Optimization of PVM Based on a User-level Communication Protocol

Jichao Zhang, Di Chang, Weimin Zheng, Huaxia Xia

Dept. of Computer Science and Technology. Tsinghua Univ.
Beijing 100084, P.R.China
zjc99@mails.tsinghua.edu.cn
chang.di@263.net

Abstract. With the introduction of high-performance Myrinet interconnection technology, building PVM on top of Myrinet has been a popular tread to improve its communication performance. There are generally two approaches to improve the efficiency of PVM: one approach is to emulate TCP/IP interface in low-level messaging layers, and base PVM on the new TCP/IP stack. The other approach is to redesign and optimize the communication mechanism of PVM, and integrate new low-level messaging layers to PVM. FPVM is a high-performance implementation of PVM designed for Myrinet-based PC clusters. By relying on the Fast Message Passing (FMP) library and optimizing communication mechanism, FPVM improves communication performance significantly. It delivers much of the underlying Myrinet performance to user-level applications. This paper introduces the design and implementation of FPVM, and presents its performance results.

Keywords: Parallel Virtual Machine; PC cluster; Myrinet; Fast Message Passing; FPVM

1 Introduction and motivation

As the emerging of advanced interconnection networks and the development of PC technologies, PC clusters have been an appealing platform for parallel processing. With the rapid development of interconnection technologies, efficient communication software are the major challenge for exploiting the performance potential of PC cluster systems. Myrinet is a high performance, highly reliable and high availability system area network (SAN), which makes it a suitable interconnection network for cluster systems. This paper introduces the experience of porting PVM to Myrinet technologies.

PVM (Parallel Virtual Machine) is one of the most popular programming environments for parallel computation [1][2]. It provides a message passing interface, makes network communication transparent, and supports flexible parallel computing in heterogeneous cluster systems. The communication levels of PVM system are illustrated in Fig. 1. In the communication path, PVM is in a middle level between the lower level communication protocols and the upper application level, and it is responsible for delivering the performance of lower level communication subsystem

to user-level applications. As a result, PVM is crucial to the user-level communication performance.

However, paying too much attention to applicability and adaptation to heterogeneous architectures in cluster systems, PVM incorporates complex communication mechanism, which incurs great communication overhead. Moreover, PVM is based on low-efficient TCP/IP protocol. TCP/IP stack is originally designed for wide area network, and it incorporate complex communication mechanism, which makes it far from taking full advantage of the high-speed physical networks. This performance loss of TCP/IP is also inherited by higher level PVM. PVM has been a bottleneck of the communication system of clusters, which limits the efficiency of parallel computing, the adaptability to applications, and the scalability of cluster systems.

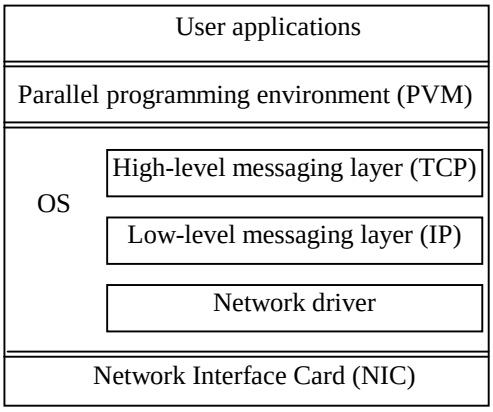

Fig. 1. Architecture of traditional PVM

Based on the above analysis, there are generally two approaches to improve the efficiency of PVM. One approach is to emulate TCP/IP interface using low-level messaging layers based on Myrinet, and then port PVM to the new TCP/IP stack without modifying its implementation mechanism. While this approach eases the portability of PVM, its limitation and restrains incurs great overhead and reduces communication performance to a great extend. The other approach is to redesign and optimize the implementation mechanism of PVM and use proprietary communication protocols designed for Myrinet. This approach takes advantage of the high performance of Myrinet network and reduces the communication overhead of PVM, and so it is promising to achieve high performance. FPVM adopts this approach, which optimizes the design mechanism of PVM and is based on FMP, a user-level communication protocol. FPVM is targeted to bridge the gap of the performance of lower level communication library and user-level applications, and to fully utilize the high performance of network hardware. FPVM retains the programming interface of PVM, which enable the available user applications using PVM ported to FPVM without modifying the source code.

The remainder of this paper is organized as follows. Section 2 analyzes the communication mechanism of PVM and points out its restraints and limitations. Section 3 introduces the main design and implementation issues of FMP. Some critical design technologies used in FPVM are discussed in Section 4. Section 5 evaluates the performance results of FPVM in terms of latency and bandwidth, which is compared with that of PVM. Section 6 gives the related research projects and compares them with FPVM. Finally, the conclusions and the future work are presented in Section 7.

2 The restrains and limitations of PVM

This section analyzes the communication mechanism of PVM and points out its limitations, which accounts for its low efficiency [3~5].

2.1 Complex communication mechanism

The architecture of PVM is illustrated in Fig. 2. In PVM, there is a daemon process named pvmd residing on each node, which servers as a maintainer and manager of the whole PVM system. And the tasks on each node undertake computation work.

As is shown in the figure, there are three kinds of communication in PVM: communication between pvmds using UDP/IP, communication between tasks in the same node using TCP with pvmd as an agent, and communication between tasks in different nodes either in normal mode through pvmd or in direct mode using TCP/IP and bypassing pvmd.

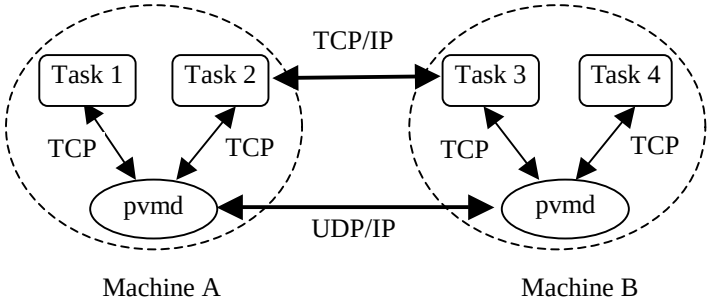

Fig. 2. Communication mechanism of PVM

Analysis of the communication mechanism of PVM reveals some factors affecting its efficiency. First of all, both TCP/IP and UDP/IP protocols are low efficient, which fail to exploit the performance potential of network hardware. Next, daemon-based communication mode increases data copies, which incurs great overhead. Moreover, even communication between tasks in different nodes in direct mode requires pvmd involvement and expensive establishment of socket connections.

2.2 Complex buffer management

In PVM, tasks or pvmds need message buffers for message transfer and PVM has a complex management of the buffers.

To send a message, a task must allocate a buffer dynamically for the outgoing message, pack the data of the message into the buffer, convert data format, send the message, and release the buffer. And in pvmd, there is also a complex manage of its buffers. The complex management of the buffers increases data copies and conversions of data format, which lead to additional communication overhead.

2.3 Low efficient multicast algorithm

Multicast operations are important for parallel programs. The multicast algorithm used in PVM is based on reliable point-to-point communication. To multicast a message, the sender repeatedly sends the message to the receivers, which is a great burden for the sender and makes the sender a bottleneck of multicast. With the increase of the number of nodes, the communication overhead will also increase.

3 The design of the Fast Message Passing Protocol

FPVM relies on FMP stack, which is a reduced user-level communication protocol for PC clusters based on Myrinet. FMP stack provids efficient transport of messages between processes, and hides the underlying hardware and software components. This section gives an overview of FMP.

PC clusters connected with high performance networks and running Windows NT operating system have gain popularity in the field of parallel computing. This tread has been driven by a combination of factors including higher performance-cost rate of Intel-based PCs compared with workstations, the popularity of Windows NT in research and business fields, and the advent of advanced network interconnections such as Myrinet. The communication protocols are crucial to the performance of clusters. FMP is designed to take full advantage of the high performance of Myrinet.

3.1 Myrinet-based PC cluster

FMP is implemented on a PC cluster based on Myrient network. Each PC is a SMP with two Pentium microprocessors running Windows NT, and a Myrinet network interface card (NIC) on PCI Bus.

Myrinet is a high performance switch network from Myrient Inc [6]. It incorporates technologies in both LAN and MPP, and has a maximum transfer rate of 1.28Gbits/s and an error rate of 10^{-15}. Myrinet transmits variable-length packets using wormhole routing, provides hardware flow control via back-pressure, in-order delivery. There is a programmable processor called LANai in the NIC that can be programmed to satisfy variable applications.

3.2 The principles and characteristics of FMP

Low-level communication protocols deliver the performance of network hardware to high-level programming environments or user applications, which are critical to the performance of user applications. FMP is targeted to reduce the communication overhead and make fully use of the performance potential of hardware. FMP removes operating system from the critical communication path, provides direct user-level access to the network interface. It achieves one-way latency of 9us and bandwidth of 83MB/s, which is a great performance leap over TCP/IP. Moreover, FMP provides

reliable and in-order message delivery and multiplexing and protective accessing of the network interface.

3.3 FMP interface

FMP provides a basic communication interface, which can be used directly by user applications and can also server as an intermediate layer for higher level communication layers such as PVM. Table 1 lists the main communication functions, which provide basic communication services. Since the interface matches the communication model of PVM, it's convenient to base PVM on top of FMP.

Communication functions	Operations
FMP_send_message(dstid, tag, buf, len)	Send messages
FMP_receive_message(srcid, tag, buf)	Receive messages
FMP_ProbeLocalCommu()	Probe local messages
FMP_ProbeNetCommu()	Probe network messages

Table 1. FMP interface

4 Critical design issues of FPVM

The analysis of PVM system in section 2 shows that the main problems of PVM are the complex communication mechanism and the low efficiency of TCP/IP protocol. FPVM optimizes the communication mechanism of PVM and is based on FMP instead of TCP/IP. This section introduces the critical design issues of FPVM.

4.1 Optimize the communication mechanism of PVM

The comparison of the architectures of PVM and FPVM is illustrated in Fig. 3.

As is shown in the figure, the communication mechanism of PVM is so complex that it's low-efficient. Moreover, the involvement of pvmd in the communication path leads to extra overhead. FPVM reduces the intermediate communication layers, and shorten the communication path, which improves efficiency. Since FMP are based on high performance Myrinet, and

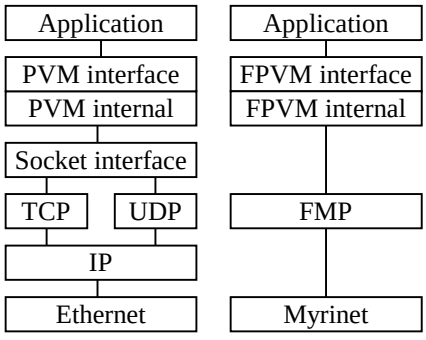

Fig. 3. The architectures of PVM and FPVM

exploits much of its performance, FPVM is possible to deliver much of the performance of Myrinet to user applications.

4.2 The translation between FMP task ID (TID) and PVM process ID (PID)

For each task in FMP, an integer ID is allocated as its identity (TID), through which the tasks are identified and can communicate with each other. However, the processes in FPVM are identified by process ID (PID). So it's necessary to translate between TIDs and PIDs. An address mapping table (PortTable) is designed to do so. Since FPVM processes have to access the PortTable for address transformation before each message transfer, it's important to implement high-efficient access of the table. PortTables are implemented in a hash structure.

Another important issue to be addressed is the update and consistency of PortTables in different nodes in the virtual machine. PortTables are implemented in shared memory that can be accessed by all the processes in a node and managed by pvmds. Pvmds are responsible for the update and consistency of PortTables by communicating with other pvmds in the virtual machine.

4.3 Simplify the message structure

There are three kinds of message structures including Message, Fragment and Package in PVM. Considering the message transfer procedure at the sender side, the data are first packed into a message that comprises several fragments, and then each fragment is segmented into several packets to be sent to network. At the receiver side, the message is received and handled in a reverse way. During the message transfer, there are at least four data copies and two conversions of data format, which incurs great overhead.

In contrast, there are only two message structures including Message and Fragment in FPVM. Each message consists of several fragments which are the unit of data transfer in network. By simplifying the message structure, data copies and conversions of data formats and are reduced. In FPVM, there are only two data copies and no conversions of data format for message transfers.

4.4 Simplify the buffer management

Since FPVM is based on FMP which provides reliable, in-order communication service, it's not necessary to incorporate the message acknowledge and retransmission mechanism which is used in PVM. By avoiding the message acknowledge and retransmission, the corresponding buffer structures are eliminated and the buffer management is more efficient, which reduces the communication overhead. Moreover, since the PCs in the cluster are homogeneous, conversions of data format used in PVM for heterogeneous architectures are eliminated in FPVM.

4.5 Improve the multicast mechanism

Though communication systems are often evaluated in terms of point-to-point communication performance, multicast communication is also important for the overall communication systems, especially for applications with many collective operations.

PVM implements multicast mechanism by unicasting a separate copy of the multicast message to every node from the source node, which is called iterative tree multicast algorithm. As is illustrated in Fig. 4, iterative tree algorithm is apparently low efficiently and far from exploiting the performance potential of Myrinet network. FPVM implements an efficient binomial tree multicast algorithm, as is illustrated in the following figure. Compared with iterative tree multicast algorithm, binomial tree algorithm reduces message relays and improves multicast efficiency. With the increase of nodes, multicast performance will also increase.

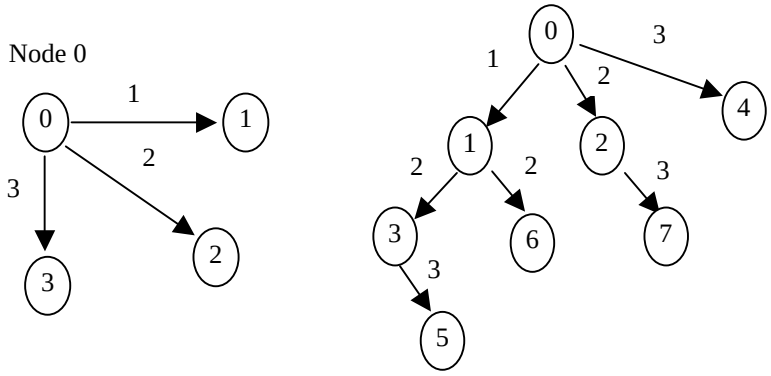

Fig. 4. Comparison of iterative tree and binomial tree multicast algorithm

5 Performance results and evaluation

Performance is the driving force of the design and implementation of FPVM. This section presents the performance results of FPVM in terms of one-way latency and bandwidth and the results are compared with that of PVM.

All the performance tests are conducted on a PC cluster based on Myrinet. Eight PCs are connected with a 1.28Gbits/s Myrinet switch, and each PC is a SMP with two Pentium III processors running Windows NT, and a Myrinet network interface card on PCI Bus.

The latency tests are in the general ping pong mode and the bandwidth in the stream mode. First of all, we compare the performance results of FMP and TCP/IP, which is plotted in Fig. 5 and Fig. 6. As is shown in the figures, FMP achieves one-way latency of 9us and bandwidth of 83MB/s, which improves the communication performance significantly compared with TCP/IP.

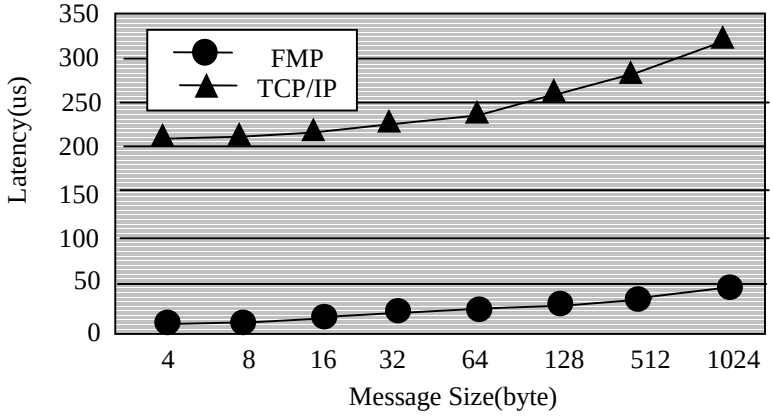

Fig. 5. Latency of FMP and TCP/IP

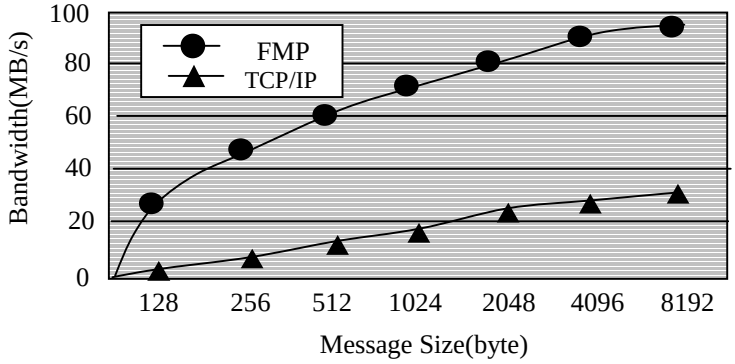

Fig. 6. Bandwidth of FMP and TCP/IP

Based on FMP and an optimized internal communication mechanism, FPVM improves communication efficiency greatly. Compared with PVM, FPVM achieve impressive communication with one-way latency of 45us and bandwidth of 48MB/s, which proves the validation of the design and implementation of FPVM, as is shown in Fig. 7 and Fig. 8.

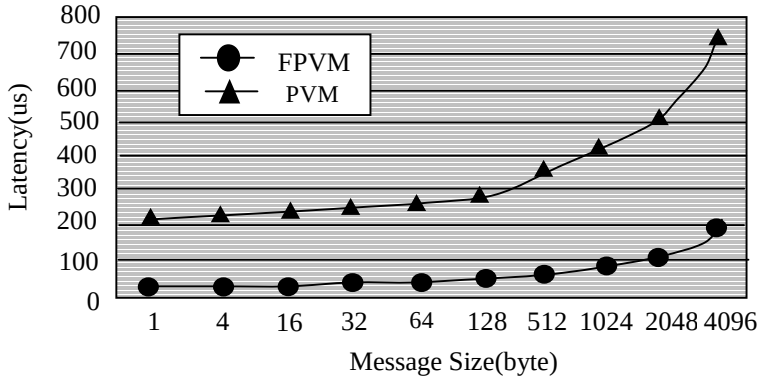

Fig. 7. Latency of FPVM and PVM

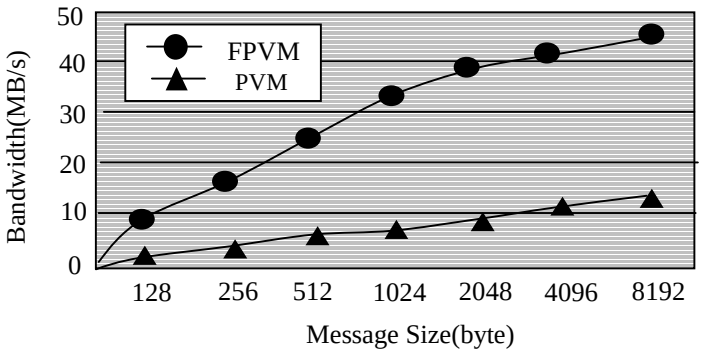

Fig. 8. Bandwidth of FPVM and PVM

6 Related work and comparison

With the popularity of Myrinet-based PC clusters as parallel computing platform, there have been many research projects on efficient communication using Myrinet. The research fields of interest include high performance communication protocols for Myrinet and providing efficient parallel programming environments such as PVM and MPI.

Generally, research projects address implementing MPI on top of newly developed protocols for its efficiency and ease of portability. While PVM is a popular programming environment for parallel computing and has a large number of users, it's needed to integrate advanced interconnection networks into PVM. There have been several projects concerning with the implementation of new PVM on new communication protocols, and some of them are listed below.

PVM-ATM from University of Minnesota is an early attempt to implement PVM on top of ATM network [7]. It uses Fore System's ATM API as low-level messaging layer and 100 Mbps TAXI interface cards. PVM-ATM achieves max. bandwidth of 27Mbps, and round trip latency of 1905us. Though PVM-ATM improves communication performance compared with PVM-TCP, it's far from satisfactory and fails to make much of the performance of underlying network hardware.

PVM-SCI project from University of Paderborn ports PVM to SCI interconnection [8]. SCI (Scalable Coherent Interface) is a high-performance interconnection network for cluster systems. The SCI protocol supports message passing as well as distributed shared memory access. By optimizing the Linux operation system, PVM-SCI achieves max. bandwidth of 60MB/s and one-way latency of 57us. And on Solaris platform, PVM-SCI gains max. bandwidth of 48MB/s and one-way latency of 57us.

PVM-GM project embeds Myrinet GM protocol to PVM [9]. PVM-GM uses linux operation system and uses the fast Myrinet transport when available but also interoperates and communicates with other cluster nodes via TCP/IP. GM is a user-level communication protocol from Myrinet Inc.. Due to the high performance of GM in gigabits, PVM-GM achieves max. bandwidth of 70MB/s and one-way latency of 50us.

According to the above analysis, the development of new interconnection networks has improved communication performance significantly. Compared with the hardware performance, there is still space left for improving the performance of communication software.

7 Conclusions and future work

With the increasing popularity of PC clusters running Windows NT, efficient parallel programming environments are required to exploit the performance of PC clusters. PVM is a widely used one. However, PVM is based on TCP/IP protocol and incorporates complex communication mechanism, which makes it fail to delivery mush of the performance of network to user applications.

FPVM is an optimized implementation of PVM, which is based on FMP, and optimizes the communication mechanism. This paper analyzes the restraints and limitations of PVM, presents the design and implementation of FPVM, and evaluates the performance results of FPVM.

Though FPVM achieves impressive performance, it's not a perfect one. We'll make further efforts to improve it. Our future work will concern the follows.

● Improve the communication performance of FMP. Though the performance of FMP is a leap over that of TCP/IP, there is still space left for the improvement of the performance of FMP.

● Optimize the interface between FMP and FPVM. FPVM exploits only about 50 percent of the performance of FMP. By optimizing the interface between FMP and FPVM, the performance gap could be reduced

References

1. Al Geist, Adam Beguelin, Jack Dongarra, Weicheng Jiang, Robert Mancheck, and Vaidy Sunderam. PVM: Parallel Virtual Machine. Scientific and Engineering Computation. MIP Press, 1994.
2. Sunderam, V. S., Geist, G. A., Dongarra, J. and Manchek, R. The PVM Concurrent Computing System: Evolution, Experiences, and Trends. Parallel Computing, Vol. 20, No. 4, April 1994.
3. Jeffrey M. Squyres. PVM? Are you joking? Paper for CSE 643: Principles of Parallel Computing, Fall 1997.
4. Geist GA, Kohl JA, Papadopoulos PM. PVM and MPI: a Comparison of Features. Calculateurs Paralleles, Vol. 8, No. 2, 1996.
5. H. Zhou and A. Geist. Faster Message Passing in PVM. In Proceedings of IPPS'95 Workshop on High Speed Networks, 1995.
6. Nanette J. Boden, Danny Cohen, Robert E. Felderman, Alan E. Kulawik, Charles L. Seitz, Jakov N. Seizovic, and Wen-King Su. Myrinet: a gigabit-per-second local area network. IEEE Micro, February 1995.

7. S. Chang, D. Du, J. Hsieh, M. Lin, and R. Tsang. Enhanced PVM Communications over a High-Speed Local Area Network. IEEE Parallel and Distributed Technology, Vol. 3, No. 3, 1995.
8. Markus Fischer, Jens Simon. Embedding SCI into PVM. Proc. of EuroPVM/MPI '97 Cracow, Poland, 1997.
9. Myricom. The gm api, December 1998. http://www.myri.com/GM.