International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:2, No:12, 2008

A High Performance MPI for Parallel and Distributed Computing

Prabu D., Vanamala V., Sanjeeb Kumar Deka, Sridharan R., Prahlada Rao B. B., and Mohanram N.

***Abstract*—**Message Passing Interface is widely used for Parallel and Distributed Computing. MPICH and LAM are popular open source MPIs available to the parallel computing community also there are commercial MPIs, which performs better than MPICH etc. In this paper, we discuss a commercial Message Passing Interface, C-MPI (C-DAC Message Passing Interface). C-MPI is an optimized MPI for CLUMPS. It is found to be faster and more robust compared to MPICH. We have compared performance of C-MPI and MPICH on Gigabit Ethernet network.

Keywords—C-MPI, C-VIA, HPC, MPICH, P-COMS, PMB

I. Introduction

THE enormous advance in the field of PCs and high-speed networks have led to low-cost clusters of personal computers, that are able to provide the computing power equivalent to that of supercomputers. According to Moore Law, the processor speed doubles every 18 months [1]. Because of this, the point of bottleneck has been shifted from the compute nodes to the interconnection network. This has driven the focus to high-speed interconnection networks are available to overcome such problems. But there might be interoperability problems when clusters with incompatible SANs are to be connected to clusters of clusters [2].

In that case, we need to have a method that will enable us to run our program across the cluster of nodes. Message passing mechanism among the processes helps to solve a particular problem in parallel environment [3]. But running distributed message-passing application in a heterogeneous environment is a challenge, as the applications have to deal with different numbers of communication interfaces, lower-level protocols, encoding schemes etc. for achieving the expected performance.

Our idea is to provide the user with a parallel programming standard, MPI (Message Passing Interface) to solve their problems in a parallel environment. A panel of parallel programming industry leaders including representatives from the national laboratories, universities and key parallel system vendors define the Message Passing Interface.

Rest of the paper is organized as follows. Section II provides a brief overview of C-MPI. Section III is the brief overview of MPICH and Gigabit Ethernet. Section IV gives the description of the experimental environment. In Section V, experimental results have been discussed and compared to that of MPICH. Conclusion and Future Work are presented in Section VI.

Manuscript received October 15, 2006

Authors are with Systems Software Development Group, Center for Development of Advanced Computing, Knowledge Park, No. I Old Madras Road, Byappanahalli, Bangalore-560038, India (e-mail: {prabud, vanamalav, sanjeebd, rsridharan, prahladab, mohan}@cdacb.ernet.in).

II. C-DAC Message Passing Interface (C-MPI)

C-MPI is a high performance implementation of the MPI standard for a Cluster of Multiprocessors (CLUMPS). By adhering to the standard, C-MPI supports the execution of the multitude of MPI applications with enhanced performance on CLUMPS. C-MPI can work with both TCP/IP using Ethernet and VIA over PARAMNet-II. It also leverages on the fact that most of the high performance networks provides a substantial communication bandwidth. It helps in performing send and receives operation of messages in a simultaneous manner, consequently reducing the no of hops in the transmission path.

In C-MPI different MPI collective communication calls have been optimized using efficient algorithms for CLUMPS architecture. The different C-MPI algorithms effectively use the shared memory communication on multiprocessor nodes to reduce the total computation time of the application. C-MPI provides the following advantages over public domain MPI [4]:

- Supports multiple protocols.
- It uses the network for remote communication while shared memory for local communication.
- Collective communication routines have been optimized for CLUMPS to achieve minimized execution time.

C-MPI is designed to achieve high performance and portability. It is layered over Abstract Device Interface (ADI) [5] to maintain portability. On C-DAC's PARAM Padma [6], C-MPI employs both TCP/IP [7] and C-VIA in the underlying ADI layer. The C-MPI functions are implemented in terms of macros and functions.

In Fig. 1, the upper layer does the communication of control information and the lower layer performs the transfer of data from one process address space to another. For achieving optimal performance characteristics, C-MPI can directly work over the SAN (System Area Network) in the user space using lightweight communication protocols. This results in decrease in communication time for MPI point-to-point communication protocols. The communication, at the lowest level, occurs in point-to-point manner as shown in the Fig. 1 below. Hence,

reduction in the communication time of the point-to-point communication calls leads to reduce communication time for the collective communication calls as well.

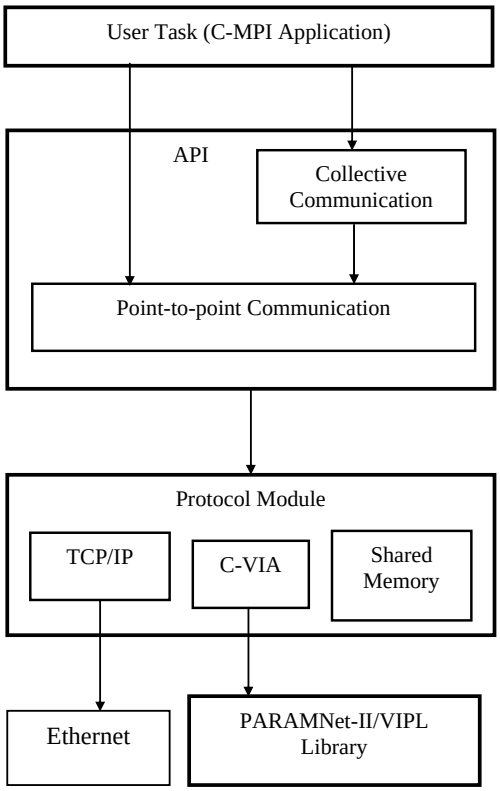

Fig. 1 C-MPI Control Flow

III. MPICH AND GIGABIT ETHERNET

A. MPICH

MPICH is an open source, portable implementation of MPI standard which is a widely used parallel programming paradigm for distributed memory applications in parallel computing. MPICH [8] is available for different flavors of Unix (including Linux) and Microsoft Windows. It is a product of Argonne National Laboratory.

MPICH supports one-side communication, dynamic processes, intercommunicator collective operations, and expanded MPI-IO functionality. Moreover, it can work over clusters consisting of both single-processor and SMP nodes. MPICH implementations are not thread safe. Its architecture is driven by two principles. Firstly, intention to increase the amount of code can be shared. Most of the code in the MPICH implementation such as MPI opaque object, including data types, groups, attributes and even communicators, are platform independent. Secondly, intention to provide a structure that makes it to be ported on to a new system quickly by replacing parts of shared code by platform specific code. This is achieved with a lower layer interface in ADI called Channel Interface. MPICH uses TCP/UDP socket interfaces to communicate messages between nodes. Because of this, there have been great efforts in reducing the overhead incurred in processing the TCP/IP stacks. To overcome this problem MPICH is now enabled to support VIA (Virtual Interface Architecture). VIA has defined different mechanisms that enable bypassing layers of protocol stacks and avoid intermediate copies of data during sending and receiving of messages. This allows significant increase in communication performance and decrease in processor utilization by the communication subsystem.

B. Gigabit Ethernet

Gigabit Ethernet [9] also known as the IEEE Standard 802.3z is the latest Ethernet technology. It is a Media Access Control (MAC) and physical layer technology. This transmission technology is based on the Ethernet frame format and protocols used in local area network. It offers a bandwidth of One Gigabit per second. To achieve this bandwidth, Gigabit Ethernet uses a modified version of the ANSI X3T11 Fiber Channel standard physical layer. It supports both half-duplex and full-duplex mode of transmission.

While operating in the full-duplex mode, Gigabit Ethernet uses buffers to store incoming and outgoing data frames until the MAC has time to pass them higher up the protocol stack. During heavy traffic, the buffers might fill up with data faster than the MAC can process them. When such situation arises, it is up to the MAC layer to prevent the upper layers from sending until some part of the buffer becomes free. Otherwise, there will be loss of data frames due to insufficient buffer space. When the receive buffers approach their maximum capacity, a high watermark interrupts the MAC control of the receiving node and sends a signal to the sender to suspend the sending process for a specified amount of time until the buffer can catch up. The sender resumes transmission after the time interval is past or it receives a new packet from the receiver with a time interval of zero. The function of the high watermark is to ensure that enough buffer remains to give the MAC time to inform the sender to suspend the transmission of data before the buffer overflows. Similarly, there exists a low watermark to notify the MAC control that there is enough space in the buffer to restart the flow of incoming data.

In the half-duplex mode, Gigabit Ethernet uses the enhanced CSMA/CD (Carrier Sense Multiple Access with Collision Detection) access method. In this technique, the channel can either transmit or receive at a time. When there is a collision between two frames, the MAC layer stops transmitting and retransmit the frame once the transmission medium is clear. But if a collision occurs after the packet is sent, then the packet is lost because the MAC layer has already discarded the frame and started to prepare for the next frame for transmission. CSMA/CD protocol is sensitive to frame length. That is why Gigabit Ethernet's performance is degraded when it operates in the half-duplex mode.

IV. Description of the Experimental Environment

The experimental environment comprises of two entities- Test bed and Benchmark. They are briefly described in the following sub-sections.

A. Specification of the Test Bed

The experimental evaluation for performance of C-MPI is done on PARAM Padma at CTSF [10], C-DAC Knowledge Park I, Bangalore, India. The table below is the specification of PARAM Padma that currently has a peak performance of nearly one Teraflop. The performance testing of C-MPI for comparison with the public domain MPI, MPICH is done on 8 and 32 4-way nodes running AIX-5.1 operating system.

TABLE I
Description of PARAM Padma

Specification	Compute node	File servers
Configuration	62 nos. of 4 way SMP and one node. of 32 way SMP	6 no. of 4 way SMP
No. of processors	248(Power 4 @1GHz)	24(UltraSparc-IV @900MHz)
Aggregate memory	0.5 Terabytes	96 Gigabytes
Internal storage	4.5 Terabytes	0.4 Terabytes
Operating system	AIX/LINUX	Solaris
Peak computing power for 62 AIX nodes	992 GF (~1 TF)	--
File system	--	QFS

Fig. 2 Picture of C-DAC's Tera-Scale Supercomputing Facility (CTSF)

B. Overview of the Benchmarks used

For comparing the performance of C-MPI to that of MPICH, we have used the following benchmarks:

- HPL (High Performance Linpack)
- PMB (Pallas MPI Benchmark)
- P-COMS (PARAM Communication Overhead Measurement Suites)

1. HPL

HPL [11] benchmark is a numerically intensive test. It is a popular benchmark suite to evaluate the performance of Super Computers and Clusters and involves solving a system of dense linear system in double precision (64 bits) arithmetic linear equations. Using HPL benchmark tests the PARAMPadma cluster efficiency.

2. PMB

PMB (Pallas MPI Benchmark)[12] is complex benchmark used for measuring MPI performance. It comprises of a concise set of benchmarks targeted at evaluating most important MPI functions. The different benchmarks under PMB are *PingPong*, *PingPing*, *Sendrecv*, *Exchange*, *Allreduce*, *Reduce*, *Reduce_scatter*, *Allgather*, *Allgatherv*, *Alltoall*, *Bcast* and *Barrier*.

3. P-COMS

P-COMS [13] comprises of a set of MPI benchmarks used for measuring communication overheads on large message passing clusters (such as PARAM 10000, PARAM Padma). The benchmarks have been implemented using MPI (Message Passing Interface) standard. The different benchmarks under P-COMS are *all*, *all1*, *ptp*, *advptp*, *cc*, *ccomp*, *gppong*, *roundtrip*, *allgring*, *oneway* and *circularshift*. The benchmarks measure the overhead time of different MPI library calls for Point-to-Point Communication, Collective Communication and Collective Communication and Computation etc. for message sizes ranging from 0 bytes to 10 Megabytes.

V. Experimental Results

Table II depicts the results of running HPL benchmark on 32 4-way nodes for C-MPI and MPICH. The sustained performance for C-MPI is found to be 274.5 Gigaflops against the calculated peak performance of 512 Gigaflops. However, it is 255.9 Gigaflops for MPICH. The results listed in the table below show that C-MPI clearly outperforms MPICH.

TABLE II
Tabular Form for Performance of HPL Benchmark

Matrix Size/ Block Size	C-MPI (Sustained/Peak Performance)	MPICH (Sustained/Peak Performance)
1280/200	0.9653 Gflops/ 512 Gflops	0.2663 Gflops/ 512 Gflops
155086/200	274.5 Gflops/ 512 Gflops	236.8 Gflops/ 512 Gflops
160528/200	273.1 Gflops/ 512 Gflops	255.9 Gflops/ 512 Gflops
165794/200	265.1 Gflops/ 512 Gflops	245.6 Gflops/ 512 Gflops

From Table II we have seen that C-MPI sustained performance is 53.61% of peak performance while for MPICH it is 49.98% of the peak value. This means C-MPI performance is better than MPICH by 7.27%. Moreover, for small problem size (1280/200) also C-MPI is 362.49% faster than the other. Hence, C-MPI is better between the two as it is evident from the result as shown is Fig. 3.

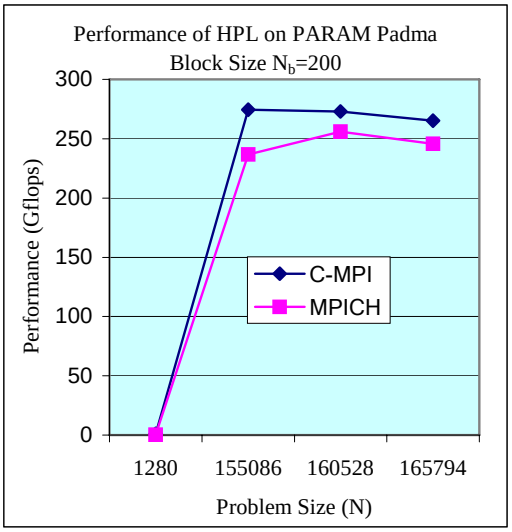

Fig. 3 Performance comparison of C-MPI and MPICH for HPL Benchmark

In Table III, output of PMB (Pallas MPI Benchmark) is depicted. The latency is obtained using Pingpong benchmark across two AIX nodes.

TABLE III
LATENCY FOR C-MPI AND MPICH

MPI Type	Latency (Microseconds)
C-MPI	24.38
MPICH	26.18

Table IV, below shows the performance of the two MPIs for P-COMS Benchmark.

TABLE IV
LATENCY & BANDWIDTH FOR C-MPI AND MPICH FOR P-COMS

Communication Overhead Parameter	C-MPI	MPICH
Latency	27.39µs	25.22µs
Bandwidth	114.69Mbps	106.13Mbps

Performance test using HPL and Pallas Benchmarks clearly reveals that C-MPI performs better than that of MPICH. From Table IV, we have seen that the bandwidth provided by C-MPI is much higher than MPICH. However, it lags behind in latency by a very small margin.

VI. CONCLUSIONS AND FUTURE WORK

The result shows that C-MPI provides better performance compared to that of the public domain MPI, MPICH over Gigabit Ethernet. The experimental result of HPC benchmarking shows that the optimized commercial C-MPI performs even better than MPICH. C-MPI is proved to be more powerful, encouraging and more robust than MPICH for the user community of high performance computing and communication.

Currently, C-MPI is enabled only for clusters. Work is in progress for Grid Enabled MPI [14].

REFERENCES

[1] Carlo Kopp, "Moore's Law and it's Implication for Information Warfare," The 3rd International Association of Old Crows (AOC) Electronic Warfare Conference Proceedings, Zurich, May 20-25, 2000. http://www.ausairpower.net/moore-iw.pdf
[2] Daniel Balkanski, Mario Trams, Wolfgang Rehm, " Communication Middleware System for Heterogeneous Clusters: A Comparative Study," Proceedings of the IEEE International Conference on Cluster Computing (CLUSTER-03) http://ieeexplore.ieee.org/iel5/8878/28041/01253359.pdf
[3] J. Silcock, A. Goscinski, " Message Passing, Remote Procedure Calls and Distributed Shared Memory as Communication Paradigm for Distributed System," Technical Report, School of Computing and Mathematics, Deakin University, Geelong, Australia. http://www.deakin.edu.au/scitech/sit/dsapp/archive/techreport/TR-C95-20.pdf
[4] W. Gropp, E. Lusk, N. Doss and A. Skjellum, "A high-performance, portable, implementation of the MPI Message Passing Interface Standard," Parallel Computing, 22:789-828,1996. http://www.globus.org/alliance/publications/papers/paper1.pdf
[5] William Gropp, Ewing Lusk " MPICH Abstract Device Interface, Version 3.3," MCSD, Argonne National Laboratory, December 2001 http://www.cse.ohio-state.edu/~panda/788/papers/ 3c_adi3man.pdf
[6] PARAM Padma Center for Development of Advanced Computing (C-DAC), Pune, India. Available at http://www.cdac.in
[7] TCP/IP. Available at http://www.ietf.org/rfc/rfc1180.txt
[8] Cornell Center for Materials Research Computing Facility. Available at http://monod.cornell.edu/docs/instructions/compilers/mpich.html
[9] Gigabit Ethernet Alliance, Gigabit Ethernet Overview. (1997) Available at http://www.gigabit-ethernet.org/
[10] Center for Development of Advanced Computing (C-DAC), Pune, India. CTSF. Available at www.cdac.in/html/ctsf/
[11] A. Petitet, R. C. Whaley, J. Dongarra, A. Cleary, "HPL- A Portable Implementation of The High-Performance Linpack Benchmark for Distributed-Memory Computers," Innovative Computing Laboratory, University of Tennessee, January 2004. http://www.netlib.org/benchmark/hpl/
[12] Pallas MPI Benchmark (PMB), Intel, http://www.pallas.com/e/products/index.htm
[13] PARAM- Communication Overhead Measurement Suites (P-OMS), Center for Development of Advanced Computing, Pune, India. http://www.cdac.in/html/betatest/hpc.asp
[14] C-DAC, GARUDA INDIA, The National Grid Computing Initiative. Available at http://www.garudaindia.in/tech_research.asp