just beginning to come on line and the high-speed HIPPI network in place in the Advanced Computing Laboratory is beginning to be used effectively for Grand Challenge applications. We anticipate continued development in this area as more applications begin to use these tools.

REFERENCES

1. A. Beguelin, J. Dongarra, A. Geist, B. Manchek and V. Sunderam, *A User's Guide to PVM, Parallel Virtual Machine*, ORNL/TM-11826, Oak Ridge National Laboratory, Oak Ridge, Tennessee, 1991.
2. R. L. Phillips, "A scientific visualization workbench," Proceedings of Supercomputing 1988, pp. 145–148, 1988.
3. R. S. Hotchkiss and C. L. Wampler, "The auditorialization of scientific information," Proceedings of Supercomputing Conference 1991, pp. 453–461, 1991.
4. R. L. Phillips, "Media View, a general multimedia digital publication system," *Communications of the ACM* **34**(7), 1991.
5. W. Lorensen and H. Cline, "A high resolution 3D surface construction algorithm," *Computer Graphics* **21**, 163–169 (1987).