Using Distributed Memory Dynamically in a Cluster of PVM-Based Linux Workstations

OP Gupta[1], Kiran Gupta[2] and Karanjeet Kahlon[2]
[1]*Punjab Agricultural University, Ludhiana, INDIA.*
[2]*Guru Nanak Dev University, Amritsar, INDIA..*
opgenius@yahoo.com, kirang123@lycos.com, karanvkahlon@yahoo.com

Abstract

Parallel Virtual Machine (PVM) has been identified as an effective system for heterogeneous computing. In this paper we investigate the performance of a heterogeneous distributed system based on PVM, where the computing nodes are connected each other through several levels of communication hierarchy and have different specifications. Matrix Computation is taken as sample case for checking the cost and performance of the heterogeneous PC cluster. Experiment shows that Data size needs to be as big as possible and communication cost increases with increase in number of hosts for fixed size problem. Heterogeneous PC cluster which is cost effective and performance oriented, can used in providing quality education and research for weather forecasting , image processing and many more areas of scientific computing.
Keywords: PVM, MPP, Matrix Computation, Cluster computing.

1. Introduction

In the quality education and research era, the need for faster computers is driven by the demands of both data-intensive applications in commerce and computational intensive applications in science and Engineering. Components of high performance parallel computers are evolving from proprietary parts e.g. CPUs disks and memories, into commodity parts. This is because technologies such commodity parts have matured enough to be used for high end compute systems. Another important trend changing the face of computing is an enormous increase in the capabilities of the networks that connects computers. These trends make it feasible to develop applications that use physically distributed resources as if they were part of the some computer. Computing on networked computers (also called Distributed computing' is not just a subfield of parallel computing, but it is deeply concerned with problems such as reliability securely and heterogeneity.

Parallel and distributed processing has merits in performance and cost. Here the two widely employed paradigms are Massively Parallel Processors (MPP) and Heterogeneous Computing (HC) [1, 2]. Even though MPP run much faster than HC, it costs more while the latter costs far less. There exists a clear tradeoff between the two computing paradigms with respect to the performance and cost. Under this condition, Parallel Virtual Machine (PVM) [7] was developed for allowing efficient heterogeneous computing. PVM divides the problem into subtasks and assign each one to be executed on the processor architecture that is best suited for that subtask.

PVM is based on message-passing model. Messages are passed between tasks over the connecting networks. The paradigm requires the user to partition the data among processor and specify interaction among processor explicitly. Of course, on a cluster of PC/ WS connected via some high latency LAN, message–passing is the only reasonable to program them.

Previous works on PVM [3, 7] have already identified its effectiveness. It has been used for solving various problems including material design, climate prediction, groundwater modeling, and so forth. It has also been found to be useful for solving large scale compute intensive problems. In these problems, matrix operations such as matrix multiplications and inverted indexing etc. are usually required to be solved. Using PVM for implementing these operations not only saves time, but also saves cost.

The rest of the paper is organized as follows. Section 2 introduces the pilot system used. Section 3 reveals about the parallel algorithm and data partitioning scheme. The experiment results are presented in Section 4 for various parameters. Section 5 provides a conclusion and the direction for the future research.

2. PC Cluster Pilot System

In earlier studies mostly a network of workstations was used as the main platform. Here the constituent nodes usually have the same (or similar) architectures and consequently same performances. They are also located closely with direct Ethernet connection, and the performances are measured while the system is used for only the problems being solved. The performance of such system will be greatly different from that of the system with the heterogeneous environment. Here the nodes are connected through several levels of communication network hierarchy, and have quite different specifications. The loads of them also dynamically change.

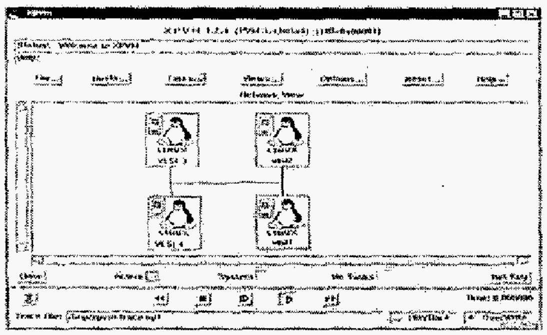


Figure. 1 PC Cluster

For the experiment, the PVM system is implemented in four computer systems [Figure.1] whose performances are widely varying. TCP/IP is used as a communication protocol. We also study the communication performance of PVM by measuring the transmission time of various size data.

It reveals that the larger the data, the more regular and efficient the communication time is.

3. Parallel Matrix Computation

Matrix multiplication [6] is a building block in many matrix operations and for solving graph theory problems. Due to its fundamental importance in sciences and engineering, Personal computer/workstation Clusters are promising candidates for future high performance computers, because of their good scalability and cost performance ratio. Thus investigating the scalability and performance of PC/Workstations based on PVM for parallel matrix multiplication is meaningful.
The parallel matrix multiplication algorithm that is taken for the implementation in the heterogeneous network environment is based on the conventional serial algorithm as given below.

```
Procedure MAT_MULT (A, B, C)
    Begin
        For I = 0 to n-1 do
            For j =0 to n-1 do
            Begin
            C [i,j]=0
            For k=0 to n-1 do
                C[i,j]=C[i,j]+A[i, k] * B[ j,k]
            End for
    End MAT_MULT
```

3.1 Parallel Implementation

The concept that is used in parallel matrix computations [1, 2] is of block matrix operation. The matrix-multiply algorithm to calculate C = AB, where C, A, and B are all square matrices, also uses block matrix operations, where m x m tasks will be used to calculate the solution. Each task will calculate a sub block of the resulting matrix C. The block size and the value of m is given as a command line argument to the program. The matrices A and B are stored as blocks distributed over the m^2 tasks The details of the algorithms at a high level is as follows:
Given:
i) A grid of m x m tasks.
ii) Each task (t_{ij} where 0<=i,j < m) initially contains blocks C_{ij}, A_{ij}, and B_{ij}.
Step 1. The tasks on the diagonal (t_{ij} where i = j) send their block A_{ij} to all the other tasks in row i.
Step 2. After the transmission of A_{ii}, all tasks calculate A_{ii} x B_{ij} and add the result into C_{ij}.
Step 3. The column blocks of B are rotated. That is, t_{ij} sends its block of B to T (i - 1) j. (Task t_{0j} sends its B block to t (m- 1) j.) The tasks now return to the first step;
Step 4. A $_{i(i+1)}$ is multicast to all other tasks in row i, and the algorithm continues. After m iterations the C matrix contains A x B, and the B matrix has been rotated back into place.

4. Experiments and Observations

In this section, we investigate the performance of a heterogeneous computing system based on PVM using the problem discussed in the earlier section and then study the communication speed of PVM system since it is a very important factor in deciding the overall performance.

4.1 Network View

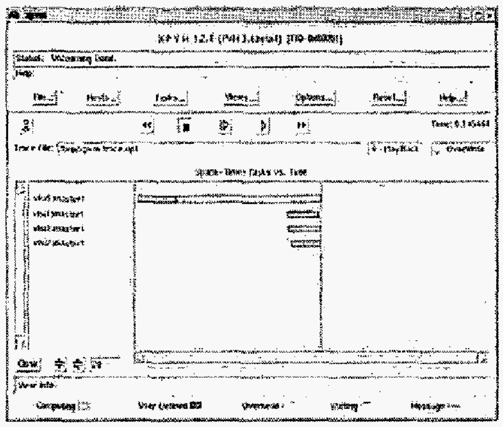
The Network view displayed the present virtual machine configuration and the activity of the hosts. Computer icons illuminated in different colors to indicate their status in executing PVM tasks. From the network view [Fig.1] we can see how well the virtual machine is being utilized by PVM application. If all the hosts are green most of the time, then machine utilization is good.

4.2 Call Trace View

It provides a textual record of the last PVM call made in each task. The list of tasks is the same as in the Space -Time view. This view is useful as a call level debugger to identify where a PVM program's execution hangs

4.3 Space - Time View

The Space -Time view displayed the activities of individual PVM tasks that are running on the virtual machine. Listed on the left-hand side of the view are the executable names of the tasks, preceded by the host they are running on.



4.4 Communication Speed

Communication speed is checked using VLSI1 as the master and other hosts as slaves (VLSI2, VLSI3, and VLSI4). As soon as the task identification (tid) from all the hosts is reached at master i.e. console i.e. connection is established among the master and slaves. Master packs a message and sends to all other hosts and in reverse hosts echoes the message back to master. Table 1 and

Table 1 VLSI1-VLSI2

Size	Send time	Pack Time
10^3	0.28	0.391
10^4	0.84	0.441
10^5	0.131	0.460
10^6	0.144	0.482

Table 2 show the time taken to it and to other slave hosts using varying message sizes (10^3, 10^4, 10^5, 10^6).

Table 2 VLSI1 -SELF

Size	Send time	Pack Time
10^3	0.122	0.660
10^4	0.184	0.710
10^5	0.133	0.910
10^6	0.147	0.998

Note from the tables that communication time vary a lot when the size of data is relative small. It became stable as the size of data increases. This shows that, for efficient communication, the data size needs to be big as possible.

4.5 Computational Performance

Computation performance is checked while problem size kept constant, while hosts are increased step by step. Time taken by the system to solve the problem is recorded in the table 3.

Table 3 Computational Speed

Hosts	Time	Speedup
VLSI1	0.689	1.00
VLSI1,VLSI2	0.581	1.19
VLSI1,VLSI2,VLSI3	0.534	1.29
VLSI1,VLSI2,VLSI3,VLSI4	0.618	1.11

These results are in conformity with our expectations since the inter-process communication overhead and synchronization overhead tends to increasingly dominate the overall execution time with increasing no. of processors values for a given value of data size.

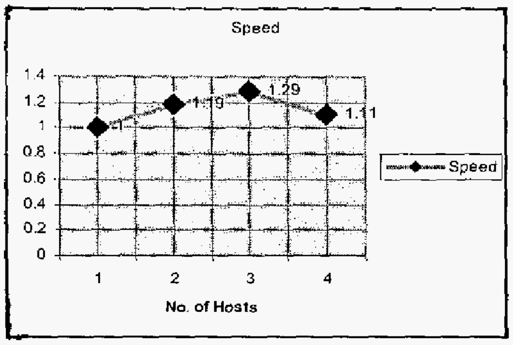


Figure 4: Speedometer

5. Conclusions

We have studied the performance of heterogeneous distributed computing based on PVM on Linux cluster. PVM can efficiently partition a job and distribute them to several hosts so that they can run in parallel. Experimental results showed that

- To be communication channel efficient, data size should as large as possible.
- Number of hosts in the Linux cluster will depend on the size of the problem.
- Data partitioning should be done according to the number of hosts.

Future research and study is open for the controlling the master process so that it should not be terminated abruptly when any one of hosts hangs or fails.

References

[1] Kumar, Vipin, Grama, Ananth, Gupta, Anshul, Karypis, George. *Introduction to Parallel Computing: Design and Analysis of Algorithms.* Addison Wesley, 1994.

[2] M. Šoch, J. Trdlîcda, P.Turidík, *PVM, Computational Geometry, and Parallel Computing Course*, Lecture Notes in Computer Science, Vol. 1156, Springer- Verlag, 1996.

[3] J. Touriño, R. Doallo, *A PVM-Based Library for Sparse Matrix Factorizations*, Lecture Notes in Computer Science, Vol. 1497, Springer-Verlag, pp.304-311, 1998.

[4] Peter Carlin. *Distributed Linear Algebra on Networks pf Workstations. Thesis.* California Institue of Technology, 1994.

[5] Paul Gray, Alan Krantz, Soeren Olesen, Vaidy Sunderam, *Advances in Heterogeneous Network Computing*, Lecture Notes in Computer Science, Vol.1497, Springer-Verlag, 1998.

[6]' Stewart G.W., Introduction to Matrix Computations, Academic Press Inc., SanDiego California 1973.

[7] A.Geist, A. Beguelin, and V. Sunderam, PVM Parallel Virtual Machine – A User's Guide and Tutorial for Networked Parallel Computing, MIT Press, Cambridge, MA, 1994.