MICROBENCHMARK PERFORMANCE COMPARISON OF HIGH-SPEED CLUSTER INTERCONNECTS

HIGH-SPEED CLUSTER INTERCONNECTS MYRINET, QUADRICS, AND INFINIBAND ACHIEVE LOW LATENCY AND HIGH BANDWIDTH WITH LOW HOST OVERHEAD. HOWEVER, THEY SHOW QUITE DIFFERENT PERFORMANCE BEHAVIORS WHEN HANDLING COMMUNICATION BUFFER REUSE PATTERNS.

Jiuxing Liu
Balasubramanian Chandrasekaran
Weikuan Yu
Jiesheng Wu
Darius Buntinas
Sushmitha Kini
Dhabaleswar K. Panda
The Ohio State University

Pete Wyckoff
Ohio Supercomputer Center

•••••• Today's distributed and high-performance applications require high computational power and high communication performance. Recently, the computational power of commodity PCs has doubled about every 18 months. At the same time, network interconnects that provide very low latency and very high bandwidth are also emerging. This is a promising trend in building high-performance computing environments by clustering—combining the computational power of commodity PCs with the communication performance of high-speed network interconnects.

There are several network interconnects that provide low latency (less than 10 μs) and high bandwidth (several gigabytes per second). Two of the leading products are Myrinet[1] and Quadrics.[2] Recently, InfiniBand[3] has entered the high-performance computing market.

All three interconnects share similarities. For one, they provide user-level access to network interface cards for performing communication; they also support access to remote processes' memory address spaces. However, they also differ in many ways. So the question arises: *How can we conduct a meaningful performance comparison among all three interconnects?*

Traditionally, researchers have used simple microbenchmarks, such as latency and bandwidth tests, to characterize a network interconnect's communication performance. Later, they proposed more sophisticated models such as LogP.[4] However, these tests and models focus on general parallel computing systems and do not address many features present in these emerging commercial interconnects.

Another way to evaluate different network interconnects is to use real-world applications. However, real applications usually run on top of a middleware layer such as the Message Passing Interface (MPI). Therefore, the application-level performance reflects not only the capability of the network interconnects, but also the quality of the MPI implementations and the design choices of the MPI implementers. Thus, to provide more insight into the communication capabilities offered by each interconnect, it is desirable to conduct tests at a lower level.

We propose an approach to evaluate and compare the performance of the three high-speed interconnects: InfiniBand, Myrinet, and Quadrics. We have designed a set of micro-benchmarks to characterize different aspects of the interconnects. Our microbenchmarks include not only the traditional performance measurements, but also those more relevant to modern networks that provide user-level access. Our benchmarks also concentrate on the remote-memory-access capabilities provided by each interconnect. We have conducted tests in an 8-node cluster system containing all three network interconnects. From the experiments, we found that although these interconnects have similar programming interfaces, their performance behavior differs significantly in terms of

- handling completion notification,
- buffer reuse patterns, and
- the level of unbalanced communication.

Interestingly, we cannot evaluate these qualities using the default latency or bandwidth tests.

Motivation

As shown in Figure 1, today's cluster systems usually consist of multiple protocol layers. Network adapters and switches connect nodes in a cluster. At each node, a communication protocol layer provides functionalities for internode communication. Upper layers—such as MPI, PVFS, and cluster-based databases or Web servers[5]—take advantage of this communication layer –to support user applications.

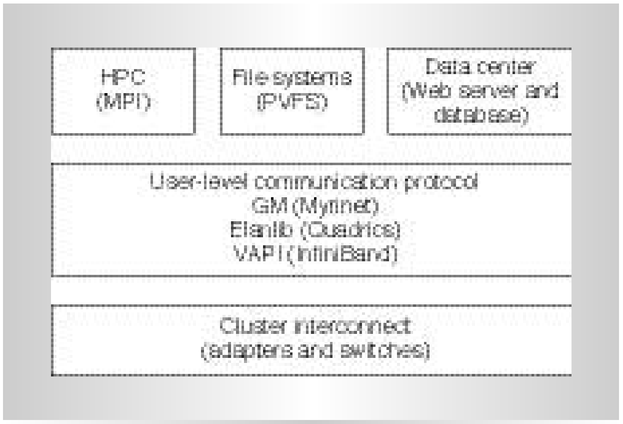
HPC (MPI)
File systems (PVFS)
Data center (Web server and database)
User-level communication protocol
GM (Myrinet)
Elanlib (Quadrics)
VAPI (InfiniBand)
Cluster interconnect (adapters and switches)

Figure 1. Typical protocol layers in clusters.

One of the key tasks for a cluster designer is to choose the interconnect that best meets the applications' communication requirements. Because applications can have very different communication characteristics, it is very important to have a thorough knowledge of different interconnects and their performance so that you know how they perform under these communication patterns. This knowledge can also help upper-layer developers better optimize their software.

Microbenchmarks can be invaluable in gaining insight into the performance of different interconnects. Two of the most commonly used types of microbenchmarks for interconnects are latency and bandwidth. However, using only latency and bandwidth to characterize performance has the following potential problems:

- These tests usually represent an ideal scenario, which might differ from the patterns in real applications. For example, these tests use a single buffer at both the sender and receiver sides. In contrast, real applications might use multiple communication buffers on each side. Therefore, the ideal scenario's results could be misleading because it only reflects the low latency of access to a single buffer by either sender or receiver.
- Modern interconnects such as Myrinet, Quadrics, and InfiniBand have the standard send/receive operations and sophisticated network adapters to offload communication tasks from host CPUs. They also incorporate new features such as user-level access to network interfaces and remote direct memory access (RDMA) operations. However, simple latency or bandwidth tests fail to explicitly take these advanced features into consideration.

To provide more realistic performance measures for these modern interconnects, we propose a new set of microbenchmarks. This microbenchmark suite accounts for not only the different communication characteristics of the upper layers, but also these advanced interconnect features. Our microbenchmarks explore many aspects of communication rather than measure performance in a few ideal cases. Therefore, these microbenchmarks can help upper-layer software designers know about the communication layer's strengths and limitations. They also help communication layer designers optimize the implementation for a given upper layer.

InfiniBand and VAPI

The InfiniBand architecture defines a network for interconnecting processing and I/O nodes. In an InfiniBand network, processing nodes connect to the communication fabric via host channel adapters (HCAs). I/O nodes use target channel adapters (TCAs).

Our InfiniBand platform consists of InfiniHost 4X HCAs and an InfiniScale switch from Mellanox. InfiniScale is a full-wire-speed switch with eight 10-Gbps ports. The InfiniHost 4X HCA connects to the host through a 133-MHz, 64-bit PCI-X bus, a higher-speed version of the Peripheral Component Interconnect. This allows for a bandwidth of up to 10 Gbps over a link. However, the bandwidth for user payloads is 8 Gbps in each direction (that is, 8 Gbps plus 8 Gbps) because of the 8b/10b encoding used in InfiniBand. (An IBM-patented method, 8b/10b encoding converts 8-bit data bytes into 10-bit transmission characters.)

The Verb-Based API (VAPI) is the software interface for InfiniHost HCAs. The interface is based on the InfiniBand verbs layer, which is an abstract description of functionalities provided by all HCAs. VAPI supports send, receive, and RDMA operations. Currently, vendors have implemented reliable connection (RC) and unreliable datagram (UD) services on InfiniBand HCAs. In this article, we focus on the RC service. In InfiniBand, application memory buffers must be registered with the HCA to be used in communication operations. Completion queues (CQs) report the completion of communication requests.

Myrinet and GM

Myricom developed Myrinet (http://www.myri.com/myrinet/overview/index.html) based on communication and switching technology originally designed for massively parallel processors (MPPs). Myrinet has a user-programmable processor in the network interface card that provides great flexibility in designing communication software.

Our Myrinet network consists of M3F-PCIXD-2 network interface cards connected by a Myrinet 2000 switch. The link bandwidth of the Myrinet network is 2 Gbps for each direction (2 Gbps plus 2 Gbps). The Myrinet 2000 switch is an 8-port crossbar switch. The network interface card uses a 133-MHz, 64-bit PCI-X interface. It has a programmable Lanai-XP processor running at 225 MHz. The Lanai processor on the network interface card can access host memory via the PCI-X bus through a host DMA controller.

GM (http://www.myri.com/scs/index.html) 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 a connectionless communication model to the upper layer and supports send/receive operations. It also has RDMA operations that can directly write or read data to a remote node's address space. Similar to VAPI, GM requires users to register their communication buffers. We have used GM version 2.0.1 for our studies.

Quadrics and Elanlib

Quadrics networks consist of Elan3 network interface cards and Elite switches. The Elan network interface cards connect to hosts via a 66-MHz, 64-bit PCI bus. Elan3 has 64 Mbytes of on-board SDRAM and a memory management unit. 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; it has a transmission bandwidth of 400 Mbytes/s in each link direction.

Elanlib is a communication library that supports protected, user-level access to Elan network interfaces. It provides a global virtual address space by integrating the address spaces of individual nodes. One node can use DMA to access a remote node's memory space. Elanlib provides a general-purpose synchronization mechanism based on events stored in Elan memory, so it can report the completion of RDMA operations through events. Unlike VAPI and GM, the Quadrics network does not require users to register communication buffers. Elan network interface cards have an on-board memory management unit. The system software is responsible for synchronizing the memory management unit table and performing address translation.

Microbenchmarks and performance

To provide more insight into the communication behavior of the three interconnects, we designed a set of microbenchmarks and performance parameters to reveal different aspects of the interconnects' communication performance. This set includes benchmarks and performance parameters traditionally used to characterize interconnect communication:

- unidirectional latency,
- unidirectional bandwidth, and
- host overhead for communication.

We also designed tests that are more representative of application communication patterns and better related to the user-level communication model used by these interconnects; they include

- bidirectional latency and bandwidth;
- cost of checking for the completion of communication operations;
- impact of buffer reuse; and
- tests for hot –spot, which are used to measure communication performance under unbalanced load

Our experimental test bed consists of 8 SuperMicro Super P4DL6 nodes with ServerWorks GC chipsets and dual Intel Xeon 2.40-GHz processors. InfiniBand, Myrinet, and Quadrics connected the machines. The InfiniHost HCA adapters and Myrinet network interface cards work under the PCI-X 64-bit 133-MHz interfaces. The Quadrics cards use 64-bit 66-MHz PCI slots. We used the Linux RedHat 7.2 operating system with 2.4 kernels.

Latency and bandwidth

Researchers have frequently used end-to-end latency to characterize interconnect performance. All interconnects we study here support access to the memory space of remote nodes via RDMA. We therefore measured the latency to finish a remote write operation. InfiniBand plus VAPI and Myrinet plus GM

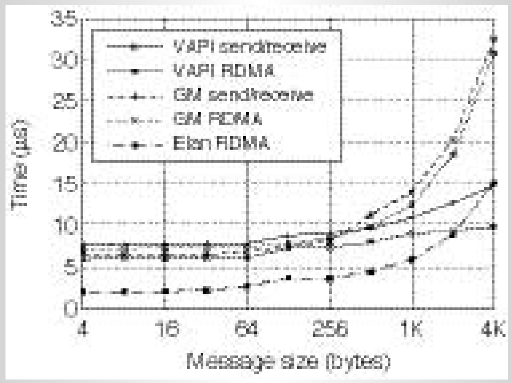

Figure 2. Latency of messaging software running over commercial networks.

also support send/receive operations. Figure 2 shows the latency results for both send/receive, and RDMA write. For small messages, Elan has the best latency, 2.0 µs. VAPI RDMA latency is around 6.0 µs and send/receive latency is around 7.8 µs. The higher latency for send/receive in VAPI is mainly due to the more complex processing at the receiver side. GM has a small message latency of about 6.5 µs for send/receive. Unlike VAPI, GM's RDMA has a slightly higher latency of 7.3 µs. As message size increases, VAPI begins to outperform both GM and Elanlib. For example, when the message size is larger than 2 Kbytes, VAPI RDMA performs best.

We used the bandwidth test to determine the maximum sustained data rate achievable at the network level. In this test, a sender keeps sending back-to-back messages to the receiver until it reaches predefined queue size *Q*. Then it waits for *Q*/2 messages to finish and sends out another *Q*/2 messages. In this way, the sender ensures that there are at least *Q*/2 and at most *Q* outstanding messages. Other research used a similar method.[10] Generally speaking, larger *Q* values lead to better bandwidth results.

Figure 3a shows the bandwidth results with a large queue size of 100. The peak bandwidth for VAPI, GM, and Elan is around 831, 236, and 314 Mbyte/s. For large messages, VAPI performs significantly better than both GM and Elan. We can also see that VAPI RDMA performs better than send/receive, while GM

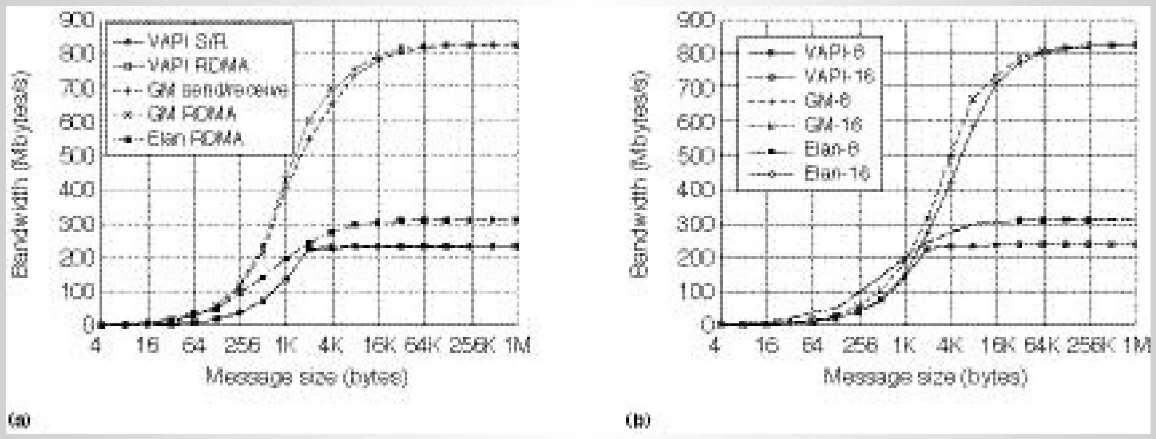

Figure 3. Unidirectional bandwidth.

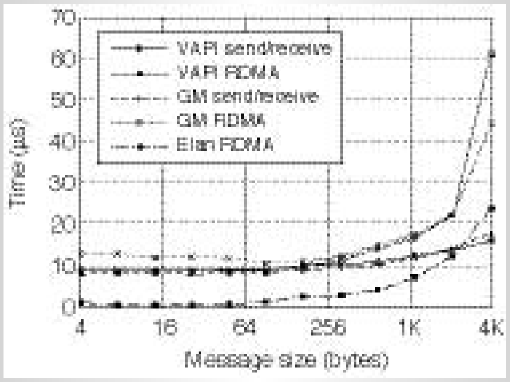

Figure 4. Bidirectional latency.

shows similar performance for both.

VAPI peak bandwidth is limited mainly by the PCI-X chipset. GM and Elan peak bandwidths are limited by their respective link bandwidths. Figure 3b shows the bandwidth for different queue sizes (eight and 16). We can see that VAPI is more sensitive to the value of queue size *Q*. However, even with a relatively small queue size (eight), all three interconnects can achieve their respective peak bandwidths.

Bidirectional latency and bandwidth

Compared with unidirectional latency and bandwidth tests, bidirectional tests put more stress on the PCI bus, the network interface cards, and the switches. They therefore might be more helpful in understanding communication bottlenecks. We carry out bidirectional tests in a way similar to that of unidirectional tests. The difference is that both sides send data simultaneously. Figure 4 shows that the bidirectional latency performance for all interconnects is worse than unidirectional performance. Elan small-message bidirectional latency is only slightly worse (2.1 µs versus 2.0 µs). However, performance degradations for VAPI and GM are more significant. For example, VAPI RDMA latency for small messages increases from 6.0 µs to 9.7 µs, and GM send/receive latency increases from 6.5 µs to 10.5 µs.

Figure 5 shows results for bidirectional bandwidth. Although InfiniBand allows for a maximum of 16 Gbps of bandwidth, the PCI-X bus becomes the bottleneck and limits the bandwidth to around 901 Mbytes/s. We can also see that VAPI RDMA operations perform significantly better than send/receive, except for very large messages. Although Elan has better unidirectional bandwidth than GM, its peak bidirectional bandwidth is only around 319 Mbytes/s. This is much less than Elan's link bandwidth or the PCI bus' peak bandwidth, indicating that the performance bottleneck lies in the Elan software or the network interface cards. On the other hand, GM performs very well in the bidirectional bandwidth test, basically doubling its unidirectional bandwidth to achieve 471 Mbytes/s. GM's link bandwidth limits its performance.

Host communication overhead

We define host communication overhead as the time the CPU spends on communication tasks. The more time it spends on communication, the less time it has for computation. This measure can therefore indicate the messaging layer's ability to overlap communication and computation. We further characterize the host communication overhead in terms of both latency and bandwidth. In the latency test, we obtain the overhead by directly measuring the CPU time spent on communication. In the bandwidth test, we insert a computation loop into the program. Increasing the time of this computation loop will eventually produce a drop in the bandwidth. The point beyond which the bandwidth drops signifies the percentage of CPU cycles that can be allocated for computation without delaying communication.

Figure 6 presents the host overhead in the latency test. VAPI has the highest overhead, around 2.0 μs. Elan overhead is around 0.7 μs. GM has the least overhead, around 0.5 μs. For all three interconnects, host communication overhead does not increase with message size, indicating that all these interconnects can offload most communication work to the network interface cards instead of increasing host overhead.

Figure 7 shows the impact of computation time on bandwidth, as indicated by tests using 256-Kbyte messages. These results show that all three interconnects can overlap communication and computation quite well—their bandwidths drop only after the CPU devotes 99 percent or more of its time to computation.

Cost of checking for communication completion

Because all three interconnects support RDMA, one way to detect the arrival of messages at the receiver side is to poll the destination buffer about its memory content. This approach minimizes receiver overhead, but is hardware dependent because it relies on the order in which the DMA controller writes to host memory, an order not specified in the Infiniband standard. So in theory, this method might not work on future Infiniband-compliant hardware that does not provide this feature.

Beside the preceding simple method, other methods are possible, but must account for the different mechanisms used by the three commercial networks to report the completion of remote memory operations. For example, VAPI uses completion queues (CQs), while

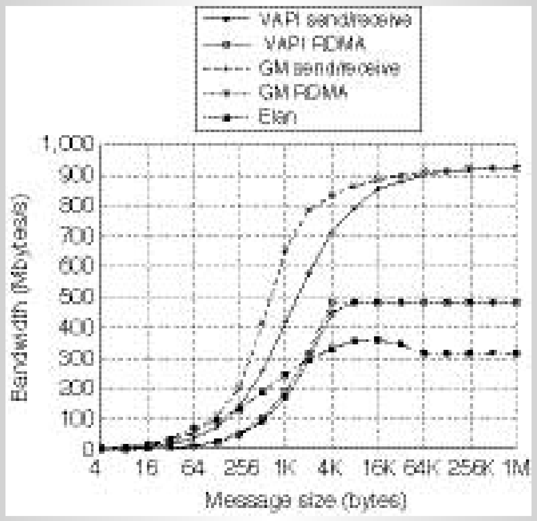

Figure 5. Bi-directional bandwidth.

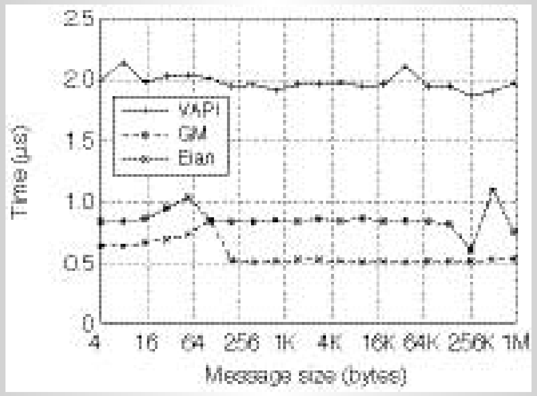

Figure 6. Host overhead in latency test.

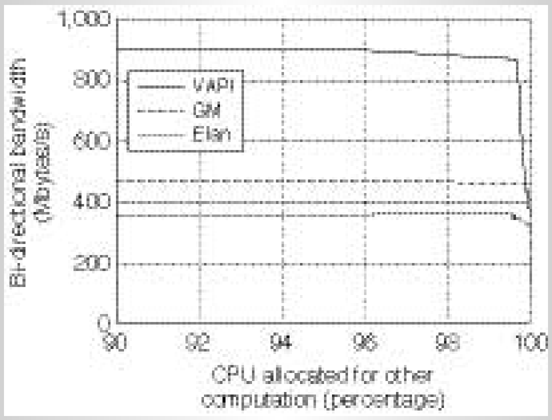

Figure 7. CPU utilization in bandwidth test.

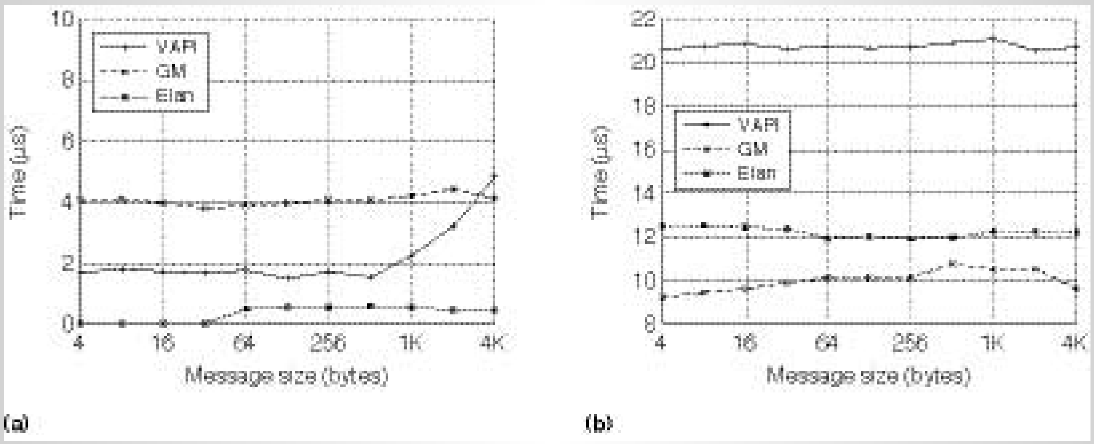

Figure 8. Overhead of checking for completion

GM and Elanlib rely on event abstractions. Figure 8a shows the increase in latency when using these mechanisms for remote memory access on the receiver side. In these tests, the host CPU is actively polling for the completion. We see that Elan has a very efficient notification mechanism, which adds only 0.4 μs of overhead for large messages. For messages less than 64 bytes, there is no extra overhead. VAPI has an overhead of around 1.8 μs. GM does not have a mechanism to notify the receiver of message arrival for RDMA operations. Therefore, we simulated the notification by using a separate send operation. This adds around 3 to 5 μs of overhead. Instead of busy polling, which constantly uses CPU to check completions, the upper layer can use blocking to wait for completions. In this case, the process that checks for incoming messages goes to sleep and is awaken by an interrupt when the message arrives. Figure 8b shows that the overheads for GM and Elan using interrupts are about 11 μs and 13 μs. VAPI has the highest overhead, more than 20 μs, indicating that VAPI has a higher interrupt processing overhead.

Impact of buffer reuse

In most microbenchmarks designed to test communication performance, the sender and receiver sides each use only one buffer. The sender or receiver keeps reusing the buffer until the test finishes. Real applications, however, usually use many buffers for communication. The buffer reuse pattern can have a significant impact on the performance of interconnects that support user-level access to network interfaces, such as those we present here. This impact arises from the address translation mechanisms used in these interconnects, an affect that test using only one buffer cannot characterize. Therefore, we designed a set of tests that have different buffer reuse patterns to study the impact of address translation on communication performance.

To capture the cost of address translation at the network interface, we designed two patterns for buffer reuse and changed the tests accordingly. The first pattern uses N buffers of the same size in FIFO order for multiple iterations. Increasing N might eventually lead to a performance drop. Basically, for each interconnect, this test measures how large the *communication working set* can be while still maintaining the best communication performance.

Figure 9 shows the bandwidth results with 512-Kbyte messages. We can see that with up to 25 buffers, GM and Elan show no performance degradation. However, VAPI performance drops at more than 10 buffers. This indicates that GM and Elan can handle several buffers at the same time without degrading communication performance.

The second pattern, based on percentage, is slightly more complicated. In this pattern, the test consists of N iterations, and we define buffer reuse percentage R. For the N test iterations, $N \times R$ iterations will use the same buffer; all other iterations will use completely different buffers. By changing buffer reuse percentage R, we see how buffer reuse patterns affect communication performance.

Figure 10a and Figure 10b show the results with the percentage pattern for both latency and bandwidth tests. For the latency test, we see that Elan is very sensitive to buffer reuse percentage; its performance drops significantly when the buffer reuse percentage decreases. VAPI and GM latencies only increase slightly when the reuse percentage decreases.

Elan performance also suffers from a low buffer reuse percentage in the bandwidth test. VAPI shows similar behavior, although its drop is not as significant as Elan. GM bandwidth performance is insensitive to the buffer reuse rate. Overall, we can see that GM handles communication patterns with low buffer reuse percentages more gracefully than both VAPI and Elan.

Hot-spot tests

Our hot-spot tests measure the ability of network interconnects (including network adapters) to handle unbalanced communication patterns, similar to tests conducted by other researchers. We used two sets of hot-spot tests; in send tests, a master node keeps sending messages to several slave nodes. In these hot-spot tests, the master node first sends a set of small messages to slave nodes to synchronize them and then receives messages from all the slave nodes. Our tests repeat this procedure for multiple iterations, and we report the latency for a single iteration. We conduct these tests for various numbers of slave nodes.

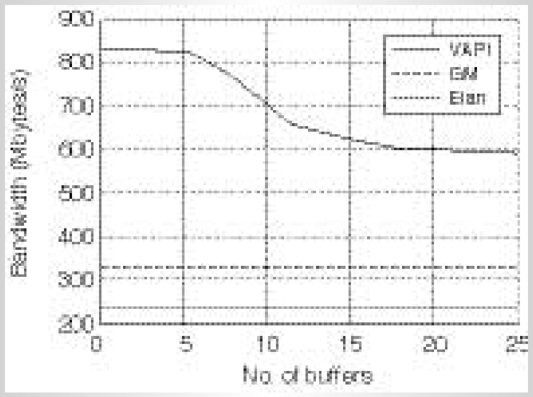

Figure 9. Bandwidth for 512-Kbyte messages, using a FIFO pattern of buffer reuse.

Figures 11a and 11b show the hot-spot performance results for 4-byte messages. We see that Elan scales very well when the number of slaves increases in both tests, and its latency only increases slightly. In the hot-spot send test, GM and VAPI show similar performance; their latencies increase from around 6 μs to 7 μs for one slave, ending up at around 26 μs for seven slaves. In these hot-spot tests, although both GM and VAPI show worse scalability than Elan, GM's latency increases more rapidly than VAPI's when the number of slaves increases.

Related work

LogP and its extension LogGP[6] are the

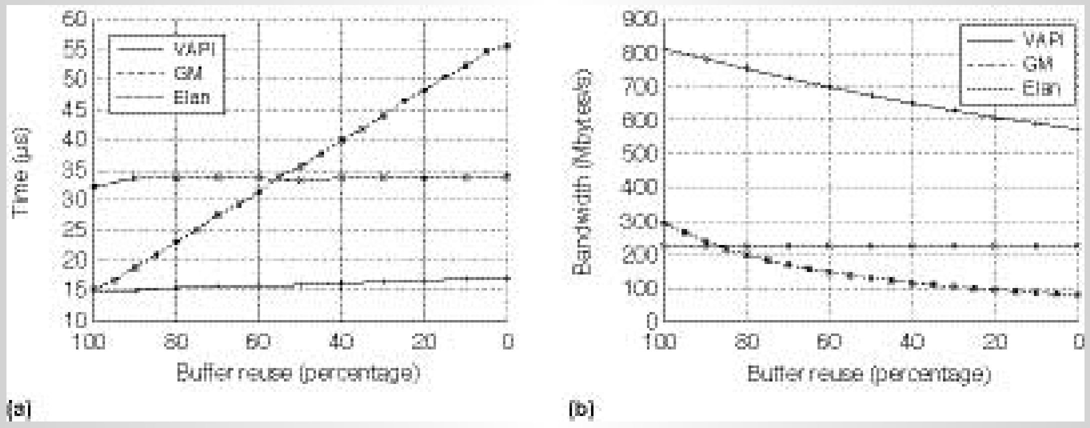

Figure 10. Effect of a buffer reuse pattern based on a percentage: latency test using 4Kbyte buffers (a) and bandwidth test using 512-Kbyte buffers (b).

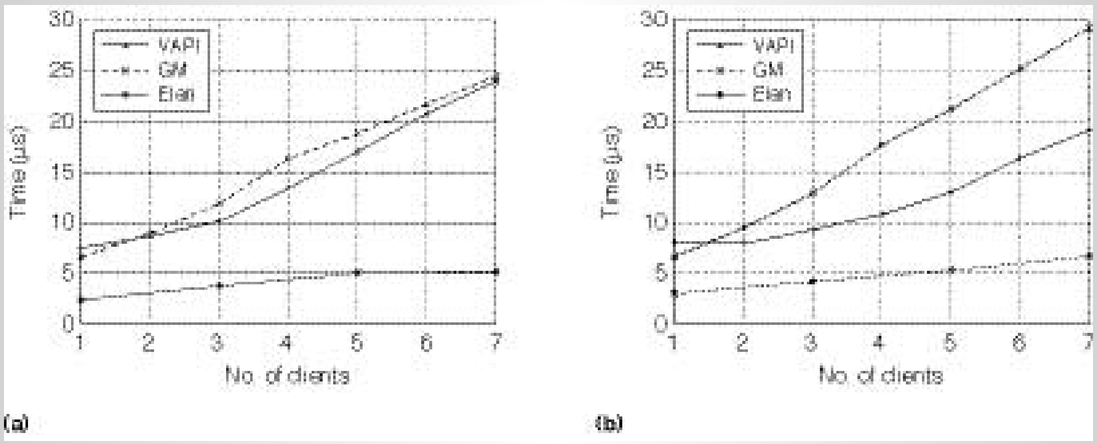

Figure 11. Host spot test results showing increase in the latency of 4-byte messages with increasing number of slave nodes for send (a) and receive (b) operations.

methodologies that researchers often use to extract performance parameters in conventional communication layers. In addition to the LogGP performance parameters, our study explored the performance impact of other advanced features available in the interconnects under study.

Other researchers have studied the performance of communication layers on the Myrinet and Quadrics networks. Our previous work[7,8] devises test suites and uses them to compare performance for Virtual Interface Architecture[9] and InfiniBand implementations. Here, we extend the work by adding several important scenarios that have strong application implication and apply them to a wider range of communication layers and networks. Bell et al.[10] evaluate communication layer performance for several parallel computing systems and networks. However, they base their evaluation on the LogGP model and used different test beds for different interconnects.

Our results show that to gain more insight into the performance characteristics of these interconnects, it is important to go beyond simple tests such as those for latency and bandwidth. Specifically, we must consider the performance impact of certain features such as remote memory access, completion notification, and address translation mechanisms in the network interface. In future, we plan to expand our micro-benchmark suite to include more tests and more interconnects. We also plan to extend our work to higher level programming models. MICRO

Acknowledgments

This research is supported in part by Saudia National Laboratory's contract #30505, Department of Energy's Grant #DE-FC02-01ER25506, and National Science Foundation's grants #EIA-9986052, #CCR-0204429 and #CCR-0311542.

References

1. N.J. Boden et al., "Myrinet: A Gigabit-per-Second Local Area Network," *IEEE Micro*, vol. 15, no. 1, Jan.-Feb. 1995, pp. 29-36.
2. F. Petrini et al., "The Quadrics Network: High-Performance Clustering Technology," *IEEE Micro*, vol. 22, no. 1, Jan.-Feb. 2002, pp. 46-57.
3. "InfiniBand Architecture Specification, Release 1.0," InfiniBand Trade Assoc., Oct. 2000; http://www.infinibandta.org/specs.
4. D.E. Culler et al., "LogP: Towards a Realistic Model of Parallel Computation," *Proc. 4th ACM SIGPLAN Symp. Principles and Practice of Parallel Programming* (PPOPP 93), ACM Press, 1993, pp. 1-12.
5. E.V. Carrera et al., "User-Level Communication in Cluster-Based Servers," *Proc. 8th Symp. High-Performance Computer Architecture* (HPCA 02), IEEE CS Press, 2002, pp. 275-288.
6. A. Alexandrov et al., "LogGP: Incorporating Long Messages into the LogP Model for Par-

allel Computation," *J. Parallel and Distributed Computing,* vol. 44, no. 1, July 1997, pp. 71-79.
7. M. Banikazemi et al., "VIBe: A Microbenchmark Suite for Evaluating Virtual Interface Architecture (VIA) Implementations," *Proc. Int'l Parallel and Distributed Processing Symp.* (IPDPS 01), 2001.
8. B. Chandrasekaran, P. Wyckoff, and D.K. Panda, "MIBA: A Microbenchmark Suite for Evaluating InfiniBand Architecture Implementations," *Performance TOOLS 2003, Lecture Notes in Computer Science,* vol 2794, Springer-Verlag Heidelberg, 2003, pp. 29-46.
9. D. Dunning et al., "The Virtual Interface Architecture," *IEEE Micro*, vol. 18, no. 2, Mar.-Apr. 1998, pp. 66-76.
10. C. Bell et al., "An Evaluation of Current High-Performance Networks," *Proc. Int'l Parallel and Distributed Processing Symp.* (IPDPS 03), IEEE CS Press, 2003.

Jiuxing Liu is a PhD student in the Computer and Information Science Department at The Ohio State University. His research interests include high-speed interconnects, cluster computing, and storage systems. Liu has a BS and an MS in computer science from Shanghai Jiao Tong University, China. He is a member of the IEEE.

Balasubramanian Chandrasekaran is a systems engineer at the Scalable Enterprise Computing Lab, Dell Computer Corp. His research interests include high-speed interconnects, high-performance computing, and virtualization of data centers. Chandrasekaran has an MS in computer science from The Ohio State University. He is a member of the IEEE and ACM.

Weikuan Yu is a PhD student in the Computer and Information Science Department at The Ohio State University. His research interests include cluster computing, resource management, and bioinformatics. Yu has an MS in molecular, cellular, and developmental biology from The Ohio State University. He is a member of the IEEE Computer Society.

Jiesheng Wu is a PhD student in the Computer and Information Science Department at The Ohio State University. His research interests include parallel and distributed computing with a focus on file, storage, and communication systems. He is a member of the IEEE and the Usenix Association.

Darius Buntinas is a postdoctoral researcher in the Mathematical and Computer Science Division of the Argonne National Laboratory. His research interests include communication support for programming models and using programmable network interface cards to improve cluster performance. Buntinas has a PhD in computer science from The Ohio State University, and an MS and a BS in computer science from Loyola University in Chicago. He is a member of the IEEE and the IEEE Computer Society.

Sushmitha Kini is a software engineer with the Windows Server group, Microsoft Corp. Her research interests include collective communication, high-performance computing, and directory services. Kini has an MS in computer and information science from The Ohio State University. She is a member of the IEEE.

Dhabaleswar K. Panda is a professor of computer science at The Ohio State University. His research interests include parallel computer architecture, high-performance computing, user-level communication protocols, interprocessor communication and synchronization, network-based computing, and quality of service. Panda has a PhD in computer engineering from the University of Southern California. He received a National Science Foundation Faculty Early Career Development Award, The Ohio State University's Lumley Research Award, and an Ameritech Faculty Fellow Award. He is a member of the IEEE Computer Society and ACM.

Pete Wyckoff is a research scientist at the Ohio Supercomputer Center. His research interests include high-performance networking and parallel file systems. Wyckoff has a PhD from the Massachusetts Institute of Technology. He is a member of the ACM.

Direct questions and comments about this article to Jiuxing Liu, 395 Dreese Lab, Computer and Information Science Department, 2015 Neil Ave., Columbus, OH 43210; liuj@cis.ohio-state.edu.