Performance Evaluation of Fast Ethernet, Giganet and Myrinet on a Cluster

Marcelo Lobosco, Vítor Santos Costa, and Claudio L. de Amorim

Programa de Engenharia de Sistemas e Computação, COPPE, UFRJ
Centro de Tecnologia, Cidade Universitária, Rio de Janeiro, Brazil
{lobosco, vitor, amorim}@cos.ufrj.br

Abstract. This paper evaluates the performance of three popular technologies used to interconnect machines on clusters: Fast Ethernet, Myrinet and Giganet. To achieve this purpose, we used the NAS Parallel Benchmarks. Surprisingly, for the LU application, the performance of Fast Ethernet was better than Myrinet. We also evaluate the performance gains provided by VIA, a user lever communication protocol, when compared with TCP/IP, a traditional, stacked-based communication protocol. The impacts caused by the use of Remote DMA Write are also evaluated. The results show that Fast Ethernet, when combined with a high performance communication protocol, such as VIA, has a good cost-benefit ratio, and can be a good choice to connect machines on a small cluster environment where bandwidth is not crucial for applications.

1 Introduction

In the last few years, we have seen a continuous improvement in the performance of networks, both in reduced latency and in increased bandwidth. These improvements have motivated interest in the development of applications that can take advantage of parallelism in clusters of standard workstations.

Fast Ethernet, Giganet [1] and Myrinet [2] are three popular interconnection technologies used to build cluster systems. Fast Ethernet is a cheap LAN technology that can deliver 100Mbps bandwidth, while maintaining the original Ethernet's transmission protocol, CSMA/CD. TCP/IP is the most popular communication protocol for Fast Ethernet, although other protocols can be used, such as VIA [3]. TCP/IP is a robust protocol set developed to connect a number of different networks designed by different vendors into a network of networks. However, the reliability provided by TCP/IP has a price in communication overhead. To ensure reliable data transfer, protocol stack implementations like TCP/IP usually require data to be copied several times among layers and that communicating nodes exchange numerous protocol-related messages during the course of data transmission and reception. The number of protocol layers that are traversed, data copies, context switches and timers directly contributes to the software overhead. Also, the multiple copies of the data that must be maintained by the sender node and intermediate nodes until receipt of the data-packet is acknowledged contributes to reduce memory resources and further slows down transmission.

P.M.A. Sloot et al. (Eds.): ICCS 2002, LNCS 2329, pp. 296–305, 2002.

Giganet and Myrinet are more expensive technologies that provide low latency, high bandwidth, end-to-end communication between nodes in a cluster. Giganet provides both a switch and a host interface. The switch is based on a proprietary implementation of ATM. Giganet's host interface is based on a hardware implementation of VIA. The Virtual Interface Architecture (VIA) is a user-level memory-mapped communication architecture that aims at achieving low latency and high bandwidth communication within clusters of servers and workstations. The main idea is to remove the kernel from the critical path of communication. The operating system is called just to control and setup the communication. Data is transferred directly to the network by the sending process and is read directly from the network by the receiving process. Even though VIA allows applications to bypass the operating system for message passing, VIA works with the operating system to protect memory so that applications use only the memory allocated to them. VIA supports two types of data transfers: Send-Receive, that is similar to the traditional message-passing model, and Remote Direct Memory Access (RDMA), where the source and destination buffers are specified by the sender, and no receiver is required. Two RDMA operations, RDMA Write and RDMA Read, respectively write and read remote data.

Myrinet is the most popular high-speed interconnect used to build clusters. Myrinet also provides both a switch and a host interface. Myrinet packets may be of any length, and thus can encapsulate other types of packets, including IP packets, without an adaptation layer. Each packet is identified by type, so that Myrinet, like Fast-Ethernet, can carry packets of many types or protocols concurrently. Thus, Myrinet supports several software interfaces. The GM communication system is the most popular communication protocol for Myrinet. It provides reliable, ordered delivery between communication endpoints, called ports. This model is connectionless in that there is no need for client software to establish a connection with a remote port in order to communicate with it. GM also provides memory protected network access. Message order is preserved only for messages of the same priority, from the same sending port, and directed to the same receiving port. Messages with differing priority never block each other.

This paper studies the impacts of these three popular cluster interconnection technologies on application performance, since previous works pointed out that interconnection technology directly impacts the performance of parallel applications [4]. To achieve this purpose, we used the NAS Parallel Benchmark (NPB) [5] to measure the performance of a cluster when using each of the interconnection networks presented previously. The main contributions of our work are a) the comparative study of three popular interconnections technologies for clusters of workstations; b) the evaluation of the performance gains provided by VIA, a user lever communication protocol, when compared with TCP/IP, a traditional, stacked-based communication protocol; c) the evaluation of the impacts caused by the use of Remote DMA Write on VIA and d) an explanation for the poor performance of the LU benchmark on Myrinet. The results show that Fast Ethernet, when combined with a high performance communication protocol, such as VIA, has a good cost-benefit ratio, and can be a good choice when connecting machines on a small cluster environment where bandwidth is not crucial for applications. This paper is organized as follows. Section 2 presents the applications used in the study and their results. Section 3 concludes the work.

2 Performance Evaluation

Our experiments were performed on a cluster of 16 SMP PCs. Each PC contains two 650 MHZ Pentium III processors. For the results presented in this paper, we used just one processor on each node. Each processor has a 256 Kb L2 cache and each node has 512 Mb of main memory. All nodes run Linux 2.2.14-5.0. Table 1 shows a summary of the interconnect specifications used in the performance evaluation. To run VIA on Fast Ethernet, we use the NESRC's M-VIA version 1.0 [6], a software implementation of VIA for Linux.

Table 1. Summary of interconnect specifications

	Fast Ethernet	Giganet	Myrinet
Switch	Micronet SP624C	cLAN 5300	M2L-SW16
Network Card	Intel EtherExpress Pro 10/100	cLAN NIC	M2L-PCI64B
Link Speed	100Mbps	1.25Gbps	1.28Gbps
Topology	Single Switch	Thin Tree	Full-Crossbar
Protocol	TCP/IP and VIA	VIA	GM
Middleware	MPICH1.2.0 and MVICH1-a5	MVICH1-a5	MPICH1.2..8

Figure 1 shows the latency and bandwidth of TCP/IP and M-VIA on the Intel eepro100 card, VIA on Giganet and GM on Myrinet. The figures show that M-VIA's latency is 70% of the TCP/IP's. Giganet's and Myrinet's latency is an order of magnitude smaller. Giganet's latency is smaller until 28 bytes; after this Myrinet's latency is smaller. The bandwidth to send 31487 bytes is 10.5 MB/s on Fast Ethernet with TCP/IP, 11.2 MB/s on Fast Ethernet with M-VIA, 98.28 MB/s on Giganet and 108.44 MB/s on Myrinet.

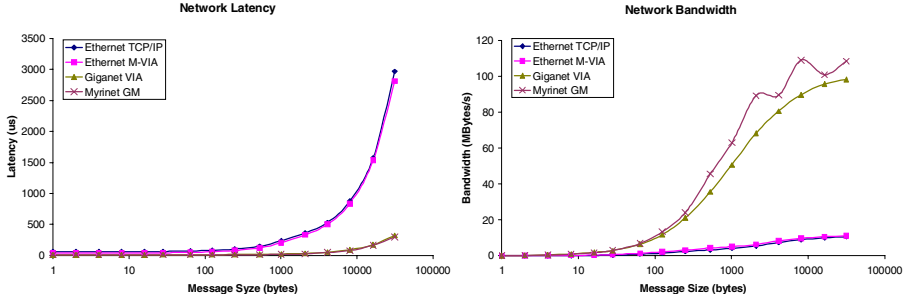

Fig. 1. Network Latency and Bandwidth

The NAS Parallel Benchmarks (NPB) are widely used to evaluate the performance of parallel machines [5]. The set consists of five kernels – EP, FT, MG, CG and IS – and three applications – BT, LU and SP – derived from computational fluid dynamics (CFD) codes. Each benchmark has five classes: S, W, A, B and C. Problems size grows from class S to class C. Our experiments used class B. We did not run FT and BT in our experiments. The sequential version of BT did not run. We could not compile FT in our experiments because it requires a Fortran 90 compiler (we used GNU g77 version 2.91.66 to compile the source codes). All NAS benchmarks use

MPI [7]. We used MPICH 1.2.0 in our experiments for TCP/IP, MPICH 1.2.8 for GM and MVICH 1-alpha 5 [8] for VIA. The implementation of MVICH is at the early stages, so it is neither completely stable nor optimized for performance. We expect MVICH results presented here to improve in future versions. We ran the applications with and without RDMA for the tests with VIA (both Fast Ethernet and Giganet). This option is enabled at compilation.

Table 2. Sequential times of applications

Program	Program Size	Seq. Time (s)	Std. Dev. (%)
IS	2^{25}, Iterations = 10	69.03	0.71
MG	256x256x256, Iterations = 20	342	0.52
CG	75,000, Iterations = 75	2,346	0.01
SP	102x102x102, Iterations = 400	6,783	0.46
EP	2^{31}, Iterations = 10	1,537	0.08
LU	102x102x102, Iterations = 250	7,553	0.30

The sequential execution times for the applications are presented in Table 2. Each application was run 5 times; the times presented are an average of these values. We also present the standard deviation of the times. All execution times are non-trivial, with LU and SP having the longest running-times. IS and MG have much shorter running-times, but still take more than a minute.

Table 3. Messages and data at 16 processors

Program	Messages	Transfers (MB)	Medium Size (KB/m)	Bw per CPU (MB/s)
IS	5,420	1,281.19	242.05	16.18
MG	42,776	667.25	15.97	1.74
CG	220,800	13,390.54	62.1	5.03
SP	153,600	14,504.17	96.69	1.82
EP	90	0.003	0.03	0.00
LU	1,212,060	3,525.70	2.97	0.45

Table 3 shows the total amount of data and messages sent by the applications, as well as medium message size (in kilobytes per message) and average bandwidth per processor, when running on 16 nodes. We can observe that the benchmarks have very different characteristics. EP sends the smallest number of messages, while LU sends the larger amount of messages. Although it takes the least time to run, IS sends the largest messages, hence requiring very high bandwidth, above the maximum provided by Fast Ethernet. SP is a long running-time application and also sends the largest volume of data, so it has moderate average bandwidth. The least bandwidth is used by EP, which only sends 90 messages in more than a thousand seconds. Table 4 presents a more detailed panorama of the message sizes sent by each application. SP and EP are opposites in that SP sends a lot of large message and EP few small messages. IS sends the same number of small and large messages. MG has a relative uniform distribution. Last, LU mostly sends medium-sized messages, and also some larger messages.

Table 5 shows the application's speedups for 16 processors. Speedup curves are presented in Figure 2. To understand the performance of the benchmarks, we used both the log and trace options available at the Multiprocessing Environment library.

Table 4. Message numbers by size

Size	IS	MG	CG	SP	EP	LU
$x < 10^1$	2,560	760	124,800	0	60	0
$10^1 \leq x < 10^2$	0	8,960	2,400	0	30	60
$10^2 \leq x < 10^3$	0	9,920	0	0	0	$3x10^5$
$10^3 \leq x < 10^4$	300	11,520	0	0	0	$9x10^5$
$10^4 \leq x < 10^5$	0	9,664	0	57,600	0	0
$x \geq 10^5$	2,560	1,952	93,600	96,000	0	12,000

Figure 3 presents the execution time breakdown of the benchmarks. We include only four communications calls (`MPI_Send`, `MPI_Recv`, `MPI_Isend` and `MPI_Irecv`) and two synchronizations calls (`MPI_Wait` and `MPI_Waitall`). All other MPI calls are constructed through the combination of these six primitives calls. We show breakdowns for the Myrinet and Giganet configurations. The breakdowns are quite similar, except for LU. Note that computation time dominates the breakdown. IS has very significant send and waitall times, and LU has very significant recv time. The dominance of computation time indicates we can expect good speedups, as it is indeed the case.

Table 5. Speedups – 16 processors

Application	MPICH		MVICH			
	Ethernet	Myrinet	Ethernet	Eth RDMA	Giganet	Gig RDMA
IS	1.43	11.85	NA	NA	NA	13.95
MG	10.95	14.28	12.36	NA	13.92	14.03
CG	7.70	14.11	9.72	NA	13.36	13.59
SP	NA	13.57	11.93	NA	13.4	13.66
EP	15.92	16.01	16.00	16.00	16.02	16.02
LU	12.76	7.22	15.16	NA	15.48	15.66

IS. Integer Sort (IS) kernel uses bucket sort to rank an unsorted sequence of keys. IS sends a total of 5,420 messages; 2,560 messages are smaller than 10 bytes and 2,560 are bigger than 10^5 bytes. IS requires a total of 16.18 MB/s of bandwidth per CPU. IS running on Myrinet spends 55% of the time on communication, while the version running on Giganet with RDMA spends 48% of time on communication. So, this is a communication bound application and just the bandwidth required per CPU by IS explains the poor performance of Fast Ethernet. We found that the difference between Giganet and Myrinet stems from the time spent in the recv and waitall primitives. For 16 nodes, Giganet spends 0.78s and 1.55s, respectively, in recv and waitall, while Myrinet spends 0.90s and 2.26s (Figure 3a). This seems to suggest better performance from the Giganet switch. VIA is effective in improving performance of Fast Ethernet. For 8 nodes, the speedup of the M-VIA version is 3.66, against 1.02 of the TCP/IP version (figure 2a). We believe this result to be quite good, considering the average bandwidth required by the application. Unfortunately, we could not make M-VIA run with 16 nodes, due to an excessive number of messages. M-VIA with RDMA version only runs with 2 nodes, so we cannot evaluate its effectiveness in improving performance.

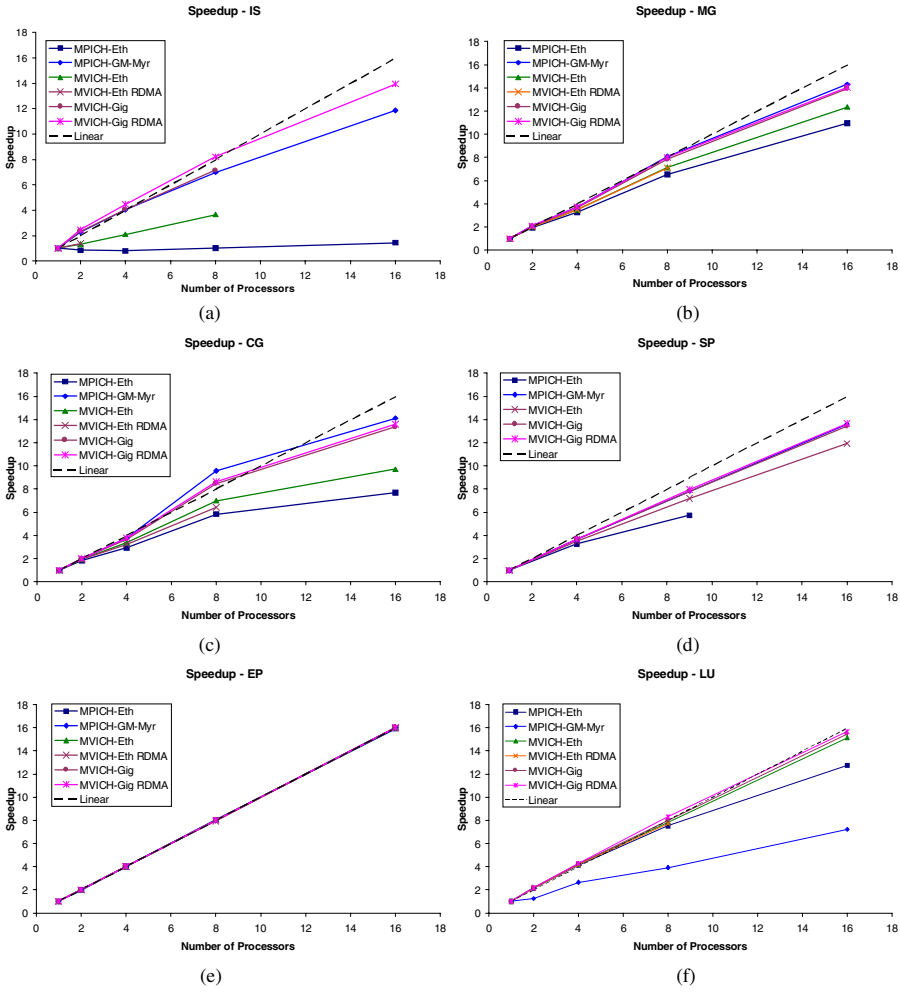

Fig. 2. Speedups: (a) IS, (b) MG, (c) CG, (d) SP, (e) EP, (f) LU

MG. MG uses a multigrid method to compute the solution of the three-dimensional scalar Poisson equation. MG sends a total of 42,776 messages. Despite of sending more messages than IS, the medium message size of MG is smaller and the benchmark runs for longer. This is reflected in the average bandwidth required of only 1.74 MB/s, smaller than the bandwidth required by IS. As shown in Figure 3b, MG is a computation bound application, spending less than 10% of the time doing communication on both Myrinet and Giganet. Fast Ethernet presented a good speedup on 16 nodes: 10.95 for TCP/IP and 12.36 for VIA. The speedups of Myrinet and Giganet are quite similar: Myrinet achieved a speedup of 14.28, against 14.03 of Giganet with the RDMA support. Without RDMA, the speedup of Giganet is 13.93

(Figure 2b). So RDMA support does not offer, for this application, a significant improvement in performance. Indeed, for 8 nodes, the performance of the version with RDMA support on Fast Ethernet was slightly worst: 7.10 against 7.15 for the version without RDMA. For 16 nodes on Fast Ethernet, the version with RDMA support has broken. The reason why Myrinet has a slightly better performance than Giganet is the time spent in the send primitive: Myrinet spends 1.4s, while Giganet spends 1.55s (Figure 3b). We believe MG may benefit from Myrinet's larger bandwidth.

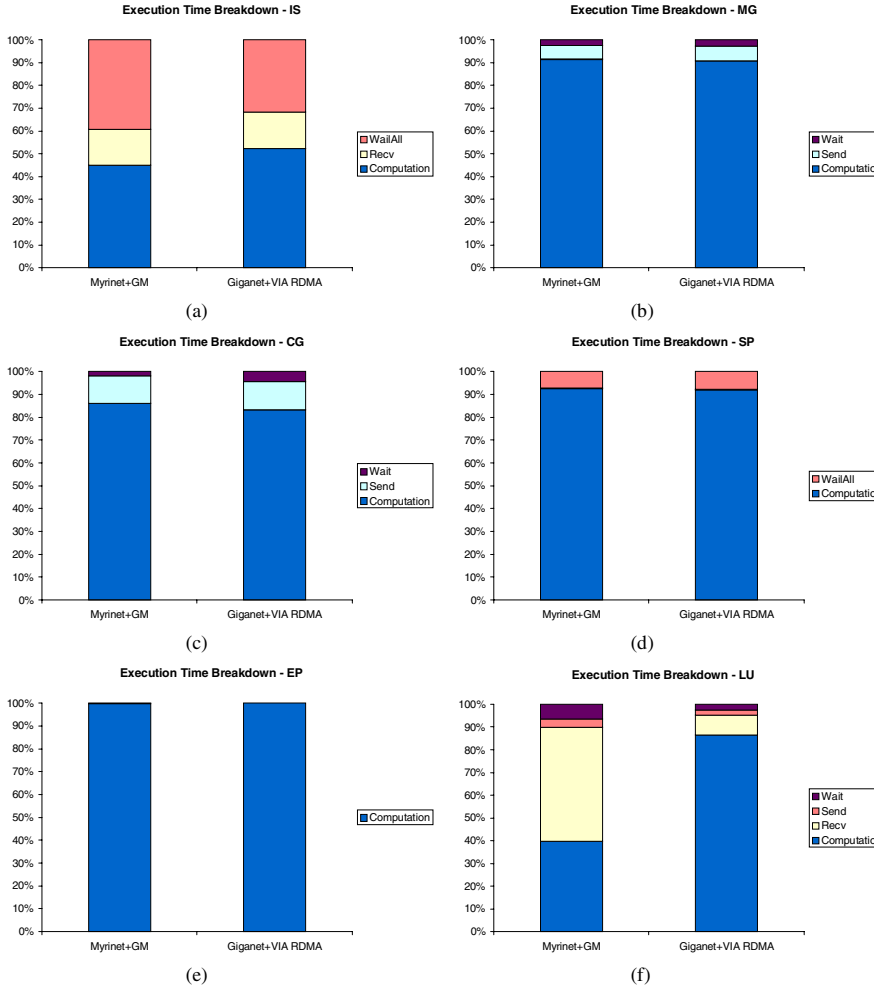

Fig. 3. Execution time breakdown - 16 nodes: (a) IS, (b) MG, (c) CG, (d) SP, (e) EP, (f) LU

CG. In the CG kernel, a conjugate gradient method is used to find an estimate of the largest eigenvalue of a large, symmetric positive definite sparse matrix with a random

pattern of nonzeros. CG sends a total of 220,800 messages; 124,800 of which are smaller than 9 bytes and 93,600 bigger than 10^5. In spite of the large number of messages sent, CG is a computation bound application: 16% of the time is spend on communication on Giganet and 14% on Myrinet. The bandwidth required by this benchmark is 5.03 MB/s. Myrinet achieves the better speedup: 14.11 on 16 nodes. Giganet achieves a speedup of 13.59 with the RDMA support and 13.36 without RDMA. TCP/IP over Fast Ethernet achieved a speedup of 7.7. Substituting TCP/IP by M-VIA makes the speedup grow up to a very reasonable 9.72 (Figure 2c). Again, M-VIA with RDMA was broken for 16 nodes. For 8 nodes, the version with RDMA has worst performance (6.42) than the version without RDMA (7.00). This time, the time spent in the wait primitive by Myrinet, 3.28s, is smaller than the time Giganet spends on it, 7.5s. Giganet also spends more time in the send primitive (21.05s) than Myrinet (19.7s), which contributed to the worst performance of Giganet on this benchmark (Figure 3c). Again, we believe Myrinet may benefit from larger bandwidth.

SP. SP solves 3 uncoupled systems of non-diagonally dominant, scalar, pentadiagonal equations using the multi-partition algorithm. In the multi-partition algorithm, each processor is responsible for several disjoint sub-blocks of points of the grid, which are called cells. The information from a cell is not sent to the next processor until all linear equations of the cell have been solved, which keeps the granularity of communications large (57,600 messages sent by SP are between 10^4 and 10^5-1 bytes, and 96,000 are bigger than 10^5 bytes). As CG and MG, SP is also a computation bound application, spending less than 8% on communication. The bandwidth required by SP is 1.82 MB/s. SP requires a square number of processors. Myrinet and Giganet have similar performance. Myrinet achieved a speedup of 13.57 on 16 nodes, and Giganet achieved a speedup of 13.66 when running with RDMA support and 13.4 without RDMA. We were surprised to find that TCP/IP over Fast Ethernet was broken for 16 nodes. M-VIA over Fast Ethernet achieved a speedup of 11.93, reasonably close to the faster networks (Figure 2d).

EP. In the Embarrassingly Parallel (EP) kernel, each processor independently generates pseudorandom numbers in batches and uses these to compute pairs of normally distributed numbers. This kernel requires virtually no inter-process communication; the communication is just needed when the tallies of all processors are combined at the end of computation. All systems achieved linear speedups in this application (only TCP/IP was slightly slower – see Figure 2e).

LU. The Lower-Upper (LU) diagonal kernel uses a symmetric successive over-relaxation (SSOR) procedure to solve a regular sparse, block (5x5) lower and upper triangular system of equations in 3 dimensions. The SSOR procedure proceeds on diagonals, which progressively sweep from one corner on the third dimension to the opposite corner of the same dimension. Communication of partition boundary data occurs after completion of computation on all diagonals that contact an adjacent partition. This results in a very large number ($1.2x10^6$) of very small messages (about 2 Kb). The bandwidth required by LU is a little: 0.45 MB/s. Surprisingly, the performance of Myrinet is terrible in this benchmark. It achieves a speedup of 7.22 on 16 nodes, while TCP/IP over Fast Ethernet achieves a speedup of 12.76. M-VIA is

effective in improving the performance of LU on Fast Ethernet, achieving a speedup of 15.16. Giganet has the better performance for LU: 15.66 for the version with RDMA, and 15.48 for the version without RDMA (Figure 2f). While Giganet spends 13.5% of the time sending messages, Myrinet spends almost 40%. The send, recv and wait primitives are responsible for the poor performance of LU on Myrinet. While Myrinet spends 522s, 39s and 67s on send, recv and wait, respectively, Giganet spends 41s, 11s and 11s (Figure 3f). Previous work [9] also pointed this poor performance of LU on Myrinet. They also used MPICH-GM on Myrinet, but used MPI/Pro on Giganet. The work attributed the poor performance of LU on Myrinet to the pooling-based approach adopted by MPICH-GM (MPI/Pro adopts a interrupt-based approach). But MVICH also adopts the pooling-based approach, and its performance is good, which indicates that the theory presented in [9] is not correct. Instead, we suggest that the problem may stem from the way GM allocates memory for small blocks. GM has a cache for small blocks. We believe that cache management for the very many small blocks created by the application may be significantly hurting performance.

Analysis. The results show that M-VIA was effective in improving the performance of all benchmarks on Fast Ethernet, when compared with the version that uses TCP/IP. The performance on Ethernet with VIA is 3.6 times better than the version with TCP/IP for IS; 1.12 times better for MG; 1.26 times better for CG; 1.25 times better for SP; and 1.18 times better for LU. The biggest difference between VIA and TCP/IP appeared in the application that has the biggest bandwidth requirement, IS, which suggests that the gains provided by VIA are related to the bandwidth required by the application: the bigger the bandwidth required, the bigger the performance difference between VIA and TCP/IP. The RDMA support provided by VIA on Giganet was effective in improving the performance of IS: it runs 1.14 times faster than the version without RDMA. For all other applications, the RDMA support contributes for a small improvement in performance (less than 2%). On Ethernet, the version with RDMA support was quite unstable due to sequence mismatch messages. This occurs because the NIC has dropped packets under heavy load: the interrupt service routine on the receiving node cannot process the packets as fast as the NIC receives them, causing the NIC to run out of buffer space. Because M-VIA supports only the unreliable delivery mode of VIA, these events do not cause a disconnect and the MVICH library gets different data than what it is expecting. The behavior of MVICH in this case is to abort. It is also interesting to note that on Ethernet, the benchmarks with RDMA support performed 8% worst on CG. In IS, for 2 nodes, the performance of the version with RDMA was 6% better than the version without RDMA. Recall that for this benchmark, the configurations with more than 4 nodes have broken. In all other benchmarks, the performance was equal. The results of both Giganet and Ethernet indicate that the RDMA support is only effective when the message size is huge. As could be expected, Ethernet, even with VIA support, performed worst than the best execution time, either Giganet's or Myrinet's, for all benchmarks. The only exception was EP, where Ethernet performed as well as Myrinet and Giganet. This happened because EP's almost does not require communication. For IS, Ethernet had its worst performance (124% slower than Giganet) due to the high bandwidth required by this application. The better performance, after EP, is LU: only 3% slower than Giganet. Not surprisingly, LU is

the application with the smaller bandwidth requirement after EP. Other performances are 15% worst for MG and SP and 45% for CG. The performance on LU and MG suggests that Ethernet is a good choice to connect machines on a small cluster environment where bandwidth is not crucial for applications, since it has a good cost-benefit ratio. Ethernet is 10 times cheaper than Giganet and Myrinet, and its performance is 2 times slower for the worst application, IS. If performance is most important, a good rule of thumb would be to choose Fast Ethernet with TCP/IP if your application requires very small bandwidth, and to use a faster protocol such as M-VIA on the same hardware if your application requires up to 5 MB/s. Giganet performed better than Myrinet for IS (18%) and LU (116%); had a similar performance for EP and SP; and performed slightly worst for MG (2%) and CG (4%). These results show the influence of the MPI implementation over performance. The raw communication numbers presented in Figure 1 could suggest that Myrinet would have the best performance, indicating that the implementation of MPICH-GM does not take full advantage of the performance of Myrinet.

3 Summary and Conclusions

This paper evaluated the impact of three popular cluster interconnection technologies, namely Fast Ethernet, Giganet and Myrinet, over the performance of the NAS Parallel Benchmarks (NPB). The results show that Fast Ethernet, when combined with a high performance communication protocol, such as VIA, has a good cost-benefit ratio, and can be a good choice to connect machines on a small cluster environment where bandwidth is not crucial for applications. We also evaluated the performance gains provided by VIA, when compared with TCP/IP, and found that VIA is quite effective in improving the performance of applications on Fast Ethernet. The RDMA support provided by VIA was evaluated, and we conclude that it is only effective when the messages exchanged by the applications are huge. Last, our results showed Giganet performing better than Myrinet on the NPB. We found the main difference in LU, where Myrinet performance is poor due to the MPI implementation for this interconnect technology, and not due to the pooling-based approach adopted by the MPI implementation, as a previous work has pointed out.

References

1. Giganet, Inc. http://www.emulex.com/
2. Myricom, Inc. http://www.myri.com/
3. Compaq, Intel, and Microsoft. VIA Specification 1.0. Available at http://www.viarch.org.
4. Wong, F, et alli. Architectural Requirements and Scalability of the NAS Parallel Benchmarks. SuperComputing'99, Nov 1999.
5. Bailey, D, et alli. The NAS Parallel Benchmarks. Tech. Report 103863, NASA, July 1993.
6. NERSC. M-VIA. http://www.nersc.gov/research/FTG/via/.
7. Message Passing Interface Forum. http://www.mpi-forum.org/.
8. NERSC. MVICH. http://www.nersc.gov/research/FTG/mvich
9. Hsieh, J, et alli. Architectural and Performance Evaluation of GigaNet and Myrinet Interconnects on Clusters of Small-Scale SMP Servers. SuperComputing'00, Nov 2000.