Network Computing : A LAN Based Parallel Computing Platform at The University of Indonesia

Bagio Budiardjo
EE Departement, Faculty of Engineering University of Indonesia

VL

Earlier version of this paper has been published in the Proceeding of Scientific Seminar FTUI, :*Design Analysis and Industrial Application"* on Electronics, New Material and Information Technology, Center for Japanese Study, October 1996

Abstract

This paper explains the development of a parallel computing platform using the available LAN at the Inter University Center for Computer Science University of Indonesia. A Software tool like Parallel Virtual Machine enables the networked stations to be harnessed as a huge computing resource, combining various type of computing nodes to run parallel computing tasks. The research was aimed at first : to develop a number of parallel high complexity algorithms such as modelling air pollution distribution based on Gaussian Air Pollution Dispersion Formula which simulates dispersion of pollutant emitted from industrial exhaust and also from vehicle exhaust on the road. Secondly, these algorithms were used to test the platform, to obtain network behaviour in supporting the implementation of parallel algorithms. From the experiments conducted, there are strong indications that the network based computing facility could be used as a suitable platform for running parallel programs, with some restrictions. Optimum performance with near linear speed-up could be obtained by carefully partitioning the problems, limiting message-passing activities and balancing the load of the participating processors.

Key words : LAN, parallel computing, data partitioning, function partitioning, homogenous, heterogeneous, speed-up, load balancing, message-passing.

1. Introduction

The demand for computing platform either using PCs or workstations at the University of Indonesia is growing rapidly. At the Inter University Center for Computer Science (IUC-CS), there are some interconnected LAN networks of both PCs and workstations, mostly linked with the Ethernet, supported by Unix, Linux or Windows Operating Systems. With the availability of Paralle Virtual Machine (PVM), this interconnected network is linked and utilised as a parallel computing platform.

PVM is a software system that permits a heterogeneous collection of Unix computers networked together to be viewed by a user's program as a single parallel computer. PVM is designed to link computing resources and to provide users with a parallel platform for running their computer programs. The users do not have to be aware of the types of the computers they are using as well as the location of these computing resources [2]. With the growing need to solve both engineering and mathematical problems, there is a strong demand for higher performance, lower cost and relatively "easy to use" computing facility. MPPs is by far the most powerful computers in the world, but the main burden or obstacle is the cost. In 1993, MPPs typical cost ranging from several hundred thousands up to a couple of million of dollars each, depending on the number of processors in the system and also the complexity of the architecture [2]. On the other hand network stations are available in most academic institutions in Indonesia mostly in the form of Local Area Network (LAN). It could be utilised as a parallel computing machine with minimum preparation cost. Furthermore, the message-passing model of distributed and parallel algorithms has been widely adopted as a viable programming paradigm. PVM fits this paradigm, with minor additional procedures to develop parallel programs. It is a relatively "easy to use" computing platform, enables users to develop parallel programs similar to traditional sequential programs, differs only in scheduling, message-passing and synchronisation procedures.

2. Network Development

Experiment to develop parallel programs on the PVM network started out in early 1993 on

PVM version 2.4 that was available as a public domain software on the Internet. The computing platform was a network of Sun Sparc +1 with a file server and 7 supporting nodes. This was a homogeneous parallel computing platform, differs only on the server's hardware configuration. The Operating system was Sun OS version 4.1 and each of the supporting nodes is having identical hardware and software configuration. Due to some hardware problems and limited memory space (8 Mbytes in each node), this homogenous parallel computing environment that was installed in late 1990 was no longer a stable platform for running parallel programs. There were also some DEC Stations (3 stations) installed at a network, but due to a problem in router, this network was isolated, could not be used as a parallel computing platform. Another network was developed, consisted of two later version of Sun Sparc stations (one Sun Sparc +10 and another Sun Sparc +20, dual processor). In early 1995, a network of PCs was also installed at IUC-CS linked to Sun SS +10 (PVM symbolic name Bariah) and Sun SS + 20 (PVM symbolic name Raden). These Sun workstations are linked with Ethernet bus, supported by SUN OS version 4.1 and PVM version 3.8; to form a heterogeneous parallel computing platform (Figure 1).

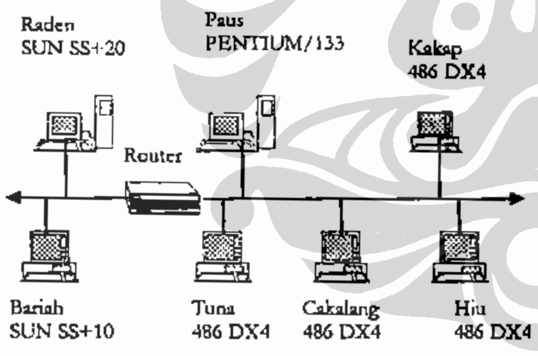

Figure 1
The heterogeneous computing platform

This platform was used to develop test programs, to obtained network behaviour in supporting implementation of parallel algorithms. A subset of this network consists of all PCs, is the operational platform, while the Sun workstations are not fully utilised due to some router and security problems. The PCs are supported by LINUX Operating System version 1.2.13 and PVM version 3.8, each of the "old" DX4 processors is having identical hardware configuration, with 16 Mbytes memory, 256 Kbytes cache. The Pentium/133 is having more memory space (64 Mbytes) and 512 Kbytes cache. This subset of network of PC's is a heterogeneous computing platform. Each of these PC is also has a unique PVM symbolic name (Kakap, Tuna, Hiu, Cakalang and Paus). A number of PCs with newer version of processors (Pentium 200 and up) has been added to this installation recently. Technologically they are almost the same as the Pentium processor already installed in the LAN.

3. Parallel Application

The first PVM based program developed at the IUC-CS network was the MDP-93 [6], which models the dispersion of particulate matters in the air, following the Newton's law. Subsequently, a number of other modelling programs were developed. They are the MDG-93 [4] which simulates the dispersion of pollutant in the air originated from single stack (industrial exhaust) based on Gaussian dispersion model, the MDG-94 [5] which was the extension of single stack model, simulates dispersion of pollutant in the air originated from multiple stack and the MDG-95 [11], the Finite Line Length Source (FLLS) based also on Gaussian dispersion model, which simulates the dispersion of pollutant over the roads or highways. MDP-93 and MDG-93 were developed and tested on homogeneous parallel computing platform (Sun Sparc +1 network), from 2 up until 4 supporting processors including the server. The MDG-94 and MDG-95 were developed and implemented on homogeneous Sun Sparc +1 network, using up until 6 supporting processors including the server. The scheduling of tasks in all the above mentioned software basically is *master - slave*, with *data partitioning* and *balancing the workload*. It is relatively easy to schedule the tasks in this scheme, since the computing algorithms are simple. In most of the tests made, server acted as the master and other nodes served as slaves. Most of the computations were done involving integer data type, few of them used floating point. The workload in all participating processors were kept balanced. All processors were arranged to start and finish their process at almost the same time. This came-out to near linear speed-up, except the MDG-93. Figure2 gives the basic idea of parallel algorithms developed on Sun Sparc +1 network.

Balancing the workload in the above mentioned algorithm means distributing equal computing steps in all participating processors.

MDG-95 was also ported on homogeneous PC network supported by up until four 486 DX-4/100 processors, and it was running well.

Other benchmark software were also developed, to test the heterogeneous LAN consisting PC's nodes. MFP-1 [8] and MFP-2 [7] are test programs experimenting the function partitioning algorithm. Introducing Pentium/133 (Paus) in the network, caused difficulties in balancing the workloads. Paus raw computing speed almost twice faster than the 486 DX4/100 and even almost 1.5 times faster than Raden. Balancing the workload to processors by distributing equal number of computing steps to each participating processor no longer yield to a good speed-up. Sahni and his co-worker [10] stated that in this heterogeneous environment, the balancing factor is no longer the computing steps or workload, but the *running time*. This means that the algorithm is said to be *balanced*, if and only if the execution time in all processors are equal. MFP-1 and MFP-2 tried to balance the workload by

```
Master
[initialisation of variables]
[initialised Processors]
sent data to all slaves;
for n = 1 to k do;
    compute data
    store results
efor;
receive data from slaves;
terminate slaves;
End Master.

Slave
[initialized variables]
receive data from Master
for m = k_i to j_i do;
    compute data
    store results
efor;
send data to master;
End Slave.
```

Figure 2
Basic master - slave algorithm

Note : If T is total number of the overall computing steps (both in master and slaves) then $T/(j_i-k_i)$ is the partitioning factor of the workload. $T/(j_i-k_i)$ is equal to the number of participating processors. scheduling MDP-93, MDG-93, MDG-94 and MDG-95 to different processors, either in pure function partitioning scheme or a combination between function and data partitioning scheme. In this scheduling scheme, each of MDP-93, MDG-93, MDG-94 and MDG-95 were treated as a single function.

CKP-96 [1] (check pointing and roll-back test) and MPC-96 [9] (multiple process creation) test programs were also developed to be exercised on the heterogeneous PC network. CKP-96 was developed to exercise the possibility of restarting parallel tasks in case of any unexpected failure during the run and MPC-96 was a test program to study the cost of multiple tasks creation form 25 up until 400 parallel tasks simultaneously. A communication test parallel program is being developed involving Raden and Bariah but the result is still inconsistent due to router problems. This program tests the capability of communication links between processors in delivering messages in various data types and sizes.

4. Lesson Learned

In the **homogeneous** Sun Sparc +1 network, MDG-94, MDG-94 and MDG-95 gave indications of network behaviour in running parallel programs. Results and analysis taken from [3] are given below.

Static Scheduling

The first data partitioning algorithm was the MDG-93, which simulates the dispersion of pollutant from a single industrial exhaust in the air. Relatively poor speed-up obtained was due to enormous ratio of T_{mp} over T_{par} (Tabel 3).

Tabel 3
Performance of MDG-93

T_{seq}	T_{par}	T_{mp}	% T_{mp}/T_{par}	Sp-up
1716.1	789.2	535.8	68.4	2.17

T_{seq}, T_{par},, T_{mp} are in milli seconds.
Speedup = T_{seq}/ T_{par}

By definition, MDG-93 algorithm is

medium grain, where the number of computing steps between two message passing activities is less than 10,000 [12].

The second data partitioning algorithm developed were the MDG-94. Speed-up in

MDG-94 with balanced workload is 5.15 using 6 processors.

Table 4
Performance of MDG-95

T_{seq}	T_{par}	T_{mp}	% T_{mp}/T_{par}	Sp-up
50621	9473	324	3.4	5.34

T_{seq}, T_{par},, T_{mp} are in seconds.
Speedup = T_{seq}/ T_{par}

Tabel 4 shows the important highlight of testing the MDG-95. It is clear that small ratio of T_{mp} over T_{par} gives better speed-up. This means that message passing activities has to be kept minimum. Selection of proper data types and providing sufficient message-passing buffer size are crucial steps in design process.
Heterogeneous PC network, has different behaviour. The imbalance in computing power of nodes demands a new task scheduling scheme. Integrating MDG-93, MDG-94 and MDG-95 and schedule each of them as a function, resulted in poor speed-up (1.11 for 3 processors, one of them is the PAUS as master) and low processor utilisation , 37 % (ratio of speed-up over number of processors, in percent). This means that load was not distributed equally, taking
into account the power of participating processors. After combining the scheduling both function and data partitioning, the performance was better (speed-up 1.22 and processor utilisation 40 % for 3 processors). There was no scheduling formula for balancing the workload, rather than best estimate based on the programmer's experience. Intuitive load balancing should be exercised in function partitioning, such that each participating processor, get the workload proportional to its computing power.

Dynamic Scheduling
The MPC-96 was aimed to simulate the distribution of workloads in the form of various kind of computing tasks or processes (assumed as *functions*) to study the effect of dynamic scheduling and communication overhead in the LAN based parallel computing platform. Creating multiple processes in network stations cost extra time, which contributes to the reduction of the overall speed-up. The tests made using MPC-96 program is highlighted in Tabel 5. The numbers presented in this table is the normalized creation time of tasks. For instance, the time to create 25 tasks in a particular configuration is taken as the reference.

Table 5
Cost of creating multiple tasks in MPC-96

# Task	2 proc	3 proc	4 proc	5 proc
25	1.00	1.00	1.00	1.00
50	2.01	1.85	1.75	1.72
100	3.77	3.63	3.62	3.36
200	8.06	7.01	6.88	7.44
400	15.57	14.71	14.58	12.63

If the time to create 25 tasks in two processors is 5.6 seconds, than time to create 400 tasks in that particular configuration is (15.57 X 5.6 seconds) = 87.192 seconds. This is a huge cost to bear for fine to medium grain parallel programs. But for coarse grain program that runs for hundreds of minutes, creating multiple tasks could be carefully considered as an alternative solution, depending on the algorithm.
The dynamic scheduling test of the MPC-96 shown promising results. It works in PC network and check pointing of parallel tasks could be done using the available PVM calls. Rolling back a terminated task should be done with some rigid procedures. Some modification should be made in CPK-96 to guarantee consistent roll-back process
in a heterogeneous network. The key factors should be considered in developing parallel programs are : ratio between message-passing and overall computing time, overhead time in creating tasks and also the balance of the workload in participating processors.

5. Further Development

The existing parallel computing platform has shown promising results. Despite the restrictions found during the experiments, some types of parallel algorithms could be implemented in the platform, that yields to a relatively good computing performance. Based on these facts, further development of the LAN based parallel computing platform is feasible. There is a chance to involve other institutions in University of Indonesia such as the Faculty of Natural Science. The fibre optic network that was developed in 1993, could be used to connect high end PC's already installed in faculties to function as a huge computing resource. In the nearest future, connection between Faculty of Engineering (40 Pentium PC's were installed in 1997) and the IUC-CS (it added more Sun Sparc +4 stations) could be realised. This network may be used as a cost effective test bed for parallel program development, that may support solutions of some high complexity problems. The remaining obstacle is the router. PVM could not distribute tasks or data through this device. We hope to overcome this problem in this nearest future.

High Speed Link
From the study we learned that message passing played a great role in determining the speed-up. The Ethernet bus speed (10 Mbit) contributed partly to this problem. We have decided to develop a new network of PCs at the Department of Electrical Engineering Faculty of Engineering, using fast Ethernet link (100 Mbit maximum transfer speed). This network will be linked to the University wide fibre network to form a larger parallel computing platform. Anyhow, if the router problem still persist, then we plan to ultilise the existing LAN at the Faculty of Engineering as our main platform.

6. Conclusion

1. In a homogenous LAN platform, a near linear speed-up could be obtained. Number of computing steps should be partitioned equally to create balanced workload. Ratio of computing and communication should be at least 10 or better, to get best result.
2. Scheduling multiple task or process on a LAN is time consuming. It is feasible only for coarse grain algorithms, where the ratio of task creation time and the overall computing time is small.
3. Balancing the *running time* in heterogeneous LAN couldn't be done by either data of function partitioning. Scheduling multiple tasks is one way to do that, provided that the tasks are coarse grain and they are independent.
4. Check pointing is feasible to be implemented in PVM platform, but rollback recovery procedure is more difficult to design. Standard PVM software doesn't support this procedure.

References

[1] Adrianto, " Implementation of Checkpointing and Rollback Recovery of PVM based Parallel Programs" Final Project, EE Dept. Faculty of Eng. Univ. of Indonesia, August (1996) (In Indonesian Lang.)

[2] Al Geist, et al, "Parallel Virtual Machine: A User Guide and Tutorials for Networked Parallel Computing", MIT Press, (1994).

[3] Bagio Budiardjo, "First Year Report of RUT III on Network Based Parallel Algorithm Development", Faculty of Eng. Univ. of Indonesia, Februari (1996) (In Indonesian Lang.)

[4] Bagio Budiardjo, Riri F. Sari, "Implementing a Parallel Algorithm for Calculating Dispersion of Pollutant from Single Stack", Research Report, Faculty of Eng. Univ. of Indonesia, December (1993) (In Indonesian Language).

[5] Bagio Budiardjo, Alexander Mario, "Implementation of a Parallel for Calculating Pollutant Dispersion from Multiple Stacks", Research Report, EE Dept. Faculty of Eng. Univ. of Indonesia, August (1994) (In Indonesian Language).

[6] Dewi Tristiana, "Implementation of a Parallel Algorithm for Caclculating Dispersion of Particalate Matters" Final Project, EE Dept. Faculty of Eng. Univ. of Indonesia, October (1993) (In Indonesian Language).

[7] Fajar Nashir, " Efficiency Analysis of a Network Based Parallel Process Scheduling", Final Project, EE Dept. Faculty of Eng. Univ. of Indonesia, Juli (1996) (In Indonesian Language).

[8] Nurfitri, "Implementing Parallel Algoritms with Function Partitioning", Final Project, EE Dept. Faculty of Eng. Univ. of Indonesia, January (1996) (In Indonesian Lang.)

[9] Satrio Budiwibowo, "Modelling Communication Overhead of Multiple Creation on a PVM Network", Final Project, EE Dept. Faculty of Eng. Univ. of Indonesia, July (1996) (In Indonesian Language).

[10] Sartaj Sahni and Venkat Thanvantri, "Performance Metrics : Keeping the Focus on Runtime", IEEE Parallel & Distributed Technology Volume 4 Number 1, pp. 43-55, Spring (1996).

[11] Wira Satyawan, "Scheduling Parallel Process for FLLS Model" Final Project, EE Dept. Faculty of Eng. Univ. of Indonesia, July (1995) (In Indonesian Language).

[12] John Van Zandt, "Parallel Processing in Information System : with examples and cases", Wiley, New York (1992).