A Comparison of 4X InfiniBand and Quadrics Elan-4 Technologies

Ron Brightwell Douglas Doerfler Keith D. Underwood

Sandia National Laboratories*
P.O. Box 5800 MS-1110
Albuquerque, NM 87185-1110
E-mail: {rbbrigh, dwdoerf, kdunder}@sandia.gov

Abstract

Quadrics Elan-4 and 4X InfiniBand have comparable performance in terms of peak bandwidth and ping-pong latency. In contrast, the two network architectures differ dramatically in details ranging from signaling technologies to programming interface design to software stacks. Both networks compete in the high performance computing marketplace, and InfiniBand is currently receiving a significant amount of attention, due mostly to its potential cost/performance advantage. This paper compares 4X InfiniBand and Quadrics Elan-4 on identical compute hardware using application benchmarks of importance to the DOE community. We use scaling efficiency as the main performance metric, and we also provide a cost analysis for different network configurations. Although our 32-node test platform is relatively small, some scaling issues are evident. In general, the Quadrics hardware scales slightly better on most of the applications tested.

1. Introduction

InfiniBand is the latest network technology competing in the large-scale, high performance computing (HPC) cluster marketplace. Unlike Quadrics, which was specifically designed to be the high-performance interconnect of a tightly-coupled cluster machine, InfiniBand was designed for more general purposes, including storage area network solutions and other data center and non-HPC needs. In spite of this, numerous members of the HPC community have begun to tout InfiniBand as the cluster interconnect technology of the future. This work studies the performance and scalability of 4X InfiniBand and compares it to the recently released Quadrics Elan-4 using scientific applications currently in use at Sandia National Laboratories.

At this time, the Quadrics Elan-4 product is the only commercial interconnect with the potential to compete with 4X InfiniBand in delivered latency and bandwidth. Both networks claim to deliver approximately 2 GB/s of bandwidth at the physical layer and use PCI-X to interface with the host. Both networks also claim to deliver sub-10μs latencies; however, in many aspects, the network architectures differ dramatically. InfiniBand uses narrow physical channels that leverage extremely high-speed serial links while Quadrics leverages a wider, slower physical layer. At the programming interface layer, InfiniBand is limited to remote DMA and queue pair semantics that are not a strong match to MPI semantics. In contrast, Quadrics provides an interface that more closely matches the MPI semantics. These differences in low-level programming interface drive dramatic differences in the MPI implementations over top of these networks.

Recently, Sandia had a rare opportunity to compare InfiniBand directly to the Quadrics Elan-4 interconnect. For a short time, Sandia was able to deploy a cluster with a 96-node InfiniBand partition and a 32-node Quadrics Elan-4 partition, with both partitions using identical compute hardware. This allowed a head-to-head comparison between the two technologies where the only difference was the high-speed interconnect, its associated drivers, and the respective MPI implementation. Using the two partitions, micro-benchmark and application benchmark data has been collected. The application benchmarks were chosen from the set of scientific applications in use at the DOE defense labs to be as representative as possible of current workloads. In this paper, we evaluate InfiniBand as an HPC network technology to determine if the differences in network architecture matter significantly.

There is a substantial amount of previous work on evaluating and comparing high-performance interconnects. Most recently, Liu et. al. presented a comparison of InfiniBand, Myrinet, and Quadrics Elan-3 on an eight-node cluster [11]. This previous study analyzed performance using several dif-

*Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

ferent communication micro-benchmarks, the NAS parallel benchmarks, and Sweep3d. For our study, we are using the next-generation Elan-4 network, which has much greater performance than Elan-3. We also compare performance out to 32 nodes and include results from an additional real-world application.

The rest of this paper is organized as follows. The next section describes the platform and benchmarks used in our evaluation. Following that, specific performance related feature differences between the InfiniBand and Quadrics network architectures are discussed in Section 3. Results that compare the two platforms are presented in Section 4. A brief cost discussion based on list prices is presented in Section 5, and conclusions are presented in Section 6. Finally, avenues for future work are presented in Section 7.

2. Platforms and Benchmarks

The InfiniBand and Quadrics data was collected using identical compute nodes, which are described in Table 1. The InfiniBand partition of the cluster consisted of 96 nodes, and the Quadrics Elan-4 partition consisted of 32 nodes. The partitions were independent in operation, but shared a common management infrastructure. Both networks were evaluated using a variety of micro-benchmarks and application benchmarks. Data was collected using both one MPI process per node (1 PPN) and two MPI processes per node (2 PPN).

2.1. Micro-benchmarks

The classic method of measuring delivered peak bandwidth and delivered minimum latency is to use ping-pong message exchanges. In this method, only two processes are involved in the exchange and only a single message is outstanding at any given time. A message is sent to a receiving process, which then re-sends the same message back to the sending process. Total time for the transaction is measured by the sending process, and the latency is calculated as the total time divided by two. Several hundred exchanges are performed and the average time is reported. The results presented in Section 4 were collected using the Pallas MPI Benchmarks PingPong routine [2].

To contrast the ping-pong method, another method using a non-blocking, streaming message passing pattern was used [12]. In this method, the sender transmits a predefined number of back-to-back messages to the receiver, which has pre-posted a matching number of corresponding receive requests. This benchmark quantifies the ability to fill the message passing pipeline.

The third micro-benchmark used to evaluate the two networks was the Effective Bandwidth (b_eff) benchmark [1, 21]. This benchmark measures the aggregate bandwidth of a network of systems, rather than the capability of a single link. Several message sizes, communication patterns, and methods are used. This benchmark accounts for the different achievable bandwidths of both short and long messages that are typical in real applications.

2.2. Application Benchmarks

The set of application benchmarks used in our evaluation consists of one production code, one DOE benchmark code, and one selection from the widely used NAS Parallel Benchmark suite. Since our test platform was only available for a limited time, we chose these particular benchmarks because they cover a broad scope of application characteristics.

2.2.1. LAMMPS

The LAMMPS (Large-scale Atomic/Molecular Massively Parallel Simulator) application is a classical molecular dynamics code designed for simulating molecular and atomic systems on parallel computers using spatial-decomposition techniques [17, 16, 18]. LAMMPS is used extensively by the materials science and molecular science research communities at Sandia, comprising a significant share of the cycles used on Sandia's parallel computing clusters.

Two example problem sets were run. The first set is an example of atomic simulations of Lennard-Jones systems (LJS), while the second is an example of a biomembrane model. Both problem sets are scaled studies, where each process is assigned an equivalent amount of computational work. In contrast, a fixed study keeps the size of the problem constant as the number of processes in the job is increased, so each process has less computational work. On an ideal machine, perfect scaling efficiency results for a scaled-size problem would result in a horizontal line.

2.2.2. Sweep3d

The Sweep3D benchmark code solves a 1-group time-independent discrete ordinates (Sn) 3D Cartesian (XYZ) geometry neutron transport problem. The XYZ geometry is represented by an IJK logically rectangular grid of cells. The angular dependence is handled by discrete angles with a spherical harmonics treatment for the scattering source. The solution involves two steps: the streaming operator is solved by sweeps for each angle and the scattering operator is solved iteratively [10]. For this study, we used a fixed-sized problem based on a 150-cubed spatial grid point data set.

Table 1. Evaluation Platforms

Node Type	Dell PowerEdge 1750 Server: Dual 3.06 GHz Intel Xeon Processors, 533MHz FSB, ServerWorks GCLE chip set, 133 MHz PCI-X bus for the high-speed interconnect
Operation System	Red Hat Linux 9, Linux kernel 2.4.22
InfiniBand Interconnect	Voltaire: HCS 400 4X host channel adapter, ISR 9600 Switch Router, 4X copper cable. MPI Implementation is the MVAPICH version release 0.9.2 from The Ohio State University [14].
Quadrics Interconnect	Quadrics QsNetII: QM500 Network Adapter, QS5A 64 port Node Level Switch. MPI Implementation is the Quadrics MPI based on MPICH, Release MPI.1.24-28.

2.2.3. NAS Parallel Benchmark CG

The NAS Parallel Benchmarks (NPB) are a set of 8 codes designed to evaluate the performance of parallel computers. The benchmarks, which are derived from computational fluid dynamics applications, consist of five computational kernels and three pseudo-applications [4, 8].

For this study, we selected the Conjugate Gradient (CG) benchmark because its computation and communication demands are most similar to those of real Sandia applications. We chose the Class A problem size so that the data would reside in cache for all of the jobs that were run. This strategy results in a low computation to communication ratio, which provides the best scaling information.

3. Network Features

InfiniBand and Quadrics (both Elan-3 and Elan-4) provide different capabilities in their network application programming interfaces (API). In this section, we provide an overview of the programming interface used by each network to implement MPI semantics, outline the important capabilities of the two networks, and discuss how these issues can impact the scalability and performance of an MPI implementation.

3.1. Tports

The MPI implementation for the Elan network uses a network programming interface called Tagged Ports, or Tports. This interface was initially developed by Meiko for their supercomputing products (circa 1994) [5] and was carried forward by Quadrics into their Elan products. Tports preceded the MPI Standard, but it provides similar two-sided message passing semantics. As such, the MPI implementation provided by Quadrics uses Tports as its underlying transport layer.

The Elan architecture allows a process to execute a thread on a processor on the network interface. This approach provides enormous flexibility for handling network protocol processing. Tports is implemented via a thread running on the network interface that processes incoming messages. The thread performs message selection by matching incoming message tags against a queue of receive requests. Upon finding a match, the network thread transfers the data directly into the application message buffer. The Tports interface also buffers messages for which there is no matching receive request. The network thread manages the buffer space needed to store these unexpected messages, and this buffering is hidden by the Tports interface.

3.2. RDMA

The InfiniBand specification does not define a standard API. Instead, it specifies functionality through an abstraction called *Verbs*. This lack of standardization has led to numerous different APIs for InfiniBand hardware. For example, the Verbs Application Programming Interface (VAPI) is provided by Mellanox for their hardware, while the InfiniBand Linux SourceForge Project provides an API known as the InfiniBand Access Layer (IBAL).

There are also efforts to standardize a programming interface for remote DMA (RDMA) operations for several networks, including InfiniBand. These interfaces provide capability similar to what the InfiniBand verbs specify. Two such efforts are the User Direct Access Programming Library (UDAPL) [7] and the Remote Direct Data Placement (RDDP) API [9] The RDMA Consortium [19] is also developing a verbs-based specification for RDMA-capable networks.

Because of the similarity of VAPI, UDAPL, and RDDP, we will refer to these collectively as RDMA, even though these interfaces include some form of two-sided message passing via a queue pair abstraction as well. Also, some of these functional interfaces offer a wide variety of different

levels of service, such as unreliable datagram, reliable datagram, and reliable connection-oriented. This evaluation is based on the service that is used for published MPI implementations for InfiniBand [14, 13, 20].

3.3. MPI Characteristics

In this section, we discuss several desirable characteristics of an MPI implementation that we believe are important to achieving scalability and performance. Comparing Quadrics and InfiniBand, we find that Quadrics has several features that support these desirable characteristics, while InifiniBand does not.

3.3.1. Connectionless

It is highly desirable that the lowest-level programming interface to a high-performance network be connectionless. In this context, connectionless is defined to be an interface that does not dynamically allocate shared network resources to maintain state and does not require an explicit connection establishment step before a data transfer. For example, a reliability protocol using sequence numbers can be connectionless if the resources for keeping track of sequence numbers are static and do not scale linearly with the number of processes that use the network.

The Tports interface for Quadrics is connectionless. While Quadrics requires that a capability be allocated for a group of processes that wish to communicate, the allocation and management of capabilities is independent of the programming interface. Conversely, InfiniBand is connection-based. Two processes that wish to communication must first go through a connection establishment phase (for queue pairs) or a key exchange (for RDMA) before data can be sent or received.

3.3.2. Memory Registration

Operating systems that support demand-paged virtual memory require pages involved in network transfers to be identified to insure that they are resident in physical memory. The overhead of this page management can be significant. Some networks, such as InfiniBand, require applications to make function calls to register and unregister memory regions in involved in data transfer operations. In contrast, Quadrics has no such limitations. The Elan network interface hardware has a memory management unit for efficient address translation and the network interface works with the operating system to maintain address mappings.

To our knowledge, there has been no in-depth comparison of the explicit user-level, host-based memory registration method used in InfiniBand with the implicit, network interface-driven registration method used by Quadrics. It can be argued that the Quadrics approach eases the burden of the application developer, but it is unknown whether this simplification can be detrimental to performance. The impact of memory registration on latency and bandwidth performance was measured in [11] by implementing a ping-pong benchmark that varies the percentage of re-use of the message buffer. Results showed that both InfiniBand and Quadrics Elan-3 are sensitive to memory registration costs. Copy blocks (sometimes called bounce buffers) can be used to mitigate the memory registration costs. Messaged are copied through these buffers (initialized at startup) when they are sent or received. As an example, such buffers are used by MPICH/GM for Myrinet for messages smaller than 16 KB, which is why the results of the buffer re-use benchmark in [11] do not vary below this message size.

3.3.3. Independent Progress

Enabling the MPI implementation to make progress independent of calls to the MPI library is desirable. The MPI Standard mandates a Progress Rule for asynchronous communication operations which states that once a non-blocking communication operation has been posted, a matching operation will complete regardless of whether the application makes further MPI library calls. However, different interpretations of this rule have emerged. From a performance standpoint, it can be more efficient for an implementation to allow for progress independent of MPI library calls, especially for large messages. Quadrics allows for independent progress, since the network interface is responsible for servicing incoming requests. The Tports thread examines incoming messages and decides how to proceed. Because InfiniBand does not have this capability, some other mechanism, such as interrupts or a user-level thread, must be used to achieve independent progress. The MVAPICH implementation used in this study does not support independent progress – it relies on MPI library calls to be made in order to make progress on outstanding communication requests.

3.3.4. Offload

A related characteristic is the ability to offload MPI matching and protocol processing to an intelligent or programmable network interface. While all OS-bypass technologies support delivering data directly into an application's address space, most networks require the host processor to perform MPI matching and queue traversal functions. Others, such as Quadrics, do not. Offload can improve application by reducing overhead [15], but it can also force the traversal of long queues on a slow processor on the network interface [22].

3.3.5. Overlap

It is also desirable to have a network programming interface that supports overlap of computation and communication, as well as one that supports overlap of communication with communication, where several data transfer requests can be outstanding. The benefit of overlap is that the host processor need not directly be involved in the transfer of data to its final destination, allowing the CPU to be dedicated to computation.

Some networks, like InfiniBand, are capable of performing RDMA read and write operations that fully overlap communication with computation. However, the target address of these operations must be known. If the MPI implementation does not support independent progress, long message transfers are not able to provide any significant overlap because the sender must wait for the receiver to respond with the address of the target buffer. If the target address is only communicated to the sender when the receiver completes the receive, all opportunity for overlap is gone. If the transfer of the target address is handled directly by the network interface, or by a user-level thread, then significant overlap is possible. Since the MVAPICH implementation does not support independent progress, it is unable to support a significant amount of overlap. Quadrics, on the other hand, is able to fully support overlap [11].

4. Results

For this study, Quadrics Elan-4 and 4X InfiniBand are compared at both the micro-benchmark and application benchmark levels. Application benchmarks are a critical portion of the evaluation process, since application performance cannot always be derived directly from micro-benchmarks. Micro-benchmarks, however, isolate specific performance issues in ways that can provide insight into the source of performance differences. Thus, both are presented here.

4.1. Micro-benchmarks

Results of the ping-pong latency test are shown in Figure 1(a). Note that the x-axis is on a log scale. The average latency for Elan-4 is approximately half of that for InfiniBand. Perhaps more importantly, the InfiniBand latency has a sharp jump between 1 KB and 2 KB messages. This is a typical shift as MPI switches from a "short" message protocol to a "long" message protocol. It should be noted, however, that shifting the break point between the two for InfiniBand involves a trade-off in system resource usage. MVAPICH maintains a set of RDMA buffers for short messages that grows with the number of processes and with the maximum size of a "short" message. The linear relationship between the number of processes and the amount of short message buffer space constrains the maximum "short" message size more tightly than on networks where the buffer space is only related to the size of "short" messages and not to the number of processes. The latency increases for both Elan-4 and InfiniBand then continue to track the increase in message size (based on bandwidth limitations).

Bandwidth measurements are somewhat more susceptible to the measurement method used. A comparison of Elan-4 and InfiniBand bandwidth using both the Pallas MPI Benchmarks and the streaming bandwidth benchmark is presented in Figure 1(b). The plot shows that the Elan-4 ping-pong network performance is better for all message sizes; however, both networks asymptotically approach similar bandwidth performance levels. The dramatic drop in bandwidth for InfiniBand using a 4 MB message size has been observed by others [3] and is reportedly due to thrashing when registering memory. It is reportedly fixed in subsequent versions of MVAPICH.

The difference in the rate at which Elan-4 and InfiniBand approach their maximum bandwidth as message sizes grow is dramatics. For example, at a message size of 8 KB, the Elan-4 and InfiniBand bandwidths are 552 MB/s and 249 MB/s respectively — a difference of a factor of two. Figure 1(c) plots the ratio of Elan-4 and InfiniBand bandwidth as a function of message size for both the streaming and ping-pong tests. At small message sizes, Elan-4 achieves over a factor of five advantage using the streaming benchmark.

Results from the b_eff benchmark are shown in Figure 1(d). The plot presents the b_eff rating normalized by the number of processes involved in the job. For an ideal machine[1], the trend line would be flat. For this test, the benchmark was run using the 1 PPN mode. It should be noted that b_eff is a logarithmic average of bandwidths measured at several different message sizes. The majority of these messages are a kilobyte or less, and the logarithmic average gives significantly greater weight to the shorter message lengths than a simple arithmetic average would. While b_eff provides a single metric as its output, it is more dominated by small message bandwidth (which are significantly affected by latency characteristics) than by long message bandwidth. Since b_eff is predominantly a measure of short-message bandwidth, the values of b_eff are low relative to peak delivered bandwidths.

4.2. Application Benchmarks

Application performance is the ultimate metric by which networks should be be measured. Low latency and high bandwidth are not necessarily sufficient to insure good ap-

[1] A machine with an ideal interconnect, zero latency and infinite bandwidth.

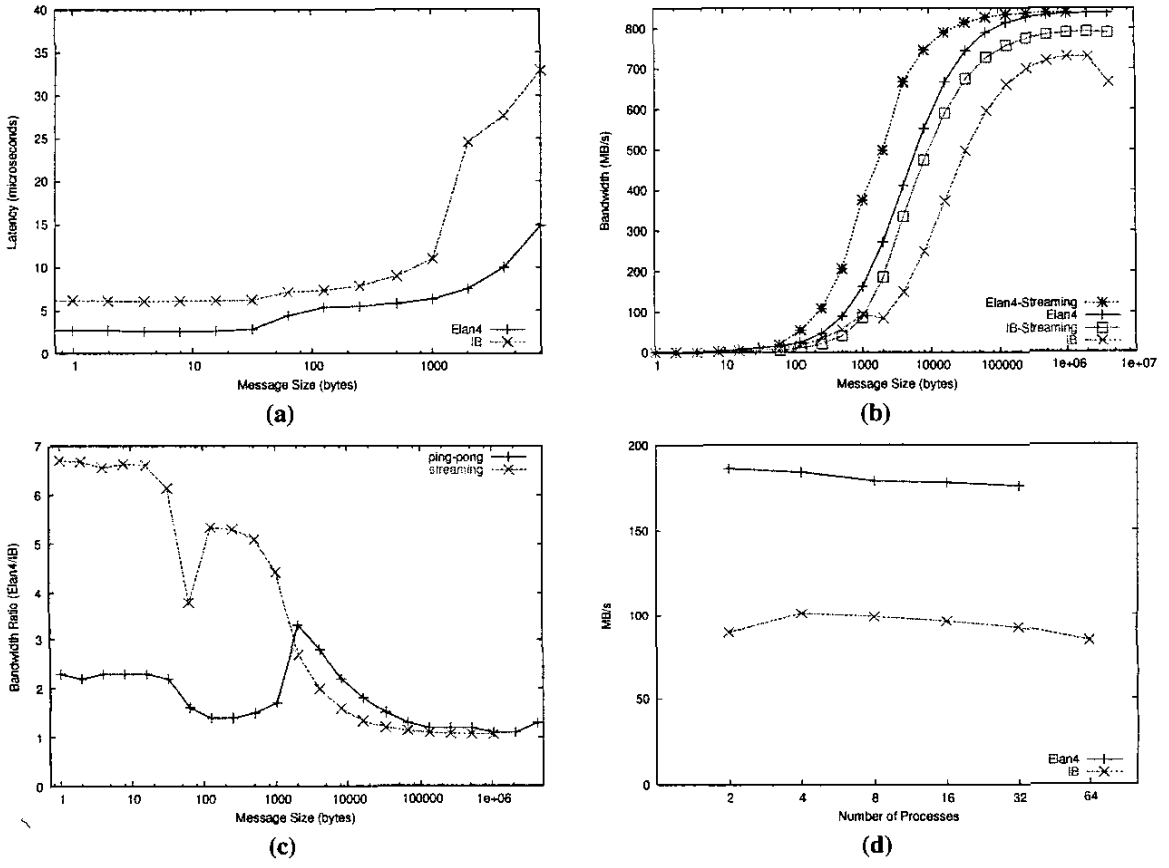


Figure 1. A comparison of 4X InfiniBand and Quadrics Elan-4 in terms of: **(a)** latency, **(b)** bandwidth, **(c)** bandwidth ratio, and **(d)** effective bandwidth.

plication performance or scalability. For example, issues such as independent progress and host overhead can make significant differences. For each benchmark, both a measure of execution time and scaling efficiency are presented. A scaling efficiency of 100% indicates a machine that is N times faster when using N more processors. These application benchmarks highlight differences in the two networks that are unlikely to be attributable to differences in latency and bandwidth alone. Each data point is the average of four benchmark runs.

4.2.1. LAMMPS

Performance for the LAMMPS application with the LJS data set is shown in Figure 2. As noted earlier (Section 2), this was a scaled speedup study. Scaled speedup studies attempt to maintain the ratio computation to communication by increasing the size of the problem in proportion to the number of processes; thus, on an ideal network, execution time (Figure 2(a)) should be flat. Figure 2(b) graphs the same data in terms of scaling efficiency.

The data in Figure 2 is approximately what would be expected. The 1 PPN mode outperforms the 2 PPN mode for both networks. In 1 PPN mode, Elan-4 outperforms InfiniBand marginally as expected, since the micro-benchmarks show that Elan-4 has slightly better performance. The interesting data in this figure, however, is the pair of 2 PPN lines. There is a much wider margin between the Elan-4 2 PPN curve and the InfiniBand 2 PPN curve. Correspondingly, there is a much wider performance margin between the InfiniBand 1 PPN curve and the InfiniBand 2 PPN curve than between the Elan-4 1 PPN curve and the Elan-4 2 PPN curve. These data points are extremely important in a market where two processors per node configurations are becoming standard.

These differences cannot be readily explained by differences in the micro-benchmark performance. Instead, they have two likely primary sources. First, the Elan-4 network provides offload of MPI semantics while InfiniBand performs these functions on the host. This means that the Elan-4 network places much less load on the host, including an associated reduction in cache pollution. This effect could become exaggerated in two processors per node scenarios. Second, the Elan-4 network (and associated MPI implementation) provides independent progress, where the MPI implementation for InfiniBand does not. This means that, for InfiniBand, MPI progress (matching, rendezvous progress, etc.) only occurs when the application makes MPI calls. When two application processes (on two host processors) contend for access to memory and network interface resources, this may impact the rate at which progress can occur.

Figure 3 shows data from the LAMMPS application running the membrane data set. Like the LJS study, the membrane problem is a scaled speedup study. Unlike the LJS data set, results from the 1 PPN and 2 PPN runs for Elan-4 are extremely close. This typically implies that the computation to communication ratio is very high; however, another possible explanation could be that the code exploits asynchronous communications and successfully leverages overlap of computation and communication. This second explanation is given more credence when examining the InfiniBand results. The InfiniBand network, with an MPI implementation that does not provide independent progress, has a much larger gap between the 1 PPN and 2 PPN curves. Further research, such as that performed in [6], is needed to discern the true cause of these differences.

A second interesting observation about runs with the LAMMPS membrane data set is that the Elan-4 curves are almost perfectly flat from 8 to 32 nodes. Scaling efficiencies of 93% for 1 PPN runs and 91% for 2 PPN runs are very respectable scores for a 32-processor configuration. In contrast, at 32 nodes, InfiniBand scaling efficiency is tailing off rapidly, achieving only 84% scaling efficiency with 32 processors in a 1 PPN configuration and 77% scaling efficiency with a 2 PPN configuration. If the trends in efficiency for both networks continue, this represents a serious limitation in the scalability of InfiniBand networks relative to Quadrics networks.

4.2.2. Sweep3d

Although the Sweep3d benchmark was run in 1 PPN and 2 PPN modes, only the 1 PPN data is presented in Figure 4, as the 2 PPN data is similar. This suggests that this benchmark has a high computation time to communication time ratio. In addition to a grind time plot, scaling efficiency is plotted. The data illustrates a weakness of fixed problem size scaling studies. As is the case with many benchmarks, Sweep3d exhibits a superlinear speedup when moving from 1 to 4 processors using this input data. This effect is typically attributable to the unscaled problem fitting in cache when the number of processors is increased.

For this benchmark, the scaling efficiency analysis is particularly important. The large change in scale on the grind time plot obscures the differences between the networks; however, moving to a plot of scaling efficiency emphasizes the significant advantage Elan-4 holds at 9 and 16 nodes. At 25 nodes, however, the scaling efficiency of InfiniBand jumps dramatically and exceeds that of Elan-4. It is unclear why this occurs. After analyzing the data, several additional data sets were run on InfiniBand, and these are plotted in Figure 5. For this plot the 4-processes data point was used as the normalization factor to compute scaling efficiency. These results indicate that for the transition from 16 to 25 nodes the scaling efficiency continued the existing trend.

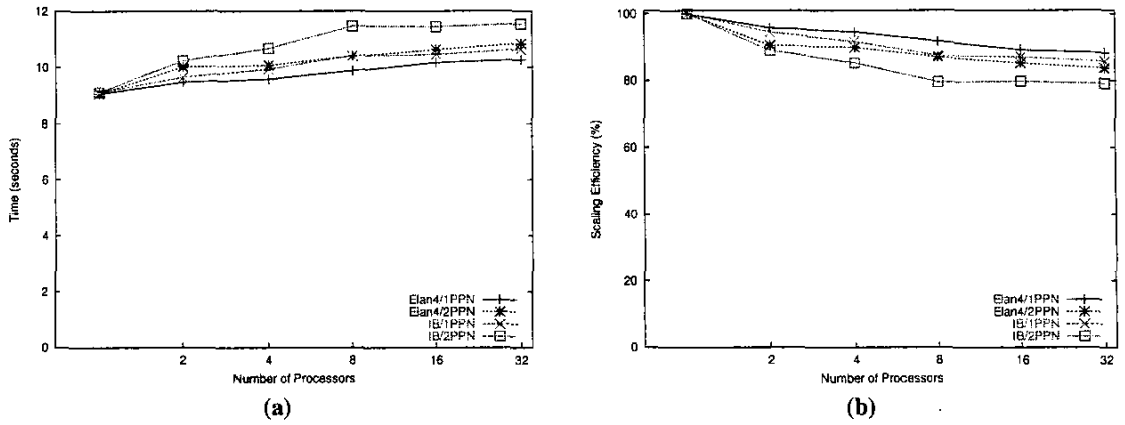


Figure 2. Performance of the LJS data set with the LAMMPS application benchmark: (a) time, and (b) scaling efficiency

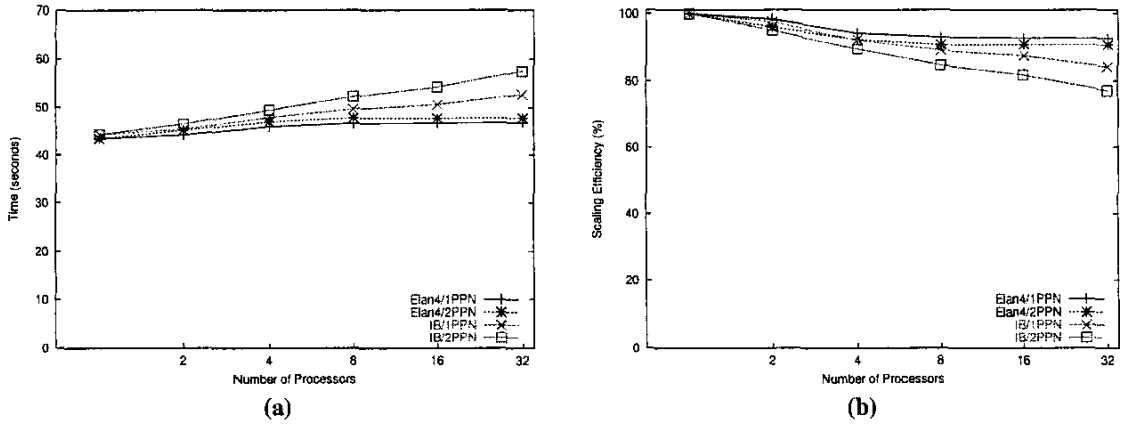


Figure 3. Performance of the membrane data set with the LAMMPS application benchmark: (a) time, and (b) scaling efficiency

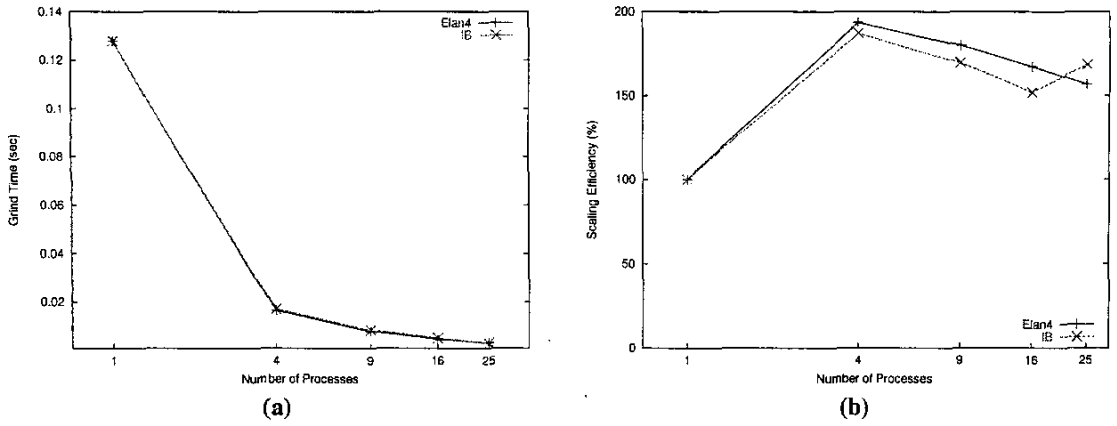


Figure 4. Performance of the Sweep3d benchmark: (a) time, and (b) scaling efficiency

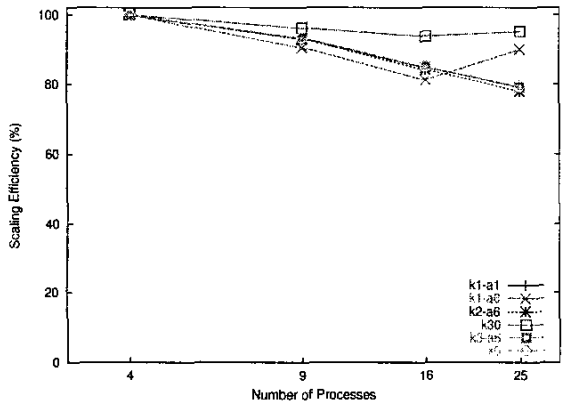


Figure 5. Performance of various input sets for Sweep3d

Table 2. InfiniBand list prices

Description	List
HCA	$995
24 Port Switch	$7,450
288 Port Switch	$196,100
288 Port Chassis	$29,950
24 Port Leaf	$10,950
96 Port Switch	$92,025
96 Port Chassis	$47,025
12 Port Leaf	$5,625
Cables for Spine	$150
Cables for Hosts	$175

Thus, it would appear that this input data is an anomaly. Unfortunately, by the time the data was analyzed, the Elan-4 test system had been disassembled and comparable tests could not be completed for Elan-4.

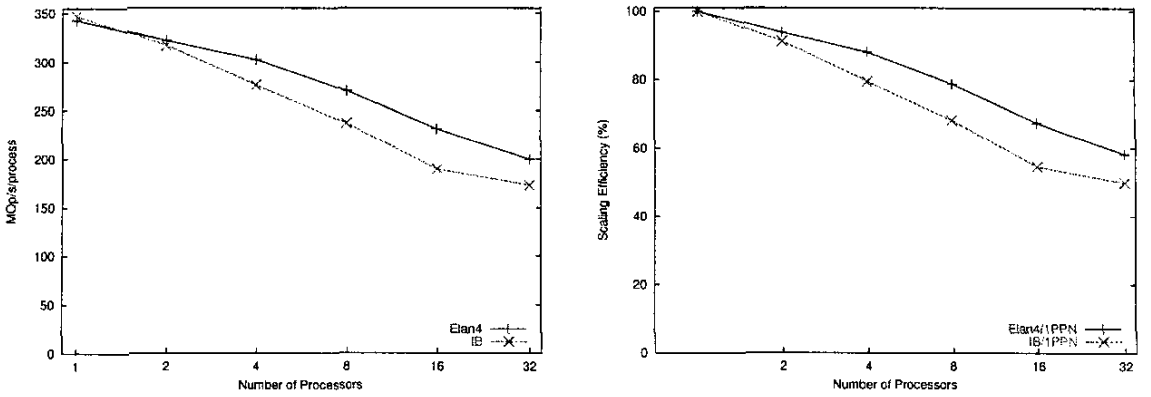
4.2.3. NAS Parallel Benchmark CG

The third benchmark chosen was the CG benchmark from the NPB suite. Since this benchmark uses a fixed problem size, the class A problem size was chose because it is known to fit in cache at all processor counts. This prevents it from exhibiting the superlinear speedups seen in the Sweep3d benchmark. This data set size also provides a communication dominated (because of the small problem size) benchmark that is representative of typical Sandia codes. Not surprisingly, both Quadrics and InfiniBand networks rapidly drop in scaling efficiency as the node count grows; however, Quadrics maintains a distinct advantage. This advantage seems to grow slightly as the node count grows, but more data would be needed to confirm the trend.

5. Cost Issues

In the cluster computing marketplace, cost is a significant factor. We provide a basic discussion of cost issues here. Tables 2 and 3 show the list prices for the basic components of the InfiniBand and Quadrics networks used for this analysis (current as of April, 2004).

Figure 7 shows four cost lines. The top line is the cost (per port) of building Quadrics Elan-4 networks of various sizes. The second line is the cost per port of networks built from 96-port InfiniBand switches (the largest available when this study began). The final pair of lines shows network costs for InfiniBand networks built using a combina-



(a) (b)

Figure 6. Performance of the NAS CG benchmark: (a) MOps/second/process, and (b) scaling efficiency

Table 3. Quadrics Elan-4 list prices

Description	List
QM-500 Network Adapter	$1,200
Node Level Chassis	$93,000
Top Level Switch	$110,500
QM580 Clock Source	$1,800
QM581-05 EOP Link Cable, 5M long	$225
QM581-03 EOP Link Cable, 3M long	$185
CAT5E Clock/Ethernet Cable, 10M long	$40

tion of 24-port and 288-port switches that are now available.

The graph indicates that Elan-4 is relatively cost competitive with InfiniBand networks built from 96-port switches. The difference in cost is comparable to the difference in application performance. When considering the newer InfiniBand switch configurations, the cost of InfiniBand drops dramatically; thus, InfiniBand seems to be a significant win. The difference in cost for the network (per node) is approximately 65% at large scale. An important note, however, is that when node cost is included (using $2500 as a lower bound of rack mounted, dual processor node cost), the difference between Elan-4 and 4X InfiniBand *total system cost* is only 4% and 51% (96-port switches and 288-port switches, respectively).

A 51% difference in price is a dramatic hurdle to overcome in cost. For markets that care most about cost, this differential is virtually insurmountable. For other markets, the need for a scalable capability resource may win out over reasonable differences in cost. Even in these markets, though, cost-performance is an issue. The improvements in performance must at least somewhat offset the increase in cost.

This study has not explored issues of large scale; how-

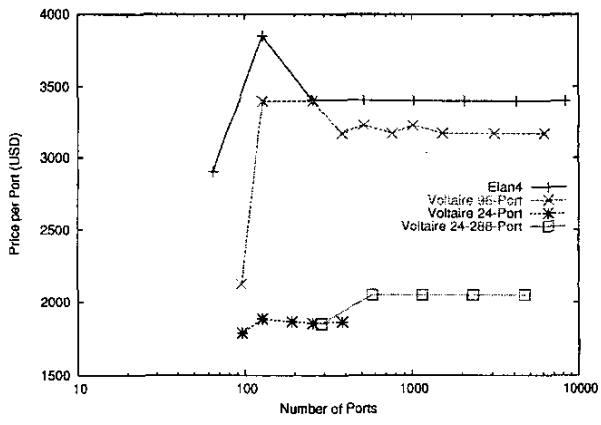


Figure 7. Cost comparison of InfiniBand and Quadrics Elan-4

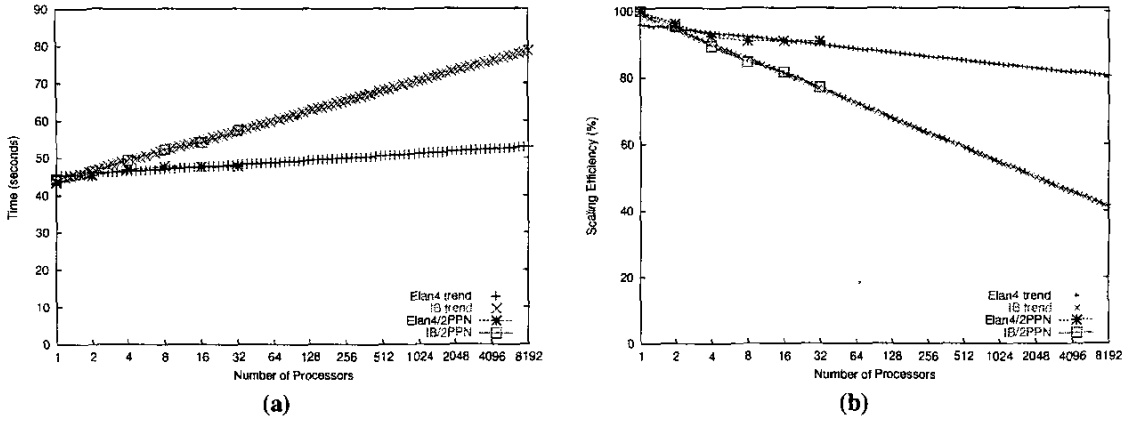


Figure 8. Extrapolated performance of the membrane data set with the LAMMPS application benchmark: (a) time, and (b) scaling efficiency

ever, we can answer a simple question: does Elan-4 have any hope of being competitive with InfiniBand in cost-performance at scale? To do this, we examine the LAMMPS membrane data set, which is the best performance case for Elan-4. Figure 8 extrapolates this data out to 8192 processors, assuming the scaling trends continue exactly as they did for the first 32 nodes. This is probably an optimistic assumption for Elan-4. Nevertheless, the result is a difference of nearly 40% in scaling efficiency at 1024 nodes. Thus, we can say, with some confidence, that Quadrics might be able to be competitive for some applications at scale, if current trends continue.

6. Conclusions

The results presented here are a preliminary look at the relative merits of the InfiniBand and Quadrics networks. Although a 32 node system is insufficient to fully analyze issues of scale, the Quadrics Elan-4 network shows indications of much better scalability than current 4X InfiniBand networks. Indeed, in all but one anomalous case, the Elan-4 network exhibited much better scaling efficiency than InfiniBand. This gap appears to grow as the number of nodes increases, but data at larger scale is needed to further assess the trends. In addition, the Quadrics programming interface is a closer match to the MPI semantics, which may be a significant factor in providing better performance and scalability.

Cost is an extremely significant issue when considering cluster computing interconnects. Emerging switch technology has made InfiniBand a much less expensive option than Quadrics Elan-4. At the scale of this study, it is impossible to conduct a thorough cost-performance analysis. However, preliminary results indicate that these two technologies *could* be cost-competitive at scale.

7. Future Work

The biggest limitations of this type of study are system scale and the number of applications evaluated. System scale is extremely difficult to address properly — large-scale procurements do not typically include two high performance networks, and different procurements tend to have slight, but significant, differences in compute hardware. Using the data presented here as a baseline, we plan to address the issues of system scale by using independent platforms and studying scaling efficiency. On these platforms, we will also be able to study a greater breadth of applications for a longer period of time at a larger scale. By using the same applications, the 4 to 32 node data can be calibrated back to measurements presented here to enhance the validity of the comparison.

The second focus for future work will be to use techniques such as those in [6] and to develop new techniques to study the exact source of differences in scaling efficiency. These differences could be as simple as current inefficiencies in the MPI implementation or could be as complex as the capability to provide independent progress through hardware offload. These will be challenging questions to answer, but the differences in application performance and scaling efficiency for two networks of seemingly similar performance indicates a need for further study.

Acknowledgments

The authors would like to acknowledge the contributions of Milton Clauser, Glenn Laguna, and William Lawry (all at Sandia National Laboratories) in the collection of this data. We would also like to thank Chris Maestas from Hewlett Packard Company and Steve Simonds of Performance Software Solutions, Inc. for help in obtaining the data presented here.

References

[1] *Effective Bandwidth (b_eff) Benchmark.* `http://www.hlrs.de/organization/par/services/models/mpi/b_eff/`.

[2] *Pallas MPI Benchmarks.* `http://http://www.pallas.com/e/products/pmb/index.htm`.

[3] Conversations. Second Annual DOE Infiniband Workship, Jan. 2004. Sonoma, CA.

[4] D. H. Bailey et al. The NAS Parallel Benchmarks. *International Journal of Supercomputer Applications*, 5(3):63–73, 1991.

[5] J. Beecroft, M. Homewood, and M. McLaren. Meiko CS-2 interconnect Elan-Elite design. *Parallel Computing*, 20(10-11):1627–1638, 1994.

[6] R. Brightwell and K. D. Underwood. An analysis of the impact of overlap and independent progress for MPI. In *Proceedings of the 2004 International Conference on Supercomputing (ICS2004)*, St. Malo, France, June 2004.

[7] DAT Collaborative. *uDAPL: User Direct Access Programming Library*, May 2003. `http://www.datcollaborative.org/uDAPLv11.pdf`.

[8] R. F. V. der Wijngaart. NAS Parallel Benchmarks Version 2.4. Technical report, October 2002.

[9] IETF RDDP Working Group. `http://www.ietf.org/html.charters/rddp-charter.html`, 2003.

[10] K. R. Koch, R. S. Baker, and R. E. Alcouffe. Solution of the first-order form of the 3-d discrete ordinates equation on a massively parallel processor. *Transactions of the American Nuclear Society*, 65(198-9), 1992.

[11] 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. In *The International Conference for High Performance Computing and Communications (SC2003)*, November 2003.

[12] J. Liu, B. Chandrasekaran, W. Yu, J. Wu, D. Buntinas, S. P. Kini, P. Wyckoff, and D. K. Panda. Micro-benchmark performance comparison of high-speed cluster interconnects. *IEEE Micro*, 24(1), January/February 2004.

[13] J. Liu, W. Jiang, P. Wyckoff, D. K. Panda, D. Ashton, D. Buntinas, W. Gropp, and B. Toonen. Design and implementation of MPICH2 over InfiniBand with RDMA support. In *Proceedings of the 2004 International Parallel and Distributed Processing Symposium*, April 2004.

[14] J. Liu, J. Wu, S. P. Kini, P. Wyckoff, and D. K. Panda. High performance RDMA-based MPI implementation over InfiniBand. In *Proceedings of the 2003 International Conference on Supercomputing (ICS-03)*, pages 295–304, New York, June 23–26 2003. ACM Press.

[15] R. P. Martin, A. M. Vahdat, D. E. Culler, and T. E. Anderson. Effects of communication latency, overhead, and bandwidth in a cluster architecture. In *Proceedings of the 24th Annual International Symposium on Computer Architecture*, June 1997.

[16] S. J. Plimpton. Fast parallel algorithms for short-range molecular dynamics. *Journal Computation Physics*, 117:1–19, 1995.

[17] S. J. Plimpton. Lammps web page, July 2003. http://www.cs.sandia.gov/ sjplimp/lammps.html.

[18] S. J. Plimpton, R. Pollock, and M. Stevens. Particle-mesh ewald and rRESPA for parallel molecular dynamics. In *Proceedings of the Eighth SIAM Conference on Parallel Processing for Scientific Computing*, Minneapolis, MN, Mar. 1997.

[19] RDMA Consortium. `http://www.rdmaconsortium.org`, 2003.

[20] W. Rehm, R. Grabner, F. Mietke, T. Mehlan, and C. Siebert. An MPICH2 channel device implementation over VAPI on InfiniBand. In *Proceedings of the 2004 Workshop on Communication Architecture for Clusters*, April 2004.

[21] K. Solchenbach. Benchmarking the balance of parallel computers. In *SPEC Workshop on Benchmarking Parallel and High-Performance Computing Systems*, Wuppertal, Germany, Sept. 1999.

[22] K. D. Underwood and R. Brightwell. The impact of mpi queue usage on mpi latency. In *submitted*, 2004.