# Performance Comparison of MPI Implementations over InfiniBand, Myrinet and Quadrics[*]

Jiuxing Liu Balasubramanian Chandrasekaran Jiesheng Wu Weihang Jiang
Sushmitha Kini Weikuan Yu Darius Buntinas Peter Wyckoff[†] D K Panda

Computer and Information Science
The Ohio State University
Columbus, OH 43210

[†]Ohio Supercomputer Center
1224 Kinnear Road
Columbus, OH 43212

## Abstract

*In this paper, we present a comprehensive performance comparison of MPI implementations over InfiniBand, Myrinet and Quadrics. Our performance evaluation consists of two major parts. The first part consists of a set of MPI level micro-benchmarks that characterize different aspects of MPI implementations. The second part of the performance evaluation consists of application level benchmarks. We have used the NAS Parallel Benchmarks and the sweep3D benchmark. We not only present the overall performance results, but also relate application communication characteristics to the information we acquired from the micro-benchmarks. Our results show that the three MPI implementations all have their advantages and disadvantages. For our 8-node cluster, InfiniBand can offer significant performance improvements for a number of applications compared with Myrinet and Quadrics when using the PCI-X bus. Even with just the PCI bus, InfiniBand can still perform better if the applications are bandwidth-bound.*

## 1 Introduction

In the past few years, the computational power of commodity PCs has been doubling about every eighteen months. At the same time, network interconnects that provide very low latency and very high bandwidth are also emerging. This trend makes it promising to build high performance computing environments by clustering, which combines the computational power of commodity PCs and the communication performance of high speed network interconnects.

Currently, there are several network interconnects that provide low latency (less than $10\mu$s) and high bandwidth (in the order of Gbps). Two of the leading products are Myrinet [14] and Quadrics [16]. In the high performance computing area, MPI [18] has been the *de facto* standard for writing parallel applications. To achieve optimal performance in a cluster, it is very important to implement MPI efficiently on top of the cluster interconnect. Myrinet and Quadrics were designed for high performance computing environments. As a result, their hardware and software are specially optimized to achieve better MPI performance. More recently, InfiniBand [7] has entered the high performance computing market. Unlike Myrinet and Quadrics, InfiniBand was initially proposed as a generic interconnect for inter-process communication and I/O. However, its rich feature set, high performance and scalability make it also attractive as a communication layer for high performance computing.

In this paper, we present a comprehensive performance comparison of MPI implementations over InfiniBand, Myrinet and Quadrics. Our objectives are to answer the following questions:

- Can InfiniBand offer good performance at the MPI level?
- How does MPI over InfiniBand perform compared to MPI over Myrinet and Quadrics?

The MPI implementations we use for Myrinet and Quadrics are those included in their respective software packages. For InfiniBand, we have used our MVAPICH [8, 9] implementation.

Our performance evaluation consists of two major parts. The first part consists of a set of MPI level micro-benchmarks. These benchmarks include traditional measurements such as latency, bandwidth and host overhead. In addition to those, we have also included the following micro-benchmarks: communication/computation over-

[*]This research is supported in part by Sandia National Laboratory's contract #30505, Department of Energy's Grant #DE-FC02-01ER25506, and National Science Foundation's grants #EIA-9986052 and #CCR-0204429.

lap, buffer reuse, memory usage, intra-node communication and collective communication. The objective behind this extended micro-benchmark suite is to characterize different aspects of the MPI implementations and get more insights into their communication behavior.

The second part of the performance evaluation consists of application level benchmarks. We have used the NAS Parallel Benchmarks [13] and the sweep3D benchmark [5]. We not only present the overall performance results, but also relate application communication characteristics to the information we got from the micro-benchmarks. We use in-depth profiling of these applications to measure their characteristics. Using these profiled data and the results obtained from the micro-benchmarks, we analyze the impact of the following factors: overlap of computation and communication, buffer reuse, collective communication, memory usage, SMP performance and scalability with system sizes, and PCI-X bus.

The main contributions of this paper are:

- We present a detailed performance study of MPI over InfiniBand, Myrinet and Quadrics, using both applications and micro-benchmarks.
- We have shown that MPI communication performance is affected by many factors. Therefore, to get more insights into different aspects of an MPI implementation, one has to go beyond simple micro-benchmarks such as latency and bandwidth.
- Our results show that for 8-node clusters, InfiniBand can offer significant performance improvements for many applications compared with Myrinet and Quadrics when using PCI-X bus. Even with the PCI bus, InfiniBand can still perform better if the applications are bandwidth-bound.

The rest of this paper is organized as follows: In Section 2 we provide an overview of the interconnects, their respective messaging software and MPI implementations. In Section 3 we present our micro-benchmarks and their performance results. The application results are presented in Section 4. We then discuss related work in Section 5 and draw conclusions in Section 6.

## 2 Overview of Interconnects and MPI Implementations

Currently one of the most popular MPI implementations is MPICH [3] from Argonne National Laboratory. MPICH uses a layered approach in its design. The platform dependent part of MPICH is encapsulated by an interface called Abstract Device Interface (ADI), which describes the communication functions used by the MPI implementation. To port MPICH to a new communication architecture, only the ADI functions need to be implemented. More sophisticated ADI functions, such as collective communication calls, are usually implemented by using point-to-point ADI functions. However, the implementation architecture of ADI is very flexible. To achieve optimal performance, collective functions can be implemented directly over the messaging layer provided by the interconnect.

To further reduce the effort of porting, MPICH introduces another interface called Channel Interface below the ADI. The Channel Interface contains only a few functions and therefore it is very easy to re-target them to anther communication architecture.

All the three MPI implementations studied in this paper are derived from MPICH. Essentially, they represent three different ADI2 (the second generation of ADI) implementations on top of InfiniBand, Myrinet and Quadrics. In this section, we provide an overview of the three interconnects, their messaging layers and their MPI implementations.

### 2.1 InfiniBand

The InfiniBand Architecture [7] defines a high speed network for interconnecting processing nodes and I/O nodes. In an InfiniBand network, processing nodes and I/O nodes are connected to the fabric by Host Channel Adapters (HCAs) and Target Channel Adapters (TCAs).

Our InfiniBand platform consists of InfiniHost HCAs and an InfiniScale switch from Mellanox [11]. InfiniScale is a full wire-speed switch with eight 10 Gbps ports. The InfiniHost MT23108 HCA connects to the host through PCI-X bus. It allows for a bandwidth of up to 10 Gbps over its ports.

VAPI is the software interface for InfiniHost HCAs. The interface is based on the InfiniBand verbs layer. It supports both send/receive operations and remote direct memory access (RDMA) operations. Currently, Reliable Connection (RC) and Unreliable Datagram (UD) services are implemented. In VAPI, user buffers must be registered before they can be used for communication. The completion of communication requests is reported through completion queues (CQs).

The details of the MPI implementation on top of VAPI have been discussed in [9]. This implementation is based on the RC service. To achieve better performance, RDMA operations are used not only for large messages, but also for small and control messages.

### 2.2 Myrinet

Myrinet was developed by Myricom [12] based on communication and packet-switching technology originally designed for massive parallel processors (MPPs). Myrinet has a user-programmable processor on the network interface card that provides much flexibility in designing communication software.

Our Myrinet network consists of Myrinet M3F-PCIXD-2 network interface cards connected by a Myrinet-2000 switch. The link bandwidth of the Myrinet network is 2Gbps in each direction. The Myrinet-2000 switch is a 8-port crossbar switch. The network interface card has a 133MHz/64bit PCI-X interface. It has a programmable LANai-XP processor running at 225 MHz with 2MB onboard SRAM. The LANai processor on the NIC can access host memory via the PCI-X bus through the DMA controller.

GM [12] is the low-level messaging layer for Myrinet clusters. It provides protected user-level access to the network interface card and ensures reliable and in-order message delivery. GM provides to the upper layer a connectionless communication model. GM supports send/receive operations. It also has a directed send operation which can directly write data to a remote node's address space. Similar to VAPI, user communication buffers must be registered in GM.

MPICH over GM [12] is implemented by re-targeting the Channel Interface to the GM messaging layer. For internode communication, GM send/receive operations are used for small and control messages while directed send operations are used for large messages.

### 2.3 Quadrics

Quadrics networks consist of Elan3 network interface cards and Elite switches[17]. The Elan network interface cards are connected to hosts via 66MHz/64Bit PCI bus. Elan3 has 64 MB on-board SDRAM and a memory management unit (MMU). An Elite switch uses a full crossbar connection and supports wormhole routing.

Our Quadrics network consists of Elan3 QM-400 network interface cards and an Elite 16 switch. The Quadrics network has a transmission bandwidth of 400MB/s in each link direction.

Quadrics provides different programming libraries for accessing the Elan3 network interface. Among all the libraries, Elan3lib provides the lowest level access to the network interface. Therefore it also offers the best performance. Elan3lib supports protected, user-level access to Elan network interfaces. It provides a global virtual address space by integrating individual node's address space. One node can use DMA to access a remote node's memory space. Unlike VAPI and GM, communication buffers do not need to be registered. Elan network interface cards have an on-board memory management unit. The system software is responsible for synchronizing the MMU table and doing the address translation.

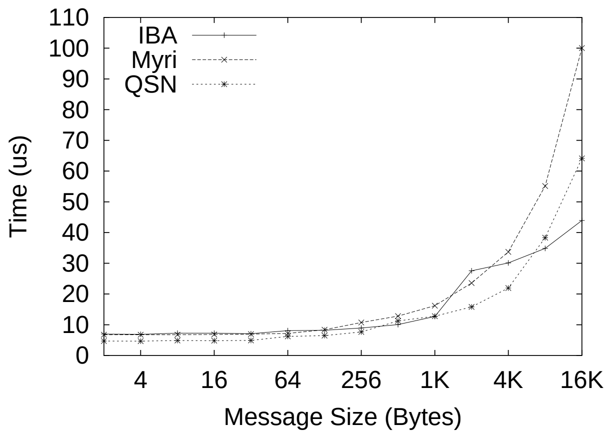

**Figure 1. MPI Latency**

Quadrics also provides a library called Tports which presents a point-to-point message passing interface. The Tports programming interface is very similar to that of MPI. MPICH over Quadrics is based on the ADI2 interface. It is implemented on top of Tports.

## 3 Micro-Benchmarks

To provide more insights into communication behavior of the three MPI implementations, we have designed a set of micro-benchmarks. They include basic measurements such as latency, bandwidth and host overhead. In addition, we use several micro-benchmarks to characterize the following aspects of an MPI implementation:

- Bi-directional communication performance.
- Ability to overlap communication with computation.
- Impact of application buffer reuse patterns on communication performance.
- Performance of intra-node communication.
- Performance of collective communication.
- Memory usage.

Our experimental testbed consists of 8 SuperMicro SUPER P4DL6 nodes with ServerWorks GC chipsets and dual Intel Xeon 2.40 GHz processors running Linux Red Hat 7.2 operating system. The same machines were connected by InfiniBand, Myrinet and Quadrics interconnects. InfiniHost HCA adapters and Myrinet cards work under PCI-X 64-bit 133MHz interfaces. Quadrics cards use 64-bit 66MHz PCI slots. MPI over InfiniBand is MVAPICH 0.9.1 with MPICH 1.2.2. MPI over GM uses GM 2.0.3 with MPICH 1.2.5. MPI over Quadrics uses MPICH 1.2.4. For the tests, we compile with Intel(R) C++ and FORTRAN Compilers for 32-bit applications Version 6.0.1 Build 20020822Z.

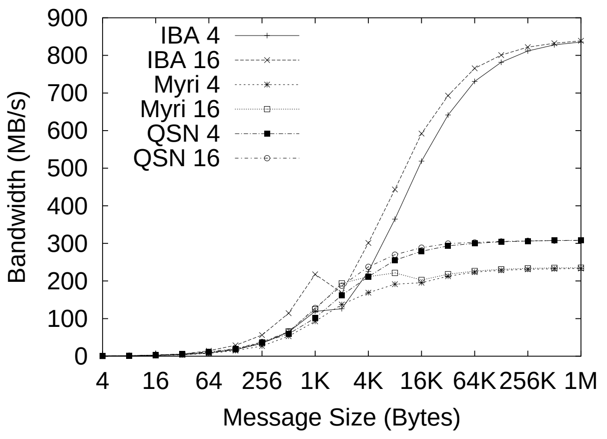

**Figure 2. MPI Bandwidth**

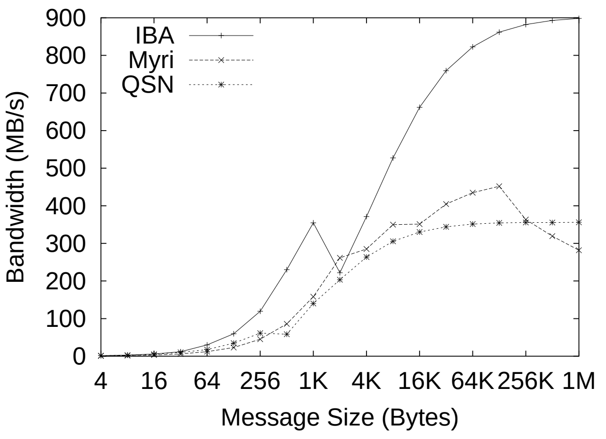

**Figure 5. MPI Bi-Directional Bandwidth**

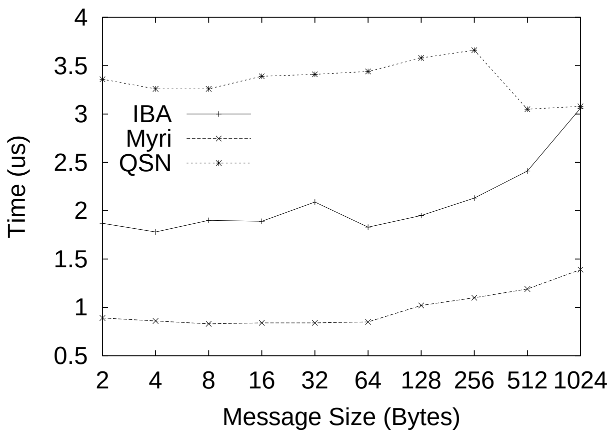

**Figure 3. MPI Host Overhead**

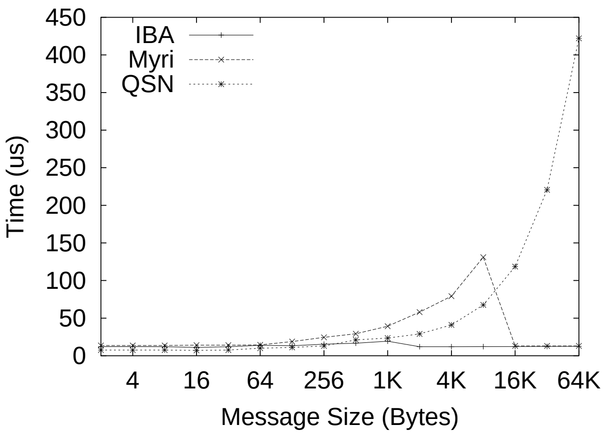

**Figure 6. Overlap Potential**

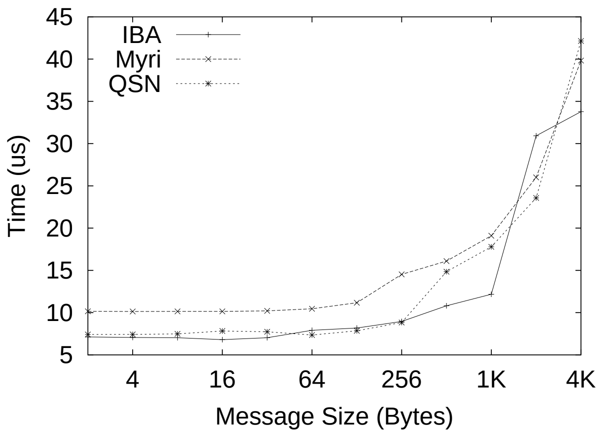

**Figure 4. MPI Bi-Directional Latency**

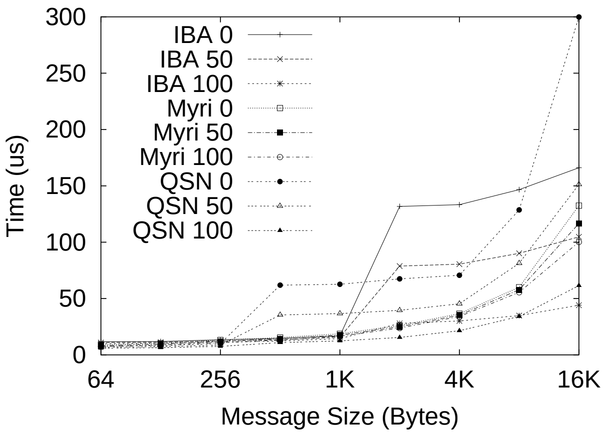

**Figure 7. MPI Latency with Buffer Reuse**

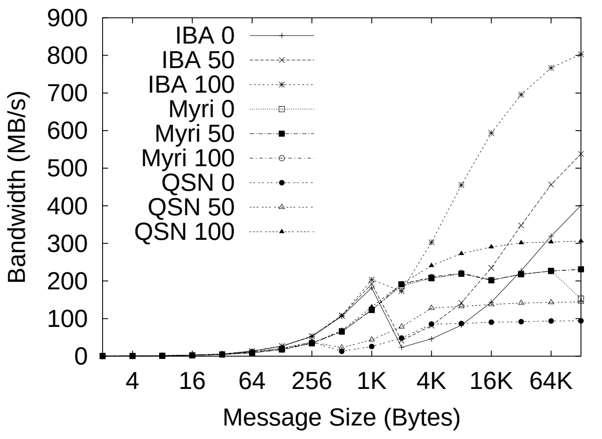

**Figure 8. MPI Bandwidth with Buffer Reuse**

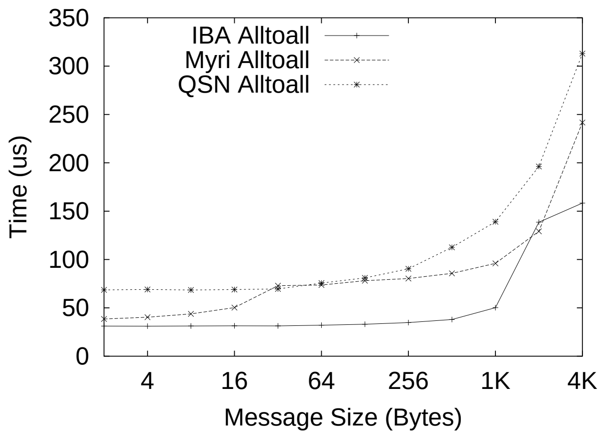

**Figure 11. MPI Alltoall**

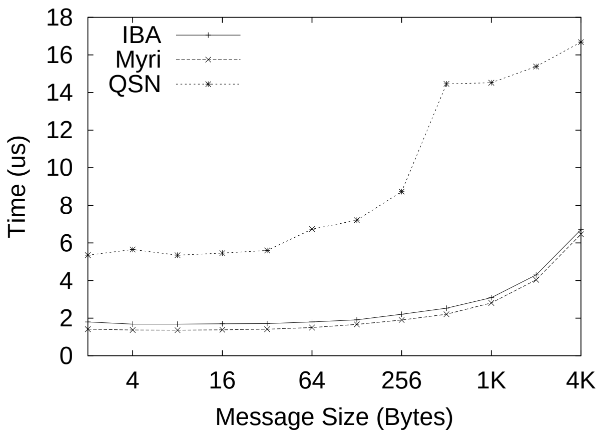

**Figure 9. MPI Intra-Node Latency**

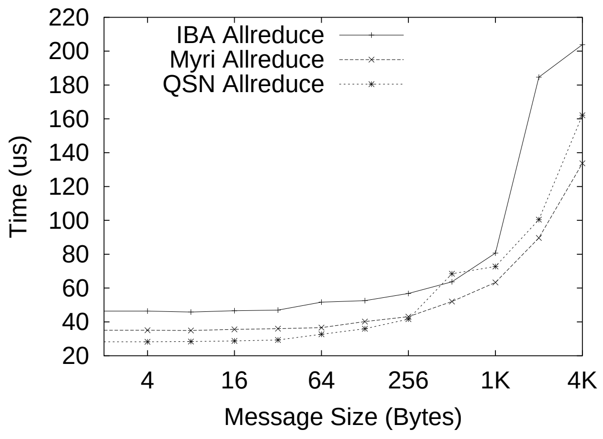

**Figure 12. MPI Allreduce**

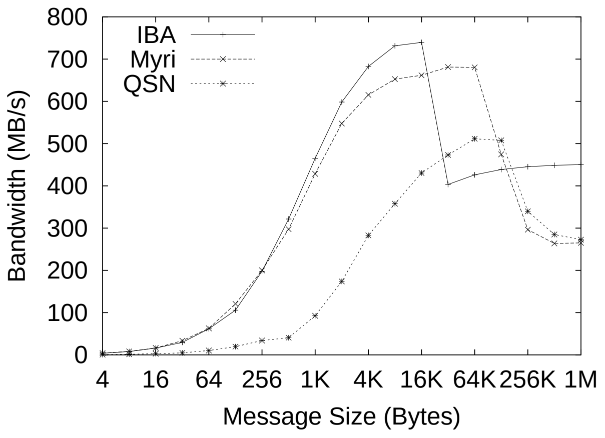

**Figure 10. MPI Intra-Node Bandwidth**

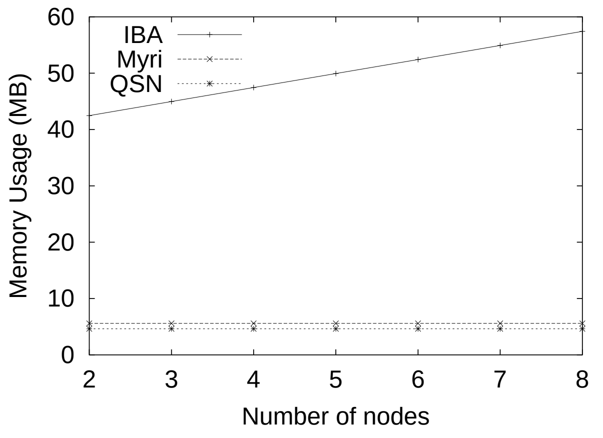

**Figure 13. MPI Memory Consumption**

### 3.1 Latency and Bandwidth

Figure 1 shows the MPI-level latency results. The test is conducted in a ping-pong fashion and the latency is derived from round-trip time. For small messages, Quadrics shows excellent latencies, which are around 4.6$\mu$s. The smallest latencies for InfiniBand and Myrinet are 6.8$\mu$s and 6.7$\mu$s, respectively. For large messages, InfiniBand has a clear advantage because of its higher bandwidth.

The bandwidth test is used to determine the maximum sustained data rate that can be achieved at the network level. Therefore, non-blocking MPI functions are used. In this test, a sender keeps sending back-to-back messages to the receiver until it has reached a pre-defined window size W. Then it waits for these messages to finish and sends out another W messages. Figure 2 shows the uni-directional bandwidth results for different W. For large messages and window size 16, InfiniBand can achieve bandwidth of over 841MB/s. (Note that unless stated otherwise, the unit MB in this paper is an abbreviation for $2^{20}$ bytes.) The bandwidth drop for 2KB messages is because the protocol switches from Eager to Rendezvous. The peak bandwidths for Quadrics and Myrinet are around 308MB/s and 235MB/s, respectively. The window size W also affects the bandwidth achieved, especially for small messages. For InfiniBand and Myrinet, their performance increases with the window size. Quadrics shows similar behavior for window size less than 16. However, its performance drops when the window size exceeds 16.

### 3.2 Host Overhead

Host overhead has been shown to have a significant impact on application performance [10]. Figure 3 presents the host overhead results for small messages in the latency test. The overhead includes both the sender side and the receiver side. It is obtained by measuring the time spent in communication. For Myrinet and InfiniBand, the overheads are around 0.8$\mu$s and 1.7$\mu$s, respectively. And their overheads increase slightly with the message size. Although Quadrics has better latency, it has higher overhead, which is around 3.3$\mu$s. Its overhead drops slightly after 256 bytes.

### 3.3 Bi-Directional Performance

Compared with uni-directional tests, bi-directional latency and bandwidth tests put more stress on the communication layer. Therefore they may be more helpful to us for understanding the bottleneck in communication. The tests are carried out in a way similar to the uni-directional ones. The difference is that both sides send data simultaneously.

Figure 4 shows the bi-directional latency results. We can see that both Quadrics and Myrinet show worse performance compared with their uni-directional latencies. For small messages, their respective bi-directional latencies are 7.4$\mu$s and 10.1$\mu$s. The latency for MPI over InfiniBand is 7.0$\mu$s. However, this number is only slightly worse than its un-directional latency (6.8$\mu$s).

Figure 5 shows the bi-directional bandwidth results. The window size of the bandwidth tests is 16. From the figure we notice that InfiniBand bandwidth increases from 841MB/s uni-directional to 942MB/s. Then it is limited by the bandwidth of the PCI-X bus. Quadrics bandwidth improves from 308MB/s to 375MB/s. Myrinet shows even more improvement. Its peak bandwidth increases from 235MB/s to 473MB/s. However, Myrinet bandwidth drops to less than 340MB/s when the message size is larger than 256KB.

### 3.4 Communication/Computation Overlap

To achieve better performance at the application level, one of the techniques MPI programmers use is to overlap communication with computation. To measure the ability to overlap computation with communication of each MPI implementation, we have designed an overlapping test using MPI non-blocking functions. The test is based on the latency test. At the sender, we use non-blocking MPI functions to start receive and send operations. Then the program enters a computation loop. After that it waits for the send and receive operations to finish. We define the potential of overlapping to be the maximum time of the computation loop that does not increase the latency.

We present the overlapping potential results in Figure 6. We can see that for small messages, InfiniBand and Myrinet have better overlapping potential compared with Quadrics because of their higher latencies and lower host overheads. However, the amount of overlapping drops at a certain point and stays as a constant. For Quadrics, the overlapping potential increases steadily with the message size.

MPI implementations usually use eager protocol for small messages and rendezvous protocol for large messages. The rendezvous protocol needs a handshake between the sender and the receiver. For MPI over InfiniBand and Myrinet, this handshake needs host intervention. Therefore, their abilities for overlapping computation and communication are limited by the rendezvous protocol. MPI over Quadrics is able to make communication progress asynchronously by taking advantages of the programmable network interface card. Thus it shows much better overlapping potential for large messages.

### 3.5 Impact of Buffer Reuse

For interconnects using user-level mode communication, application buffer reuse patterns can have significant impact on performance. This is due to the following reasons:

- Interconnects such as InfiniBand and Myrinet require that communication buffers be registered before communication. Therefore user buffers need to be registered in order to achieve zero-copy communication. To reduce the number of registration and de-registration (events), MPI implementations usually use techniques similar to pin-down cache [4] to de-register buffers in a lazy fashion. Thus, application buffer reuse patterns directly affect the hit rate of pin-down cache.

- Modern high speed network interfaces such as those studied in this paper usually have DMA engines to access host memory. An address translation mechanism is needed to translate user buffer virtual addresses to DMA addresses. Application buffer reuse patterns also affect the performance of this address translation process.

Our buffer reuse benchmark consists of N iterations of communication. We define a buffer reuse percentage R. For the N iterations of the test, N*R iterations will use the same buffer, while all other iterations will use completely different buffers. By changing buffer reuse percentage R, we can see how communication performance is affected by buffer reuse patterns. Figures 7 and 8 show the latency and bandwidth results for different buffer reuse percentage, respectively. We can see that all three MPI implementation are sensitive to the buffer reuse pattern. When the reuse percentage decreases, their performance drops significantly. For message sizes greater than 1KB, the IB latency suffers greatly when buffers are not reused. Quadrics also sees a steep rise in latency with lack of buffer reuse starting for all messages. Myrinet latency is not significantly affected until the message size reaches 16kB.

### 3.6 Intra-Node Communication

For SMP machines, it is possible to improve intra-node communication performance by taking advantage of shared memory mechanism. In this section, we present intra-node MPI performance for the three implementations. Figures 9 and 10 show the latency and bandwidth performance results. From the figures we can see that Quadrics does not perform well in SMP mode. Its intra-node latency is even higher than inter-node latency. The small message latencies for Myrinet and InfiniBand are about 1.3$\mu$s and 1.6$\mu$s, respectively. Bandwidth for both Myrinet and Quadrics drops for large messages because of cache thrashing. MPI over InfiniBand only uses shared memory for small messages (less than 16KB). Its bandwidth is over 450MB/s for large messages, which is significantly better than Myrinet and Quadrics.

### 3.7 Collective Communication

MPI collective communications can be implemented by using point-to-point MPI functions. However, to achieve optimal performance, we can also implement them directly over the message passing layer. This is even more desirable when an interconnect has special support for collective communications.

Two of the most frequently used MPI collective operations are MPI_Alltoall and MPI_Allreduce [21]. Figures 11 and 12 shows the performance of MPI_Alltoall and MPI_Allreduce for all three MPI implementations on 8 nodes. The Pallas MPI Benchmarks [15] have been used for these tests. For MPI_Alltoall operations, InfiniBand performs better than Quadrics and Myrinet, with a latency of 31$\mu$s for small messages compared with 67$\mu$s and 36$\mu$s for Quadrics and Myrinet, respectively. Quadrics achieves a latency of 28$\mu$s for small message MPI_Allreduce operations, which is better than Myrinet (35$\mu$s) and InfiniBand (46$\mu$s).

### 3.8 Memory Usage

One aspect of an MPI implementation often ignored by many micro-benchmarks is memory usage. The more memory allocated by the MPI implementation, the more likely it will adversely affect application performance. We run a simple MPI barrier program and measure the amount of memory it consumes. The memory data is obtained through the proc file systems in Linux.

The results are presented in Figure 13. We can see that MPI over Quadrics and Myrinet consume relatively small amount of memory, which does not increase with the number of nodes. Memory consumption for MPI over InfiniBand increases with the number of nodes. The reason for this increase is that the current implementation is built on top of InfiniBand Reliable Connection service. During initialization, a connection is set up between every two nodes and a certain amount of memory is reserved for each connection. Therefore, total memory consumption increases with the number of connections. This problem can be alleviated by using InfiniBand Reliable Datagram service or techniques like on-demand connection[23].

## 4 Applications

In this section, we compare the three MPI implementations using the NAS Parallel Benchmarks [13] and the sweep3D [5] benchmark. Basic MPI performance parameters such as latency, bandwidth and overhead play an important role in determining application performance. However, depending on the application, other factors in MPI implementation such as computation/communication overlapping, collective communication, memory usage and buffer reuse can have great impact as well. To better understand the relationship between application performance and MPI implementations, we have done profiling for the applications under study. By relating application communication characteristics and different aspects of MPI implementations, we can get much more insights into the communication behavior of these applications. The profiling data is obtained through the MPICH logging interface [3]. We

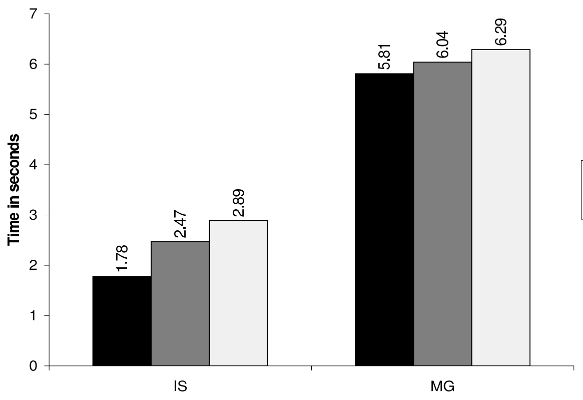

**Figure 14. IS and MG on 8 Nodes**

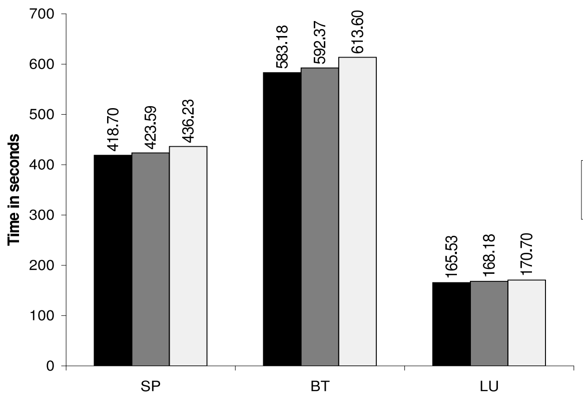

**Figure 15. SP and BT on 4 Nodes and LU on 8 Nodes**

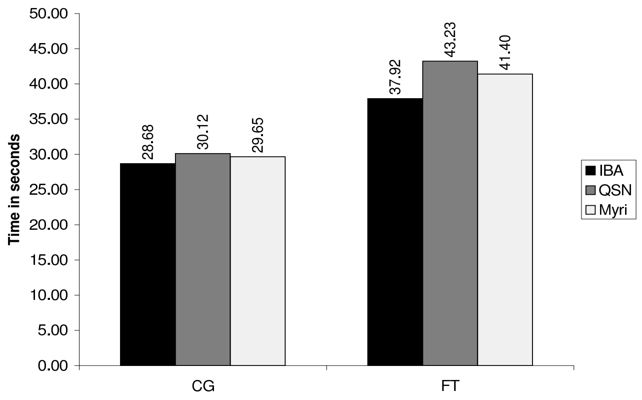

**Figure 16. CG and FT on 8 Nodes**

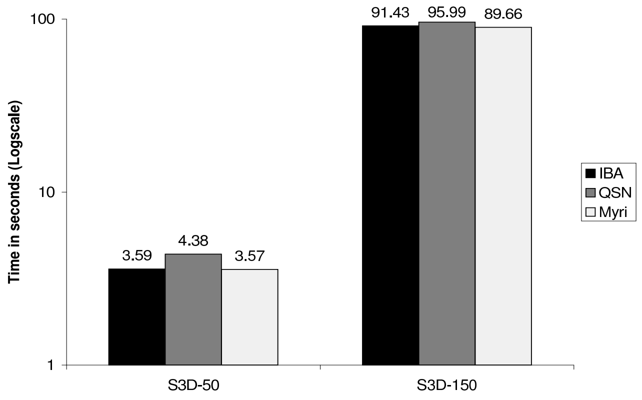

**Figure 17. Sweep3D on 8 Nodes**

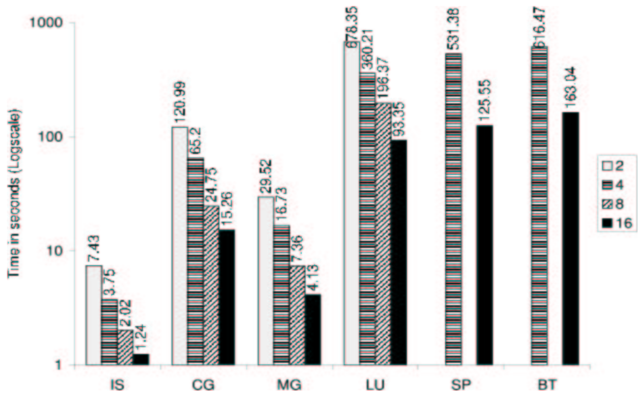

**Figure 18. Scalability with System Sizes for a 16-Node System at Topspin**

modified its source code to log more information such as buffer reuse patterns.

We also present application performance for SMP mode. To study the impact of PCI-X bus on the performance of MPI over InfiniBand, we make the InfiniBand HCAs run with 66MHz PCI bus and compare the performance with 133MHz PCI-X bus.

## 4.1 Application Performance Results

Figures 14, 15, 16 and 17 show the application running time for class B NAS parallel benchmarks and sweep3D. We use two input sizes for sweep3D: 50 and 150. We present 8 nodes results for IS, CG, MG, LU and FT. SP and BT require square number of nodes, therefore we only show results on 4 nodes for them. We can see that MPI over InfiniBand performs better than the other two implementations for all NAS benchmarks. The largest improvement comes from IS, which uses very large messages, as shown in Table 1. The much higher bandwidth of InfiniBand gives it a clear advantage. It performs 28% and 38%

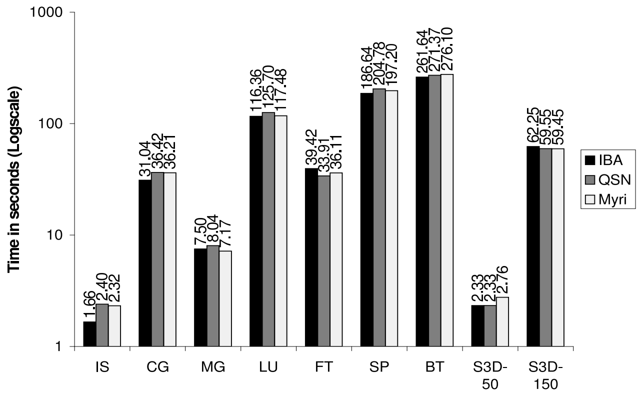

**Figure 19. SMP Performance (16 Processes on 8 Nodes at OSU with block mapping)**

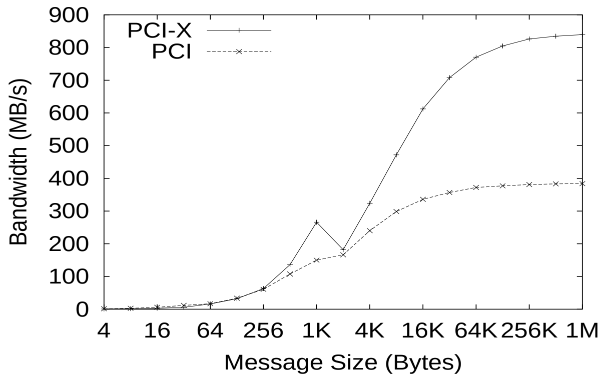

**Figure 22. MPI over InfiniBand Bandwidth with PCI**

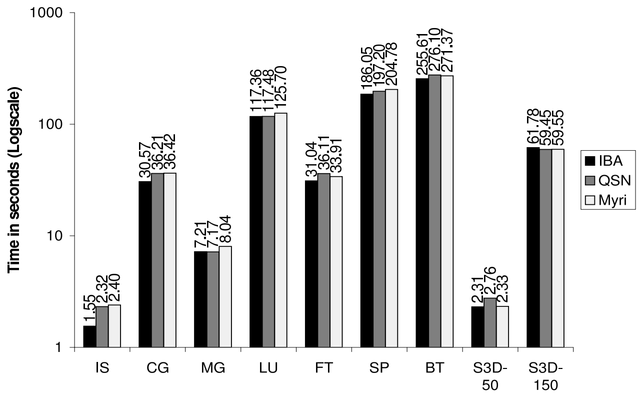

**Figure 20. SMP Performance (16 Processes on 8 Nodes at OSU with cyclic mapping)**

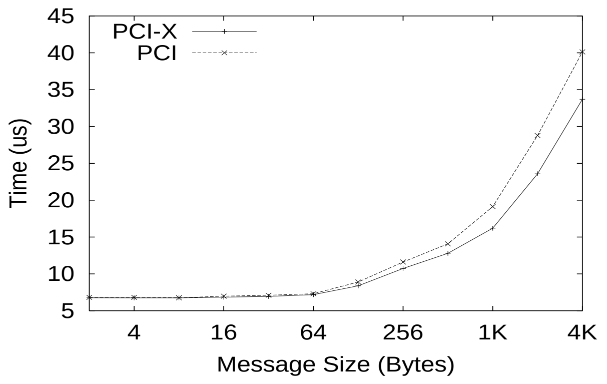

**Figure 23. MPI over Myrinet Latency with PCI**

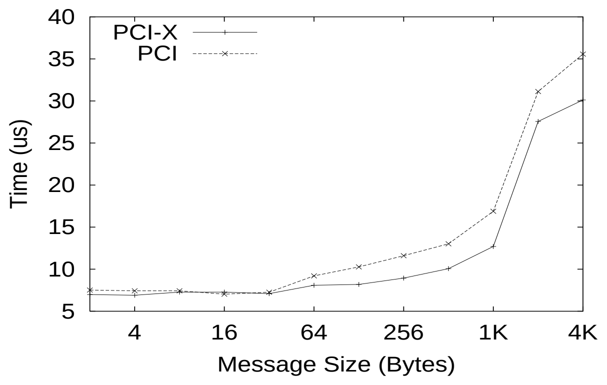

**Figure 21. MPI over InfiniBand Latency with PCI**

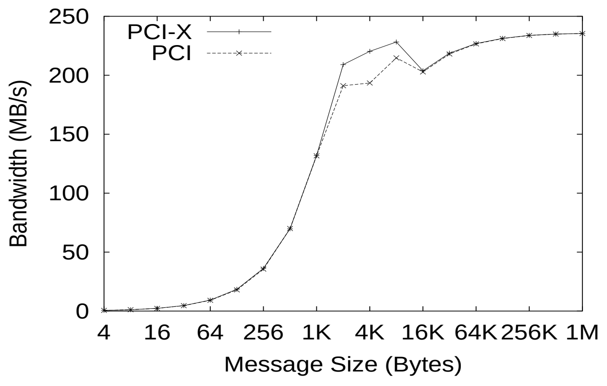

**Figure 24. MPI over Myrinet Bandwidth with PCI**

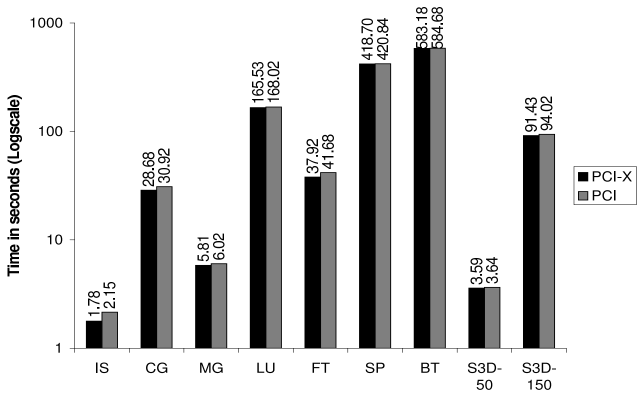

**Figure 25. MPI over InfiniBand Performance with PCI**

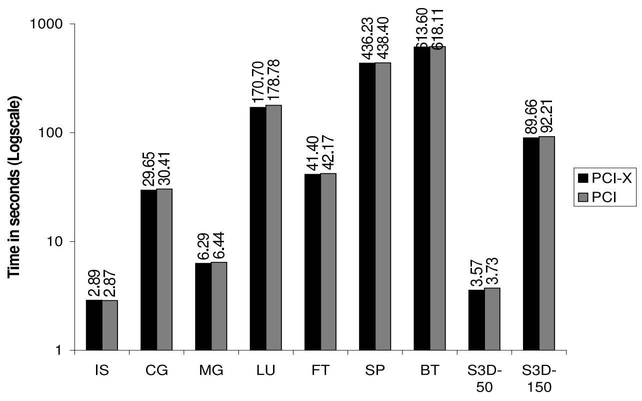

**Figure 26. MPI over Myrinet Performance with PCI**

better than Quadrics and Myrinet, respectively. For other applications which use many large messages, such as FT and CG, InfiniBand also performs significantly better. For SP, BT and MG, MPI over InfiniBand also outperforms the other two. For applications that mostly use small messages, like LU, Quadrics and Myrinet performance is more comparable with InfiniBand.

**Table 1. Message Size Distribution**

| **Apps** | <2K | 2K-16K | 16K-1M | >1M |
|---|---|---|---|---|
| IS | 14 | 11 | 0 | 11 |
| CG | 16113 | 0 | 11856 | 0 |
| MG | 1607 | 630 | 3702 | 0 |
| LU | 100021 | 0 | 1008 | 0 |
| FT | 24 | 0 | 0 | 22 |
| SP | 9 | 0 | 9636 | 0 |
| BT | 9 | 0 | 4836 | 0 |
| S3d-50 | 19236 | 0 | 0 | 0 |
| S3d-150 | 28836 | 28800 | 0 | 0 |

For the sweep3D benchmarks, Quadrics performs worse than InfiniBand and Myrinet for input size 50. The three implementations perform comparably for input size 150.

## 4.2 Scalability with System Size

To study the scalability of the MPI implementations, we have measured application performance for 2, 4 and 8 processes in our 8 node cluster. (Due to the problem size, FT and sweep3D with input 150 do not run on 2 nodes.) We also measure performance of MPI over InfiniBand on a 16 node Topspin InfiniBand cluster [20], which is connected through a Topspin 360 24 port 4x InfiniBand switch. The HCAs are Topspin InfiniBand 4x HCAs. And the hosts are Microway dual 2.4GHz P4 Xeon systems with 2GB of memory based on a Tyan 2721-533 motherboard. The results are shown in Table 2 and Figure 18. We can observe that all three MPI implementations have good scalability, with some applications like MG and CG showing super-linear speedup. For IS, which uses very large messages, MPI over InfiniBand still shows almost linear speedup. However, Myrinet and Quadrics do not perform as well as InfiniBand for IS.

## 4.3 Impact of Computation/Communication Overlap

The effect of computation and communication overlap in real applications is difficult to characterize. As an approximation, we have collected information for non-blocking MPI calls in the applications. The results are shown in Table 3. (Average sizes are in bytes.) We can see that different applications use non-blocking MPI functions very differently. FT and sweep3D do not use them at all. MG,

**Table 2. Scalability with System Sizes for Three Networks (Execution times are in seconds.)**

| Apps | IBA | | | Myri | | | QSN | | |
|---|---|---|---|---|---|---|---|---|---|
| | 2 | 4 | 8 | 2 | 4 | 8 | 2 | 4 | 8 |
| IS | 6.73 | 3.30 | 1.78 | 7.86 | 4.99 | 2.89 | 7.04 | 4.71 | 2.47 |
| CG | 132.26 | 81.64 | 28.68 | 135.76 | 74.36 | 29.65 | 135.05 | 73.10 | 30.12 |
| MG | 23.60 | 13.41 | 5.81 | 25.77 | 14.87 | 6.29 | 24.07 | 13.75 | 6.04 |
| LU | 648.53 | 319.57 | 165.53 | 708.43 | 338.70 | 170.70 | 667/30 | 314.55 | 168.18 |
| FT | - | 75.50 | 37.92 | - | 82.74 | 41.40 | - | 81.89 | 43.23 |
| S3d-50 | 13.58 | 7.18 | 3.59 | 13.33 | 6.96 | 3.57 | 14.94 | 7.37 | 4.38 |
| S3d-150 | - | 179.35 | 91.43 | - | 176.94 | 89.66 | - | 177.66 | 95.99 |

LU and CG only use non-blocking receive functions. SP and BT use both non-blocking send and non-blocking receive operations. We also noticed that the average sizes for non-blocking functions are very large. Therefore, it gives an advantage to MPI over Quadrics, which has better computation/communication overlap for large messages. As a result, for the applications with non-blocking operations, MPI over Quadrics performs more comparably with MPI over InfiniBand, as seen in the plots for SP and BT in Figure 15.

**Table 3. Non-Blocking MPI Calls**

| Apps | Isend | | Irecv | |
|---|---|---|---|---|
| | # calls | Avg Size | # calls | Avg Size |
| IS | 0 | 0 | 0 | 0 |
| CG | 0 | 0 | 13984 | 63591 |
| MG | 0 | 0 | 2922 | 270400 |
| LU | 0 | 0 | 508 | 311692 |
| FT | 0 | 0 | 0 | 0 |
| SP | 4818 | 263970 | 4818 | 263970 |
| BT | 2418 | 293108 | 2418 | 293108 |
| S3d-50 | 0 | 0 | 0 | 0 |
| S3d-150 | 0 | 0 | 0 | 0 |

## 4.4 Impact of Buffer Reuse

In Figures 7 and 8, we have shown that buffer reuse patterns have a significant impact on the performance of all the three MPI implementations. We define buffer reuse rate to be the percentage of accesses to previously used buffers. Table 4 shows buffer reuse rates and buffer reuse rates weighted by buffer sizes for all applications. One conclusion we can draw from the table is that in these applications, buffer reuse rates are very high. Therefore, although we have seen the MPI implementations can have different performance for different buffer reuse patterns, its impact to these applications is small. However, for other applications which have more dynamic memory usage patterns, this conclusion might not hold.

**Table 4. Buffer Reuse Rate**

| Apps | Buffer Reuse | |
|---|---|---|
| | % Reuse | Wt % Reuse |
| IS | 81.08 | 27.40 |
| CG | 99.99 | 99.98 |
| MG | 99.80 | 99.83 |
| LU | 99.99 | 99.80 |
| FT | 86.00 | 91.30 |
| SP | 99.92 | 99.89 |
| BT | 99.87 | 99.83 |
| S3d-50 | 99.96 | 99.99 |
| S3d-150 | 99.99 | 99.99 |

## 4.5 Impact of Collective Communication

In Table 5 we show the usage of MPI collective operations in these applications. We have measured the total number of collective calls, the percentage in terms of total MPI calls, and the percentage in terms of total communication volume. We can observe that different applications have very different usage patterns for collective operations. Applications like IS and FT almost exclusively use collective operations. On the other hand, in applications such as CG, collective functions are seldom used. However, overall it indicates that it is worthwhile to use special support in the hardware to improve the performance of collective operations. We also notice that the most frequently used collective operations are MPI_Alltoall and MPI_Allreduce. Therefore, priorities should be given to them when optimizing collective operations.

## 4.6 Impact of Intra-Node performance

To study the impact of running applications on an SMP, we have run the applications with 16 processes on 8 nodes. Figure 20 presents the performance results. We can see that except for MG and sweep3D with input 150, MPI over InfiniBand performs better than the other two implementations. (We should note that intra-node communication patterns depend on the mapping of processes to nodes. We have used "cyclic" mapping in our tests.)

Table 5. MPI Collective Calls

| Apps | MPI Collective Calls | | |
|---|---|---|---|
| | # calls | % calls | % Volume |
| IS | 35 | 97.22 | 100.00 |
| CG | 2 | 0.01 | 0.00 |
| MG | 101 | 1.70 | 0.03 |
| LU | 18 | 0.02 | 0.00 |
| FT | 47 | 100.00 | 100.00 |
| SP | 11 | 0.09 | 0.02 |
| BT | 11 | 0.22 | 0.01 |
| S3d-50 | 39 | 0.20 | 0.00 |
| S3d-150 | 39 | 0.07 | 0.00 |

Table 6. Intra-Node Point-Point Communication for Cylic Mapping

| Apps | Intra-node statistics | | |
|---|---|---|---|
| | # calls | % calls | % Volume |
| IS | 0 | 0.00 | 0.00 |
| CG | 0 | 0.00 | 0.00 |
| MG | 672 | 0.73 | 0.00 |
| LU | 0 | 0.00 | 0.00 |
| FT | 0 | 0.00 | 0.00 |
| SP | 0 | 0.00 | 0.00 |
| BT | 0 | 0.00 | 0.00 |
| S3d-50 | 0 | 0.00 | 0.00 |
| S3d-150 | 0 | 0.00 | 0.00 |

Table 7. Intra-Node Point-Point Communication for Block Mapping

| Apps | Intra-node statistics | | |
|---|---|---|---|
| | # calls | % calls | % Volume |
| IS | 16 | 100.00 | 100.00 |
| CG | 192128 | 42.93 | 33.41 |
| MG | 14912 | 16.25 | 1.43 |
| LU | 804044 | 33.16 | 21.89 |
| FT | 0 | 0.00 | 0.00 |
| SP | 70608 | 16.41 | 16.26 |
| BT | 25760 | 16.31 | 16.21 |
| S3d-50 | 153600 | 33.29 | 33.11 |
| S3d-150 | 460800 | 33.32 | 33.47 |

Table 7 presents the Intra-node profiling data for the applications with 16 processes on 8 nodes. The data is only for point-to-point communications. We can see that intra-node communication plays a very important role in many applications. For sweep3D, the point-to-point communication performance is solely determined by the efficiency of intra-node communication.

### 4.7 Impact of PCI-X Bus on MPI over InfiniBand

In previous experiments, the InfiniBand cards worked with PCI-X bus, which has higher frequency (133 MHz vs. 66MHz) and theoretical bandwidth (1024MB/s vs. 512MB/s) than PCI bus. To study the impact of PCI-X bus, we conducted experiments by forcing InfiniBand to use PCI bus. Figures 21 and 22 show the latency and bandwidth results. The latency for small messages is not significantly affected and only increases by about 0.6$\mu$s. However, the bandwidth decreases and only reaches 378MB/s. The drop in bandwidth also increases latency for large messages.

Figure 25 shows class B NAS benchmark performance on 8 nodes. (SP and BT results are for 4 nodes.) We can see that on the average, the performance degrades by about 11% compared with PCI-X. Comparing with Myrinet and Quadrics, InfiniBand with PCI performs still better for IS, FT and CG. All these applications are bandwidth-bound and use many large messages. However, for other applications InfiniBand with PCI performs worse.

## 5 Related Work

An interesting evaluation of current high performance networks was carried out in [1]. The authors used LogP model to evaluate a wide variety of interconnects at both the MPI level and the low level messaging software level. However, they did not include InfiniBand and the tests were done only at micro-benchmark level. The networks they studied were in different systems. We have done a performance evaluation in a single cluster for different interconnects, which makes it possible to compare them with minimum impact from other parts of the system.

Work done in [6] used both micro-benchmarks and the NAS Parallel Benchmarks to study the performance of Giganet and Myrinet on clusters of SMP servers. Our work follows a similar approach. However, we have greatly expanded the set of micro-benchmarks and studied the relationship between application communication characteristics and different performance aspects of MPI.

The LogP model was proposed in [2], and a study of application performance sensitivity to LogP parameters were carried out in [10]. In our micro-benchmarks, we include not only measurements similar to those in the LogP model, but also additional tests to characterize other performance aspects of MPI implementations.

There have also been many studies about communication patterns for parallel applications. Studies of the NAS

Parallel Benchmarks have done in [22, 19]. Another excellent study on communication characteristics of large scale scientific applications was conducted in [21]. The focus of our work is to compare the three different MPI implementations. Therefore, we have used the communication pattern information to study the impact of different MPI implementations on application performance.

## 6 Conclusions

In this paper, we have presented a detailed performance study of MPI over InfiniBand, Myrinet and Quadrics, using both applications and micro-benchmarks. We have shown that MPI communication performance is affected by many factors. Therefore, to get more insights into different aspects of an MPI implementation, one has to go beyond simple micro-benchmarks such as latency and bandwidth. For example, we found that all the three MPI implementations are sensitive to buffer reuse patterns. We also found that MPI over Quadrics has better ability for overlapping computation and communication, and MPI over GM offers the best intra-node communication performance. None of these can be revealed by simple inter-node latency and bandwidth tests.

Our study also shows that although InfiniBand is relatively young in the HPC market, it is able to give very good performance at the MPI level. The performance gains of InfiniBand are not only due to its using a PCI-X bus, where as seen in the experiments, InfiniBand on a PCI bus can still outperform other interconnects.

### Acknowledgments

We would like to thank Kevin Deierling, Jeff Kirk, and Ezra Silvera from Mellanox Technologies for their support with the InfiniBand hardware and software. Thanks to Brad Benton from Fabric Networks for providing us their hardware and software and for having many interesting discussions related to InfiniBand. Thanks to Fabrizio Petrini from LANL and David Addison from Quadrics for having many discussions related to Quadrics. We would also like to thank Ben Eiref, Robert Starmer, and Lorne Boden from Topspin Communications for all their efforts in providing us access to their 16 node InfiniBand cluster.

## References

[1] C. Bell, D. Bonachea, Y. Cote, J. Duell, P. Hargrove, P. Husbands, C. Iancu, M. Welcome, and K. Yelick. An Evaluation of Current High-Performance Networks. In *International Parallel and Distributed Processing Symposium (IPDPS'03)*, April 2003.

[2] D. E. Culler, R. M. Karp, D. A. Patterson, A. Sahay, K. E. Schauser, E. Santos, R. Subramonian, and T. von Eicken. Logp: Towards a realistic model of parallel computation. In *Principles Practice of Parallel Programming*, pages 1–12, 1993.

[3] W. Gropp, E. Lusk, N. Doss, and A. Skjellum. A high-performance, portable implementation of the MPI message passing interface standard. *Parallel Computing*, 22(6):789–828, 1996.

[4] H. Tezuka and F. O'Carroll and A. Hori and Y. Ishikawa. Pin-down Cache: A Virtual Memory Management Technique for Zero-copy Communication. In Proceedings of 12th IPPS.

[5] A. Hoisie, O. Lubeck, H. Wasserman, F. Petrini, and H. Alme. A General Predictive Performance Model for Wavefront Algorithms on Cluster of SMPs. In *ICPP 2000*, 2000.

[6] J. Hsieh, T. Leng, V. Mashayekhi, and R. Rooholamini. Architectural and performance evaluation of giganet and myrinet interconnects on clusters of small-scale SMP servers. In *Supercomputing*, 2000.

[7] InfiniBand Trade Association. InfiniBand Architecture Specification, Release 1.0, October 24 2000.

[8] J. Liu, J. Wu, S. P. Kini, D. Buntinas, W. Yu, B. Chandrasekaran, R. Noronha, P. Wyckoff, and D. K. Panda. MPI over InfiniBand: Early Experiences. Technical Report, OSU-CISRC-10/02-TR25, Computer and Information Science department, the Ohio State University, January 2003.

[9] J. Liu, J. Wu, S. P. Kini, P. Wyckoff, and D. K. Panda. High Performance RDMA-Based MPI Implementation over InfiniBand. In *17th Annual ACM International Conference on Supercomputing (ICS '03)*, June 2003.

[10] R. Martin, A. Vahdat, D. Culler, and T. Anderson. Effects of Communication Latency, Overhead, and Bandwidth in a Cluster Architecture. In *Proceedings of the International Symposium on Computer Architecture*, 1997.

[11] Mellanox Technologies. Mellanox InfiniBand InfiniHost Adapters, July 2002.

[12] Myricom. Myricom, Inc. http://www.myri.com.

[13] NASA. NAS Parallel Benchmarks.

[14] N.J. Boden, D. Cohen, R.E. Felderman, A.E. Kulawik, C.L. Seitz, J.N. Seizovic, and W. Su. Myrinet: A Gigabit-per-second Local Area Network. *IEEE Micro*, 15(1):29–36, February 1995.

[15] Pallas. Pallas MPI Benchmarks. http://www.pallas.com/e/products/pmb/.

[16] F. Petrini, W. Feng, A. Hoisie, S. Coll, and E. Frachtenberg. The Quadrics Network: High-Performance Clustering Technology. *IEEE Micro*, 22(1):46–57, 2002.

[17] Quadrics. Quadrics, Ltd. http://www.quadrics.com.

[18] M. Snir, S. Otto, S. Huss-Lederman, D. Walker, and J. Dongarra. *MPI–The Complete Reference. Volume 1 - The MPI-1 Core, 2nd edition.* The MIT Press, 1998.

[19] T. Tabe and Q. F. Stout. The use of the MPI communication library in the NAS parallel benchmarks. Technical Report CSE-TR-386-99, University of Michgan, 1999.

[20] Topspin. Topspin InfiniBand Products. http://www.topspin.com.

[21] J. S. Vetter and F. Mueller. Communication Characteristics of Large-Scale Scientific Applications for Contemporary Cluster Architectures. In *IPDPS 02*, April 2002.

[22] F. C. Wong, R. P. Martin, R. H. Arpaci-Dusseau, and D. E. Culler. Architectural requirements and scalability of the nas parallel benchmarks. In *In the Proceedings of Supercomputing'99*, 1999.

[23] J. Wu, J. Liu, P. Wyckoff, and D. K. Panda. Impact of On-Demand Connection Management in MPI over VIA. In *Proceedings of the IEEE International Conference on Cluster Computing*, 2002.