MiMPI: A Multithread-Safe Implementation of MPI

F. García, A. Calderón, J. Carretero

Facultad de Informática, Universidad Politécnica de Madrid (UPM) Campus de Montegancedo, Boadilla del Monte, 28660 Madrid, Spain
fgarcia@fi.upm.es

Abstract. In this paper, we present a new thread-safe implementation of MPI, called MiMPI, that allows efficient development of thread-safe parallel programs using the message-passing standard MPI. MiMPI uses multithread operations to increase the performance of collective communications. The paper describes the main design goals of MiMPI and the current implementation. Finally we present some performance results, obtained on an IBM SP2, comparing the performance with others MPI implementations.

Keywords: Parallel programming, message-passing, MPI, threads.

1 Introduction

MPI [MPI95] defines a standard interface for the message-passing model of parallel computation. MPI was intended to facilitate widespread portability of programs among diverse parallel architectures. With this aim, the primary goals of the MPI specification are: efficiency, portability, and functionality.

Threads are useful to improve application performance. A program with only one thread of control must wait each time it requests a service from the operating system. Using more than one thread lets a process overlap processing with one or more I/O requests.

Using threads in conjunction with message passing and the use of a multi-thread implementation of MPI can be extremely convenient, for several reasons:

- Threads provide a natural implementation of nonblocking communication operations. A thread can be created to do a blocking receive operation. This operation blocks only the thread and not the process. The same applies to sends.
- Threads can increase the efficiency of the implementation of collective operations.
- Threads provide a natural way to implement operations required for the shared-memory operations. A separate set of threads, one for each process, can handle the data management functions on behalf of a process while a main thread in each process performs the main computation.
- Threads are becoming the parallel programming model for symmetric multiprocessing shared-memory machines.
Threads are specially important for client/servers applications. Server programs in client/servers applications may get multiple requests from independent clients simultaneously.

Threads can improve performance by helping to reduce communication latency.

MPI specification defines a thread-safe semantic. Thread safety means that multiple threads can be executing message-passing library calls without interfering with one another. MiMPI is a multithread implementation of MPI, that uses threads to increase the performance of some operations and provides an implementation that can be used in multithread applications.

Currently MiMPI implements a subset of all MPI functions, and have been used successfully in ParFiSys [Garcia98], a parallel file system developed at the UPM. MiMPI is available for cluster of workstations and IBM SP2 platforms.

The rest of this paper is organized as follows. Section 2 surveys related work about MPI implementations. In section 3, we discuss our thread-safe implementation of MPI. Section 4 provides some experimental results comparing MiMPI with other implementations. The experiments have been made on an IBM SP2. Finally, section 5 summarizes our conclusions.

2 Related Work

MPI [MPI95] [Gropp95] (message-passing interface) is a message-passing application programmer interface, together with protocol and semantic specifications for how its features must behave in any implementation.

MPI includes point to point message passing and collective operations. A collective operation must be performed by all the processes in a computation. MPI supports two kinds of collective operations: data movement operations and collective computation operations. Others features included in MPI are: virtual topologies, debugging and profiling, communication modes, and support for heterogeneous networks.

In 1997 appears MPI-2. MPI-2 contains clarifications and corrections to the initial standard and describes additions to this standard. These include miscellaneous topics, process creation and management, one-sided communications, extended collective operations, external interfaces, I/O, and additional language bindings.

Currently, there are many implementations of MPI, with both free available and vendor-supplied implementations, but few implementations provide a fully thread-safe semantic. Some of these systems are as follows:

- LAM [Burns94] is available from the Ohio Supercomputer Center and runs on heterogeneous networks of Sun, DEC, SGI, IBM, and HP workstations.
- CHIMP-MPI [Alasdair94] is available from the Edinburgh Parallel Computing Center and runs on Sun, SGI, DEC, IBM, and HP workstations, the Meiko Computing Surface machines. This implementation is based on CHIMP [Clarke94].