Evaluation of Process Migration for Parallel Heterogeneous Workstation Clusters

M.A.R. Dantas

Computer Science Department,
University of Brasilia (UnB),
Brasilia, 70910-900, Brazil
mardantas@computer.org

Abstract. It is recognised that the availability of a large number of workstations connected through a network can represent an attractive option for many organisations to provide to application programmers an alternative environment for high-performance computing. The original specification of the Message-Passing Interface (MPI) standard was not designed as a comprehensive parallel programming environment and some researchers agree that the standard should be preserved as simple and clean as possible. Nevertheless, a software environment such as MPI should have somehow a scheduling mechanism for the effective submission of parallel applications on network of workstations. This paper presents the performance results and benefits of an alternative lightweight approach called *Selective - MPI* (S-MPI), which was designed to enhance the efficiency of the scheduling of applications on parallel workstation cluster environments.

1 Introduction

The standard message-passing interface known as MPI (Message Passing Interface) is a specification which was originally designed for writing applications and libraries for distributed memory environments. The advantages of a standard message-passing interface are portability, ease-of-use and providing a clearly defined set of routines to the vendors that they can design their implementations efficiently, or provide hardware and low-level system support.Computation on workstation clusters when using the MPI software environment might result in low performance, because there is no elaborate scheduling mechanism to efficiently submit processes to these environments. Consequently, although a cluster could have available resources (e.g. lightly-loaded workstations and memory), these parameters are not completely considered and the parallel execution might be inefficient.This is a potential problem interfering with the performance of parallel applications, which could be efficiently processed by other machines of the cluster.

In this paper we consider the *mpich* [2] as a case study implementation of MPI. In addition, we are using ordinary Unix tools therefore concepts employed

in the proposed system can be extended for similar environments. An approach called **Selective - MPI (S-MPI)** [1], is an alternative design to provide MPI users with the same attractive features of the existing environment and covers some of the functions of a high-level task scheduler. Experimental results of demonstrator applications indicate that the proposed solution enhances successfully the performance of applications with a significant reduction on the elapsed-time. The present paper is organised as follows. In section 2, we outline some details of the design philosophy of the S-MPI. Experimental results are presented in section 3. Finally, section 4 presents conclusions.

2 S-MPI Architecture

Figure 1, illustrates the architecture of the S-MPI environment compared to the *mpich* implementation. Difference between the two environments are mainly the layer where the user interface sits and the introduction of a more elaborate scheduling mechanism.

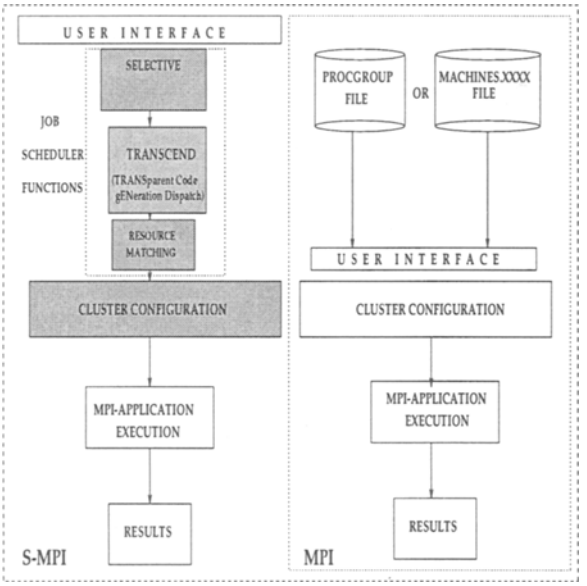

Fig. 1. Architecture differences between S-MPI and *mpich*.

The S-MPI approach covers functions of a high-level task scheduler monitoring workstations workload, automatically compiling (or recompiling) the parallel code, matching processes requirements with resources available and finally selecting dynamically an appropriate cluster configuration to execute parallel applications efficiently.

3 Experimental Results

Configurations considered in this paper use a distributed parallel configuration composed of heterogeneous clusters of SUNs and SGIs (Silicon Graphics) workstations, in both cases using the *mpich* implementation. These workstations were private machines which could have idle cycles (i.e. considering a specific threshold of the average CPU queue length load index of the workstations) and memory available. Operating systems and hardware characteristics of these workstations are illustrated by table 1.

Table 1. General charateristics of the SUN and SGI clusters.

CLUSTERS	SUN	SGI
Machine Architecture	**SUN4u - Ultra**	**IP22**
Clock(MHz)	**143-167**	**133-250**
Data/Instruction Cache (KB)	**64**	**32-1024**
Memory(MB)	**32-250**	**32-250**
Operating System	**SOLARIS 5.5.1**	**IRIX 6.2**
MPI version	***mpich* 1.1**	***mpich* 1.1**

We have implemented two demonstrator applications with different ratio between computation and communication, The first application, was a gaussian algorithm to solve a system of equations. The second algorithm implemented was a matrix multiplication, which is also an essential component in many applications. Our algorithm is implemented using a *master-slave* interaction.

The improved performance trend on the S-MPI environment executing the matrix multiplication is presented in figure 2. This figure demonstrates that the S-MPI has a better performance than the MPI configurations. The SUN-MPI and SGI-MPI environments represent the use of the specific machines defined in the static *machines.xxx* files. The SUN/SGI-MPI configuration shows results of the use of the *procgroup* file. This static file was formed with the address of heterogeneous machines from both SUN and SGI clusters. Finally, the results of SUN/SGI-SMPI illustrated the use of the automatic cluster configuration approach of the S-MPI.

Figure 3 shows the enhanced performance of the S-MPI approach compared to the MPI for solving a system of 1000 equations. This figure shows a downward trend of the elapsed-time for the S-MPI environment in comparison to the conventional SUN and SGI MPI environments.

4 Conclusions

In this paper we have evaluated the performance results and benefits of a lightweight model called **Selective - MPI (S-MPI)**. The system co-exists cleanly with the

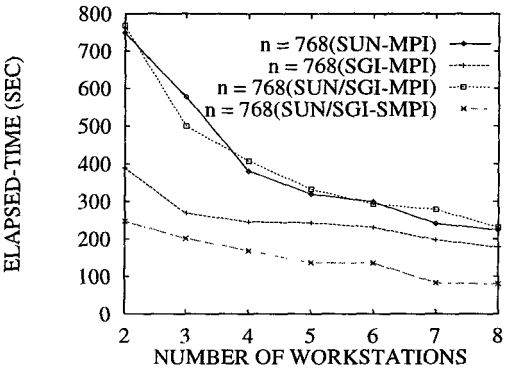

Fig. 2. Elapsed-time of the matrix multiplication on MPI and S-MPI environments.

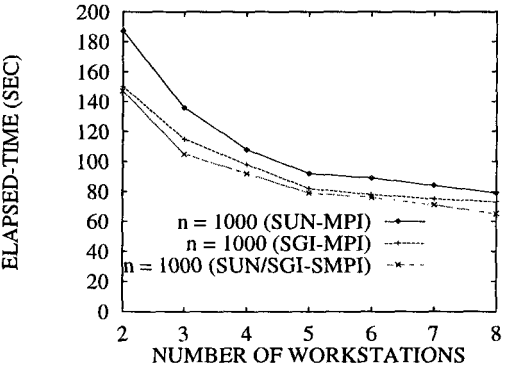

Fig. 3. Elapsed-time of the Gaussian algorithm on MPI and S-MPI environments.

existing concepts of the MPI and preserves the original characteristics of the software environment. In addition, S-MPI provides application programmers with a more transparent and enhanced environment to submit MPI parallel applications without layers of additional complexities for heterogeneous workstation clusters. Moreover, the concept clearly has advantages providing users with the same friendly interface and ensuring that all available resources will be evaluated before configuring a cluster to run MPI applications.

References

1. M.A.R. Dantas and E.J. Zaluska. Efficient Scheduling of MPI Applications on Network of Workstations. *Accepted Paper on Future Generation Computer Systems Journal*, 1997.
2. William Gropp Patrick Bridges, Nathan Doss. Users' Guide to mpich, a Portable Implementation of MPI. *Argonne National Laboratory* `http://www.mcs.anl.gov`, 1994.