Performance Evaluation of a Parallel I/O Architecture

Sandra Johnson Baylor
Caroline Benveniste
Yarsun Hsu

IBM T. J. Watson Research Center
P. O. Box 218
Yorktown Heights, NY 10594

Abstract

Presented are the results of a study conducted to evaluate the performance of parallel I/O on a massively parallel processor (MPP). The network traversal and total processing times are calculated for I/O reads and writes while varying the I/O and non-I/O request rates and the request size. Also studied is the performance impact of I/O and non-I/O traffic on each other. The results show that the system is scalable for I/O loads considered; however, the scalability is limited by I/O node saturation or considerable network contention.

I. Introduction

The advent of massively parallel processors (MPPs) has resulted in the potential for revolutionary performance in computers. MPPs appeal to an array of scientific and commercial applications including seismic processing, global climate modelling, satellite/earth station processing, and multimedia applications. These applications require high-bandwidth, low-latency access to very large amounts of data (gigabytes to petabytes). Much effort has been expended on the design and performance evaluation of processors and memory for MPPs; however, relatively little work has been done on the I/O subsystems of these machines. This has resulted in large gaps between I/O, processor, and memory access times. Since a balance has to exist between these subsystems to achieve optimum performance, I/O is widely viewed as a prohibitive bottleneck in MPPs. Designing MPPs to address this bottleneck will improve MPP performance and enable programmers to write innovative parallel applications for these machines.

This paper presents the results of a study conducted to evaluate the performance of some architectural alternatives for the parallel I/O subsystem of an MPP that can reduce the access bottleneck. The average processing times for I/O reads and writes are calculated as the system size is varied from 16 to 128 nodes and as the compute node to I/O node ratio is varied from 1 to 4. Several request rates are used for both I/O and non-I/O message traffic to determine the impact of I/O traffic on other types of network traffic under various loads as well as the impact of I/O traffic on processor-processor traffic. The evaluation methodology used is trace-driven simulation.

Figure 1 presents a logical view of a parallel I/O subsystem for MPPs. The compute nodes are used primarily for computation and therefore no I/O data is located on disks that may be included on the node (these disks may be used for paging space). The I/O data is distributed across the I/O nodes and is accessed in parallel by the compute nodes through the high bandwidth, low latency network. The compute nodes, interconnection network, and I/O nodes form the internal parallel I/O subsystem. The gateway nodes are used to transfer data between the internal parallel I/O subsystem and an archival storage system. Several commercial machines exhibit this type of high level architecture including the IBM SP1 and SP2 [1], the Thinking Machines CM5 [2], the Intel Paragon [3], the nCUBE hypercubes [4], and the Tera [5]. This paper focuses on the performance of internal parallel I/O architectures.

Some initial work has been done on the design and performance evaluation of parallel I/O subsystems. Reddy and Banerjee [6] studied I/O embedding in hypercube architectures, concentrating on I/O node placement in up to 16 nodes. Two studies were conducted by Baylor, *et. al.*, [7, 8]. The first study evaluated the parallel I/O performance of the Vulcan MPP [9], focusing on I/O node placement, the effect of varying the I/O read/write ratios scalability, and other issues for systems of up to 512 nodes. The second study included a detailed evaluation of the relationship between I/O and non-I/O message traffic traversing the network and the performance sensitivity to the request size for parallel I/O architectures. The results of both of these studies included the effects of I/O node blocking (the I/O node fails to process any subsequent requests until the processing of a write miss has completed at the I/O node) [10]. More interesting and realistic results would either not include I/O node blocking or would include I/O node blocking only when the I/O node can no longer buffer incoming requests. The results presented in this paper differs from the results of these two studies in the following ways: 1) they do not include I/O node blocking and 2) they have improved disk specifications and switch cycle times, resulting in a more realistic quantification of parallel I/O performance.

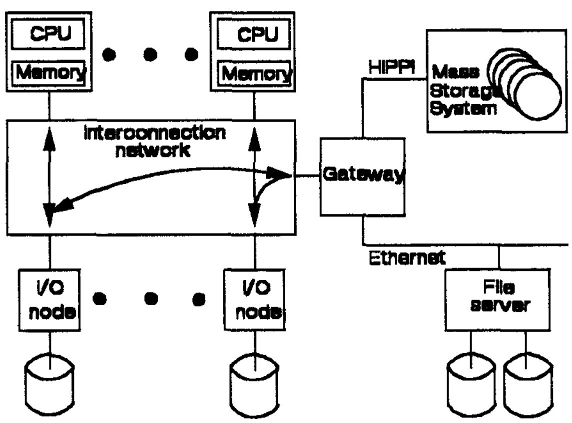

Figure 1. MPP I/O Architecture.

Presented in the next section is an overview of the MPP architecture assumed. In section III there is a discussion of the simulation methodology used and in sections IV and V we present the results and conclusions, respectively.

II. MPP Architecture

The MPP architecture includes a number of compute nodes and I/O nodes interconnected by a high performance switching network. The compute node is assumed to have up to 512 megabytes of local memory and a disk capacity of up to 4 gigabytes. The I/O node is assumed to have up to 2 gigabytes of local memory and a disk capacity of up to 8 gigabytes. The total number of nodes varies from 16 to 128; resulting in total disk storage of 16 gigabytes to 1 terabyte and peak performance ranging from 4 to 34 GFLOPS for a 66.7 MHz clock.

The interconnection network is a clustered multistage omega network composed of 4x4 bi-directional switches [9]. This packet-switched network has a bandwidth of 150 megabytes/sec. A microprocessor-based network adapter is used as a communication coprocessor, packetizing and depacketizing messages from the nodes. The adapter has a bandwidth of at least 85 megabytes/sec and it provides DMA access, error detection and correction, and reliability. Messages are broken into 512-byte packets which traverse the network as a unit.

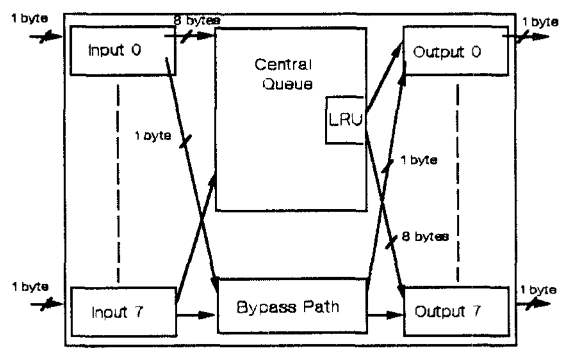

Figure 2. The Network Switch.

Figure 2 presents a diagram of the switch, which includes a central queue, a bypass path, and input and output ports. A packet traversing the switch may take either the bypass path or the central queue. If the packet's destination output port is free and no other packets are queued for that port, it will take the bypass path. Otherwise, the packet will arbitrate, using an LRU policy, for placement in the central queue. The switch cycle time is 13.3 ns [11].

While this architecture is similar to that of the IBM SP2, there are many specifics that differentiate these architectures. For example, the network topology, routing strategy, and latency used are not those of the SP2. These differences (and others not elaborated) can be a major factor in system performance. Therefore, the results presented in this paper should not in any way be considered SP2 results.

III. Simulation Methodology

A two-step process is used to model the MPP parallel I/O subsystem. First, a queueing model was developed to evaluate the performance of the I/O node using the RESQME [12] simulator. The average I/O read and write processing times are obtained from this model. Second, these processing times are used as inputs to PIOS, a trace-driven parallel I/O simulator [10] which integrates the network simulator and the I/O node model.

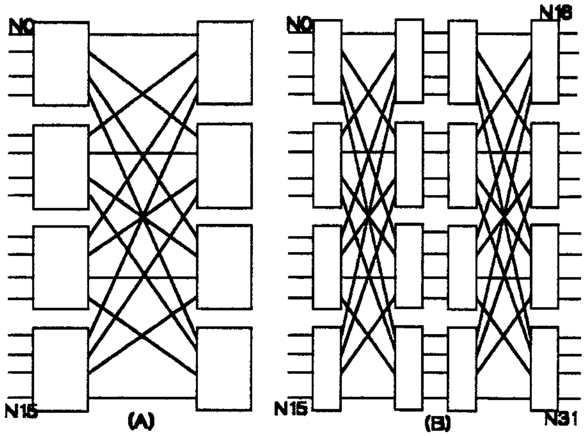

Figure 3. Interconnection Topologies for the 16-node (A) and 32-node (B) MPP.

There has been one recent study on the I/O workload characteristics of parallel systems [13]. This study obtained workload characteristics from a production-based Intel iPSC/860 machine with applications that used the Concurrent File System [14]. This differs from the architecture of the subsystem we modeled; therefore, we decided to drive PIOS with stochastic traces. For each compute node, a trace is generated containing I/O and non-I/O requests to destinations selected by the trace generator with equal probability. Two message types are specified; the I/O requests targeted for I/O nodes and non-I/O messages targeted for compute nodes. A request destined for a compute node (either a read response or a non-I/O message) is assumed to be consumed immediately upon receipt. In reality, this may not occur every time a request reaches its compute node destination. Therefore, the results presented do not include any queueing effects that may occur if the requests are blocked at the compute node. PIOS immediately processes all I/O requests reaching the I/O node with one exception. If the buffers at the I/O node are full, then the I/O node cannot process any further requests until buffer space is available. For the results simulated, the I/O node buffer space is never full, so no I/O node blocking occurs.

Figures 3, 4, and 5 illustrate the interconnection topologies used for the system sizes evaluated. Although more than one path may exist between a source-destination pair, in our simulator we use static routing, and therefore do not route packets through multiple paths.

In this study, half the total number of nodes used are always compute nodes and the ratio of compute nodes to I/O nodes is varied from 1 to 4. For example, in a 16-node system, the number of compute nodes remains constant at 8 and the number of I/O nodes is varied from 2 to 8. The I/O nodes are evenly distributed among the total number of nodes in the system. In cases where the number of compute nodes and I/O nodes do not add up to the total number of nodes in the system, the remaining nodes are assumed to be idle. This is done to evaluate the impact of increasing I/O parallelism on system performance. A number of request sizes, disk cache miss ratios, read probabilities, and disks are simulated in this study. The general behavior of the results for all of these variables are similar. Therefore, only a subset of the values for these variables are presented. For example, the results of two request sizes (4K and 32K bytes), one disk cache miss ratio (0.3), and one I/O read probability (0.67) are presented. Also, the I/O request rate is varied from 5 to over 850 requests/sec and six disks are simulated per node. The read request size is 200 bytes, the non-I/O message sizes are 2K bytes and 32K bytes, and the non-I/O loads vary from 0.1 to 0.4. A load of 1.0 means that at each cycle, each processor generates one byte of a request. Request loads of 0.1 and 0.4 correspond to request rates of 1116 and 4464 requests/sec, respectively.

If the total number of I/O requests processed remains constant, as the number of I/O nodes is doubled, the I/O request rate per I/O node is reduced by one-half. To reduce the complexity in presenting the results, the *FLOPS/byte* parameter is used instead of the I/O request rate for the results. The FLOPS/byte (F/B) is the ratio of the rate of executing floating point operations to the rate of performing I/O. (We assumed that a floating point operation occurs in one switch cycle). An F/B of 1 means that one byte of I/O is done for each floating point operation. Therefore, for a given number of compute nodes and I/O requests, as the number of I/O nodes is varied, the I/O request rate at each I/O node will vary, but the F/B ratio at the compute node will remain constant. However, for a given number of compute nodes and I/O nodes, decreasing the F/B ratio corresponds to increasing the I/O request rate. Table 1 lists the F/B ratios used and the corresponding I/O request rates for all compute node to I/O node ratios considered and for a 32K byte request.

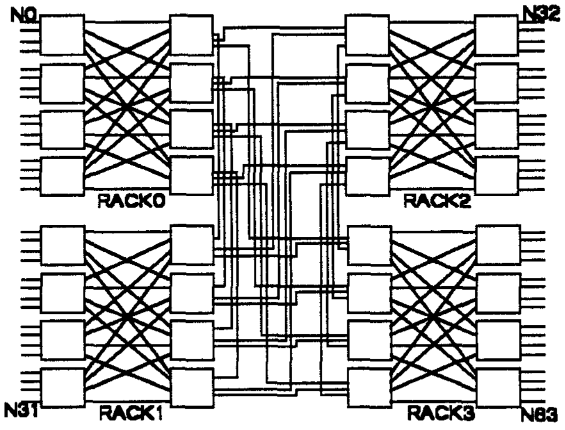

Figure 4. A Interconnection Topology for the 64-node MPP.

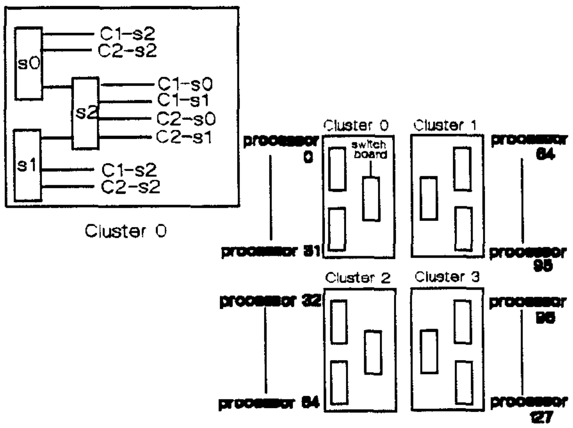

Figure 5. A Interconnection Topology for the 128-node MPP. Cx - Cluster x, Sy - Switch y.

The performance metrics used include the average I/O read request time, the average I/O read response time, the average I/O read processing time, and the average I/O write processing time. The read request time is calculated from the time the request is first generated to the time the first byte arrives at the I/O node. The read response time is calculated from the time the first byte of the response leaves the I/O node to the time the last byte is received at the requesting node. The read processing time is the sum of the request time, the response time, and the time to process the read request at the I/O node. The write processing time is calculated from the time the request is first generated to the time the last byte is placed in the disk buffer. Also calculated is the message processing time for non-I/O messages. This time is calculated from the time the request is generated to the time the last byte is received by the compute node.

Table 1. Assocation Between F/B Ratio and I/O Node Request Rates.

F/B at Compute Node	Compute Node-to-I/O Node Ratio	I/O Request Rate at each I/O Node (requests/sec)
114.5	4	80.00
	3	40.00
	2	26.67
	1	20.00
57.25	4	160.00
	3	80.00
	2	53.33
	1	40.00
28.63	4	320.00
	3	160.00
	2	106.67
	1	80.00
14.32	4	640.00
	3	320.00
	2	213.33
	1	160.00
7.16	4	1280.00
	3	640.00
	2	426.67
	1	320.00

IV. Results

I/O Node Results. Figure 6 presents the average I/O read and write processing times for disk cache hits and misses versus the I/O request rate for the I/O node. For both read and write hits, the processing times are relatively small (hundreds of microseconds for 4K bytes and approximately 3 milliseconds for 32K bytes) for request rates less than 650 requests/sec. For misses, the processing times increase by an order of magnitude to between 14 and 25 milliseconds. This is a result of the relatively large disk access time. Also, for the 32K byte request size, the processing time is much larger than that of the 4K byte request size.

The hit processing times are independent of the I/O request rate presented, showing that even for up to 650 requests/sec, an effective disk cache can minimize any potential I/O-related contention. For request rates greater than 850 requests/sec, the I/O node begins to saturate for the 32K byte request size. For disk cache misses, as the I/O request rate increases, the processing times increase; however the rate of increase is relatively small. For the 32K byte request size, the read miss processing time is slightly larger than the write miss processing time because the process includes transferring a 200 byte request size to the node, processing the request, and then preparing to send a 32K byte response to the requester once the data is accessed from the disk. During a write, 32K bytes of data are placed on the disk which is less time-consuming.

I/O-Only Traffic. All graphs presented are for 64 nodes and for 32K byte request sizes unless specified otherwise. The results of simulations with other values for these parameters exhibited similar behavior. Also, the data point reflecting the smallest number of I/O nodes for the lowest F/B ratio is not presented in any results because the I/O node begins to saturate, substantially degrading performance.

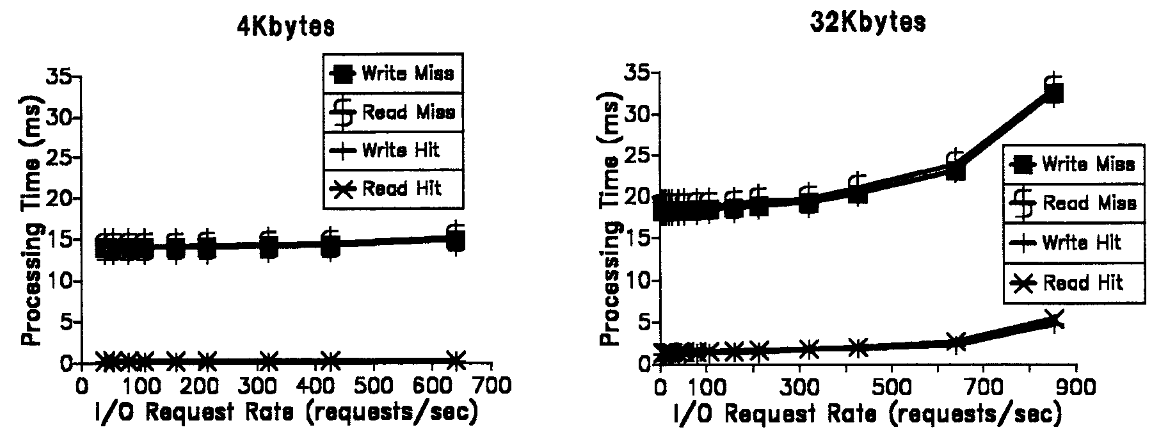

Figure 6. I/O Node Processing Times versus the I/O Request Rate and Request Sizes of 4K Bytes (left) and 32K Bytes (right).

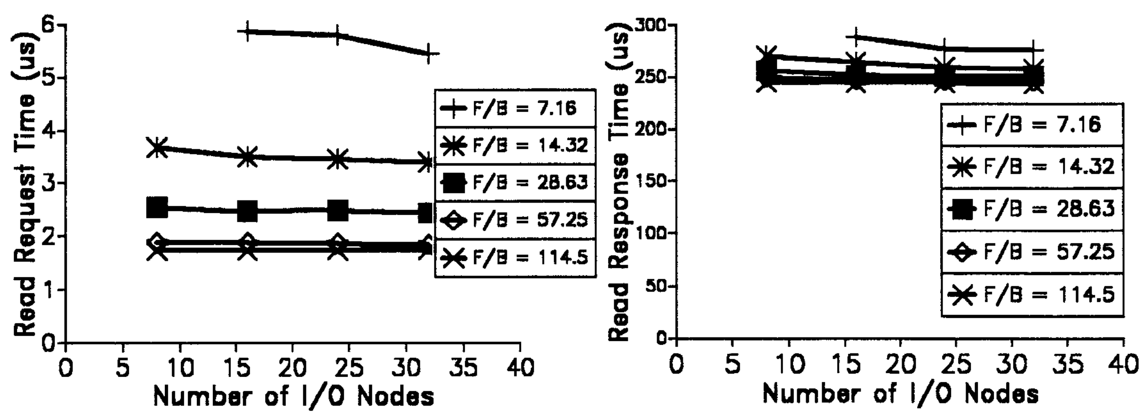

Figure 7. Network Traversal Times for I/O Reads versus the Number of I/O Nodes and the F/B Ratio. 64-node System, 32K Byte Request.

Illustrated in Figure 7 are the read request and response times versus the number of I/O nodes and the F/B ratio. The read request time is relatively independent of the number of I/O nodes and there are small decreases in the read response time as the number of I/O nodes increases. For a given F/B ratio, there is minimal performance gain achieved by increasing the number of I/O nodes because the I/O node can process all requests upon receipt. The small performance advantages for the read response time as the number of I/O nodes increases is attributed to reduced switch contention at the network stage to which the I/O nodes are attached. For a given total number of I/O requests processed, increasing the number of I/O nodes results in less work per I/O node and therefore a smaller number of requests and responses traversing the switches closest to the I/O nodes. The performance increases are not reflected in the read request time because the request size is 200 bytes and there is little or no contention associated with this small request size. Additional network contention is introduced as the F/B ratio decreases. As a result, the read request and response times increase as the F/B ratio decreases. The rate of this increase also increases as the F/B ratio decreases.

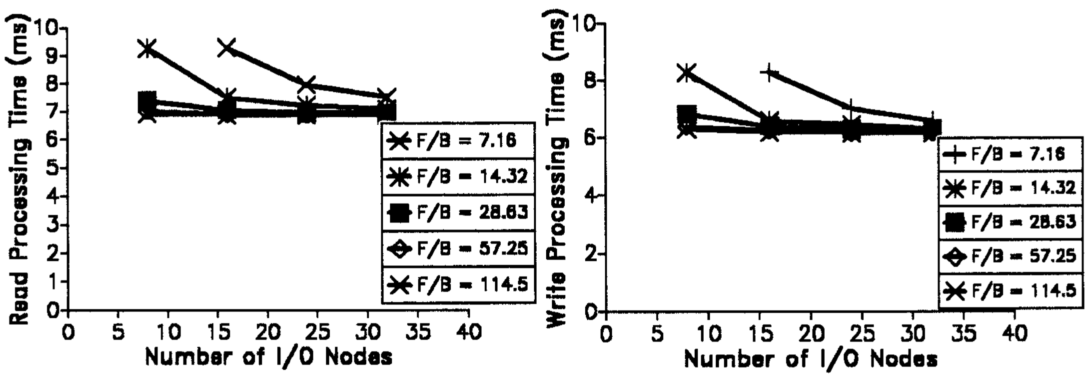

Figure 8. I/O Processing Times versus the Number of I/O Nodes and the F/B Ratio. 64-Node System, 32K Byte Request.

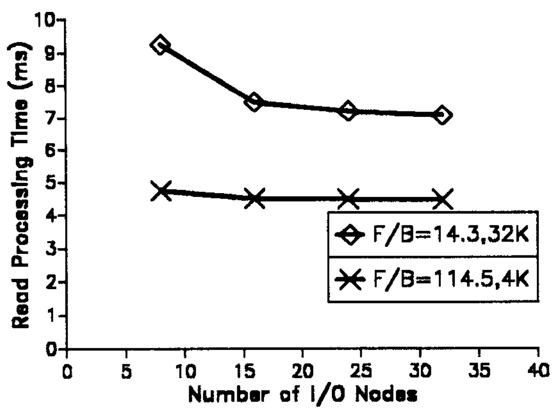

Figure 9. Read Processing Time versus the Number of I/O Nodes, the F/B Ratio and the Request Size for a 64-node System.

Presented in Figure 8 is the read processing time versus the number of I/O nodes and the F/B ratio. These parameters decrease as the number of I/O nodes increases for F/B ratios smaller than 29. They are relatively independent of the number of I/O nodes for larger F/B ratios. Processing the smaller F/B ratios results in more I/O node contention, particularly for 8 and 16 I/O nodes. It therefore takes more time to process a request at the I/O node. The total read processing time is on the order of milliseconds while the read request and response processing times are on the order of microseconds. This shows that the total read processing time is dominated by the processing of the request at the I/O node.

Figure 9 shows the read processing time versus the number of I/O nodes for the 4K byte and 32K byte request sizes. The F/B ratio is 14.3 and 114.5 for the 4K byte and 32K byte request sizes, respectively. This results in an equivalent amount of I/O data processed (in bytes) for both request sizes. The read processing time decreases as the number of I/O nodes increases for the 4K byte request size. For the 32K byte request size, the metric decreases slightly as the number of I/O nodes increases from 8 to 16 but it is independent of the metric for 16 or more I/O nodes. The performance advantages shown for the 4K byte request size occur because the I/O request rate is eight times the rate for the 32K byte request size. In fact, for 8 I/O nodes, the per I/O node request rate is such that the node begins to saturate. As the number of I/O nodes increases, the per node contention is reduced, resulting in better performance. For 32K byte request sizes, the I/O node is never close to saturation for the request rates considered. There is little or no I/O node contention for 16 or more I/O nodes for this request size, resulting in near optimum processing times.

The read and write processing times versus the total number of nodes are presented in Figure 10 for compute node to I/O node (CN-to-ION) ratios of 1 and 4 and for a 32K byte request size. The F/B ratio varies from 7.16 to 57.25. This corresponds to request rates ranging from 40 to 640 requests/sec (see Table 1). The processing times are relatively independent of the system size, indicating scalability.

While all the processing times are larger for the CN-to-ION ratio of 4, they are visibly larger for the smaller F/B ratios. This is attributed to increased contention at the I/O node because with a CN-to-ION ratio of 4, there are fewer I/O nodes and therefore more requests to process per node. For a given CN-to-ION ratio, reducing the F/B ratio results in increasing the I/O request rate per node. Both of these factors cause more contention at the I/O nodes and therefore longer processing times. However, the increased contention is more prevalent as the CN-to-ION ratio is increased.

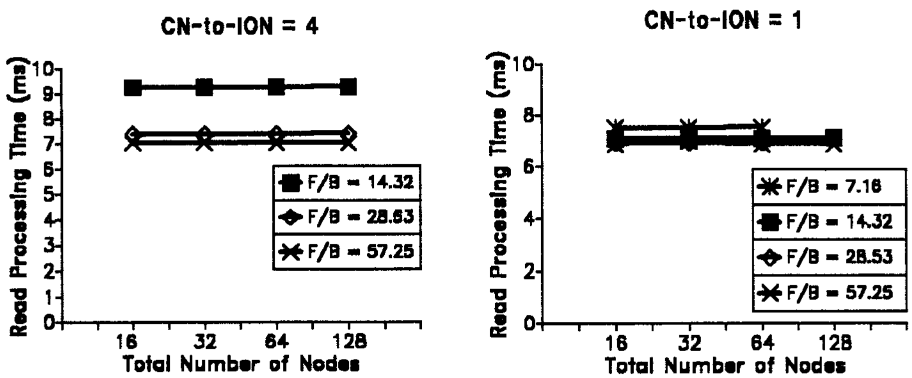

Figure 10. I/O Processing Times versus Total Number of Nodes, F/B Ratio, and Compute Node-to-I/O Node Ratio, 32K byte Requests.

Non-I/O Message Effects. To study the impact of non-I/O message traffic on the performance of I/O traffic, we ran PIOS with both I/O and non-I/O message traffic. Two message sizes were used for the non-I/O message traffic, 2048 and 32K bytes. Small messages made up 95% of the non-I/O messages issued. Unless otherwise specified, the results presented for this part of the study assume a 32-node system. All other system sizes evaluated exhibited similar behavior.

Presented in Figure 11 are the I/O read request and response times versus the number of I/O nodes and the non-I/O traffic loads of 0.0 (I/O-only traffic), 0.1, and 0.4 for a 32-node system (loads of 0.1 and 0.4 correspond to request rates of 1116 and 4464 requests/sec, respectively). Adding non-I/O traffic with a load of 0.1 doubles the read request time and increasing the non-I/O traffic load to 0.4 results in an order of magnitude increase in this metric. For all of the I/O loads considered, the read request time is independent of the number of I/O nodes for I/O-only traffic and for non-I/O loads of 0.1. However, at the non-I/O traffic load of 0.4, there is a reduction in the read request time as the number of I/O nodes increases to 12. Further increasing the number of I/O nodes from 12 to 16 results in a performance degradation for this metric. Increasing the number of I/O nodes results in less I/O traffic per node for a given total number of I/O requests processed. This reduces network contention at the switches closest to the I/O nodes which in turn reduces the time for read requests to reach these nodes. This explains the read request time behavior for 4 to 12 I/O nodes in Figure 11 (non-I/O loads of 0.4).

However, increasing the number of I/O nodes in the system also results in an increase in the number of switch ports to which an I/O node is connected. Therefore, as the number of I/O nodes is increased, there is additional contention at the the switches attached to the extra I/O nodes. This works to offset the performance gains described earlier. When the number of I/O nodes increases from 12 to 16 in Figure 11 (non-I/O loads of 0.4) the additional switch contention of the new I/O nodes is larger than any performance gains associated with reduced contention at the switches associated with the I/O nodes already present in the system. Hence, the read request time increases in this case. The increased performance seen by small increases in the number of

I/O nodes and the reduced performance seen by larger increases in the number of I/O nodes is not present for the I/O-only and smaller non-I/O loads. This is because there are fewer numbers of bytes traversing the network and therefore less contention. This suggests that increasing the number of I/O nodes in the system from a small to moderate amount has some performance advantages with very high network loads.

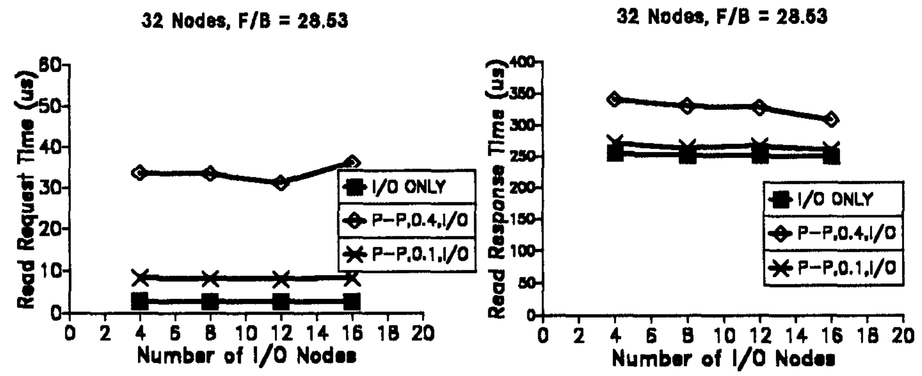

Figure 11. Read Request and Response Times versus the Number of I/O Nodes and the Non-I/O (P-P) Traffic Load, 32K byte Request.

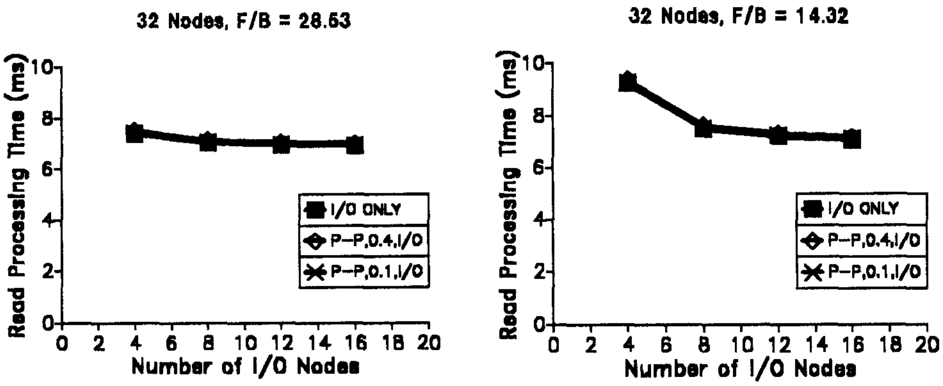

Figure 12. Read Processing Time versus the Number of I/O Nodes and the non-I/O (P-P) Traffic Load, 32K byte Request.

The read response time increases with additional network load and the rate of increase also increases with the higher network load. However, compared to the read request time, the performance degradation is not as severe. This is because the size of the response (32K bytes) is much larger than the size of the majority of the non-I/O messages (2K bytes). This is not the case for the I/O read request size of 200 bytes. When the smaller read request message contends with a non-I/O message, the percentage increase in network traversal times are much greater when compared to the read response messages.

The read response time is relatively independent of the number of I/O nodes when there is no non-I/O message traffic or when the non-I/O message traffic is not high. When the non-I/O message traffic is high, this time decreases as the number of I/O nodes increases.

Shown in Figure 12 is the read processing time versus the number of I/O nodes, the F/B ratio and the non-I/O message traffic load. All of these parameters are relatively independent of the non-I/O message traffic load. This is because the processing times are dominated by the time to process the request at the I/O node. Therefore, any performance advantages or disadvantages resulting from reduced or increased switch contention, respectively, is negligible compared to the I/O node processing times.

Effects of I/O traffic on non-I/O message traffic. We evaluated the effects of I/O traffic on non-I/O message traffic by using PIOS to simulate non-I/O message traffic in systems with and without I/O nodes. Presented in Figure 13 are the normalized message processing times versus the number of I/O nodes, the non-I/O message traffic load, and the F/B ratio for 32 nodes. The normalized time is calculated by dividing the average non-I/O message processing time with I/O traffic by the average non-I/O message processing time without I/O traffic. The results show that the non-I/O message passing time degrades as the F/B ratio decreases and as the non-I/O message traffic load increases. The degradation ranges from 2% to 20%.

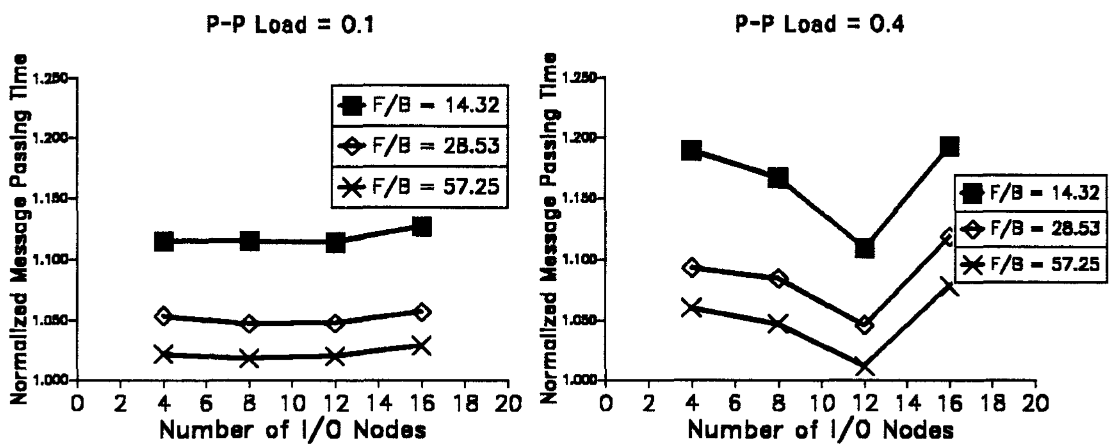

Figure 13. Non-I/O Traffic Message Passing Times versus the Number of I/O Nodes and Total Number of Nodes, 32K byte Request.

For the high non-I/O message traffic load of 0.4, the normalized parameter decreases as the number of I/O nodes increases up to 12. Increasing the number of I/O nodes beyond 16 results in an increase in the normalized non-I/O message processing time. The increased switch contention at the switches directly associated with the additional I/O nodes counteract potential performance improvements for more than 12 I/O nodes. This behavior is also seen in the results of non-I/O message traffic loads of 0.1, however the effect is smaller.

All of these results show that the architecture assumed is highly scalable, even when I/O and non-I/O traffic traverse the same network. Adding I/O nodes to the system is advantageous under two conditions: 1) when the request rates at the I/O nodes are high, causing the I/O node to approach saturation, and 2) when the non-I/O traffic in the network is very high, causing considerable network contention. These are also the two factors that may limit the scalability of the system.

V. Conclusion

The parallel I/O subsystem of the MPP assumed is evaluated using various performance metrics including the I/O and non-I/O processing times. For I/O-only traffic, the network traversal time is relatively independent of the number of I/O nodes in the system for a given I/O load. These same times increase as the I/O request rate increases. Also, the total I/O processing times are relatively independent of the number of nodes in the machine for low to high I/O loads. As processing at the I/O node reaches saturation, increasing the number of I/O nodes in the system improves the I/O processing times. Introducing non-I/O message traffic in the network does impact I/O performance, particularly at high loads, with performance degradation ranging from 2% to an order of magnitude. I/O node saturation and considerable network contention are the two factors that may limit the scalability of the architecture assumed.

All of these results are calculated based on stochastically generated requests. Future work will concentrate on obtaining realistic I/O workload characteristics from applications written for MPPs (this may include much larger request sizes than presented in this paper as well as very different request rates). This study did not assume that the data for a given I/O request is striped across multiple I/O nodes. Therefore, the subject of future work includes varying the percentage and size of I/O requests striped across multiple nodes to determine the performance impact of the resulting burstiness in I/O traffic. Improved I/O node data management strategies, including increased disk cache hit ratios would greatly improve the performance of parallel I/O subsystems. All of these issues are central to the effectiveness of parallel I/O.

REFERENCES

[1] International Business Machines Corporation, Scalable POWERparallel Systems High-Performance Technical Computing Solutions, Publication # GH23-2485-00, March 1994.

[2] Leiserson, C.E., et. al., "The Network Architecture of the Connection Machine CM-5," *4th Symposium on Parallel Algorithms and Architectures*, pp. 272-285, June 1992.

[3] Messina, P., "The Concurrent Supercomputing Consortium: Year 1," *IEEE Parallel and Distributed Technology*, vol. 1, no. 1, pp. 9-16, February 1993.

[4] Hayes, J.P., Mudge, T., Stout, Q.F., Colley, S., and Palmer, J., "A Microprocessor-based Hypercube Supercomputer," *IEEE Micro*, vol. 6, no. 5, pp. 6-17, October 1986.

[5] Alverson, R., Callahan, D., Cummings, B., Koblenz, B., Porterfield, A., and Smith, B., "The Tera Computer System," *International Conference on Supercomputing*, pp. 1-6, June 1990.

[6] Reddy, A.L.N. and P. Banerjee, "Design, Analysis, and Simulation of I/O Architectures for Hypercube Multiprocessors," *IEEE Transactions on Parallel and Distributed Systems*, vol. 1, no. 2, pp. 140-151, April 1990.

[7] Baylor, S. J., Benveniste, C., and Hsu, Y., "Performance Evaluation of a Massively Parallel I/O Subsystem," *Proc. of the IPPS Workshop on Parallel I/O (also appeared in Computer Architecture News, pp. 5-10, September, 1994)*, pp. 1-8, April 1994.

[8] Baylor, S. J., Benveniste, C., and Hsu, Y., "Performance Evaluation of a Parallel I/O Subsystem," *IBM Research Report #RC 19749*, pp. 1-8, September 21, 1994.

[9] Stunkel, C.B., et. al., "Architecture and Implementation of Vulcan," *Proc. of the International Parallel Processing Symposium*, pp. 268-274, April 1994.

[10] Baylor, S. J., Benveniste, C., and Beolhouwer, L. J., "A Methodology for Evaluating Parallel I/O Performance for Massively Parallel Processors," *Proc. 27th Annual Simulation Symposium*, pp. 31-40, April 1994.

[11] Stunkel, C.B., D.G. Shea, D.G. Grice, P.H. Hochschild, and M. Tsao, "The SP1 High-Performance Switch," *Proc. of the Scalable High Performance Computing Conference*, pp. 150-157, May 1994.

[12] Gordon, R.F., and MacNair, E.A., The RESearch Queueing Package Modeling Environment (RESQME), Yorktown Heights, NY: IBM, Revised, September 14 1993.

[13] Kotz, D. and Nieuwejaar, N., "Dynamic File-Access Characteristics of a Production Parallel Scientific Workload," *Supercomputing '94*, pp. 640-649, November 1994.

[14] Pierce, P., "A Concurrent File System for a Highly Parallel Mass Storage Subsystem," *Proceedings of the Fourth Conference on Hypercube Computers and Applications*, pp. 155-160, 1989.