Communication Software Performance for Linux Clusters with Mesh Connections

Jie Chen and William Watson III
High Performance Computing Group
Thomas Jefferson National Accelerator Facility
12000, Jefferson Ave.
Newport News, Virginia 23606, USA
chen@jlab.org

Abstract

Recent progress in copper based commodity Gigabit Ethernet interconnects enables constructing clusters to achieve extremely high I/O bandwidth at low cost with mesh connections. However, the TCP/IP protocol stack cannot match the improved performance of Gigabit Ethernet networks especially in the case of multiple interconnects on a single host. In this paper, we evaluate and compare the performance characteristics of TCP/IP and M-VIA [1] software that is an implementation of VIA [2]. In particular, we focus on the performance of the software systems for a mesh communication architecture and demonstrate the feasibility of using multiple Gigabit Ethernet cards on one host to achieve aggregated bandwidth and latency that are not only better than what TCP provides but also compare favorably to some of the special purpose high-speed networks. In addition, implementation of a new M-VIA driver for one type of Gigabit Ethernet card will be discussed.

Keywords: Linux cluster, M-VIA, Gigabit Ethernet, Myrinet, Bandwidth, Latency.

1 Introduction

The growth of performance of personal computers coupled with readily available high-speed interconnects has made high performance clusters an excellent choice for parallel computing in recent years. To achieve good results on a parallel computing cluster requires both hardware and software of the network interconnects (NICs) to be scalable, reliable and having high data throughput and low latency. Gigabit Ethernet over copper provides similar hardware performance compared to special purpose high-speed NICs such as Myrinet [3], at a fractional cost. However, the performance of Gigabit Ethernet is rarely realized for user applications because of the communication overheads of using traditional communication software such as TCP, due to the multiple memory copies of messages to/from the operating system (OS). Even with tremendous improvement in TCP performance in the latest Linux kernels along with much improved performance of PCI buses and memory subsystems for PCs, communication software systems based on TCP still provide inadequate bandwidth and latency for high performance parallel computing.

Currently, the overheads of using TCP come from two major effects. First, the actual TCP networking protocols themselves are complicated. Secondly, the protection and resource management provided by the operating system for the TCP stack requires multiple data copies and quite substantial overhead of system calls [4]. Several methods have been in place to reduce these overheads [5]. One of the approaches, dubbed "user-level networking" or ULN, removes the kernel from the critical paths of network send or receive and thus reduces the overheads greatly.

At present a few implementations of ULN exist, including FM [6], GM [7], Berkely VIA [8] and M-VIA [1], the latter two implement a industrial standard called Virtual Interface Architecture (VIA) [2]. These implementations have been quite successful and have proved ULN to be a sound concept. Of these M-VIA is the only one not to limit itself to use a Myrinet NIC and to have a nice framework within which a modified driver for a Gigabit Ethernet card can be developed.

In order to achieve so called "user level networking", user processes have to be able to access network cards directly without assistance from the operating system. A Myrinet interface card, which can be operated at 1.28 Gb/s full-duplex mode in a switched network environment, facilitates the ULN behavior by allowing direct manipulation in the NIC's RAM. With the rapid decline in price for personal computers, using Myrinet network cards as building blocks for a cluster becomes relative expensive due to high prices of Myrinet networks. Even though a single Gigabit Ethernet link may not provide as much bandwidth as a Myrinet link, multiple Gigabit Ethernet links in mesh connections provide better bandwidth and reasonable latency with much less expense.

In this paper we evaluate and compare the performance of TCP and M-VIA, with references to the performances of a Myrinet network, for a mesh-based Linux cluster using Gigabit Ethernet links. We pay special attention to whether mesh connections using multiple Gigabit Ethernet NICs on one host can provide high bandwidth and low latency with M-VIA software. In addition, we demonstrate how to develop a M-VIA driver from an existing Gigabit Ethernet driver for a Gigabit Ethernet card

within the M-VIA framework.

2 VI Architecture

In a traditional network architecture such as TCP/IP, the operating system virtulizes the network hardware into a set of logical communication endpoints available to applications. The OS multiplexes access to the hardware among these endpoints. Even though this approach simplifies the interface between the network hardware and the operating system, it introduces overheads due to the operating system involvement, such as system calls and data copies, in the critical paths of network data transfers.

The VI architecture eliminates the operating system overhead of the traditional model by providing each application process with a protected, directly accessible interface to the network hardware - a Virtual Interface. Each VI represents a communication endpoint, and pairs of such VIs can be connected to form a communication channel for bi-directional point-to-point data transfers. The OS may be involved in setting up and tearing down communication channels, but it is no longer in the way of the critical paths of data transfers. The network adapter is responsible for the tasks of multiplexing, de-multiplexing and data transfer scheduling normally performed by an OS.

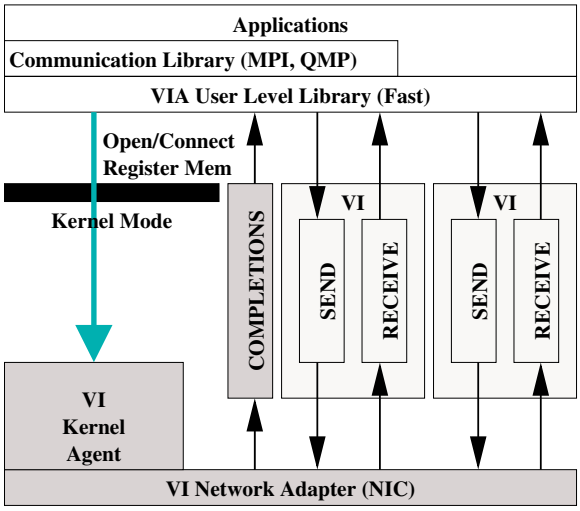

Figure 1: The VI Architectural Model.

The VI Architecture (VIA) contains four basic components: Virtual Interfaces (VI), Completion Queues, VI Providers, and VI Consumers. The VI Provider is composed of a physical network adapter and a software Kernel Agent. The VI Consumer is generally composed of a user process and an OS communication library. The organization of these components is shown in Figure 1.

In VIA, the OS is only responsible for endpoint (VI) creation/destruction, connection establishment, and memory registration. Like all other ULN implementations, VIA can send and receive data without involving the OS at all. To transmit data, an application first has a data buffer within the **registered memory**, that is pinned down (not swappable) by the host OS, so that the NIC is able to safely DMA data into and out of it. Second, the application constructs a **descriptor** that contains information about the data buffer. This descriptor is then added to a host managed and VI owned send or receive queue. Finally a **doorbell** is posted to the NIC to notify the NIC that a data buffer is ready to be sent or a buffer is ready to receive data. Once the data is sent or received, the VIA NIC marks the descriptor status directly.

A VIA doorbell is basically a pointer to a data transfer descriptor and it may be implemented in different ways. It is usually a control register or a memory mapped NIC's RAM. Unfortunately, due to a lack of programmability for Gigabit Ethernet cards, the M-VIA uses a **fast-trap** interrupt to the kernel agent to emulate a doorbell.

VIA supports three levels of communication reliability at the NIC level: Unreliable delivery, Reliable Delivery and Reliable Reception. Currently the M-VIA implementation only supports the first two reliability levels.

3 Testing Environment

All performance data are collected on a pair of PCs with an Intel Pentium 4 Xeon 1.8GHz running Redhat Linux 7.3 using kernel 2.4.20smp. There are 3 Intel Pro 1000MT Dual Port Gigabit Ethernet cards on each host. Each port is connected to a port on the other host using a category 5/6 copper cable directly. Two cards are plugged into two 133Mhz PCI-X slots and the other one is plugged into a 100Mhz PCI-X slot. Performance data of Myrinet networks are obtained using the same pair of hosts with a single LaNai9 Myrinet card on each host connected through a Myrinet 2000 switch. The M-VIA is version 1.2 and the M-VIA driver for the Gigabit Ethernet cards is based on Intel e1000 version 4.4.19.

4 Performance Results

In order to understand how M-VIA performs in a cluster with mesh connections, we present and compare the point-to-point latency and throughput for both TCP and M-VIA. We also compare aggregated bandwidth using multiple Gigabit Ethernet cards with the bandwidth from a single Myrinet card which is still more expensive than the multiple Gigabit Ethernet cards. Finally we present LogP [9] parameters to reveal what can be improved in the current implementation of M-VIA.

4.1 Small Packet Latency

The latency benchmark measures how long it takes for a single packet to travel from one node to another node. The performance data is obtained by taking half the average round-trip time for various message sizes. Figure 2 presents the results of latency for M-VIA, TCP and Myrinet networks.

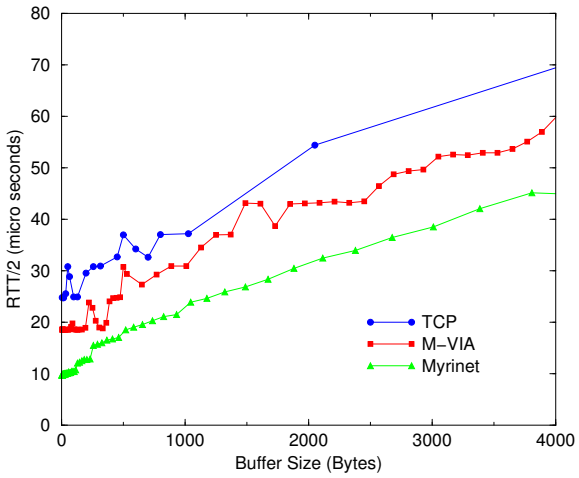

Figure 2: Application-to-Application Latency.

The performance of M-VIA for this benchmark is consistent better than what TCP can offer for all the test data sizes. The latency of TCP is at least 30% higher than the M-VIA latency which is around 18μs. This illustrates that M-VIA can indeed deliver latency needed for communication intensive applications. However the latency of M-VIA does not fair very well against that of GM. This is expected since the Myrinet network card is a special purpose high-speed NIC which offers memory-mapped NIC RAM and control registers that can be accessed directly by user applications, along with a programmable RISC processor enabling data transfers without generating any interrupt to the OS.

4.2 Point-to-Point Bandwidth

We use three types of bandwidth that capture different communication patterns occurring in typical user applications. In **bidirectional ping-pong bandwidth**, data flows in both directions alternatively, in a ping-pong fashion. In **Unidirectional bandwidth**, data flows in one direction only. Finally **bidirectional simultaneous bandwidth** simulates data transfers in both directions simultaneously. Figure 3 shows results of these types of bandwidth for M-VIA and TCP.

Clearly the bandwidth results of M-VIA are overall better than that of TCP. The M-VIA unidirectional bandwidth of 119MB/s actually approaches the hardware limit of Gigabit Ethernet. The simultaneous bidirectional bandwidth of M-VIA is 30% better than that of TCP in comparison to marginal better results for the other two types of bandwidth. This is not surprising since this type of test really generates simultaneous send and receive interrupts on a host and any reduction in sending and receiving overheads produces obvious differences in the results. However, TCP performs better in the unidirectional tests within small message range because of its better buffering mechanism which M-VIA is lacking.

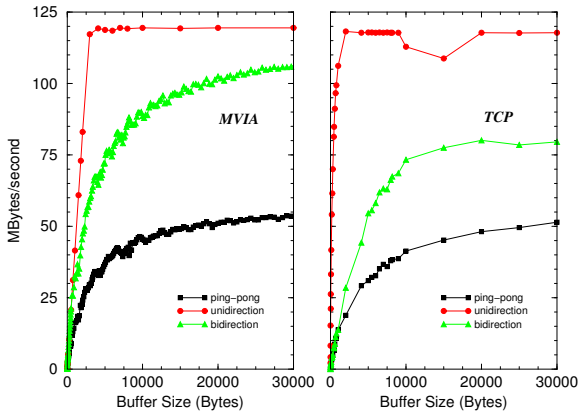

Figure 3: Point-to-Point Bandwidth.

4.3 Host Overheads of M-VIA

In order to understand performance characteristic of M-VIA, we measure or deduct LogP [9] parameters which reveal the time a message spends between the source and the destination network interfaces(L), the time the host processor is involved in sending or receiving a message (O_s, O_r). Figure 4 shows the send and receive overheads for different size of messages.

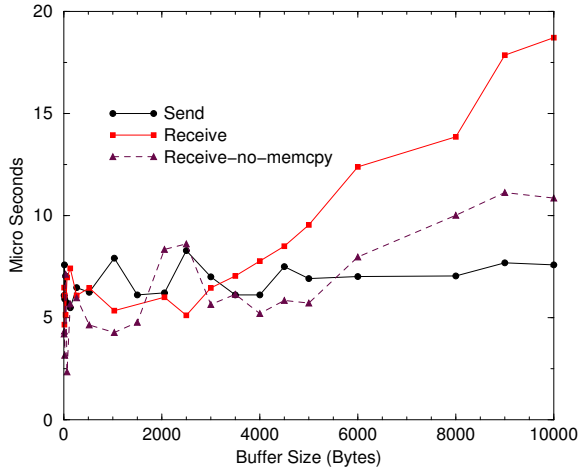

Figure 4: Send/Receive Overheads.

The OS sending overhead increases very slowly as the message size increases. This is expected since there are no memory copies in M-VIA in any sending procedure during which data are directly DMAed into the NIC. Nevertheless, the cost of fragmenting messages into segments of MTU (4096 Bytes) size contributes to the slow increases of the overhead. On the other hand, the OS receiving overhead increases slowly for message size below MTU but increases considerably for larger messages. This is because of one memory copy for receiving a message since there is no Gigabit Ethernet support to directly deposit the message into a user allocated memory buffer. The dashed line, which shows the receiving overheads without memory copies, highlights the above

points. Moreover the inverse of the slope for the receiving overhead is around 500MB/s which is smaller than the memory bandwidth because the overlap of copying one packet with the arrival of the subsequent packet (DMA). Finally the sending and receiving overhead of small messages reveal the hardware latency for the pair of NICs is around 7μs.

4.4 Aggregated Bandwidth

For some application the aggregated bandwidth from one node to all neighboring nodes is very important. In this experiment we use three cards on one machine connected to the other three cards on a different machine directly to simulate simultaneous communication with 3 neighboring nodes. The aggregated bandwidth is the sum of the simultaneous bidirectional bandwidth of each card within a single user process. Figure 5 presents these results for multiple connections.

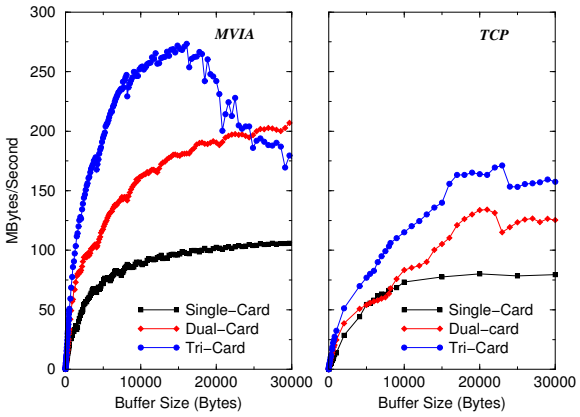

Figure 5: Aggregated Bandwidth.

The aggregated bandwidth for two connections is simply double of the bandwidth of a single connection, which reveals that two cards can be operated in full capacity concurrently. However this conclusion cannot hold for three connections especially for large messages. This is caused by difficulties to fully pipelining the three cards in a single process. Nevertheless the aggregated bandwidth of M-VIA is not only much better than that of TCP but also much better than that of a switched Myrinet network (120 MByte/s).

5 M-VIA Driver Development

The regular Linux driver for Intel Pro1000MT cards is distributed from Intel. It has to be modified to work under the M-VIA framework. Fortunately M-VIA has provided a lot of implementation handling packet fragmentation, packet reassemble, reliable delivery, software emulation of *doorbell* and a framework to modify Ethernet drivers. This framework consists of 5 driver macros that handle DMA gathering of data pointers, initiating data transmissions and differentiating VIA packets from IP packets. Of course the overall development cost of a M-VIA driver depends on various factors such as quality of the original driver source code and documents of the network hardware.

6 Conclusions

In this paper we evaluate and compare the performance characteristics of M-VIA and TCP for clusters with mesh connections using Gigabit Ethernet links. Indeed M-VIA provides excellent bandwidth and adequate latency for parallel computing even on par with the performance of Myrinet networks. Especially the aggregated bandwidth for multiple connections is much better than the bandwidth of switched Myrinet networks at a fraction of the cost. Currently preliminary studies on reducing M-VIA receiving overheads using either a larger OS pagesize or no memory copies is underway.

7 Acknowledgment

This work is supported by the Department of Energy, Contract DE-AC05-84ER40150.

References

[1] M-VIA: A High Performance Modula VIA for Linux, www.nersc.gov/research/FTG/via.

[2] Greg Regnier: Virtual Interface Architecture, Intel Press, April 2002.

[3] N. J. Boden, D. Cohen, R. E. Felderman, A. E. Kulawik, C. L. Seitz, J. N. Seizovic, and W.-K. Su. Myrinet: A gigabit-per-second local area network. *IEEE Micro*, 15(1):29-36, Feb. 1995.

[4] R. P. Martin, A. M. Vahdat, D. E. Culler, T. E. Anderson: Effects of Communication Latency, Overhead, and Bandwidth in a Cluster Architecture. In *Proc. 24th Annual Int'l Symp. on Computer Architecture (ISCA)*. 1997.

[5] C. Kurmann, F. Rauth, T. M. Stricker: Speculative Defragmentation - Leading Gigabit Ethernet to True Zero-Copy Communication. In *Cluster Computing 4*, pp. 7-18, 2001

[6] S. Pakin, V. Karamcheti, A. A. Chien: Fast messages: efficient, portable communication for workstation clusters and MPPs", In *IEEE Concurrency*, pp. 60-72, Vol. 5, No. 2, IEEE, April-June 1997.

[7] GM the low-level message-passing system for Myrinet networks: www.myri.com/scs/index.html.

[8] P. Buonadonna, A. Geweke, D. E. Culler: An Implementation and Analysis of the Virtual Interface Architecture, In *Proceedings of SC98*, Orlando, Florida, Nov. 1998.

[9] D. Culler, L. Liu, R. P. Martin, and C. Yoshikawa. LogP performance assessment of fast network interfaces. *IEEE Micro*, 1996.