Parallel Numerics '05, 207-216 M. Vajteršic, R. Trobec, P. Zinterhof, A. Uhl (Eds.)
Chapter 7: Systems and Simulation ISBN 961-6303-67-8

Tuning Communication in Gigabit Ethernet Cluster

Igor Rozman*, Roman Trobec, Marjan Šterk

Jožef Stefan Institute
Jamova 39, SI-1000 Ljubljana, Slovenia

A computing cluster built of seventeen dual-processor nodes connected in a toroidal 4-mesh with Gigabit Ethernet communication links was recently installed at the Jožef Stefan Institute. The initial performance of intra-cluster communication, resulting from default settings was unexpectedly poor. Substantial tuning of network adapter drivers was needed to improve the communication speed. In this paper the hardware configuration of the cluster and its interconnection network is described. Various aspects of network configurations that influence communication performance are shown. Point-to-point and collective communication were tested in synchronous and asynchronous mode on a ring topology. Results obtained after tuning indicate a significant improvement over the default configuration.

1 Introduction

Computer simulations are nowadays essential part of the investigation processes and science development because they are often the only viable option, less expensive [3], safer [10] and easier to perform as laboratory experiments. Such simulations need a significant amount of computing. High performance parallel computers provide the computational rates necessary for computer simulations based on parallel numerical methods, however, investigators can always opt for more complex models of larger systems, so that the computing resources will always be too small.

Parallel computers are composed of fast, unified computers connected with fast, dedicated communication links. The time of running the application at parallel computer consists of computation and communication time. By

*Corresponding author. E-mail: igor.rozman@ijs.si

increasing the number of processors the ratio of communication time compared to computation time is usually increased.

The speed-up is defined as the ratio of the execution time on a single processor to that on a parallel computer. The execution time is a sum of calculation time and communication time and further the communication time is a sum of set-up time and transfer time. The set-up time dominates with short messages while transfer time dominates with long messages. Bandwidth is defined as the amount of data transferred divided by communication time. It is usually supposed that processor performance, i.e. calculation time, has to be improved in order to improve the overall performance of parallel algorithms. However, the speed of communication is important as well. In some problems, e.g. molecular dynamics [9], a significant amount of global communication is needed. In such cases the optimal performance of intra-cluster communication is particularly important.

The rest of the paper is organized as follows. In Section 2, the computing cluster and its interconnection network is described. Next, custom driver settings for Intel network cards are described and two subsections are devoted to test results for point-to-point communication in synchronous and asynchronous mode, and broadcast communication in asynchronous mode, respectively. In the conclusions, possible future directions for further communication tuning of network cards are examined.

2 Cluster Description

A computing cluster was recently installed at Jožef Stefan Institute to run different parallel computer applications from the area of medicine, chemistry, telecommunications, satellite systems, etc. In order to fulfill our expectations, the cluster configuration was limited by the following requirements:

1. high computation power,
2. possibility of embedding various network topologies,
3. low price.

A cluster built of seventeen dual-processor nodes connected in a toroidal 4-mesh with Gigabit Ethernet communication links was chosen, as shown in Figure 1. Each node contains two 64-bit processors (AMD Opteron 244), 1024 MB RAM (PC3200 ECC DDR DIMM), 160 GB hard disc (Hitachi 7K250), six Gigabit Ethernet ports (2 $\times$ Broadcom BCM5704C + 4 $\times$ Intel Pro/1000 MT) and one Fast Ethernet port. Broadcom and Fast Ethernet ports are integrated on the motherboard (Tyan S2882G3NR), while two PCI-X dual port cards provide the four Intel Gigabit ports. Fedora Core 2 Linux [6] was

installed and the kernel upgraded to 2.6.8-1.521 smp. MPICH version 1.2.6 [12] is used for programming parallel applications. The listed hardware and software choices provided computing cluster with high performance for a low price [1].

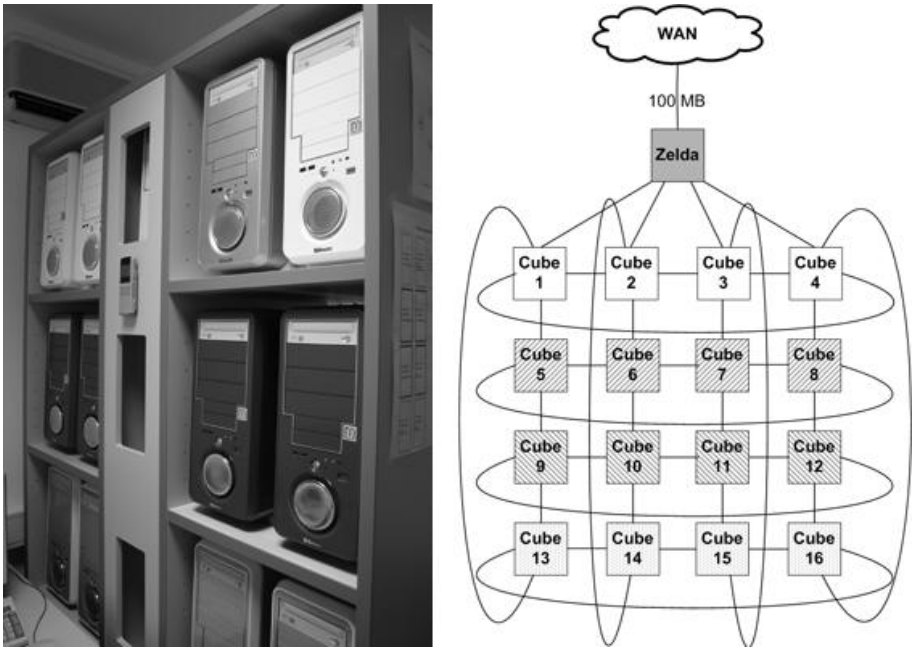

Figure 1: Photo of the cluster (left) and its interconnection network (right). Communication switch is not shown.

The sixteen nodes (named as Cube1, Cube2,..., Cube16) of the computing cluster are connected in a toroidal 4-mesh using Gigabit Ethernet connections through four Intel cards. The host node, named Zelda, is connected to the first four nodes of the cluster and to the WAN by a Fast Ethernet network card. Beside regular computation tasks, the main purpose of Zelda is enabling remote connections and running cluster applications. All nodes are also connected to the Gigabit switch (Level One GSW-2451T) using Broadcom network cards. The cluster photo and interconnection topology are shown in Figure 1.

The following topologies can be directly embedded into the described interconnections:

- rings of even lengths up to 16,
- toroidal 4-meshes of any size up to 4×4,
- hypercubes of dimensions up to 4.

Using Zelda rings can have also odd lengths up to 17. There still remains one Gigabit port that can be used to build customized topologies.

3 Tuning Communication Bandwidth

After the cluster was installed and first tests were executed, intra-cluster communication performance was found to be unexpectedly poor. The difficulties occurred at Intel network cards, where driver tuning was done by two settings [11] that have significantly shorten the communication time.

The first setting changes the allowed interrupt throttle rate (ITR). Its default value was 8000 interrupts per second, which in other words means that the network card handles up to 8000 messages per second. Following the driver instructions, ITR value was set at 100000, allowing up to 100000 interrupts per second, meaning that more short messages can be handled adequately. Consequently, the set-up time was shorten from 250μs to 25μs.

For large messages the communication remained slow. Internally, messages are decomposed into Ethernet frames (EF). Small frames usually mean more CPU interrupts and more processing overhead for the data transfer, because the per-packet processing overhead often limits the TCP performance in the LAN environment. The second settings changes the size of the EF. The Jumbo EF support is provided to break larger messages into fewer packets, therefore reducing CPU utilization and enhancing networking throughput. A Jumbo EF is anything larger than the standard 1500-byte EF, often 9000-byte is offered by hardware providers. The Media Transmission Unit (MTU) is a software driver parameter that specifies the EF size with default value 1500. It can be set independently for each Ethernet port.

The optimal size of MTU for our computing cluster was obtained by testing different kinds of communication patterns. Communication bandwidth was measured with our custom benchmark program implemented in C and some selected functions from the MPI communication library. A similar program can be found in [13].

3.1 Point-to-Point Communication

The Intel Pro/1000 MT network card supports MTU values up to 16000 bytes. Communication performance was tested at different MTU values (1500, 4000, 7000, 10000, 13000 and 16000) and different message sizes from 8 - 2^{21} bytes. The graphs in Figures 2 and 3 show the bandwidth of point-to-point communication in synchronous and asynchronous mode using the MPI functions `MPI_Send/MPI_Recv` and `MPI_Isend/MPI_Irecv`, respectively. The averages of 10 runs, which did not differ significantly, are shown.

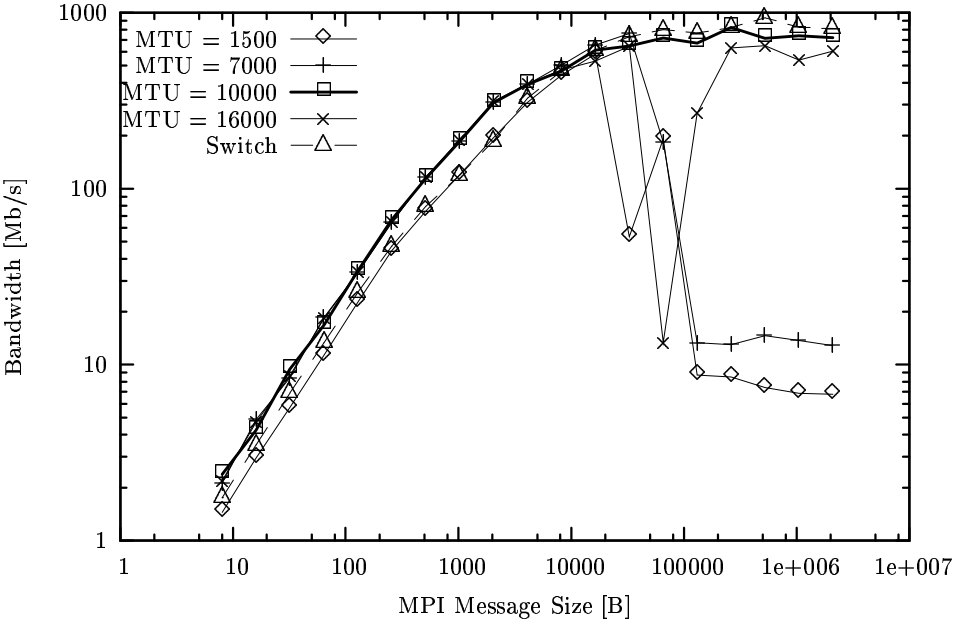

Figure 2: Bandwidth of point-to-point communication in synchronous mode.

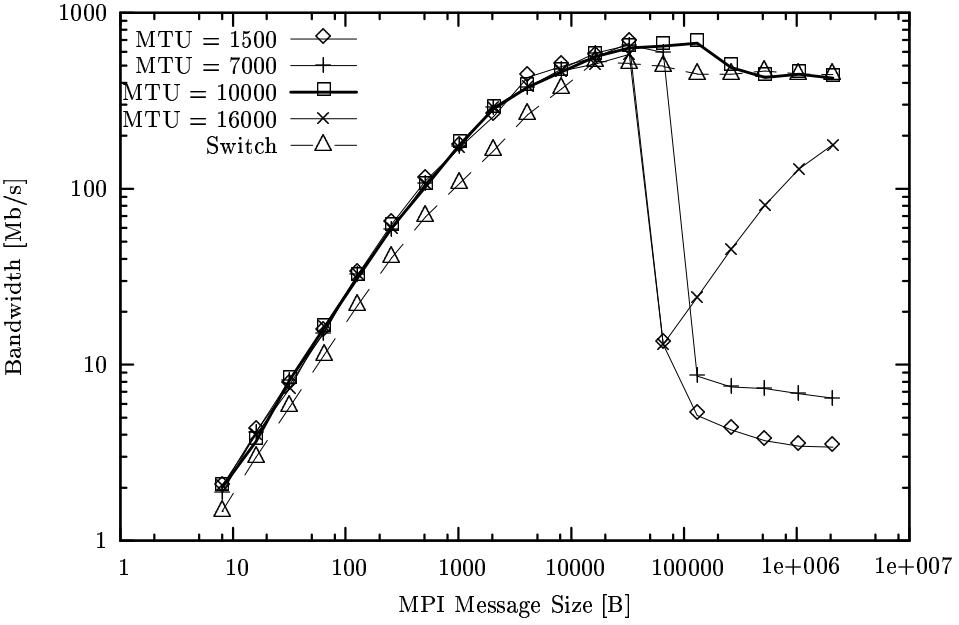

Figure 3: Bandwidth of point-to-point communication in asynchronous mode.

Theoretically, the bandwidth for short messages is reciprocal to the set-up time. The bandwidth curve will then rise with message size and asymptotically approach the maximal bandwidth. For messages larger than 16 KB the viable MTU setting was 10000 for both synchronous and asynchronous mode. The poor performance at other settings is disproportional to the increase in the number of packets, therefore we suspect that either the network drivers or the Linux kernel handle such packet sizes in a suboptimal way. Using this value the performance of communication cards is similar to the communication performance of the Broadcom cards using the switch. Unfortunately, the Broadcom cards do not support Jumbo EF therefore their MTU could not be set to larger values than 1500.

3.2 Collective Communication

Ring topology is often used in computer simulations where each node simultaneously exchanges the data with its left and right neighbors. It could be implemented either by communication using the direct links of the torodial 4-mesh or the communication switch. In both cases the MPI functions `MPI_Isend`/`MPI_Irecv` were used. The meaning of the bandwidth is now slightly different because data are transferred in parallel on all 16 pairs of processors. The actual timing was measured on a single node. Results are shown in Figure 4 for communication using Intel cards with various MTU values and for communication through the switch. The results show that the bandwidth of ring communication is the fastest with MTU = 10000. We can see also that the corresponding graph of switch bandwidth is shifted down indicating that the total switch capacity is exceeded by extensive simultaneous communication.

Finally, one-to-all communication was tested. Static routing in toroidal 4-mesh was chosen for maximal performance of intra-cluster communication [4, 5, 2], using the following two rules:

- if the node distance is one or two hops, data is sent through direct connections of the toroidal mesh,
- otherwise the data is sent through the switch.

The first processor of the node Cube1 sent the same data to all other Cubes using MPI function `MPI_Bcast`. For the communication time a maximal value among all communication delays was taken. The obtained results are shown in Figure 5. According to our previous experience we did not expect that the setting of MTU to 10000 will provide one of the poorer choices. We suspect that `MPI_Bcast` function is not tailored well to our cluster and should be customized for maximal performance.

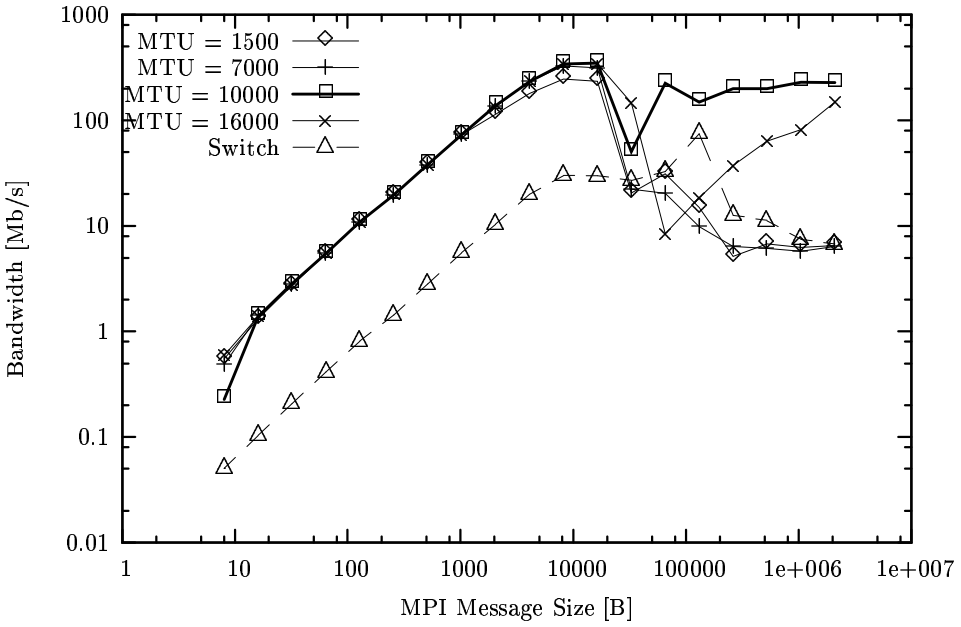

Figure 4: Bandwidth of communication between neighbors in ring topology.

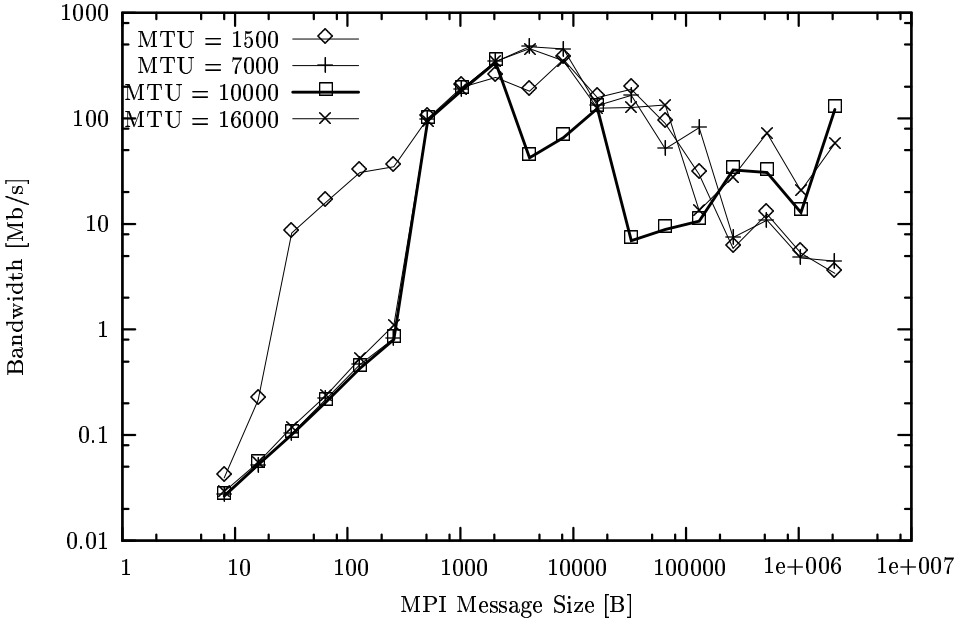

Figure 5: Bandwidth of one-to-all communication in toroidal 4-mesh.

4 Conclusions

Based on measured communication results at different ITR and MTU values the fastest communication with nearest neighbors was expected at values ITR = 100000 and MTU = 10000.

Experiments with different ITR values show surprisingly that maximal values are not always the best. Theoretically, more allowed interrupts in a processor should enable more messages to be processed, whereas in practice better results were obtained at some lower values for ITR. We find out also that MTU = 10000 is not optimal for one-to-all communication in a 4-mesh. Further investigation is needed in this direction.

In order to show that this result is reproducible also in real applications, we run a large parallel heat transfer simulation [8]. The simulation is based on the finite difference method, implemented by one-dimensional domain decomposition on a ring topology. The simulation was first run at default settings MTU = 1500 on all 32 processors. The communication was so slow that the program failed due to timeouts. After setting the MTU to 10000 the simulation succeeded with a speed-up of 16. This result was close to expected, which means that communications no longer represents the serious bottle neck.

In future work some others values of MTU will be tested, supposedly closely to 10000. More collective communication patterns will be tested in the 4-mesh topology [7], for example, each node will send data to all four neighbors at the same time. We expect that in such a complex communication pattern a big advantage of direct links will be proven as compared to the switch. In addition to synthetic performance measurements, real parallel applications will be tested where point-to-point communication is usually combined with collective communication for implementing stopping criteria or load balancing.

It was shown that default settings of Intel cards was not optimal for our computing cluster. On the other hand Broadcom cards with default settings offer close to optimal performance. One possible reason is that Intel network adapters are optimized for Intel processors, while the Broadcom adapters are supposedly optimized for the board onto which they are integrated. We suspect that the Intel adapters' low default ITR results from the default configuration that may be geared toward use in web servers or similar applications.

Acknowledgement

The purchase of the computing cluster was partially funded by the Ministry of Education, Science and Sports of the Republic of Slovenia.

References

[1] A. A. C. Braga, Technical aspects of Beowulf cluster construction, *Quimica Nova 26 (3)*, May-June 2003, pages 401-406.

[2] I. Jerebic, R. Trobec: Optimal Routing in Toroidal Networks, *Information Processing Letters 43 (6)*, October, 1992, pages 285-291.

[3] J. M. Krodkiewski, J. S. Faragher, Stabilization of Motion of Helicopter Rotor Blades Using Delayed Feedback - Modelling, Computer Simulation and Experimental Verification, *Journal Of Sound And Vibration 234 (4)*, 2000, pages 591-610.

[4] M. M. H. Rahman, S. Horiguchi: Dynamic communication performance of a Hierarchical Torus Network under non-uniform traffic patterns, *IEICE Transactions on Information and Systems E87D (7)*, July, 2004, pages 1887-1896.

[5] M. M. H. Rahman, S. Horiguchi: A new hierarchical interconnection network for massively parallel computers, *IEICE Transactions on Information and Systems E86D (9)*, September, 2003, pages 1479-1486.

[6] C. Stanton, A. Rizwan, F. Yung-Chin and H. A. Munira: "Installing Linux High-Performance Computing Clusters"; *Dell PowerSolutions 4*, 2001.

[7] R. Trobec: Two-dimensional regular d-meshes, *Parallel Computing 26 (13-14)*, December, 2000, pages 1945-1953.

[8] R. Trobec, M. Šterk, S. AlMawed, M. Veselko: Computer Simulation Of Topical Knee Cooling, *Proceedings of International IASTED Conference PDCP*, Innsbruck, 2005, pages 573-577.

[9] R. Trobec, M. Šterk, M. Praprotnik, D. Janežič, Parallel programming library for molecular dynamics simulations, *Int. j. quant. chem. 96(6)*, 2004, pages 530-536.

[10] P. Trunk, B. Gersak, R. Trobec, Topical Cardiac Cooling - Computer Simulation of Myocardial Temperature Changes, *Comput. biol. med. 33*, 2003, pages 203-214.

[11] Linux* Base Driver for the Intel(R) Pro/1000 Family of Adapters: ftp://aiedownload.intel.com/df-support/2897/ENG/README.txt.

[12] MPICH - A Portable Implementation of MPI: http://www-unix.mcs.anl.gov/mpi/mpich/.

[13] Benchmarking point to point performance: http://www-unix.mcs.anl.gov/mpi/tutorial/mpiexmpl/src3/pingpong/C/main.html.