Myrinet Networks: A Performance Study

Ying Qian Ahmad Afsahi Reza Zamani
Department of Electrical and Computer Engineering
Queen's University
Kingston, ON, Canada K7L 3N6
{qiany, ahmad, zamanir}@ee.queensu.ca

Abstract

As network computing become commonplace, the interconnection networks and the communication system software become critical in achieving high performance. Thus, it is essential to systematically assess the features and performance of the new networks. Recently, Myricom has introduced a two-port "E-card" Myrinet/PCI-X interface. In this paper, we present the basic performance of its GM2.1 messaging layer, as well as a set of microbenchmarks designed to assess the quality of MPI implementation on top of GM. These microbenchmarks measure the latency, bandwidth, intra-node performance, computation/communication overlap, parameters of the LogP model, buffer reuse impact, different traffic patterns, and collective communications.

We have discovered that the MPI basic performance is close to those offered at the GM. We find that the host overhead is very small in our system. The Myrinet network is shown to be sensitive to the buffer reuse patterns. However, it provides opportunities for overlapping computation with communication. The Myrinet network is able to deliver up to 2000MB/s bandwidth for the permutation patterns.

1. Introduction

In different phases of distributed computations, hosts may exchange a large number of short and long messages over the interconnection network. To achieve performance on network-based computing systems, the interconnection network and the communication system software must provide mechanisms to support efficient communications. Meanwhile, in a distributed environment, applications usually run on top of a standard middleware such as the Message Passing Interface (MPI), which itself runs on top of a user-level messaging layer. To determine if the applications can benefit from a particular interconnect, it is essential to assess the various features and the performance of the interconnect at both the user and middleware levels.

Several high-performance interconnects have been recently introduced including the Quadrics QsNet [15], and QsNet II [1, 17], InfiniBand [13], Myrinet [4], Giganet [18], and Sun Fire Link [16]. Each one of these interconnects provides different levels of performance, programmability, and integration with the operating systems. Recently, Myricom [14] has introduced its dual-port Myrinet "E-Card" for higher performance with a new messaging layer (GM2.1).

In this work, we evaluate the performance of the Myrinet two-port networks at the user-level (GM2.1), and the MPI level (due to limited space, we report the application performance in another work). To the best of our knowledge, this is the first study of the Myrinet two-port networks. This paper first contributes by presenting the basic performance of the GM2.1. The second part is a set of microbenchmarks designed to assess the quality of MPI implementation on top of GM. These microbenchmarks measure the latency, bandwidth, intra-node performance, computation/communication overlap, LogP parameters, buffer reuse impact, different traffic patterns, and collective communications.

Our experiments show that the two-port Myrinet network outperforms its previous (one-port) generation; and that its performance is comparable to the other high-performance system area networks. We found that the host overhead is very small, and applications, using non-blocking function calls, can benefit from overlapping their computations with communications.

The rest of the paper is organized as follows. In Section 2, we provide an overview of the Myrinet, GM, and MPICH over GM. We describe our experimental framework in Section 3. Section 4 presents the basic GM performance. Our MPI microbenchmarks are discussed in Section 5. Related work is presented in section 6. Finally, we conclude our paper in section 7.

2. Overview of Myrinet, GM, and MPICH-GM

Myrinet [4] is based on packet-switching technology, where the packets are wormhole-routed through a network consisting of switching elements and *network interface cards* (NIC). Each NIC, attached to the host's I/O bus, contains a programmable processor and on-board memory that is used to store the control program and data. This programmability provides much flexibility in designing communication software.

The new M3F2-PCIXE-2 Myrinet/PCI-X interface card (E-card) has the ability to double the throughput and provide high availability. The E-card has a 64-bit, 133MHz PCI-X interface, and has a programmable Lanai-2XP RISC processor operating at 333MHz with 2MB local memory. Each port has a 2.0+2.0 Gbps data rate. The standard firmware executing in the Lanai-2XP distributes packets adaptively across the two ports, such that the two ports act as a single 4.0+4.0 Gbps data-rate port. It also provides multi-path dispersive routing in the Myrinet switch fabric that diffuses hot spots and increases the utilization of the network bisection for large network. The high-availability operation is instantaneous; by removing the fiber cable from either port, communication will continue using the remaining port.

GM [14] is a commercial open source user-level networking protocol from Myricom, which runs on top of the Myrinet network. It provides a protected user-level interface to the NIC so that multiple user processes can share it simultaneously. GM supports a connectionless communication model, and provides reliable and ordered delivery between the communication endpoints. The Myrinet software support for the latest E-card requires GM 2.1.0 or later (or MX, yet to be released). GM supports both *send/receive* and *Remote Direct Memory Access* (RDMA) operations. The send/receive mode is a two-sided operation, where both the sender and the receiver of a message are involved in the communication. However, RDMA is a one-sided operation, where a node has access to the remote memory of other nodes. Operations finish without the remote side being involved.

MPICH [7] is a portable implementation of the MPI. MPICH-GM is implemented by targeting its Channel Interface to the GM messaging layer. MPICH-GM uses the *eager* protocol for sending small (less than 16K), and control messages via GM send/receive operations. It uses the *rendez-vous* protocol for sending large messages via GM one-sided *put* operation.

3. Experimental Framework

In this paper, we present the communication performance of the Myrinet E-card with its two ports operational. We first present the send/receive performance at the GM level (we will report the RDMA performance in a future work). In order to gain a better understanding of the communication subsystem in Myrinet SMP clusters, we have designed and implemented a set of microbenchmarks at the MPI level. Our microbenchmarks include latency and bandwidth measurements at the intra-node and inter-node levels, buffer reuse impact on performance, computation/communication overlap potential, LogP parameters, uniform and permutation traffic patterns, and collective communications.

We ran our experiments on an 8-node dedicated SMP cluster interconnected with Myrinet E-cards, and a Myrinet-2000 16-port switch. Each node is a Dell PowerEdge 2650 that has two Intel Xeon MP 2.0 GHz Processors and 1GB RAM running Linux RedHat 9 with SMP kernel version 2.4.24. Each node has a two-port Myrinet E-card on a 64-bit, 133 MHz PCI-X slot. Our PCI-X/DMA performance for a bus read, bus write, and bus read and write bandwidth is 775MB/s, 1040MB/s, and 873MB/s, respectively. We used the MPICH-GM version 1.2.5..10 as the message-passing library on top of the GM2.1.0 messaging layer.

4. GM Performance

The short-message latency and bandwidth are two important metrics for many distributed computations. We measure the *unidirectional latency/bandwidth* with different message sizes. In this test, the sender transmits a message repeatedly to the receiver, and then waits for the last message to be acknowledged. The *bidirectional latency* test is the ping-pong test where the sender sends a message and the receiver upon receiving the message, immediately replies with the same message size. In the *both-way bandwidth* test, both the sender and receiver send data simultaneously. This test puts more pressure on the communication subsystem, and the PCI-X bus.

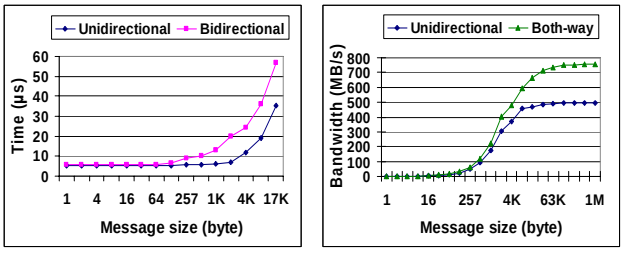

Figure 1. GM send/receive performance.

Figure 1 illustrates the GM send/receive latency and bandwidth, using the "all-size" utility. GM has small message latencies of 5.25, and 5.53 microseconds for unidirectional and bidirectional, respectively. The peak

unidirectional bandwidth is 496MB/s, while the sustained both-way bandwidth achieves 755MB/s. This is a major improvement over the Myrinet networks using the one-port D-card (471MB/s peak bandwidth [10]).

5. MPI Microbenchmarks

In this section, we present our microbenchmarks to evaluate the quality of the MPI layer on top of the GM messaging layer.

5.1. Latency

In deriving the bidirectional latency (and bandwidth in Section 5.2), we used matching pairs of sends and receives under different MPI send modes; that is, the *standard* mode, the *synchronous* mode, the *buffered* mode, and the *ready* mode. In the synchronous mode, the send call returns when the receiver has received the message. In the buffered mode, the message is buffered and the send call returns without waiting for the receiver. In the standard mode, MPI implementation chooses to buffer or to wait for the receiver. In the ready mode, matching receive must have been issued before the send.

The latency for on-node communication is shown in Figure 2. The on-node latency for 1-byte message remains at 1.38µs for unidirectional ping, 1.46µs for standard, and ready modes, 2.29µs for buffered, and 4.51µs for the synchronous mode. The latency for off-node communication is also shown in Figure 2. An overall observation satisfies the argument that on-node latency should normally be better than the off-node. However, in our system, on-node messages larger than 128KB have larger latencies than their counterparts. This could be due to insufficient memory in our SMP nodes, or unoptimized intra-node communication in MPICH-GM.

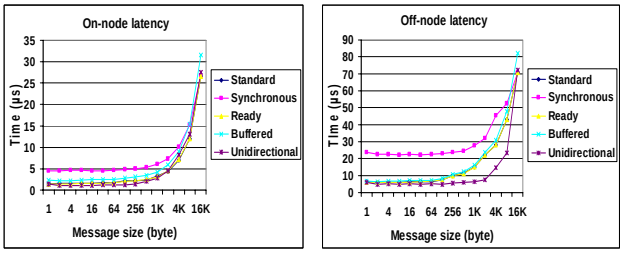

Figure 2. MPI on-node, off-node latency.

The off-node synchronous mode has a large latency, 23.6µs for up to 512-byte messages. This suggests that synchronous mode should be avoided for short message transfers. Off-node latency for 1-byte message remains at 5.7µs for unidirectional ping (compared to 5.25µs for GM), 6.1µs for standard (compared to 5.53µs for GM) and ready modes, and 6.84µs for the buffered mode. This shows MPICH has been ported to GM very well. A quick comparison with other high-performance interconnect are as follows: Quadrics QsNet: 4.6µs [15], and QsNet II: ~3µs [1, 17], Myrinet D-card: 6.7µs [10], InfiniBand: 6.8µs [10], and Sun Fire Link: 5µs [16].

5.2. Bandwidth

Figure 3 presents the on-node and off-node bandwidth for unidirectional, bidirectional, and both-way communications. The on-node bandwidth arrives at a maximum of 672MB/s, 702MB/s, and 870MB/s for unidirectional, standard bidirectional, and both-way, respectively. However, it suddenly drops afterwards.

The off-node unidirectional and bidirectional bandwidth (except for the buffered mode) achieves 440MB/s. MPICH-GM delivers a both-way bandwidth of 690MB/s equal to 91% of the GM performance. We see a drop at 16KB for the bandwidth due to the protocol switch. A quick comparison with other high-performance interconnect are as follows: Quadrics QsNet: 308MB/s [15], and QsNet II: ~900MB/s [1, 17], Myrinet D-card: 471MB/s [10], InfiniBand: 841MB/s [10], and Sun Fire Link: 792MB/s [16].

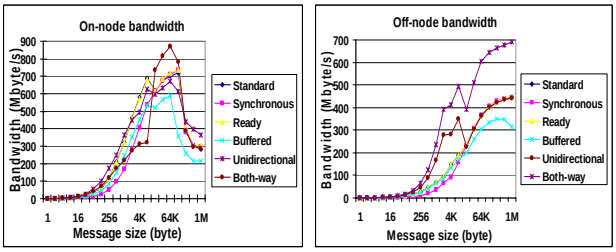

Figure 3. MPI on-node, off-node bandwidth.

5.3. LogP Parameters

The time spent for communication at the source and destination is a major contributing factor to the performance of applications in network-computing systems. *LogP* model has been proposed to gain insights into different components of a communication step [6]. LogP models sequences of point-to-point communications of short messages. L is the network hardware latency for one-word message transfer. O is the combined overhead in processing the message at the sender (o_s) and receiver (o_r). P is the number of processors. The gap, g, is the minimum time interval between two consecutive message transmission from a processor. *LogGP* [2] extends LogP to also cover long messages. The Gap per byte for long messages, G, is defined as the time per byte for a long message.

An efficient method for measurement of LogP parameters has been proposed in [9]. The method is called *parameterized LogP* and subsumes both LogP, and LogGP models. Figure 4, in logarithmic scales, shows the o_s, o_r, and g for different message sizes in our Myrinet cluster. Note that both o_s and o_r have a sudden increase at 16KB (from 5 to 70 microseconds) due to protocol switch in MPICH-GM. The combined host overhead for a 1-byte message is 0.8µs; gap is 2.3µs, and L is 4.2µs.

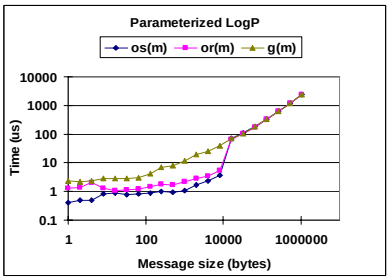

Figure 4. LogP parameters.

5.4. Buffer Reuse Impact

Applications employing different message buffers may need to register/deregister their buffers to achieve zero-copy message transfer in the Myrinet. Pinning/unpinning is expensive, thus, application buffer reuse pattern may have significant impact on the communication performance. The buffer reuse pattern exposes the MPI buffer management cost as well as virtual-to-physical address translation cost at the NIC.

Our test in this section is similar to [10]. We ran our standard latency/bandwidth test for thousand times but with a buffer reuse pattern of 0%, 25%, 50%, 75%, and 100%. A 100% buffer reuse pattern means that only one buffer is used for all message transfers. A 0% buffer reuse pattern means that each time a new buffer is used. We experimented with different message sizes.

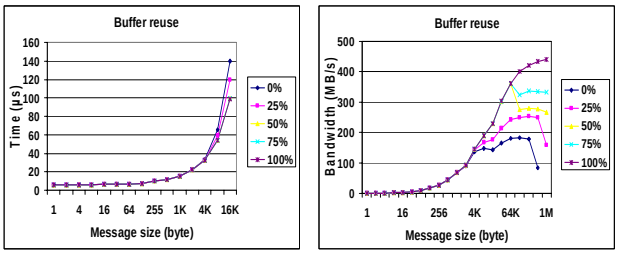

Figure 5. Buffer reuse impact on MPI.

Figure 5 shows the impact of buffer reuse pattern on the MPI latency and bandwidth. It is clear that with decreasing buffer reuse percentage, the performance drops. Specifically, the latency starts increasing with decreasing buffer reuse percentage after 4KB messages. The higher buffer reuse percentages (50% and 75%) affect the bandwidth starting at 64KB messages, while this happens at small messages (4KB) for lower buffer reuse percentages (0% and 25%).

5.5. Computation/Communication Overlap

Overlapping computation with communication can improve the application performance. A programmable NIC, such as the Myrinet E-card, offloads some of the host's functionality in a message transfer. Therefore, there will be opportunities for overlapping computation with the communication.

We use the latency test but with non-blocking send and receive function calls. However, we insert computation in between until the original latency increases. Figure 6 shows the overlap potential for different message sizes. There is about 6µs overlap for messages up to 256 bytes. This increases until a drop is observed due to the protocol switch in MPICH-GM. From then on, the overlap increases again.

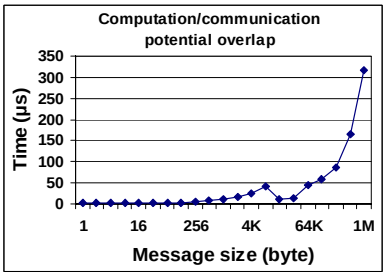

Figure 6. Computation/communication overlap.

5.6. Uniform and Permutation Traffic

Our intension is to analyze the network performance under uniform and permutation traffic patterns, where each sender selects a random or a fixed destination, respectively [15]. However, note that these patterns may generate on-node and/or off-node traffics.

5.6.1. Uniform Traffic. Each sender selects its destination randomly with a uniform distribution. The message sizes (identified by "S" in Figure 7) and message inter-arrival times ("T" in Figure 7) are randomly generated using uniform and exponential distributions. Figure 7 shows the accepted bandwidth under the uniform traffic. 10KB is chosen as the mean message size. The off-node accepted bandwidths are staying almost the same for all message sizes and inter-arrival times with different distributions. The accepted bandwidth is around 1600MB/s.

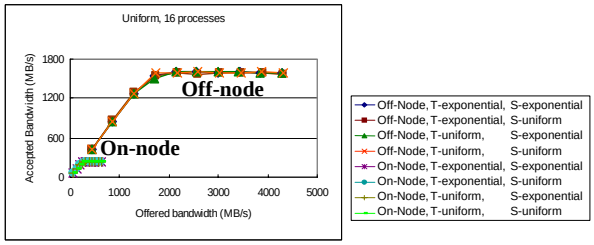

Figure 7. Uniform traffic accepted bandwidth.

5.6.2. Permutation Traffic. In these patterns, each sender communicates with a fixed destination, identified using one of the following traffics:

- ***Baseline:*** the *i*th baseline permutation is defined by $\beta_i(a_{n-1}, \ldots, a_{i+1}, a_i, a_{i-1}, \ldots, a_1, a_0) =$ $a_{n-1}, \ldots, a_{i+1}, a_0, a_i, a_{i-1}, \ldots, a_1$ $(0 \le i \le n-1)$.
- ***Bit-reversal*:** the process with binary coordinates $a_{n-1}, a_{n-2}, \ldots, a_1, a_0$ always communicates with the process $a_0, a_1, \ldots, a_{n-2}, a_{n-1}$.
- ***Butterfly*:** the *i*th butterfly permutation is defined by $\beta_i(a_{n-1}, \ldots, a_{i+1}, a_i, a_{i-1}, \ldots, a_0) =$ $a_{n-1}, \ldots, a_{i+1}, a_0, a_{i-1}, \ldots, a_i$ $(0 \le i \le n-1)$.
- ***Complement*:** the process with binary coordinates $a_{n-1}, a_{n-2}, \ldots, a_1, a_0$ always communicates with the process $\overline{a_{n-1}}, \overline{a_{n-2}}, \ldots, \overline{a_1}, \overline{a_0}$.
- ***Cube:*** the *i*th cube permutation is defined by $\beta_i(a_{n-1}, \ldots, a_{i+1}, a_i, a_{i-1}, \ldots, a_0) =$ $a_{n-1}, \ldots, \overline{a_{i+1}}, a_i, a_{i-1}, \ldots, a_0$ $(0 \le i \le n-1)$.
- ***Matrix transpose*:** the process with binary coordinates $a_{n-1}, a_{n-2}, \ldots, a_1, a_0$ always communicates with the process $a_{n/2-1}, \ldots, a_0, a_{n-1}, \ldots, a_{n/2}$.
- ***Neighbor:*** processes are divided into pairs. Each pair consists of two adjacent processes. Process 0 communicates with process 1, process 2 with process 3, and so on.
- ***Perfect-shuffle:*** the process with binary coordinates $a_{n-1}, a_{n-2}, \ldots, a_1, a_0$ always communicates with the process $a_{n-2}, a_{n-3}, \ldots, a_0, a_{n-1}$.

Figure 8 shows the accepted bandwidth for 16 processes for the Myrinet. Note that the *Butterfly*, *Cube*, and *Baseline* have *single-stage* and *multi-stage* patterns. *Single-stage* is the highest stage permutation, and *multi-stage* is the full stage permutation. It is clear that the Myrinet is able to accept up to 2000MB/s bandwidth for the permutation patterns.

5.7. Collective Communications

Efficient implementation of collective communication algorithms is one of the keys to the performance of applications. We have chosen *broadcast, gather, scatter, reduce, allreduce, alltoall, alltoallv,* and *barrier* operations as representatives of the mostly used collective communication operations in parallel applications. Our experiments are done with processes located on the same node and/or on different nodes. In the off-node case, we evenly divided the processes among the eight nodes. We measure the performance in terms of their completion time, as shown in Figure 9. Broadcast has the best performance, and allreduce has the largest completion time, followed by alltoallv and reduce. The barrier operation (not shown here) takes 9.8µs, 19.1µs, 29.5µs, and 52.4µs long, for 2, 4, 8, and 16 processes, respectively.

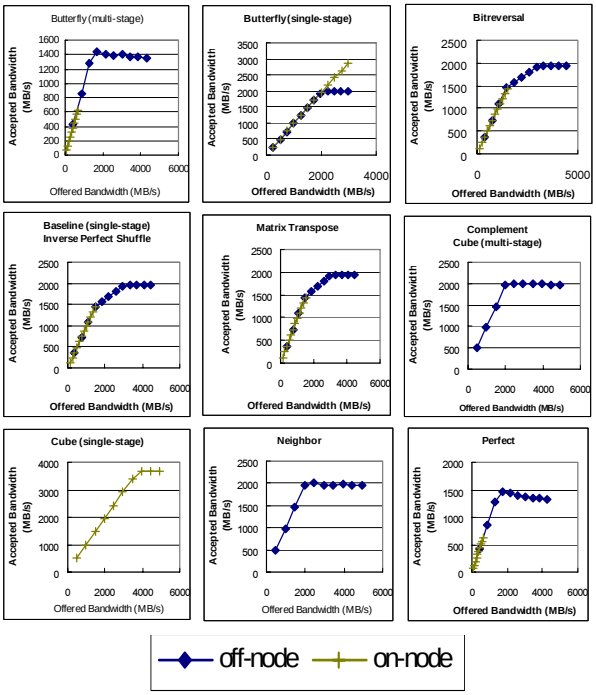

Figure 8. Permutation pattern accepted bandwidth.

6. Related Work

Research groups in academia and industry have been studying the performance of the interconnection networks at different levels [10, 11, 1, 15, 3, 8, 16, 5]. In [10], the authors have assessed the performance of the MPI implementation on InfiniBand, Myrinet, and Quadrics. They studied microbenchmarks at the MPI level as well as NAS benchmarks. Our work is similar to their work. However, we have investigated the recently introduced Myrinet two-port networks at the GM and MPI levels. We have also extended the microbenchmarks by including different traffic patterns. The authors in [1, 15] studied the Quadrics QsNet, and the QsNet II. Our work in this paper on the traffic patterns is similar to the work in [15]. However, we expanded the set of patterns.

Performance of Myrinet and Giganet were evaluated in [8] using microbenchmarks, and the NAS benchmarks. The authors in [11] studied the user-level messaging layers of the InfiniBand, Myrinet, and Quadrics. User-level microbenchmarks were used in [5] to study the InfiniBand. Current interconnects including the SP switch, and Cray T3E were studied using both user-level and MPI-level microbenchmarks [3]. Sun Fire Link was evaluated at the RSM, MPI, and application level in [16]. LogP and its variants were proposed to gain insights into the different components of a communication step [6, 2, 9]. Sensitivity of applications to latency, and overhead was studied in [12].

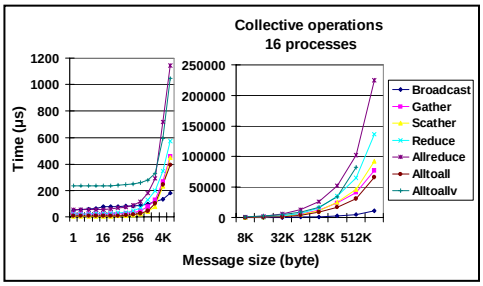

Figure 9. Collective communication performance.

7. Conclusions

In this paper, we studied the recently released two-port Myrinet network with its new GM layer. We have presented its performance at the GM level, and the MPI level using an extended set of microbenchmarks. Our performance results include detailed latency and bandwidth measurements. We discovered the sensitivity of the Myrinet network to the buffer reuse pattern, as well as the potential for computation/communication overlap. Parameters of the LogP model, different traffic patterns, and collective communications were also studied in this work. We found that the host overhead is very small. Meanwhile, the completion times for collectives demonstrate the need for their efficient implementation.

Acknowledgments

This work was supported by grants from the Natural Sciences and Engineering Research Council of Canada (NSERC), Canada Foundation for Innovation (CFI), and Ontario Innovation Trust (OIT).

References

[1] D. Addison, J. Beecroft, D. Hewson, M. McLaren and F. Petrini, "Quadrics QsNet II: A Network for Supercomputing Applications". In *Hot Chips 15*, August 2003.

[2] A. Alexandrov, M. Ionescu, K. E. Schauser, and C. Scheiman, "LogGP: Incorporating Long Messages into the LogP Model - One Step Closer Towards a Realistic Model for Parallel Computation", *7th Annual ACM Symposium on Parallel Algorithms and Architecture (SPAA'95)*, July 1995, pp. 95-105.

[3] C. Bell, D. Bonachea, Y. Cote, J. Duell, P. Hargrove, P. Husbands, C. Iancu, M. Welcome, and K. Yelik, "An Evaluation of Current High-Performance Networks", *17th International Parallel and Distributed Processing Symposium (IPDPS 2003)*, April 2003.

[4] N. J. Boden, D. Cohen, R. E. Kulawik, C. L. Seitz, J. N. Seizovic, and W-K Su, "Myrinet: A Gigabit-per-Second Local Area Network", *IEEE Micro*, Vol.15, No.1, February 1995, pp. 29-36.

[5] A. Cohen, "A Performance Analysis of 4X InfiniBand Data Transfer Operations", *17th International Parallel and Distributed Processing Symposium (IPDPS 2003)*, 2003.

[6] D. E. Culler, R. M. Karp, D. A. Patterson, A. Sahay, K. E. Schauser, E. Santos, R.Subramonian, and T. von Eiken, "LogP: Towards a Realistic Model of Parallel Computation", *4th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming*, 1993, pp. 1-12.

[7] 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.

[8] 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", *Supercomputing Conference (SC'00), 2000.*

[9] T. Kielmann, H. E. Bal, and K. Verstoep, " Fast Measurement of LogP Parameters for Message Passing Platforms", *4th Workshop on Runtime Systems for Parallel Programming (RTSPP), held in conjunction with IPDPS 2000*, 2000, pp. 1176-1183.

[10] J. Liu, B. Chandrasekaran, J. Wu, W. Jiang, S. Kini, W. Yu, D. Buntinas, P. Wyckoff, and D. K. Panda, "Performance comparison of MPI implementations over InfiniBand, Myrinet, and Quadrics", *Supercomputing Conference (SC'03)*, 2003.

[11] J. Liu, B. Chandrasekaran, Weikuan Yu, Jiesheng Wu, Darius Buntinas, Sushmitha Kini, Dhabaleswar K. Panda, and Pete Wyckoff, "Microbenchmark Performance Comparison of High-Speed Cluster Interconnects", *IEEE Micro*, 2004, Vol. 24, No.1, pp. 42-51.

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

[13] Mellanox Technologies, Available: http://www.mellanox.com.

[14] Myricom. Available: http://www.myrinet.com.

[15] F. Petrini, S. Coll, E. Frachtenberg, and A. Hoisie, "Performance Evaluation of the Quadrics Interconnection Network", *Journal of Cluster Computing*, 2003, pp. 125-142.

[16] Y. Qian, A. Afsahi, N. R. Fredrickson, and R. Zamani, "Performance Evaluation of the Sun Fire Link SMP Clusters", *18th International Symposium on High Performance Computing Systems and Applications, HPCS 2004*, May 2004, pp. 145-156.

[17] Quadrics. Available: http://www.quadrics.com.

[18] W. Vogels, D. Follett, J. Hsieh, D. Lifka, and D. Stern, "Tree-saturation control in the AC3 velocity cluster", *Hot Interconnect 8*, 2000.