PAMUKKALE ÜNİVERSİTESİ MÜHENDİSLİK FAKÜLTESİ
PAMUKKALE UNIVERSITY ENGINEERING COLLEGE
MÜHENDİSLİK BİLİMLERİ DERGİSİ
JOURNAL OF ENGINEERING SCIENCES
YIL : 2001
CİLT : 7
SAYI : 2
SAYFA : 229-233

ADAPTATION OF PARALLEL VIRTUAL MACHINES MECHANISMS TO PARALLEL SYSTEMS

***Zafer DEMİR, **Aşkın DEMİRKOL**
*Sakarya University, Engineering Faculty, Electrical-Electronic Engineering Department, Sakarya
**Sakarya University, Engineering Faculty, Computer Engineering Department, Sakarya

Geliş Tarihi : 29.01.2001

ABSTRACT

In this study, at first, Parallel Virtual Machine is reviewed. Since It is based upon parallel processing, it is similar to parallel systems in principle in terms of architecture. Parallel Virtual Machine is neither an operating system nor a programming language. It is a specific software tool that supports heterogeneous parallel systems. However, it takes advantage of the features of both to make users close to parallel systems. Since tasks can be executed in parallel on parallel systems by Parallel Virtual Machine, there is an important similarity between PVM and distributed systems and multiple processors. In this study, the relations in question are examined by making use of Master-Slave programming technique. In conclusion, the PVM is tested with a simple factorial computation on a distributed system to observe its adaptation to parallel architects.

Key Words : PVM, Parallel systems, Parallel processor, Multiple processor, Distributed system

PARALEL SANAL MAKİNE MEKANİZMALARININ PARALEL SİSTEMLERE UYARLANMASI

ÖZET

Bu çalışmada, paralel mimariler için geliştirilmiş Paralel Virtual Makineler incelenmiştir. Bu makineler, çalışma prensibi itibari ile mimari açıdan paralel sistemlere benzemektedirler. Yapı olarak ne bir işletim sistemi ne de bir programlama dilidir. Daha ziyade, özel yeteneklere sahip bir yazılım, bir malzeme olarak heterogen yapıdaki paralel sistemleri desteklemektedir. Bununla beraber, her ikisinin de özelliklerinden yararlanarak kullanıcıları, paralel sistemlere yaklaştırabilme yeteneğindedir. Paralel Virtual Makine ile dağıtılmış ve çoklu işlemci sistemleri arasında önemli bir benzerlik dikkati çekmektedir. Bu çalışmada, söz konusu benzerlik ilişkileri, Parallel Virtual Makinelerin Master-Slave prensibine göre analiz edilmiştir. Basit bir faktöryel hesabının paralel sistemlere adaptasyonu ile ilgili benzerlik test edilmiştir.

Anahtar Kelimeler : PVM, Paralel sistemler, Paralel işlemci, Çoklu işlemci, Dağıtılmış sistem

1. INTRODUCTION

After rapid advances in computing technologies, in particular, researches in scientific and engineering fields need more stronger solutions than classic technologies such as processing speed (Mano, 1993). In many fields, processors with high speed have begun to be used unavoidably. Because data or process should be executed as quickly as possible to have an optimum and quick result. Therefore, classic systems with single processor were turned into parallel processors inevitably. In general, parallel systems are divided into two parts as distributed and multiprocessor systems (Sunderam, 1990).

Distributed systems involving many independent computers are connected to one another in a network system (Geist and Sunderam, 1992). They can run the processes separately. Besides, they can be

organised in a way that they can execute the current assigned job such as one by one in parallel.

Multiprocessors include many independent and identical processors (Kumar et all., 1994). The process in the system in consideration is separated into the identical processors mentioned by using a suitable operation system. Since in the end, all tasks are performed in parallel at the same time, the response of the system is naturally expected more quickly. If the tasks in question can also be assigned equally, the performance can be risen more than expected (Tabak, 1990).

Parallel Virtual System (PVM) is an important and useful tool to be improved for parallel programming (Dongarra et all., 1993). PVM is able to run the present processes in parallel by supporting current parallel hardware. It provides an appropriate environment for many applications to be performed (Beguelin et all., 1995).

If PVM is examined together with either distributed or multiprocessor, it is realised that there is an appreciable, even important similarity among them. First of all., these relations are reviewed in this study. Furthermore, distributed system is considered as a parallel computer to examine and compare the related properties by PVM. Consequently, the adaptation of PVM to distributed systems is tested in this respect by an array application.

2. PARALLEL SYSTEMS

Parallel systems are split into multiprocessor and distributed structures in terms of hardware. While distributed system is a collection of independent computers that appear to the users of the system as a single computer, multiprocessors are composed of identical processors that execute present job in parallel (Tanenbaum, 1995). In conclusion, as long as the hardware system is organised by many different or same computer, or identical processors to be run at the same time, in general, it is called parallel system (Feitelson, 1997). It is implemented either distributed or multiprocessor system. As a shared multiprocessor system, the architecture of a general parallel processor system in consideration is shown in Figure 1 (Gokhale and Holmes, 1995).

The parallel system above ought to be supported with an appropriate operation system. While the parallel system is acted, any job separated into tasks is executed by assigning to the present processors. In this way, the performance of such a system is expected to be much more, compared to classic with single processor. The performance mentioned is supposed to be the number of processor in the system theoretically (Nanda,1992).

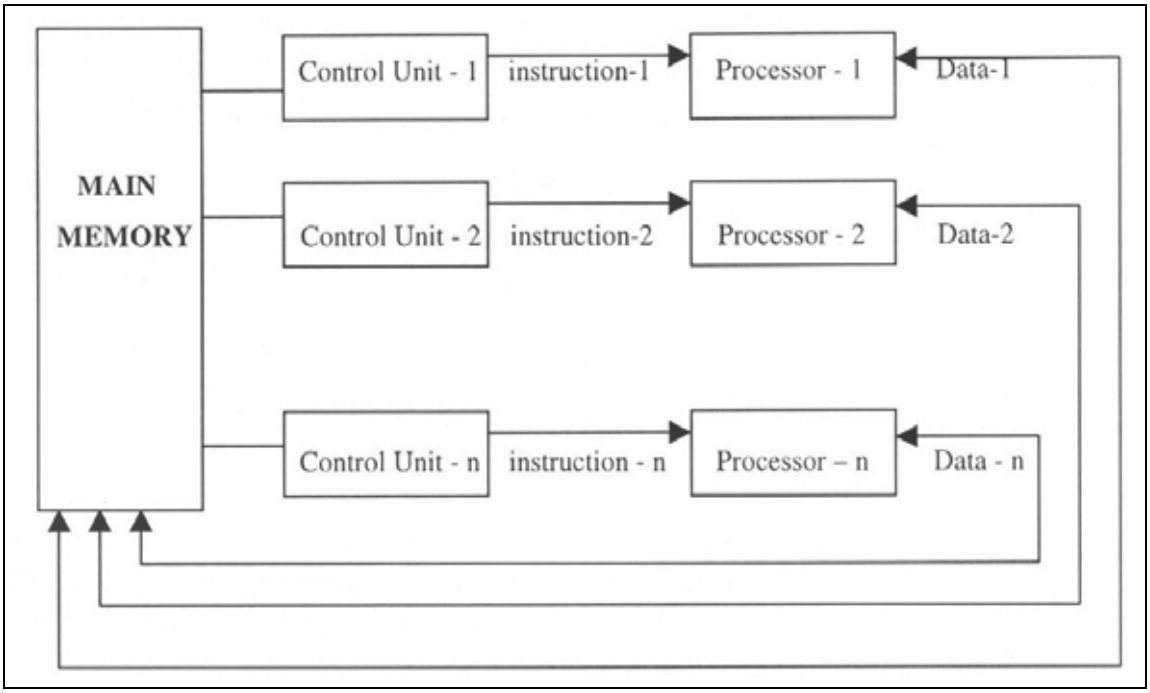

Figure 1. A parallel processor system

3. PARALLEL VIRTUAL MACHINES

Parallel Virtual Machine (PVM) is a software tool that provides users with a parallel environment to compile their programmes. In other words, it is neither programming language nor operation system. Although PVM seems like a sort of parallel programming software, as a tool, basically it has more private features. It is developed according to massage-passing principle by taking advantage of Unix. In addition, PVM is also not just only parallel programming language, in any case it is designed by using C programming language. As a result, PVM, further than operation system, is based upon an operation system and as a specific software, is stronger than a programming language (Fagg and Dongarra, 1994).

PVM is used either a network system based upon Unix or multiprocessor such as distributed-memory or shared-memory. In other words, it is a tool to be improved parallel systems (Buguelin et all., 1992). As, the Unix computers are viewed a single computer by PVM, users can make use of its power. PVM is a very useful project which is collaborated between Oak Ridge National Laboratory, the University of Tennessee, Emory University and Carnegie Mellon University in 1989. The first draft of it began to be used in the same year as PVM 1.0. Then, after the other releases, in the end, the last version is drafted in 1993 such as PVM 3.3 (Geist et all., 1994). After its production completed, it has been distributed to Universities and related researching centres to be utilised.

PVM is so important and useful that it is able to organise any network involving computers with different properties. The property in question is called heterogeneity such as architecture, data format, computational speed and network load (Sunderam et all., 1994). PVM also supports heterogeneous computers located as network. PVM is designed to link computing resources and provide users with a parallel platform for running their computer applications. Consequently, when PVM is worked, it is capable of harnessing the combined resources of typically heterogeneous networked computing platforms to deliver high levels of performance and functionality.

The significant operations executed on a PVM are the following (Sunderam et all., 1994a);

1. Writing a program
2. Building PVM on a system
3. Starting PVM on a set of machines
4. Debugging a PVM application

Before PVM is run, at first, any program is formed by the PVM editor. In this respect, the program can be coded by any suitable C editor. Then, PVM is adapted on the current parallel system. After these, the present program belonged to users is performed on related parallel configuration by the support of the PVM.

4. ADAPTATION OF PVM TO PARALLEL SYSTEMS

There are strong relations between PVM and parallel systems. PVM is developed to encourage researches in parallel processing field. First of all., it is only designed as a distributed systems. However, with the later versions, in addition to many new improvements, it is redesigned to broaden multiprocessor systems too (Geist and Sunderam, 1993). Although a multiprocessor system generally comprises identical processors, this is not important in PVM. Actually, PVM is explored for heterogeneous platforms. Therefore, properties of computers or processors in a parallel system do not need a further procedure, while they are adapted to PVM. Power of the PVM arises from heterogeneity. In other words, since non homogeneous is included, PVM is a strong tool.

Master-Slaves Paradigm of the general programming techniques of the PVM mentioned is the following (Lewis and Oline, 1993).

As shown, a Master-Slave technique on the PVM is comprises of two parts. They are just master and slaves. Slaves are not considered as two parts. Although they appear to be as such, both receiver and sender slaves are the same parts of the PVM. They are shown this way to imagine the communication between master and slaves.

The slaves in question function as both receiver and sender. Basically, slaves form parallel computer system at the same time. First of all., any written program is assigned or, sent to slaves as tasks. This group of slaves are now called “receiver”. When they complete their processes’ , they are renamed as “sender”, as the executed tasks are sent back. While whole parallel responsibility on the target program is provided by PVM until obtaining the results, PVM is supported by Unix as operation system.

The detailed situation of a Master-Slave technique in Figure 2 concerning PVM is in Figure 3 below (Geist,1994);

As shown below, in fact, receiver and sender slavers are the same part of PVM. They are single and the

same. Nevertheless, they perform different functions. Since they wait for the tasks to start, they are at receiver position. After they finish their works, they are at sender position. In conclusion, both receiver and sender slaves form the parallel computer or processor system in consideration.

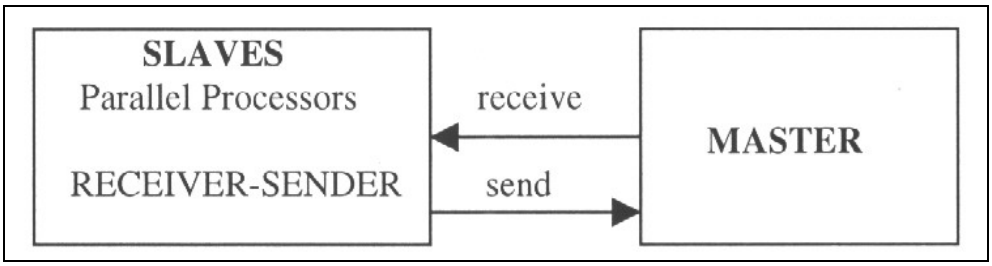

Figure 2. General view of a master-slave for PVM system

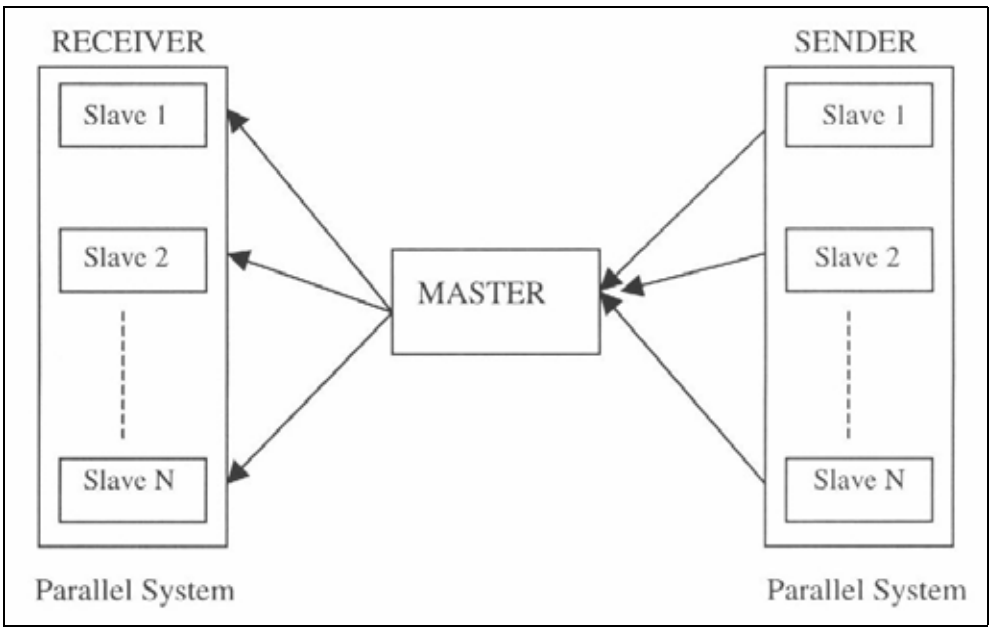

Figure 3. Working principle of a PVM system

In the end, a simple example is tested by PVM to observe its performance. Problem is based on a factorial calculation. Computation of factorial of 20 is aimed with such a system. Thus, any array consisting of 20 numbers is considered. Four processors are planned for this process. Each one takes responsibility for 5 consecutive numbers. Once the numbers mentioned are computed by each processor, the results are sent Master to compound them by multiplying. The result implemented in each processor are as the following ;

As shown in Figure 4, when these results are sent to the Master, it finalises the process by multiplying the four numbers from slaves, or processors.

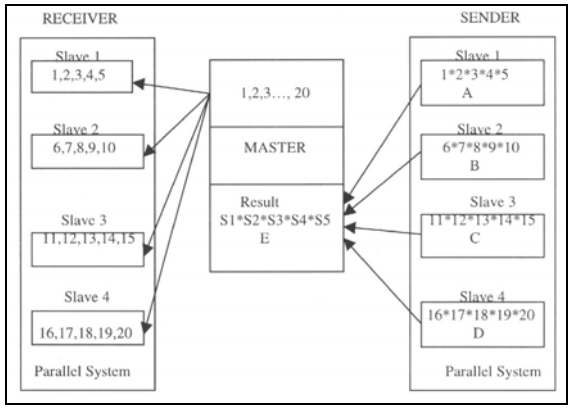

Figure 4. Computation of factorial 20

5. CONCLUSION

In this study, a Parallel Virtual Machine is reviewed. Similarities and present relations are searched between parallel systems and PVM. It is observed that PVM is a strong tool to develop for the distributed and multiprocessors. Besides, since it supports heterogeneous system that involves different computers or processors, it makes easy parallel processing applications for users as well. In addition, as it can be used together with Unix, it provides further advantage, because, in this way, there is no necessary any special operation system. As a result, all of these properties mentioned in the PVM are tested practically by a simple factorial computation by using Master-Slave technique.

6. REFERENCES

Beguelin, A., Dongarra, J., Geist, A., Marchek, R., Moore, K., Wade, R., Plank, J., Sunderam, V. S. 1992. "A Users' Guide Version 2.0", Univ., of Tennessee Computer Science Dept., CS92-157, February.

Beguelin, A., Geist, G. A., Marchek, R., Sunderam, V. S. 1995. "Recent Enhancements to PVM", International Journal for Supercomputer Applications, 9 (2), 108-127.

Dongarra, J. J., Geist, G. A., Marchek, R., Sunderam, V. S. 1993. "Supporting Heterogeneous Network Computing: PVM", Chemical Design Automation News, September/October 1993, 8, (9/10), 1-16.

Fagg, G. E., Dongarra, J. J. 1994, "PVMI: An Integration of the PVM and MPI Systems", Technical Report, University of Tennessee.

Feitelson, D. G. 1997. " Job Scheduling in Multiprogrammed Parallel Systems", IBM Research Report, RC19970 Second revision 1997, p. 1-42.

Geist, G. A. Sunderam, V. S. 1992. "Network Based Concurrent Computing on the PVM System", Journal of Concurrency: Practice and Experience, 4 (4), 293-311.

Geist, G. A, Sunderam, V. S. 1993. " The Evolution of the PVM Concurrent Computing System", Proceedings-26th IEEE Compcon Symposium, pp. 471-478.

Geist, A. 1994. Parallel Virtual Machines-A User's Guide and Tutorial for Networked Parallel Computing, The MIT Press.

Geist, A., Buguelin, A., Dongarra, J., Jiang, W., Marchek, R., Sunderam, V. 1994. "PVM : Parallel Virtual Machine A User's Guide and Tutorial for Networked Parallel Computing", Scientific and Engineering Series, MIT Press, pp. 1-2.

Gokhale,M., Holmes, B. 1995. " Processing in Memory ", IEEE Computer, 28 (4), April.

Kumar, V., Grama, A., Gupta, A., Karypis, G. 1994. "Introduction to Parallel Computing, Design and Analysis of Algorithms", pp.16-23.

Lewis, J. M., Oline Jr, E. R. 1993. "PVM Comunication in Switched FDDI Heterogeneous Distributed Computing Environments", Proceedings of the IEEE Workshop on Advances in Parallel and Distributed Systems, pp.13-19, October.

Mano, M. M. 1993. Computer System Architecture, p.128-135.

Nanda, A. 1992. A framework for Multiprocessor Performance Characterisation and Calibration, Phd thesis, Michigan State University, East Larging, Department of Computer Science, p.1-60.

Sunderam, V.S. 1990. "PVM : A Framework for Parallel Distributed Computing", Concurrency: Practice and Experience, 2 (4), 315-339.

Sunderam, V. S., Geist, G. A., Dangarra, J. and Marchek, R. 1994. "The PVM Concurrent Computing System : Evolution, Experiences, and Trends", Parallel Computing, 20 (4), 531-546.

Sunderam, V. S., Geist, G. A., Dangarra, J. and Marchek, R. 1994a. "PVM 3 User's Guide and Reference Manual", Technical Report, ORNL/TM-12187, Oak Ridge National Laboratory, Oak Ridge, Tennessee, 37831, pp. 33-39.

Tanenbaum, A. S. 1995. Distributed Operating Systems, p. 1-14.