A Preliminary Analysis of the InfiniPath and XD1 Network Interfaces

Ron Brightwell, Doug Doerfler, Keith D. Underwood

Center for Computation, Computers, Information, and Math
Sandia National Laboratories*
Albuquerque, NM 87185-1110
{rbbrigh,dwdoerf,kdunder}@sandia.gov

Abstract

Two recently delivered systems have begun a new trend in cluster interconnects. Both the InfiniPath network from PathScale, Inc., and the RapidArray fabric in the XD1 system from Cray, Inc., leverage commodity network fabrics while customizing the network interface in an attempt to add value specifically for the high performance computing (HPC) cluster market. Both network interfaces are compatible with standard InfiniBand (IB) switches, but neither use the traditional programming interfaces to support MPI. Another fundamental difference between these networks and other modern network adapters is that much of the processing needed for the network protocol stack is performed on the host processor(s) rather than by the network interface itself. This approach stands in stark contrast to the current direction of most high-performance networking activities, which is to offload as much protocol processing as possible to the network interface. In this paper, we provide an initial performance comparison of the two partially custom networks (PathScale's InfiniPath and Cray's XD1) with a more commodity network (standard IB) and a more custom network (Quadrics Elan4). Our evaluation includes several micro-benchmark results as well as some initial application performance data.

1. Introduction

Two recent networks may point to the beginning of two trends in networking for high performance computing (HPC) clusters. The first is an encouraging trend: the leveraging of commodity technology while providing some hardware innovation to deliver more performance than commodity parts. The InfiniPath network adapter from PathScale, Inc., and the RapidArray fabric in the Cray XD1 both leverage a commodity network while customizing the network interface to provide additional performance for HPC. Both networks connect to the host through the HyperTransport standard interface and both networks forego the weight of a traditional IB Verbs API to provide a lighter-weight interface for MPI. This leveraging of commodity technologies while providing selective innovation reduces both the cost and time-to-market for a high performance interconnect.

The PathScale network also lays the foundation for a more contrarian trend: the movement of processing from the network interface to the host. Past research has indicated that there are significant advantages of network interface-based processing (offload[18] and independent progress[8, 7]). Recent trends in high performance networking [23, 20, 2, 5] have confirmed the perceived importance of these features; however, even with an opportunity to customize at both the hardware and API levels, the InfiniPath network interface has chosen to put the processing burden on the host processor(s).

This paper provides an initial comparison between the XD1 and InfiniPath networks and solutions at both ends of the spectrum: purely commodity InfiniBand (IB) and fully custom Quadrics Elan-4. One would expect commodity networks, such as baseline IB, to focus on priorities other than achieving the level of performance that is typically expected from an HPC cluster interconnect; thus, it would not be expected that they would perform as well as networks targeted to the HPC market. Similarly, initial expectations would be that a custom network that has matured over multiple generations, such as Elan-4, would have a significant advantage over networks based on commodity switches.

The results, however, are somewhat surprising. As expected, all three of the networks with custom network in-

*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.

1-4244-0054-6/06/$20.00

terfaces outperformed baseline IB by a factor of three in latency and an order of magnitude in MPI message rate; however, baseline IB held a noticeable advantage in some collectives at larger message sizes. Another surprise was that, despite being a fully custom network, Elan-4 has neither a latency or a bandwidth advantage. Indeed, it was at a significant disadvantage for message rate for small MPI messages. Elan-4 has numerous advantages over InfiniPath and XD1, including messaging offload and true independent progress, and was available earlier than the XD1 or InfiniPath interconnects. Elan-4 also has a particular handicap from using an older bus (PCI-X), but the results do give reason to pause and consider how much customization is needed and how much work should be handled on the NIC.

The following section provides an overview of the features of the PathScale and XD1 network interfaces. Section 3 presents an overview of the evaluation platforms and describes the benchmarks used in our analysis. Performance results are analyzed in Section 4, and the important conclusions of this study are stated in Section 6. Relevant related research is summarized in Section 5, and Section 7 describes our plans for work related to this study.

2. Background

In a previous work [4], we performed an analysis of Quadrics Elan-4 and InfiniBand with respect to various features that they provide as well as their performance. We continue the theme of that paper by extending the analysis and comparison to include the InfiniPath and Cray XD1 adapters.

2.1. Capabilities

Few details about the inner workings of the InfiniPath network interface have been published [11], but PathScale has several whitepapers [22] that describe their general philosophy and approach. The main philosophical message that differentiates this network is that functionality typically performed by the network interface has been relegated to the host processor(s). The reasoning for this approach is that the speed of protocol processing on the host is much greater than on the network interface. Another differentiating factor is that there are no transmit DMA engines on the network interface. The host processor must move data from host memory to memory on the network interface. Indeed, the bulk of the area on the network interface is high-speed memory. In order to initiate a transfer, memory on the network interface is mapped into a user-level process, which simply writes directly to those memory locations. As soon as the network interface recognizes that data is being written to NIC memory, it can begin streaming data out to the network. On the receive side, the network interface writes messages up into host memory directly and records where incoming messages have been written in a circular event queue. The destination address for an incoming message can either be explicit in the message or anonymous. In the latter case, the message is simply written into a circular buffer of incoming messages. The host is responsible for recognizing errors in incoming messages and for performing the necessary reliability and flow control functions to insure the incoming data is valid.

Even less is publicly known about the XD1 network interface, called the RapidArray interconnect (RAI). According the data sheet on the RAI from Cray [9], the network interface has processors that offload and accelerate core network functions to unburden the host processor(s) and allow for overlap of computation and communication. It is unknown how much the RAI processors differ from the ASICs used on traditional IB network cards. For small message transfers, the MPI implementation clearly performs a memory copy type of operation. For long messages, however, the RAI appears to have a transmit DMA engine, unlike the InfiniPath interface.

2.2. Programming Interface

InfiniPath and RAI both support remote DMA operations. Unlike InfiniBand and Quadrics, there is no programming interface for InfiniPath. The user application simply accesses memory locations that have been mapped into its address space in order to transfer data. This approach is much less complex. PathScale supports the OpenIB [21] programming interface, which enables porting of IB applications, but it is expected that applications requiring the highest performance, like MPI, will continue to access the network interface directly.

The programming interface for the RAI appears to be very similar to the standard IB programming interfaces, such as VAPI [19] and uDAPL [10]. Unlike many other modern networks[23, 20, 6], neither network provides an API or hardware support for offloading more complex operations, such as MPI tag matching.

2.3. Connections

Like traditional IB, the RAI is connection-oriented, requiring an explicit connection establishment phase before data transfer can begin. InfiniPath, however, is connectionless. Traditional IB adapters not only require an explicit connection, but they also require that some application memory be committed to a connection in order to transfer data. In order to support a fully connected model like MPI, this can lead to an extremely large amount of memory. Initial measurements on the Thunderbird cluster at Sandia, which is a 4096-node (8192 processor) IB cluster, indicate

that more than 1 GB of memory per node needs to be committed to a single job that spans that entire machine.

2.4. Memory Registration

Another important way in which InfiniPath differs from traditional network interfaces, and the RAI, is that it does not require memory used for data transmits to be registered, or pinned, with the network interface. Traditional network interfaces that use DMA engines to transmit data must insure that the host memory being accessed is resident in physical memory and must have a coherent mapping from virtual memory to physical memory. On the receive side, however, zero-copy techniques are used that still require registered memory. Standard IB interfaces and the RAI programming interface provide explicit memory registration operations that allow the operating system to insure memory pages used for transfers are resident and to update the interfaces virtual-to-physical mappings. These are required on both the send and receive sides. In contrast, the Quadrics network interface has an on-board memory management unit that functions much like an additional coherent processor. The network interface works with the operating system to insure that pages are resident and appropriately mapped without any explicit involvement of the application. Because the InfiniPath interface does not use DMA engines, but rather accesses interface memory via programmed I/O, there is no need to explicitly validate and map memory for transmits.

2.5. Progress, Offload, and Overlap

Our previous analysis of IB and Quadrics included a discussion of three important MPI implementation characteristics. The first of these is independent progress, which allows the transfer of data, once enabled, to complete without the application making explicit calls to MPI. Implementations of MPI for IB do not support independent progress for long messages, while Quadrics does. Neither InfiniPath nor the RAI support independent progress, since the MPI posted receive queue is kept in user-space.

A second desirable characteristic of MPI is to support offloading of some MPI functionality, such as MPI tag matching, to the network interface. Obviously the design philosophy of the InfiniPath adapter is such that it supports no offload capability whatsoever. As mentioned above, the RAI does not support any sophisticated offload functionality either. In contrast, the Quadrics Elan-4 interconnect supports full offload of MPI matching (as do other interfaces such as Myricom's MX interface[20] and the Portals interface on the Cray XT3[6]).

Lastly, the ability to overlap communication with computation is also a desirable quality for an MPI implementation. Again, the approach of InfiniPath, where the host processor is responsible for moving all data, limits the amount of overlap that can be achieved. Because the host side of the interface is faster than the network side, for relatively small network transfers, the host can complete its work (copying data to the network interface) before the network has transferred all of the data. This only allows a very small amount of overlap. The RAI does support overlap via its native programming interface. However, like all RDMA interfaces, overlap for MPI is severely hampered for long messages, because code in the MPI library at the receiver must be invoked before the final destination of the data can be determined. On traditional IB, InfiniPath and RAI implementations, this requires the application to participate by entering the MPI library at least once after the communication has been enabled in order for the data to move.

3. Platforms and Benchmarks

Table 1 shows the specifications for each of the machines used in our evaluation. The Emerald cluster is a 144-node test machine administered by AMD. The Cray XD1 platform is administered by Oak Ridge National Laboratory.

We chose several different micro-benchmarks and one application for our evaluation. We used several point-to-point and collective micro-benchmarks from the Pallas MPI benchmark suite [1] version 2.2.1, as well as a streaming bandwidth micro-benchmark to compare raw interconnect performance. The streaming bandwidth test initiates several send operations in sequence and then waits for a single response message to return when all messages have arrived at the receiver. The results used in our analysis are from initiating 160 send operations in sequence. The streaming bandwidth results can also be converted to an MPI message rate by dividing the data rate by the message size.

We used the polling method in the COMB [14] benchmark suite to measure CPU availability. This method uses a ping-pong communication strategy with messages flowing in both directions. Each process polls waiting for messages to arrive and propogates replacement messages. After a predetermined amount of computation, bandwidth and CPU availability are computed. The polling interval can be adjusted to demonstrate the tradeoff between bandwidth and CPU availability.

We used the LAMMPS [24] molecular dynamics simulation code as our application benchmark. This is one of the few Sandia applications that is not subject to export control restrictions and can be easily evaluated on non-Sandia systems. Two implementations of LAMMPS exist – one written in Fortran90 and one written in C++. For our analysis, we chose two different problem sets for LAMMPS. We chose the "Stouch" and "LJ" problem sets because they provide a good balance of computation and communication

Table 1. Overview of Test Platforms

	Emerald	Red Squall	Thunderbird	Cray XD1
Interconnect	4x InfiniPath	Elan-4	4x InfiniBand	Dual 4x InfiniBand
Host Interface	HyperTransport	PCI-X	x8 PCI-Express	HyperTransport
Peak Link BW	2 GB/s	2.133 GB/s	2 GB/s	4 GB/s
Host Interface Peak BW	6.4 GB/s	1.064 GB/s	4 GB/s	3.2 GB/s
Host Processor(s)	quad 2.2 GHz Opteron	dual 2.2 GHz Opteron	dual 3.4 GHz Xeon EM64T	dual 2.2 GHz Opteron
Memory Speed	dual DDR-400	dual DDR-333	dual DDR-400	dual DDR-400
Operating System	Linux (RedHat EL-4)	Linux (SUSE 9.1 Pro)	Linux (SUSE 9.1 Pro)	Linux (SUSE SLES 9)
Compilers	PathScale 2.2	PathScale 2.1	PathScale 2.1	PGI 6.0.5
MPI Software	InfiniPath 1.1	MPICH QsNet 1.24-43	MVAPICH 0.92	MPICH 1.2.6
Nodes	144	256	4096	72

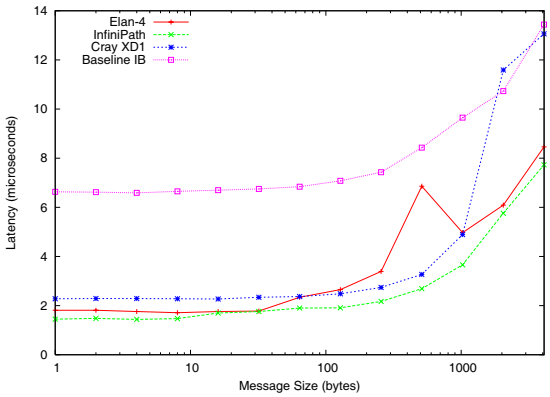

Figure 1. Ping-pong latency

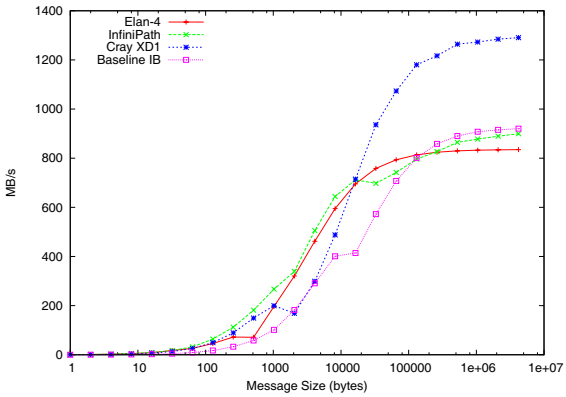

Figure 2. Ping-pong bandwidth

and, as such, are good at exposing scalability issues associated with network performance.

4. Results

The traditional first point of comparison for network evaluation has been ping-pong latency and bandwidth. Figure 1 compares the ping-pong latency for the four networks being evaluated. For all practical purposes, Quadrics Elan-4, PathScale's InfiniPath, and Cray's XD1 deliver identical short message latencies. The slight apparent advantages for InfiniPath can be contributed almost exclusively to the faster processors used in that system. However, the difference between the baseline IB results and the two custom network interfaces using the same switch fabric highlight the gains to be had by customizing both the network interface and the API layer.

Moving to the ping-pong bandwidth results in Figure 2 shows a similar picture with the custom network interfaces having a significantly faster bandwidth ramp versus message size. The Cray XD1 (the first network delivering an integrated dual-rail IB solution) shows a significant advantage over the other networks as its peak bandwidth reaches 1.3 GB/s, which is over 35% better than the other systems. This advantage is actually limited by the speed of the HyperTransport (HT) interface on the FPGAs used for the XD1 network, which have a peak unidirectional bandwidth of 1.6 GB/s. Another notable feature of the graph in Figure 2 is the relative smoothness of the Elan-4 performance. The Elan-4 stack has been in production significantly longer than the other networks; thus, it has a somewhat better tuned software stack. In each of the networks, we can see clear protocol switches as hitches in the graph. These are more severe for both the XD1 and InfiniPath networks.

The bi-directional bandwidth results in Figure 3 highlight the advantages of a modern interface between the NIC and the host processor. Elan-4 is clearly limited by the PCI-X bus it uses. The XD1 is clearly the winner in total bandwidth with a peak over 2 GB/s. This illustrates both the advantages of having two IB links and the disadvantages of using the slower HT interface provided by an FPGA. Interestingly, both InfiniPath and the XD1 appear to have protocol switches that create a dip in the bi-directional bandwidth. For the XD1, this is likely to be a switch from an

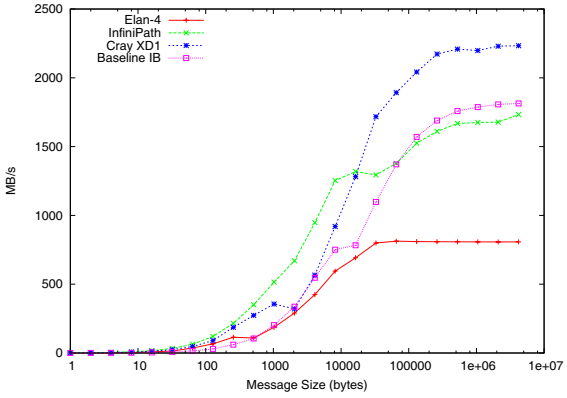

Figure 3. Send-receive bandwidth

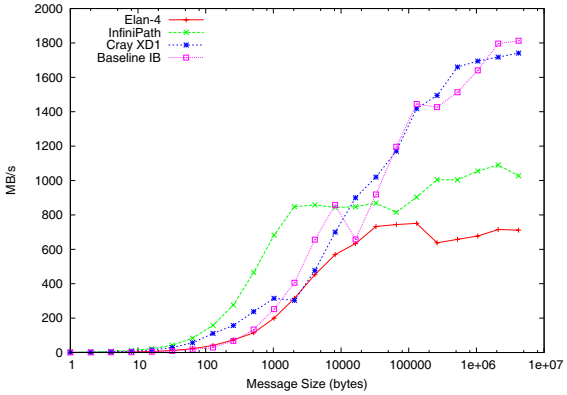

Figure 4. 32-Node exchange bandwidth

eager send using programmed I/O to a rendezvous send using the DMA engine. It is less clear what this change might be for InfiniPath (the curve shape suggests it might be more than just a poorly tuned switch from eager to rendezvous sends).

The other Pallas benchmarks were run at 32 nodes. The Pallas exchange benchmark yielded somewhat surprising results. While the curves for XD1, generic IB, and Elan-4 all follow the trends expected, InfiniPath suffers from significantly lower performance at larger message sizes. This could be caused by an interplay between the demands on the processor and the use of non-blocking sends in the Exchange benchmark; however, discussions with Pathscale suggest that it is simply a software bug.

Broadcast, Allreduce, and Alltoall performance (Figures 5, 6, and 7) behave generally as expected. The only notable exceptions are that Elan-4 has a noticeably steeper curve for Allreduce and the XD1 has a steeper slope for Alltoall. This differs from the earlier results at 16 nodes (on potentially older software) that implied that baseline IB

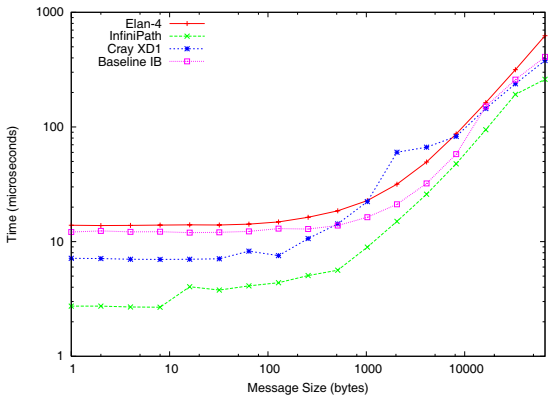

Figure 5. 32-Node broadcast performance

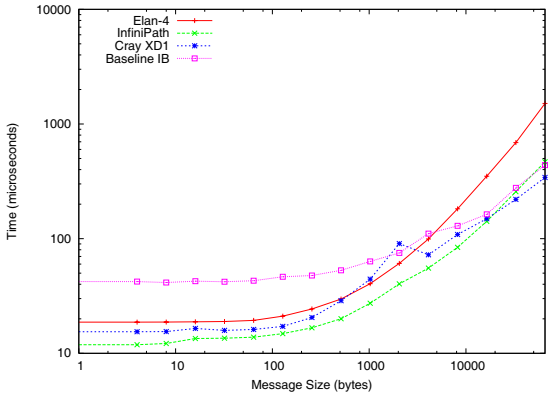

Figure 6. 32-Node allreduce performance

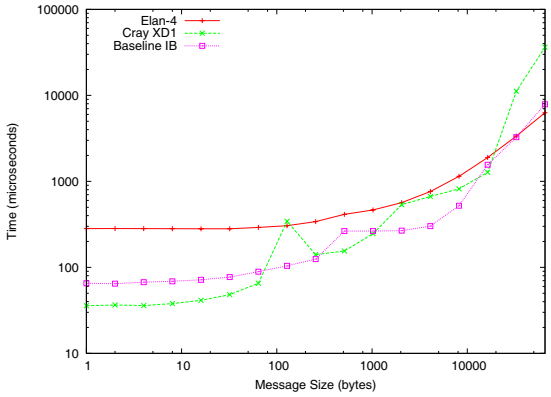

Figure 7. 32-Node alltoall performance

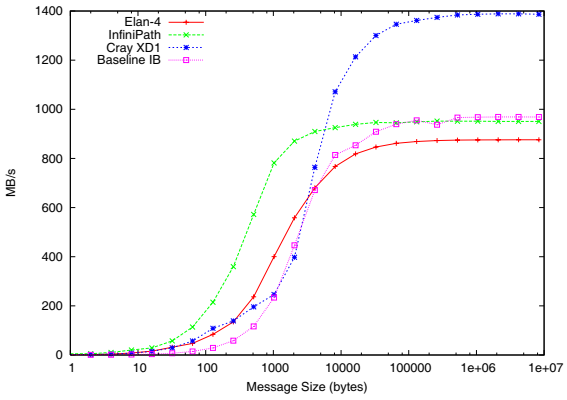

Figure 8. Streaming bandwidth

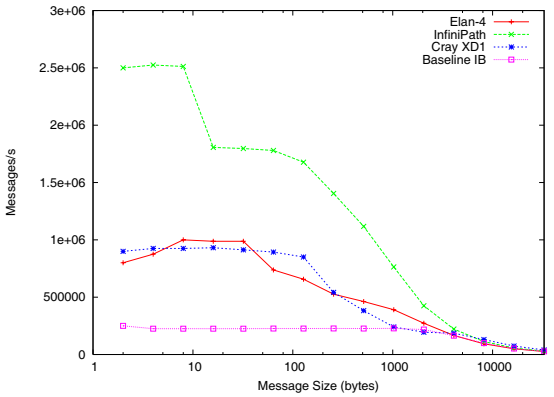

Figure 9. Message rate

may be deriving additional benefits from the optimized collectives in the MVAPICH library[13, 17, 25] that have not (at the time) been propagated to other networks.

The final network micro-benchmark examines the rate at which the network can pass MPI messages. The traditional metric of this is "streaming bandwidth" as shown in Figure 8. InfiniPath has a somewhat higher message rate than any of the other networks as seen both in the bandwidth and the message rate (Figure 9). Here, message rate was obtained by dividing the streaming bandwidth by the message size. The Elan-4 and XD1 networks outperform the generic IB network by a wide margin, but still lose to the InfiniPath adapter by over a factor of 2. In fairness, however, the Elan-4 network interface predates the higher speed, lower latency HyperTransport and PCI-Express interfaces. Thus, it may improve significantly when it moves to a newer interface. The XD1 data represents a significant drop in throughput from earlier measurements on a platform with a slower processor.

In stark contrast to the InfiniPath and XD1 networks, all

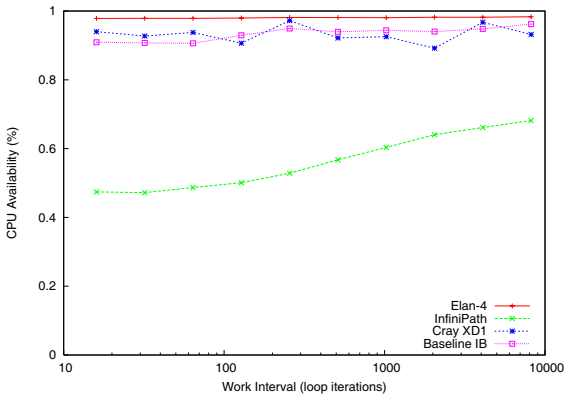

Figure 10. CPU availability (100KB messages)

of the match processing and progress engine lie on the Elan-4 NIC. The use of an embedded processor for such things as MPI match processing may be a significant contributor to the slightly slower message rate on the Elan-4 relative to InfiniPath. That difference, however, brings unique capabilities to Elan-4 (offload and independent progress) that InfiniPath and XD1 cannot match.

Figure 10 shows CPU availability results for 100 KB messages using the polling method of the COMB benchmark suite. This graph clearly shows the benefit of offloading a significant amount of processing to the network interface. The Elan-4 system is able to provide nearly all of the available host processor cycles to computation. The Cray XD1 and IB systems are also able to leave a large percentage of cycles available for computation, since they are able to leverage DMA engines on the network interface for moving data to and from the network. The impact of using host processor cycles to move data is clearly evident for the InfiniPath system, which initially uses nearly half of its available cycles for network processing.

Figure 11 presents scaling efficiency data from some preliminary runs of the LAMMPS molecular dynamics application out to 128 nodes using both one process per node (1ppn) and two processes per node (2ppn) with a scaled-size problem. The three platforms tested all scale reasonably well. For the 1ppn Fortran90 version of the code, the baseline IB system and the Elan-4 system scale slightly better than the InfiniPath system, which begins to decline at eight nodes. For the runs, the efficiency of the three systems is very similar, with the efficiency of the baseline IB system being slightly less than the other two systems.

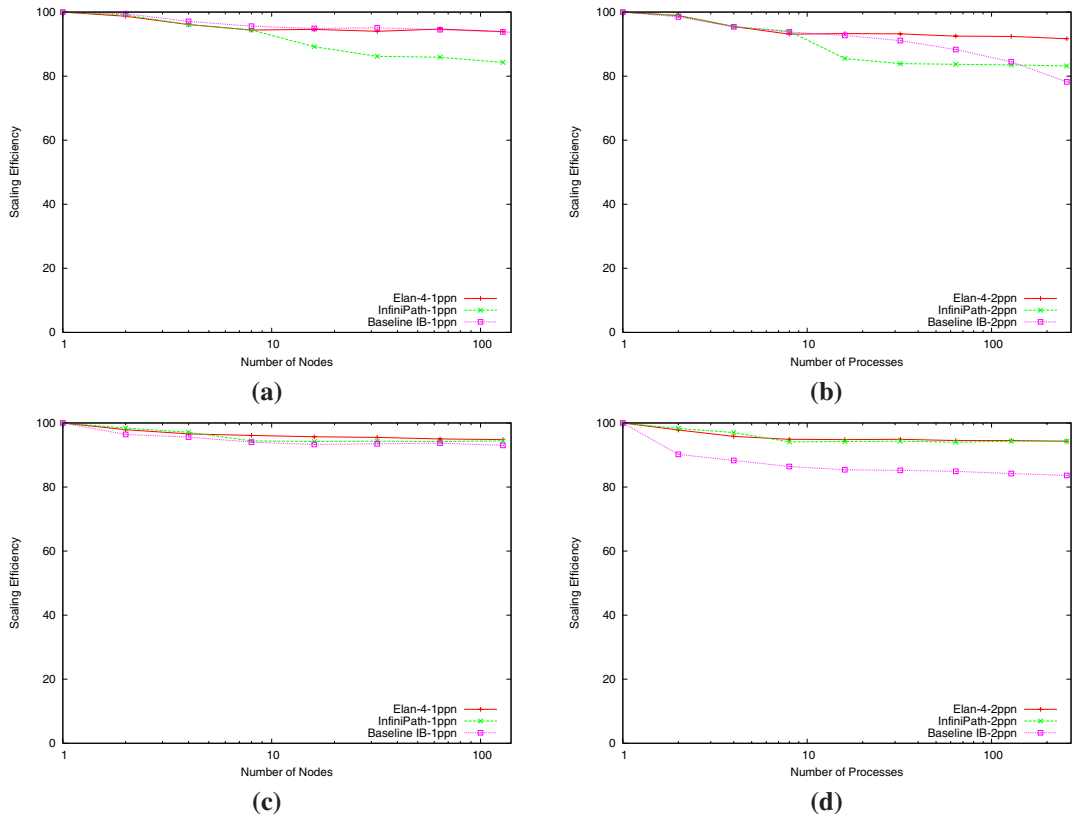

Figure 11. Efficiency for LAMMPS 2001 ((a) and (b)) LAMMPS 2005 ((c) and (d)

5. Related Work

As new networks are become available, it has been common for independent parties to benchmark them for the community. This usually begins with microbenchmarks and some preliminary application analysis. Examples of such work include a comparison of InfiniBand, Myrinet, and Quadrics Elan-3 platforms at the lowest level API [16] and at the MPI level [15]. Similar work has been performed for the Elan-4[4] and Cray XT3 networks[6, 3, 5]. While an early evaluation of the Cray XD1 has been presented[12], this paper broadens the set of network specific benchmarks evaluated and adds the important context of other high performance cluster interconnects. In addition, this paper adds an evaluation of the PathScale InfiniPath network, which, to our knowledge, has not been presented before.

6. Conclusion

While the software stacks have room to mature, both the InfiniPath and the XD1 interconnects demonstrate remarkably good performance. Both networks succeed in delivering comparable latency to a fully custom interconnect and dramatically better latency than the pure commodity interface. These networks are also able to deliver significantly better message rates than the Elan4 network and an order of magnitude better message rates than traditional InfiniBand. They achieve this by adding innovation to the network interface while relying on commodity components to provide network level signaling and switching. In the process of developing a new NIC and new API, both the InfiniPath and XD1 interconnects made an unusual choice to push functionality back to the host. The InfiniPath interconnect even goes to the extreme of eliminating transmit DMA functionality and replacing it with host processing. With all of network research trending toward more functionality on the NIC, it is questionable whether this is the right choice; however, many microbenchmarks certainly indicate that this is a win.

7. Future Work

Microbenchmarks are only part of the overall picture. Application performance and scalability have been shown to be affected by factors such as offload and independent progress, which neither of these new networks can provide.

Future work will focus on evaluating a broader group of applications to understand whether factors the much better message rates available with InfiniPath and XD1 can compensate for the lack of features provided by traditional high performance network interfaces.

8. Acknowledgments

The authors would like to express their gratitude to Greg Lindahl of PathScale for his helpful comments regarding the InfiniPath adapter. We would also like acknowledge the AMD Development Center (`http://devcenter.amd.com`) for providing access to their Emerald InfiniPath machine. The XD1 machine is a resource of the National Center for Computational Sciences at Oak Ridge National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC05-00OR22725.

References

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

[2] R. Alverson. Red Storm. In *Invited Talk, Hot Chips 15*, August 2003.

[3] R. Brightwell. A comparison of three MPI implementations for Red Storm. In B. D. Martino, D. Kranzlmuller, and J. Dongarra, editors, *Recent Advances in Parallel Virtual Machine and Message Passing Interface: 12th European PVM/MPI Users' Group Meeting, Sorrento, Italy, September 2005 Proceedings*, volume 3666 of *Lecture Notes in Computer Science*, pages 425–432. Springer-Verlag, 2005.

[4] R. Brightwell, D. Doerfler, and K. D. Underwood. A comparison of 4x InfiniBand and Quadrics Elan-4 technologies. In *Proceedings of the 2004 International Conference on Cluster Computing*, September 2004.

[5] R. Brightwell, T. Hudson, K. Pedretti, R. Riesen, and K. Underwood. Implementation and performance of Portals 3.3 on the Cray XT3. In *Proceedings of the 2005 IEEE International Conference on Cluster Computing*, September 2005.

[6] R. Brightwell, K. Pedretti, and K. Underwood. Initial performance evaluation of the Cray SeaStar interconnect. In *Proceedings of the 13th IEEE Symposium on High-Performance Interconnects*, August 2005.

[7] 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.

[8] R. Brightwell and K. D. Underwood. An initial analysis of the impact of overlap and independent progress for mpi. In *Recent Advances in Parallel Virtual Machine and Message Passing Interface: 9th European PVM/MPI Users' Group Meeting*, Budapest, Hungary, Sept. 2004.

[9] Cray, Inc. *http://http://www.cray.com/products/xd1/*, 2004.

[10] DAT Collaborative. *uDAPL: User Direct Access Programming Library*, May 2003.

[11] L. Dickman, G. Lindahl, D. Olson, J. Rubin, and J. Broughton. PathScale InfiniPath: A first look. In *Proceedings of the 13th Symposium on High Performance Interconnects (HOTI'05)*, August 2005.

[12] M. R. Fahey, S. Alam, J. Thomas H. Dunigan, J. S. Vetter, and P. H. Worley. Early evaluation of the Cray XD1. In *Cray User Group Annual Technical Conference*, May 2005.

[13] R. Gupta, P. Balaji, D. K. Panda, and J. Nieplocha. Efficient collective operations using remote memory operations on VIA-based clusters. In *Proceedings of the International Parallel and Distributed Processing Symposium*, April 2003.

[14] W. Lawry, C. Wilson, A. B. Maccabe, and R. Brightwell. COMB: A portable benchmark suite for assessing MPI overlap. In *IEEE International Conference on Cluster Computing*, September 2002. Poster paper.

[15] 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.

[16] 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.

[17] A. Mamidala, J. Liu, and D. K. Panda. Efficient barrier and allreduce on IBA clusters using hardware multicast and adaptive algorithms. In *Proceedings of the 2004 IEEE International Conference on Cluster Computing*, September 2004.

[18] 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.

[19] Mellanox Technologies. *Mellanox IB-Verbs API (VAPI): Mellanox Software Programmer's Interface for InfiniBand Verbs*, 2001.

[20] Myricom, Inc. Myrinet Express (MX): A high performance, low-level, message-passing interface for Myrinet, July 2003.

[21] Open InfiniBand Alliance. *http://www.openib.org*, 2004.

[22] PathScale, Inc. *http://pathscale.com/whitepapers.html*, January 2006.

[23] F. Petrini, W. chun Feng, A. Hoisie, S. Coll, and E. Frachtenberg. The Quadrics network: High-performance clustering technology. *IEEE Micro*, 22(1):46–57, January/February 2002.

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

[25] S. Sur, H.-W. Jin, and D. K. Panda. Efficient and scalable all-to-all exchange for infiniband-based clusters. In *Proceedings of the International Conference on Parallel Processing (ICPP)*, Montreal, Canada, August 2004.