A Portable InfiniBand Module for MPICH2/Nemesis: Design and Evaluation [*]

Miao Luo, Ping Lai, Sreeram Potluri, Emilio P. Mancini,
Hari Subramoni, Krishna Kandalla, Dhabaleswar K. Panda

Department of Computer Science and Engineering, The Ohio State University

{luom, laipi, potluri, mancini, subramon, kandalla, panda}@cse.ohio-state.edu

Abstract

With the emergence of multi-core-based processors, it is becoming significantly important to optimize both intra-node and inter-node communication in an MPI stack. MPICH2 group has recently introduced a new Nemesis-based MPI stack which provides highly optimized design for intra-node communication. It also provides modular design for different inter-node networks. Currently, the MPICH2/Nemesis stack has support for TCP/IP and Myrinet only. The TCP/IP interface allows this stack to run on the emerging InfiniBand network with IPoIB support. However, this approach does not deliver good performance and can not exploit the novel mechanisms and features provided by InfiniBand.

In this paper, we take on the challenge of designing a portable InfiniBand network module (IB-netmod) for Nemesis. The IB-netmod is designed over the Verbs-level interface of InfiniBand and can take advantage of all features and mechanisms of InfiniBand. A complete design of the IB-netmod with the associated challenges are presented. A comprehensive performance evaluation (micro-benchmarks, collectives and applications) of the new Nemesis-IB design is carried out against the Nemesis TCP/IP (with IPoIB support on InfiniBand) and the native IB support of the MVAPICH2 stack. The new IB-netmod is able to deliver comparable performance to that of the native IB support of MVAPICH2. Compared to the MPICH2/IPoIB support for InfiniBand, the new design is able to deliver significant performance benefits. For NAMD application with 256 cores, the new IB-netmod is able to deliver 4% improvement compared to the latest MVAPICH2 release. To the best of our knowledge, this is the first IB-netmod design for the MPICH2/Nemesis framework. The next release of MVAPICH2 will be having this new IB-netmod support.

1 Introduction

Rapid growth in the need for high performance computing has resulted in the deployment of increasing numbers of super computing installations around the world. Commodity clusters, built on top of multi-core computing platforms and high performance networking interconnects, are increasingly used for the newer installations. As high performance interconnects such as InfiniBand [10] and 10-Gigabit Ethernet [9] aim at improving the latency and bandwidth between the nodes, multi-core processors are also being developed to increase the computing power inside a node. In

[*] This research is supported in part by U.S. Department of Energy grants #DE-FC02-06ER25749 and #DE-FC02-06ER25755; National Science Foundation grants #CNS-0403342, #CCF-0702675, #CCF-0833169, #CCF-0916302 and #OCI-0926691; grant from Wright Center for Innovation #WCI04-010-OSU-0; grants from Intel, Mellanox, Cisco, QLogic, and Sun Microsystems; Equipment donations from Intel, Mellanox, AMD, Advanced Clustering, Appro, QLogic, and Sun Microsystems.

this scenario, a software stack that can utilize and maximize the benefits of both high performance interconnects and multi-core systems is an urgent demand for HPC application designers.

The Message Passing Interface (MPI) is the dominant parallel programming model for cluster computing. MPICH2 [15], a very popular implementation of the MPI-2 standard, has recently been re-designed to run over the Nemesis communication framework. The main goals of Nemesis design are scalability, high-performance intra-node communication, high-performance inter-node communication, and multi-network inter-node communication etc. As shown from the ranking the goals, Nemesis designers strive to minimize the overhead for intra-node communication, even if this comes at some penalty for inter-node communication [8, 7].

The Nemesis communication framework relies on various network modules (netmod's) to achieve multi-network inter-node communication. Currently, Nemesis only has two network moudles: TCP/IP and Myrinet. As InifiBand is emerging as a popular cluster interconnect, it is also important to support Nemesis on it. Whereas using the current network modules, it can only run TCP/IP netmod on top of IPoIB which does not perform well. Therefore, it is necessary for us to design and implement a new high performance *netmod* based on IB. This leads to the following problems:

- Can a high performance and scalable netmod be designed with InfiniBand Verbs-level interface under the MPICH2/Nemesis framework?
- How much performance gain can be achieved by the new netmod compared to the current TCP/IP netmod to work for InfiniBand with IPoIB?
- Can the performance and scalability of the new netmod be equal or better to those of the non-Nemesis-based designs for InfiniBand at the Verbs layer?

In this paper, we take on this challenge to design and implement a new InfiniBand (IB) netmod to support Nemesis communication framework. We design several schemes according to the new structure provided by Nemesis, as well as incorporating previous techniques that have been proved beneficial for MPI applications over InfiniBand into this new netmod. We also evaluate the performance of new IB netmod for Nemesis using standard MPI benchmarks by comparing it with MPICH2 1.2 Nemesis over TCP/IP and MVAPICH2 1.4 (which is the latest release of MVAPICH2) over IB. From the comparison, a lower latency intra-node communication is observed in the new netmod, while the performance of InfiniBand inter-node communication is successfully kept. Application evaluation shows almost the same performance between the new netmod and MVAPICH2 1.4 for NAS benchmarks with 64 processes. Another application benchmark, NAMD, is also used for comparison with up to 256 processes. And the same or even slightly better performance of the new IB netmod also indicate a potential good scalability.

The remaining parts of the paper is organized as follow: In Section 2, we provide an introduction to the necessary background of InfiniBand and MPICH2 Nemesis design; In Section 3, we describe the new design and utilized optimization techniques in the new IB netmod for nemesis; Section 4 and Section 5 give the microbenchmarks and application level evaluation on the new IB netmod; Related works are introduced in Section 6; We conclude and point out future work in Section 7.

2 Background

2.1 InfiniBand

The InfiniBand Architecture is a switched fabric that designed for interconnecting processing nodes and I/O nodes [10]. In recent years, it has been primarily used in high-performance computing area, as a high-speed, general-purpose I/O interconnect to connect commodity machines in large clusters. There are two sets of communication semantics of InfiniBand: channel and memory semantics. Channel semantics include send and receive operations that are common in traditional interfaces, such as sockets, where both sides of sender and receiver must be aware of communication. Memory semantics include one-sided operations, which are referred as Remote Direct Memory Access (RDMA). These operations can allow a host to access memory from a remote node without a posted receive. Both of the two semantics require communication memory to be registered with InfiniBand hardware and pinned in memory.

In order to support popular TCP/IP network protocol stack, the IP over IB (IPoIB) is defined by the IETF Internet Area IPoIB working group [1]. IPoIB can provide standardized IP encapsulation over InfiniBand fabrics. When IPoIB is applied, an InniBand device is assigned an IP address and accessed just like any regular TCP/IP hardware device.

2.2 MPICH2 Nemesis overview

MPICH is a freely available, complete implementation of the MPI specification developed by ANL, which provides support for the MPI-1 standard. As one of the most popular MPI implementation, it's designed to be portable and efficient. After MPI-2 standard released by MPI Forum, MPICH2 was started as a successor of MPICH, aiming to support not only MPI-1 standard, but also new features in MPI-2 standard, such as dynamic process management, one-sided communication and MPI I/O. Figure 1 shows the implementation structure of MPICH2. The ADI3 (the third generation of the Abstract Device Interface) provides a portability layer for MPICH2 to connect the application layer above it and the device layer below it. CH3 is a layer that implements the ADI3 functions, and provides an interface consisting of only a dozen functions. A "channel" implements the CH3 interface. Channels exist for different communication architectures such as sockets, shmem and so on. By implementing a channel, a new communication subsystems can be ported for MPICH2, with implementing fewer functions than in ADI3 interface. As showing in Figure 1, Nemesis is ported to MPICH2 by implementing as a CH3 channel for current design. Nemesis communication subsystem was designed to provide scalability, high-performance intra-node communication, high-performance inter-node communication, and multi-network inter-node communication for MPICH2, with features listed in order of priority.

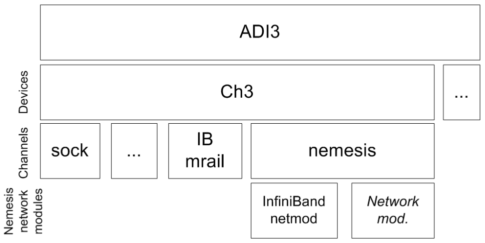

Figure 1. The Nemesis architecture

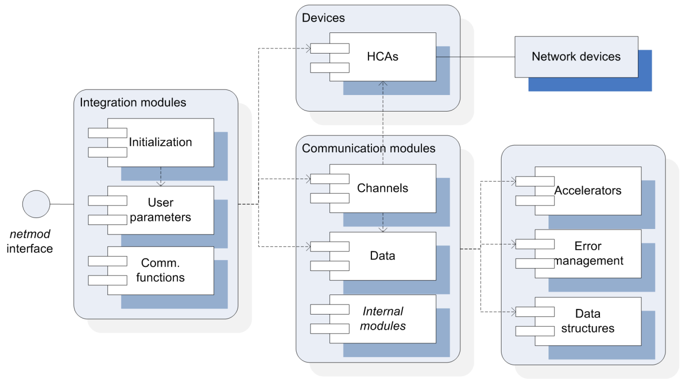

Figure 2. The main modules of InfiniBand network layer.

Initially Nemesis uses lock-free queues in shared memory to build the basis for intra-node communication [7, 8]. In addition to this, Nemesis further introduces a kernel-based mechanism KNEM specially optimized for intra-node large message transfer [6].

2.3 MVAPICH2

MVAPICH2 [16] is a popular open-source MPI implementation over InfiniBand. It is based on MPICH2 and MVICH [3]. The latest release is MVAPICH2 1.4. To implement IB netmod for Nemesis, we merged latest MVAPICH2 with MPICH2 1.2 and incorporate IB netmod into the new version of MVAPICH2. In the following subsections, MVAPICH2 nemesis-ib refers to the new IB netmod for Nemesis design. And we also choose MVAPICH2 1.4 as a reference, in order to compare the performance of the new IB netmod.

3 Designing IB Support for Nemesis

Nemesis uses internal network modules (netmod) to access to the network devices. All the network modules connect with Nemesis channel through specific interface functions [2]. Following this specification, we designed of a new module to use the InfiniBand advanced functionalities for the inter-node communication. As shown in Figure 2, the Infiniband network unit is modularized in order to decouple it from the upper layers. It interfaces the Nemesis components only through the integration modules. These depend on communication modules and then on HCAs management components. The effort of decoupling the single functional components has also the goal to make the netmod independent from the MPI middleware, to improve the reusability.

3.1 Credit-based InfiniBand Netmod Header

type	remote_credits rdma_credits	vc_addr	packet from Nemesis layer

Figure 3. IB netmod header structure.

The InfiniBand network use buffers on each pair of processes to receive incoming packets. To avoid the error raised up due to no free buffer, IB netmod must maintain a flow-control mechanism. Our design realizes this by a credit-based structure kept between each pair of sender and receiver. The credit information, which represents local and remote buffer availability, is attached to each packet and thus being transferred along data message. As the packet type passed from upper Nemesis layer is fixed, the IB netmod generate a new header for the transfer of credits the message delivery.

3.2 Eager Poking

In Nemesis design, as MPI_Isend function returns a request without doing any poking, the performance of a high performance network such as InfiniBand is limited when multiple MPI_Isend functions are called before MPI_Wait. Even a credit-update piggybacking message has been received at the network of sender side, since no poking occurs, the credits and buffers on sender side can only be consumed without refreshing/release. After credits being used up, new posted messages can only be stored in backlog and sent out later when

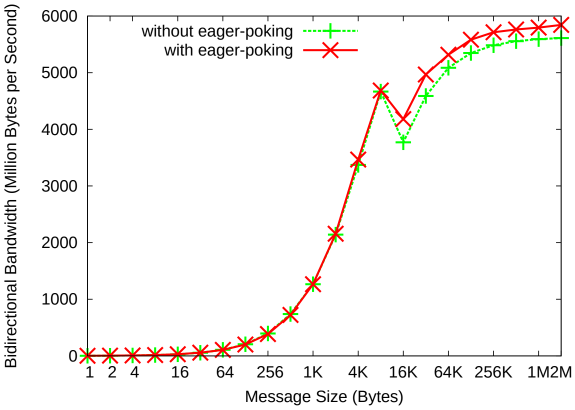

Figure 4. The Effect of Eager Poking

credits are readjusted, that is, in MPI_Wait function. To avoid a congestion of message and utilize the network efficiently, an "eager poking" is added at the end of send interface functions: iSendContig and iStartContigMsg. By eager poking, both credits and buffers information can be updated promptly, which helps the bandwidth of underlying network being utilized more efficiently. Figure 4 shows the effect of eager_poking on bi-directional bandwidth for inter-node communication.

3.3 Other Optimization Techniques

In the Nemesis over IB netmod implementation, we also utilized some designs that have been proved beneficial to inter-node communication on high-performance networks. These include RDMA fast path, header caching and Shared Receive Queue (SRQ). More details on these optimization can be found in [12], [13] and [17].

4 Micro-benchmarks evaluation

4.1 Experimental Testbed

Multiple clusters were used for our experimental evaluation. Cluster A consists of 4 Intel Nehalem machines equipped with ConnectX QDR HCAs. Each node has 8 Intel Xeon 5500 processors organized into two sockets of four cores each clocked at 2.40 GHz with 12 GB of main memory. Cluster B consists of 32 Intel Clovertown based systems equipped with ConnectX DDR HCAs. Each node in this cluster is has 8 Intel Xeon processors each organized into two sockets of four cores each clocked at 2.33 GHz with 6 GB of main memory. RedHat Enterprise Linux Server 5 was used on all machines along with OFED version 1.4.2. Cluster A was used for all the micro-benchmark level experiments, while Cluster B was used for the IBM and application level evaualtion.

4.2 Intra-node Communication Performance

Before looking into the performance of inter-node communication, we want first show the performance improvement of intra-node communication brought by Nemesis subsystem.

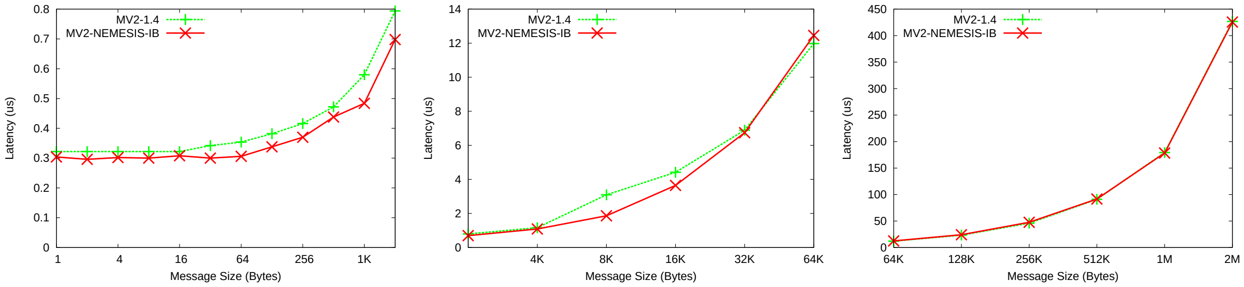

Figure 5. Intra-node latency: (a) small messages, (b) medium messages, and (c) large messages

Figure 5 shows the intra-node latency performance comparison between MVAPICH2 1.4 and MVAPICH2 nemesis-ib, which are represented by MV2-1.4 and MV2-NEMESIS-IB in the figures. From Figure 5 (a), we notice that the intra-node communication design helps reduce the latency for small message by 30 ns. For message size of 8k, the improvement reaches nearly 40%, due to different eager threshold in Nemesis and MVAPICH2 intra-node communication. This latency result and later bandwidth result show that for middle size message less than 64K, shared memory performs better than kernel-based message transfer mechanism. Figure 6 shows the performance difference of bandwidth and bi-directional bandwidth between MVAPICH2 1.4

and MVAPICH2 nemesis-ib. From Figure 6(a), the bandwidth of large messages in MVAPICH2 nemesis-ib performs slightly better than in MVAPICH2 1.4, which is introduced by the difference mechanism of KNEM and LiMIC2 design. However, when looking into the bi-directional bandwidth in Figure 6(b), a drop starting from message size of 32k shows that MVAPICH2 nemesis-ib performs even worse between message 32k to 256k. The reason should be due to an un-smoothly change from shared memory transfer to KNEM mechanism.

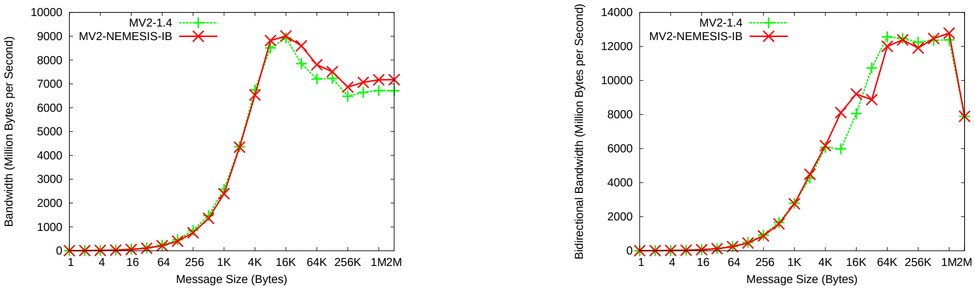

Figure 6. Intra-node (a) bandwidth and (b) bi-directional bandwidth performance

4.3 Inter-node Communication Performance

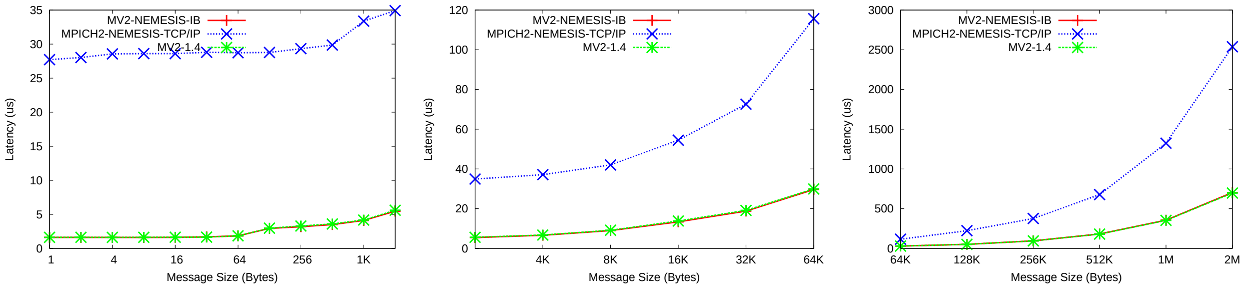

Figure 7. Inter-node latency: (a) small messages, (b) medium messages and, (c) large messages

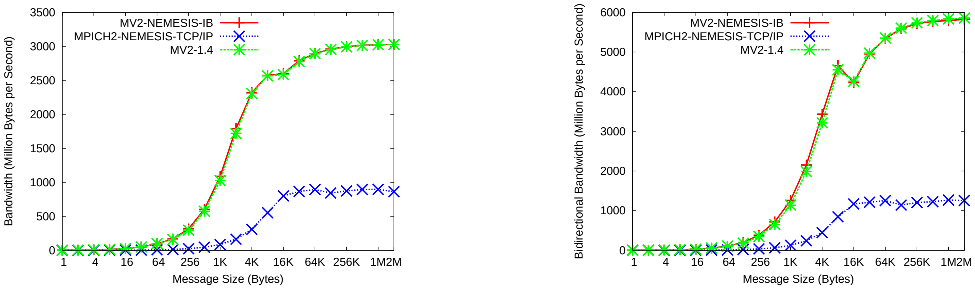

Figure 8. Inter-node (a) bandwidth and (b) bi-directional bandwidth performance

In Figure 7 and 8, we compare the inter-node performance between MPICH2 1.2 over IPoIB, MVAPICH2 Nemesis-IB and MVAPICH2 1.4, which are represented by MPICH2-NEMESIS-TCP/IP, MV2-NEMESIS-IB and MV2-1.4, respectively. As introduced in Section 2, IPoIB allows MPICH2 1.2 using TCP/IP netmod running over InfiniBand fabrics. From the results, it can be observed that new IB netmod helps reduces the short message latency of Nemesis from 27ms by using TCP/IP through IPoIB to 1.5ms by using InfiniBand, as while increases the large message bandwidth from less than 1000MB/s to more than 3000MB/s. It clearly shows that the IB netmod efficiently utilizes the high performance feature of InfiniBand network.

And when we compare the performance between MVAPICH2 Nemesis-IB and MVAPICH2 1.4, the very similar results indicate that the IB netmod for Nemesis has not only incorporated Nemesis

Subsystem's intra-node design, but also successfully remains InfiniBand's original excellent performance by utilizing various optimization techniques. For midium-sized messages, the new IB netmod even performs slightly better, due to small optimizations in header caching.

4.4 IMB Benchmark Performance

show results up to 64 128 256 cores

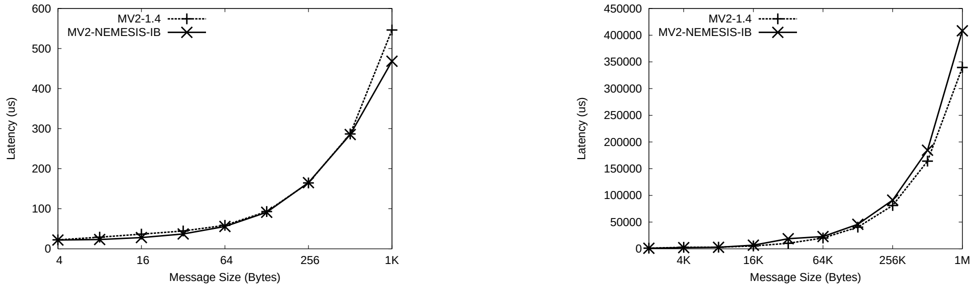

Figure 9. MPI_Allgather latency: (a) small message and (b) large message

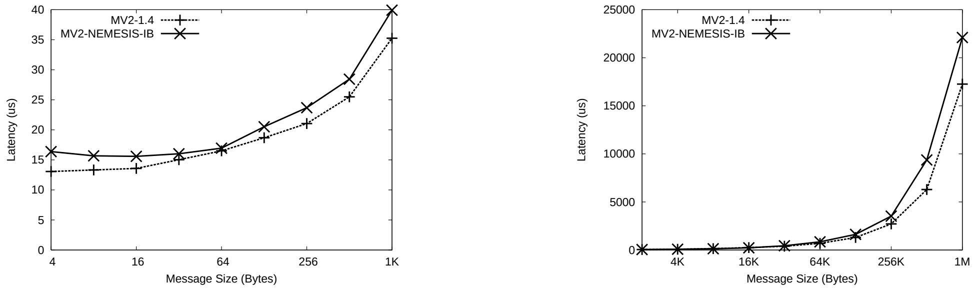

Figure 10. MPI_Allreduce latency: (a) small messages and (b) large message

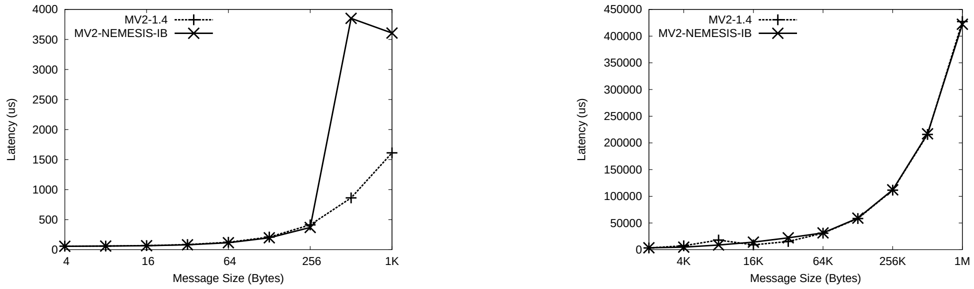

Figure 11. MPI_Alltoall latency: (a) small message and (b) large message

5 Application-level Evaluation

5.1 NAS Benchmark Performance

The NAS Parallel Benchmarks [5] are a selection of kernels that are typical in various Computational Fluid Dynamics (CFD) applications. Thus, they are widely used as a good tool to evaluate the performance of MPI library and parallel machines. Clearly, the results shown in Figure 16 exhibit that the performance of MVAPICH2 nemesis-ib and MVAPICH2 1.4 with 64 cores for various different NAS Benchmarks achieved almost the same performance.

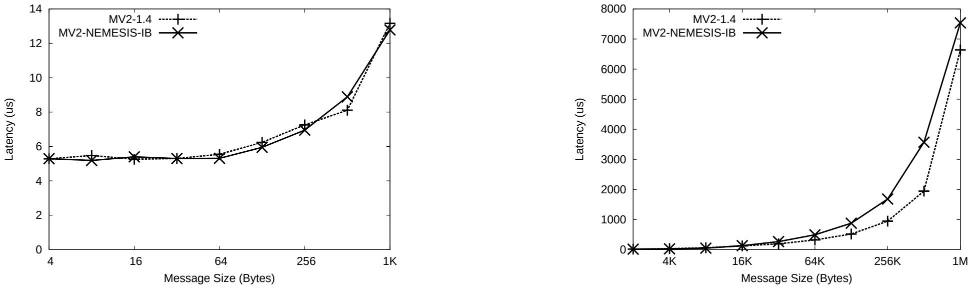

Figure 12. MPI_Bcast latency: (a) small messages and (b) large message

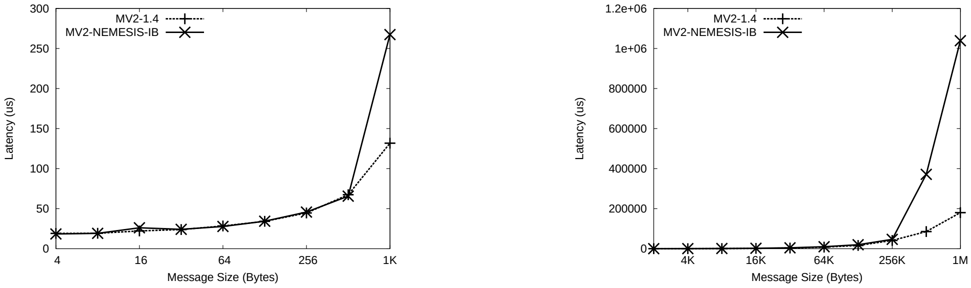

Figure 13. MPI_Gather latency: (a) small message and (b) large message

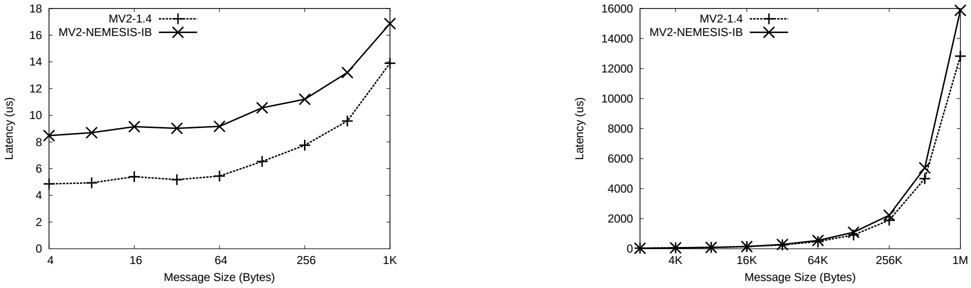

Figure 14. MPI_Reduce latency: (a) small message and (b) large message

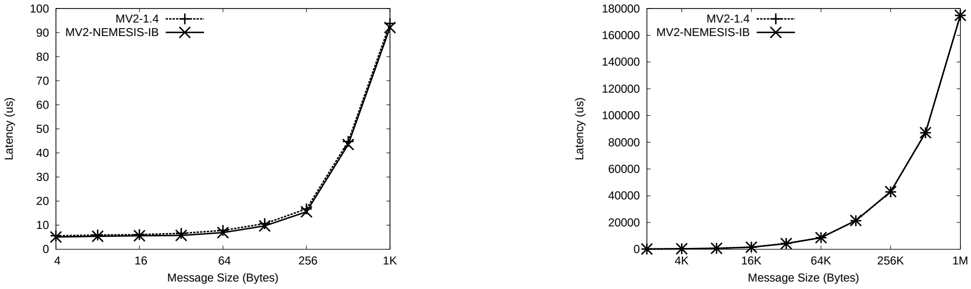

Figure 15. MPI_Scatter latency: (a) small message and (b) large message

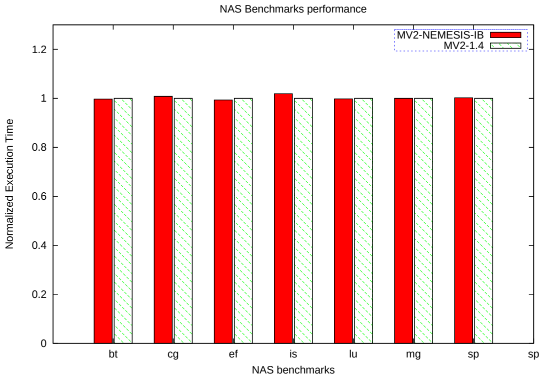

Figure 16. NAS Benchmarks

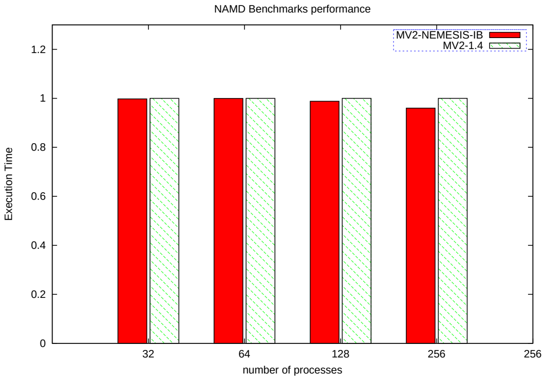

Figure 17. NAMD Benchmarks

5.2 NAMD Benchmark Performance

NAMD is a fully-featured, production molecular dynamics program for high performance simulation of large bio molecular systems [11]. NAMD is based on Charm++ parallel objects, which is a machine independent parallel programming system. There are various datasets in NAMD, we choose the apoa1 dataset, which is generally used for benchmarking. From Figure 17, we can observe that for 32 processes and 64 processes, MVAPICH2 nemesis-ib and 1.4 get the same performance. As scale increases, new IB netmod for Nemesis slightly performs better than 1.4, with 4% improvement when 256 processes are involved. The result shows that, the new IB netmod for Nemesis can achieve the same or even slightly better performance when the number of processes increases, which suggests a potentially good scalability in MVAPICH2 nemesis-ib design.

6 Related Work

Buntinas *et al.* describe the design of the MPICH2-Nemesis subsystem in [6, 7, 8]. They developed Nemesis to optimize the intra-node communication using a shared memory based approach. In this paper, we present the design of an InfiniBand network interface for Nemesis that exploits the advanced features of modern network technologies for inter-node communication. Mercier *et al.* [14] describe the integration of Nemesis into the NewMadeleine communication library. Like our design, NewMadeleine optimizes the inter-node communication by utilizing new advanced features, while focusing on different network technologies (GM/Myrinet, MX/Myrinet etc) [4].

7 Conclusion

In this paper, we implemented and evaluated a new IB netmod to support MPICH2 Nemesis over InfiniBand, in order to complement a native high-performance network support for inter-node communication to the excellent intra-node communication design of Nemesis subsystem. We adjust the structure of the new IB netmod into a more modularized way, as well as design and integrate multiple optimized techniques, in order to keep the performance of InfiniBand network while support MPICH2-Nemesis design. In the future work, we will improve the netmod with more scalability. Utilization of fine-grained multi-threading, which has just added into MPICH2-1.2.1p1 will also be included in the future design of Nemesis over IB netmod.

Software Distribution: The proposed design is planned to be included in the next MVAPICH2 [**?**] release.

References

[1] IP over InfiniBand Working Group. http://www.ietf.org/html.charters/ipoib-charter.html.

[2] MPICH2 Wiki. http://wiki.mcs.anl.gov/mpich2/index.php/Nemesis_Network_Module_API.

[3] MVICH: MPI for Virtual Interface Architecture. http://linas.org/mirrors/www.nersc.gov/2001.02.13/research/FTG/via/download_info.html.

[4] O. Aumage, E. Brunet, O. Aumage, E. Brunet, N. Furmento, and R. Namyst. Newmadeleine: a fast communication scheduling engine for high performance networks. In *in CAC 2007: Workshop on Communication Architecture for Clusters, held in conjunction with IPDPS 2007*, 2007.

[5] D. H. Bailey, E. Barszcz, J. T. Barton, D. S. Browning, R. L. Carter, D. Dagum, R. A. Fatoohi, P. O. Frederickson, T. A. Lasinski, R. S. Schreiber, H. D. Simon, V. Venkatakrishnan, and S. K. Weeratunga. The NAS Parallel Benchmarks. *The Int. J. of Supercomputer Applications*, 5(3):63–73, Fall 1991.

[6] D. Buntinas, B. Goglin, D. Goodell, G. Mercier, and S. Moreaud. Cache-Efficient, Intranode, Large-Message MPI Communication with MPICH2-Nemesis. In *Int. Conf. on Parallel Proc. (ICPP)*, 2009.

[7] D. Buntinas, G. Mercier, and W. Gropp. Design and Evaluation of Nemesis, a Scalable, Low-Latency, Message-Passing Communication Subsystem. In *Int. Symp. on Cluster Comp. and the Grid (CC-GRID'06)*, 2006.

[8] D. Buntinas, G. Mercier, and W. Gropp. Implementation and Shared-Memory Evaluation of MPICH2 over the Nemesis Communication Subsystem. In *Euro PVM/MPI 2006 Conference*, 2006.

[9] IEEE 802.3 Ethernet Working Group. IEEE 802.3. http://www.ieee802.org/3/.

[10] Infiniband Trade Association. Infiniband architecture specification release 1.2.1. http://www.infinibandta.org, January 2008.

[11] J.C.Phillips, G.Zheng, S.Kumar, and L.V.Kale. NAMD: Biomolecular Simulation on Thousands of processors. In Supercomputing, 2002.

[12] J. Liu, A. Vishnu, and D. K. Panda. Building multirail infiniband clusters: Mpi-level design and performance evaluation. *SC Conference*, 0:33, 2004.

[13] J. Liu, J. Wu, S. P. Kini, P. Wyckoff, and D. K. Panda. High Performance RDMA-Based MPI Implementation over InfiniBand. In *17th Annual ACM Int. Conf. on Supercomputing*, June 2003.

[14] G. Mercier, F. Trahay, E. Brunet, and D. Buntinas. NewMadeleine: An efficient support for high-performance networks in MPICH2. In *IEEE Int. Sym. on Parallel&Distributed Processing (IPDPS09)*, pages 1–12, Washington, DC, USA, 2009. IEEE Computer Society.

[15] MPICH2: High Performance portable MPI implementation. http://www.mcs.anl.gov/research/projects/mpich2.

[16] MVAPICH2: High Performance MPI over InfiniBand and iWARP. http://mvapich.cse.ohio-state.edu/.

[17] S. Sur, L. Chai, H.-W. Jin, and D. K. Panda. Shared receive queue based scalable mpi design for infiniband clusters. In *Int'l Parallel and Distributed Processing Symposium (IPDPS '06)*, 2006.