Characterization of MPI Communication Primitives on a Heterogeneous Cluster

Siti Arpah Ahmad[1,3], Mohamed Faidz Mohamed Said[1]
Norazan Mohamed Ramli[1] and Mohd Nasir Taib[2]

[1]Faculty of Computer and Mathematical Sciences
[2]Faculty of Electrical Engineering
Universiti Teknologi MARA (UiTM), Malaysia
[3]Email: sitiarpahahmad@yahoo.com

ABSTRACT

This paper focuses on the performance of basic communication primitives, namely the overlap of message transfer with computation in the point-to-point communication within a small cluster of four nodes. The mpptest has been implemented to measure the basic performance of MPI message passing routines with a variety of message sizes. The mpptest is capable of measuring performance with many participating processes thus exposing contention and scalability problems. This enables programmers to select message sizes in order to isolate and evaluate sudden changes in performance. Investigating these matters is interesting in that non-blocking calls have the advantage of allowing the system to schedule communications even when many processes are running simultaneously. On the other hand, understanding the characteristics of computation and communication overlap is significant, because high-performance kernels often strive to achieve this, since it is both advantageous with respect to data transfer and latency hiding. The results indicate that certain overlap sizes utilize greater node processing power either in blocking send and receive operations or non-blocking send and receive operations. The results have elucidated a detailed MPI characterization of the performance regarding the overlap of message transfer with computation in a small cluster system.

ISSN 1675-7009

***Keyword**: Point-to-point communication, message passing interface, parallel computing, MPI characterization*

Introduction

Parallel computing is an exciting and promising modern area of exploration due to the decreasing cost of computer hardware and increasing processing power. Furthermore many organizations and government departments tend to have many old PCs, which are still in a good condition and have the potential to be converted into a parallel computer cluster. Such a cluster could be used by final year undergraduate project students for the purposes of research.

MPI (Message Passing Interface) has become a *de facto* standard for implementing parallel programs. MPI defines two basic communication primitives; point-to-point and broadcast communications [1]. The research presented investigates the round-trip characteristics of mpptest-ing in a heterogeneous cluster of 4 personnel computer. The aim is to quantify the potential overlap and examine the possible performance benefits. The overlap of computation and communication is of interest, because high performance kernels often strive to perform overlapping, which is advantageous with respect to data transfer and latency hiding [2].

Background and Related Works

The motivation for this research is to disseminate and promote the concept of parallel computing research in the country and amongst university students. The exponential increase in microprocessor power and storage capacity has made personal computers extremely powerful resources; this coupled with decreasing hardware costs and increasing speed means that there is great and realistic potential for the broad implementation of parallel computing [3-5].

Parallel computing has been made easier by MPI through the simplification the concurrent software development, since it has been separated from the hardware architecture [1]. The MPI programming model sets a series of tasks, which use the local memory during computation. The tasks then exchange data through the sending and receiving of messages within the cluster. Research related to measuring parallel computing performance is usually based on speed up factor, efficiency and scalability [6].

MPI libraries provide routines to initiate and configure message environments for the sending and receiving of data packets. Communication delays effect the determination of the completion time of the program; it is therefore important to investigate the delays associated with the communication primitives in MPI [1]. To address such delays, a software technique, termed overlapping of communication and computation is introduced. This technique has been explored by many researchers, who have mainly focused on improving application performance [7]. Research on MPI communication primitives have also been performed by various researchers [1, 7-10]. The purpose and consequent benefit of understanding the communication overhead of MPI communication primitives is that the programmer is able to write efficient parallel software as well as obtain a model, which can be used to assess the overhead introduced by the communication types for different message sizes and processor numbers [1]. The efficiency of overlap communication and computation is a well known technique to hide latency and improve performance for applications in high-end computer systems [9].

All communication in MPI is controlled by the process engine. This mechanism is responsible for making progress on pending requests in the library. Conceptually, the process engine loops all the incomplete requests and attempts to send or receive as much of each message as possible. In maintaining progress of each request the engine invokes the appropriate send or receive routine in order to access the operating system and/or the network hardware. If the MPI call is blocked, the progress engine continues to loop through all the pending requests until the specified requests are completed and the blocking call is reinstated. Conversely, in the instance of a non-blocking call, the progress engine loops through all the pending requests once and then returns regardless of the state of any request [10].

This research presented in this paper investigates the performance of point-to-point communications with respect to blocking and non-blocking types, and the consequent comparison of average time and bandwidth required during computation.

Methodology

The experiments were conducted on four heterogeneous nodes. The master node consists of an Intel Pentium 1.6 GHz CPU with 512 MB RAM. The three slaves consist of an Intel Celeron 1.72 GHz CPU with 256 MB RAM, an AMD Sempron 220 CPU with 512 MB RAM and an Intel Pentium 4 CPU with 256 MB RAM. The four nodes were connected using a 10/100 Mbps D-Link switch. The operating system was Linux Redhat 9.0 and an MPI library (mpich-1.2.0). The mpptest utility provided with mpich-1.2.0 was used to perform all the experiments.

The investigation of the MPI performance of the heterogeneous cluster was performed in a series of systematic steps as follows: firstly, testing of the blocking send and receiving operations. The purpose of this test is to observe the time difference between blocking and non-blocking operations. Secondly, evaluation of the round-trip average times (ìs) between blocking and non-blocking operations. The purpose of this test is to ascertain the effect of increasing the number of nodes on the time taken between blocking and non-blocking. Thirdly, the CPU usage is measured using a top application program. The peak measurement of the CPU usage is recorded manually based on real-time readings from the display. The purpose of this test is to enable evaluation of CPU usage with respect to message size.

Result

The results of the first set of tests are presented in Figures 1 and 2. The round-trip average time results plotted are similar for both blocked and unblocked overlap message transfer with computation. There is insignificant variation in the round-trip average times with increasing message size for both blocked (Figure 1) and unblocked (Figure 2) overlap message transfer.

The results for the second series of tests are presented in Figures 3, 4 and 5. Figure 3 shows the effect of increasing the number of nodes is to decrease the average non-blocking versus blocking time, however for a message of 100,000 bytes the non-blocking versus blocking time increases markedly. For message sizes greater than 100,000 bytes increasing the number of nodes from 3 to 4 decreases the average non-blocking versus blocking time significantly, Figure 4.

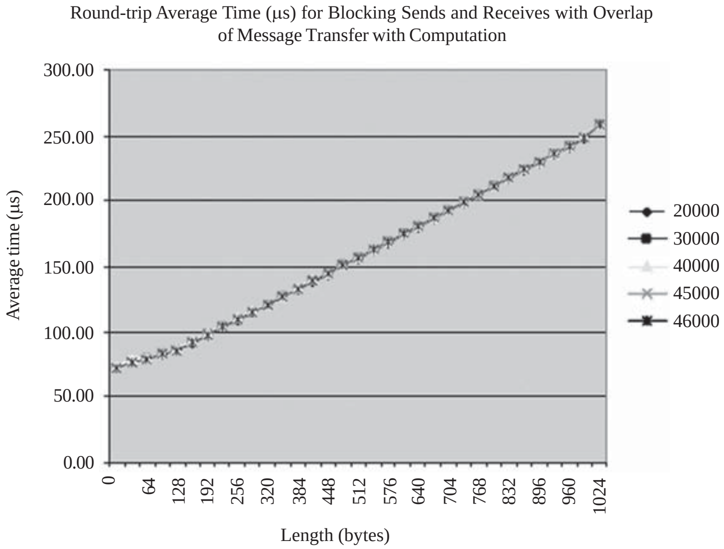

Figure 1: Round-trip Average Time (μs) for Blocking Sends and Receives with Overlap of Message Transfer with Computation

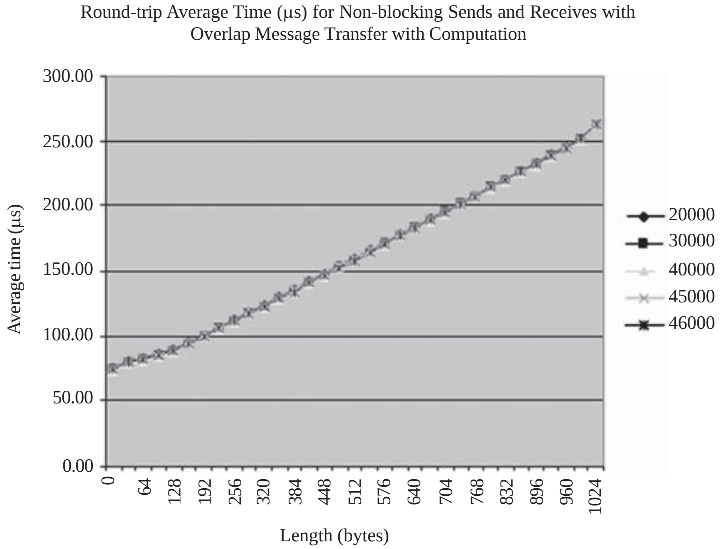

Figure 2: Round-trip Average Time (μs) for Non-blocking Sends and Receives with Overlap Message Transfer with Computation

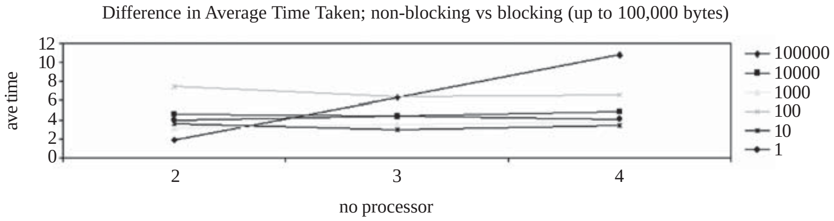

Figure 3: Difference in Average Time Taken for Blocking versus Non-blocking (up to 100,000 bytes)

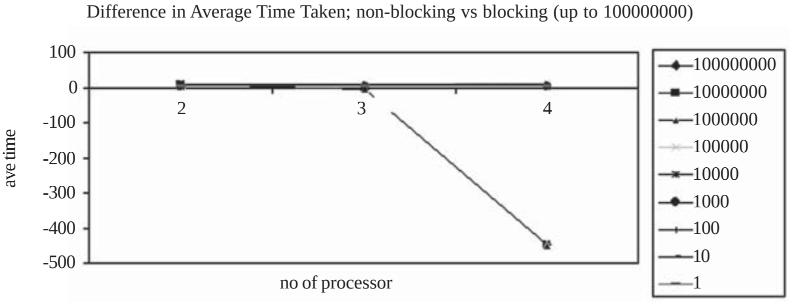

Figure 4: Difference in Average Time Taken for Blocking versus Non-blocking (up to 100,000,000 bytes)

The bandwidth required for a message decreases with increasing number of nodes, however for a message of 100,000 bytes there is no apparent change in the bandwidth, above 100,000 bytes there is a significant increase in the quantity of bandwidth required as the number of nodes increases from 3 to 4, Figure 5.

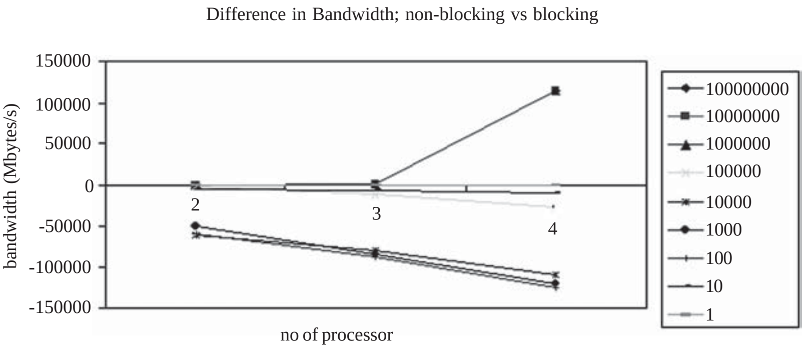

Figure 5: Difference in Bandwidth for Blocking versus Non-blocking

The results of the third test are presented in Figure 6 and indicate that as the message size increases more CPU resources are required. Non-blocking sending and receiving consistently requires greater CPU resources than for blocking.

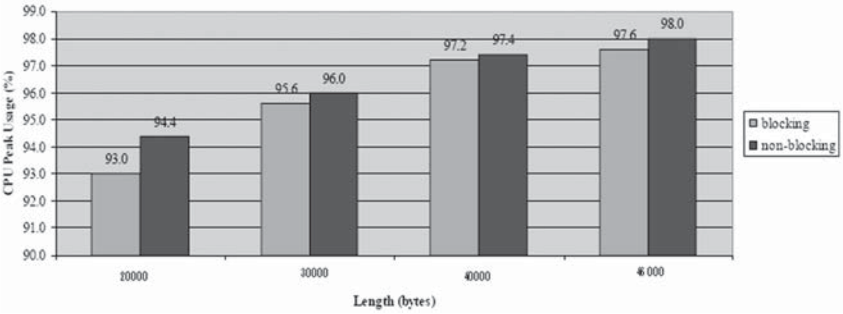

Figure 6: Comparison of CPU Usage (%) Between Blocking and Non-blocking as the Message Sizes is Increasing

Conclusion

This paper presents the findings for the mpptest-ing of a heterogeneous computer cluster with respect to the measurement of overlapping computation with communication. The results demonstrate that the network has a high potential tolerance to network latency and bandwidth, thus allowing significant relaxation of network requirements without a consequent degradation of application performance.

References

[1] A. Sinha and N. Das, 2004. A Comparative Study of the MPI Communication Primitives on a Cluster, *IEEE International Conference on High Performing Computing* (HiPC2004).

[2] http://wwwunix.mcs.anl.gov/mpi/mpptest/hownot.html, accessed on 5th Mac 09

[3] A. Abbas, *Grid Computing: A practical Guide to Technology and Applications*, Charles River Media. 2003.

[4] M. Danelutto, 2003. HPC the easy way: new technologies for high performance application development and deployment, *Journal of Systems Architecture* vol. 49(10-11) pp. 399-419.

[5] http://wwwunix.mcs.anl.gov/mpi/mpptest/ accessed on 5th Mac 09

[6] Blaise Barney, Livermore Computing, Lawrence Livermore National Laboratory,
https://computing.llnl.gov/tutorials/parallel_comp/#top, accessed on 30th Jan 2009.

[7] J. C. Sancho, K. J. Barker and D. J. Kerbyson, 2006. Quantifying the Potential Benefit of Overlapping Communication and Computation in Large-Scale Scientific Applications, *Proceeding of the 2006 ACM/IEEE conference on Supercomputing*.

[8] M. J. Rashti and A. Afsahi, 2008. .Improving Communication Progress and Overlap in MPI Rendezvous Protocol over RDMA-enable Interconnects, *22nd International Symposium on High Performing Computing System and Applications (HPCS 2008"),* DOI 10.1109.

[9] G. Anirudda, Shet and P. Sadayappan, 2008. A framework for characterization overlap of communication and computation in parallel applications, *Cluster Computer*, 11:75-90.

[10] S. Majumder and S. Rixner, 2004. An Event-driven Architecture for MPI Libraries, *The Loa Amos Computer Science International Symposium.*