ANL/RA/CP--88061
CONF-960482--3

To be presented at the High Performance Computing '96 Meeting
March 31 - April 4, 1996, New Orleans, Louisiana.

Study of Parallel Efficiency in Message Passing Environments

by

Ulf R. Hanebutte, and Masahiro Tatsumi*

Reactor Analysis Division
Argonne National Laboratory
9700 South Cass Avenue
Argonne, IL 60439 USA
(708) 252-1866

*Department of Nuclear Engineering
Osaka University
2-1, Yamadaoka, Suita, Japan

* Work supported by the U.S. Department of Energy, Nuclear Energy Programs under Contract W-31-109-ENG-38.

Study of Parallel Efficiency in Message Passing Environments *

Masahiro Tatsumi † Ulf R. Hanebutte ††

† Department of Nuclear Engineering
Osaka University
2-1, Yamadaoka, Suita, Japan

†† Reactor Analysis Division
Argonne National Laboratory
Argonne, IL 60439

Abstract

A benchmark test using the Message Passing Interface (MPI, an emerging standard for writing message passing programs) has been developed, to study parallel performance in message passing environments. The test is comprised of a computational task of independent calculations followed by a round-robin data communication step. Performance data as a function of computational granularity and message passing requirements are presented for the IBM SPx at Argonne National Laboratory and for a cluster of quasi-dedicated SUN SPARC Station 20's. In the later portion of the paper a widely accepted communication cost model combined with Amdahl's law is used to obtain performance predictions for uneven distributed computational work loads.

1 Introduction

The parallelization of engineering and physics codes that contain computational tasks of relatively fine granularity interlaced by data communication is not a trivial task. It is important to obtain an understanding of the behavior of parallel efficiency as a function of the computational granularity and the message passing requirements. What parallel efficiency can be expected for an algorithm, given its computational and communicational requirements? How rapidly does the efficiency of the parallel algorithm decrease, if the cost for message passing cannot be compensated by the computational costs? What changes to the algorithm can be proposed to increase the efficiency? As an example, a parallel Monte Carlo Eigenvalue solver [1] achieves good performance by calculating many hundreds of histories on each processor before data is communicated, rather than just a single history. A separate question is, which should be the target architecture for the parallelization, a dedicated parallel computer with high performance communication or a cluster of workstations? In order to study these issues a benchmark test using the Message Passing Interface (MPI) [2] has been developed and performance data on both the IBM SPx at Argonne National Laboratory and a cluster of quasi-dedicated SUN SPARC Station 20's have been collected.

*Work supported by the U.S. Department of Energy, Nuclear Energy Programs, under Contract W-31-109-ENG-38.

2 Numerical Experiments

2.1 Specification

The Message Passing Interface has been selected for the presented case study because it provides an efficient and portable framework for writing message passing programs. The objective of the study is to collect measurements of the parallel efficiency as a function of the computational granularity and the

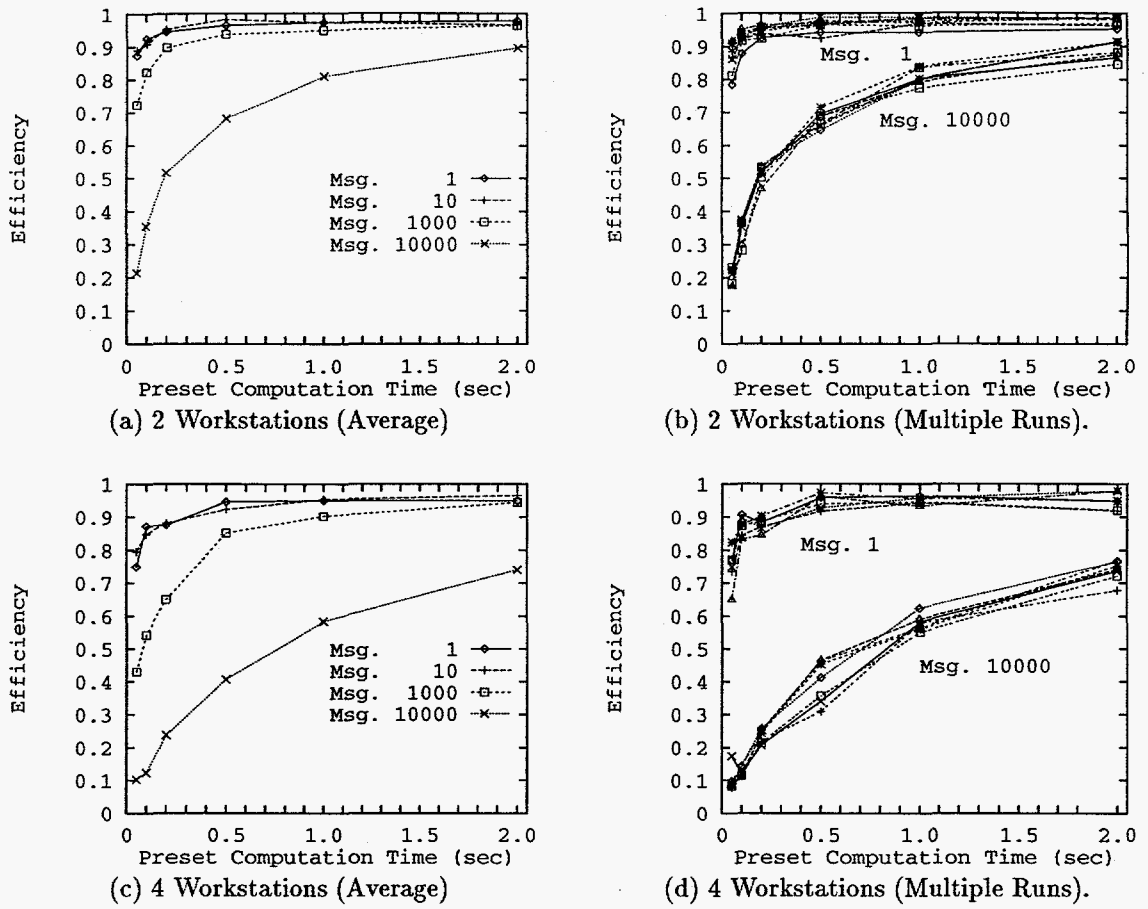

Figure 1: Results obtained on 2 and 4 SUN SS-20 workstations connected by Ethernet

message size of the data communication. The computational task is simulated by a subroutine, which consumes precisely a preset amount of CPU (central processing unit) time, T. For the first part of the study the total computing time is divided into equal fractions for each processor. In other words, each processor spends T/n_p time computing, where n_p denotes the number of processors. Speed-up can be expressed as a factor by which the execution time is reduced by a parallel execution on n_p processors compared to the serial execution:

$$\text{Speed-up} = \frac{T}{T/n_p + T_{\text{com}}} \quad , \qquad (1)$$

where T_{com} denotes the communication time. To compare the parallel performance of an algorithm for various number of processors, it is convenient to plot efficiency rather than Speed-up. Efficiency is defined as

$$\text{Efficiency} = \frac{Speed - up}{n_p} = \frac{T}{T + n_p T_{\text{com}}} \quad . \qquad (2)$$

Following the computation, a round-robin data exchange is performed. Each processor has to send one message of fixed size to its "right-neighbor" processor and receives one message from its "left-neighbor". We have chosen the common paradigm, where half the processors are sending simultaneously, while the other half is receiving. The message size is being varied throughout the study. The data is sent as FORTRAN Integers, which are a 4-byte data type. Thus, a message size of one equals 4 bytes.

In the second part of the study, load imbalance is investigated. Based on Amdahl's law [3], the total computing cost is divided into a serial and parallel fraction. By varying the serial fraction, load imbalance is modeled.

2.2 Environments

The following environments were chosen for the benchmark study. 1.) The portable implementation of MPI called MPICH, which has been developed jointly by Argonne National Laboratory and Mississippi State University. 2.) The IBM SPx system, located at ANL, with a communication network consisting of high performance switches. 3.) A cluster of SUN SPARC Station 20 workstations on Ethernet. A scheduler guarantees dedicated operations on the IBM SPx. To simulate a quasi-dedicated workstation cluster, the SUN SS-20 measurements were performed several times over night, while all batch job queues were disabled. A relatively small number of processors (2 and 4) are utilized for the case study, since small sets of powerful workstations connected by Ethernet are most commonly found in today's engineering offices. During each benchmark execution, timing measurements were taken for 20 iterations and average values are recorded.

3 Results and discussion

3.1 Cluster of quasi-dedicated SUN SPARC Station 20's

Results obtained on a quasi-dedicated SUN SS-20's for the case of perfect load balance are shown in Fig. 1. While Fig. 1(a) and 1(c) give the average values over all performed benchmark runs, Fig. 1(b) and 1(d) show the average results for each individual run performed throughout the night. For each case, the individual results form a clear band of non-negligible bandwidth. The limitations of the system consisting of workstations connected by Ethernet are clearly visible. The start-up latencies for sending messages give rise to a rapid decrease in the parallel efficiency for small total computing times. The limited bandwidth (1Mb/s) of the Ethernet further affects the efficiency negatively in the case of large message sizes. (The break-even points are 50% and 25% parallel efficiency for the two and four processor case respectively.)

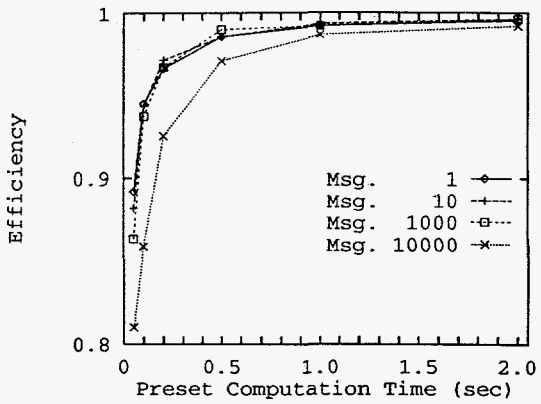

Figure 2: Result on ANL's IBM SPx using 2 processors.

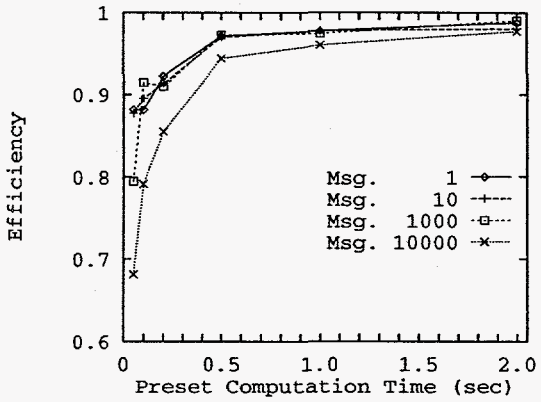

Figure 3: Result on ANL's IBM SPx using 4 processors.

3.2 ANL's IBM SPx

The measurements for the IBM SPx are given in Fig. 2 and 3. The advantage of having a scalable high-speed network is clearly visible if compared to the workstation results presented in Fig. 1. Note that the efficiency scales for Fig. 2 and 3 start at 0.8 and 0.6 respectively, compared to 0.0 for all other plots presented in this paper. However, the qualitative behavior of the parallel efficiency as a function of computational grain and message size is similar. For messages of 1,000 words and less, the efficiency results lie in a very narrow band, only the large message of 10,000 words causes a clear drop in performance.

3.3 Performance Model

To discuss the effect of uneven distributed work load, a performance model will be introduced. The communication cost $T_{\rm com}$ for the above described round-robin data exchange can be estimated for an Ethernet communication network [4], by

$$T_{\rm com} = 2(t_{start} + t_w \frac{n_p}{2} L) \quad . \tag{3}$$

The coefficients t_{start} and t_w denote the message startup time and the transfer time per 4-byte word respectively. Approximate parameter values can be found in the literature, e.g. Refs. [4], [5], and [6], or be obtained by performing a simple "ping-pong" message exchange between two processors. For the SUN workstation cluster connected on Ethernet t_{start} and t_w assume the value of $1,500\,\mu sec$ and $5\,\mu sec$ respectively [4]. Although not needed here, more complex and accurate communication models, such as given in Ref. [7], may be substituted for Eq.3. The t_{start} and t_w values cited above represent "best achievable" communication performance and thereby give a lower bound for the communication costs, which in turn leads to over-estimating the achievable parallel efficiency.

According to Amdahl's Law, total computing time can be divided into two parts,

$$T = T_s + T_p \quad . \tag{4}$$

$$T_s = s \cdot T, \quad T_p = p \cdot T \quad . \tag{5}$$

Here s and $p = 1 - s$ represent serial and parallel fraction, respectively. The parallel efficiency can be expressed by:

$$\text{Efficiency} = \frac{T}{pT + n_p(sT + T_{\rm com})} \quad , \tag{6}$$

with $T_{\rm com}$ given by Eq.3. Fig. 4 and 5 depict parallel efficiency results based on Eq.6. The left column of Fig. 4 gives efficiencies for 2, 4, and 8 processors, if the message size is only one word, while the right column gives the results for a message length of 10,000 words. As expected, the adverse effect of load-imbalance is stronger for the low communication case. A small serial fraction of only 1% causes a noticeable drop in efficiency for 4 processors, as seen in Fig. 4(c). For large messages, the high communication costs on the bandwidth limited Ethernet lead to poor performance, even when the computation time is in the range of 1 to 2 seconds. In Fig. 5, a three-dimensional plot provides a summary of the performance model results for the case of perfect load-balance.

4 Conclusion

Experiments for measuring parallelization efficiency with simulated computation time were performed on both a cluster of quasi-dedicated SPARC-20s and the IBM SPx system. If parallelization is implemented with long computation time and less frequent message passing, it almost reaches ideal efficiency. However, with short total computation and large number of processors it could become impossible to parallelize efficiently because of the time required for message passing, even if one uses dedicated systems such as the IBM SPx. This may indicate, that a shared memory multi processor system might be needed in such cases to obtain good efficiencies.

It is also important to maximize load balancing for each processor, because an increase of the serial fraction rapidly decreases the efficiency even for moderate numbers of processors.

A quasi-dedicated workstation cluster has been utilized for the present study. A discussion of parallel performance for non-dedicated clusters can be found in Ref. [8]; however, that study has no data communication between parallel tasks. The present study examines a typical computation/communication pattern found in many engineering and physics codes. The performance figures presented in our study may assist a parallel algorithm developer in making the appropriate parallel implementation strategy decisions.

Acknowledgments

The authors gratefully acknowledge use of the Argonne National Laboratory High-Performance Computing Research Facility (HPCRF). The HPCRF is funded principally by the U.S. Department of Energy Office of Scientific Computing.

References

[1] Matsuura, S., Brown, F.B., and Blomquist, R.N. (1994). *Parallel Monte Carlo Eigenvalue Calculations.* Proc. 1994 ANS Winter Meeting, Washington, D.C., November 13-17, 1994.

[2] Gropp, W., Lusk, E., Skjellum, A. (1994). Using MPI. MIT Press, Cambridge, MA, 1994.

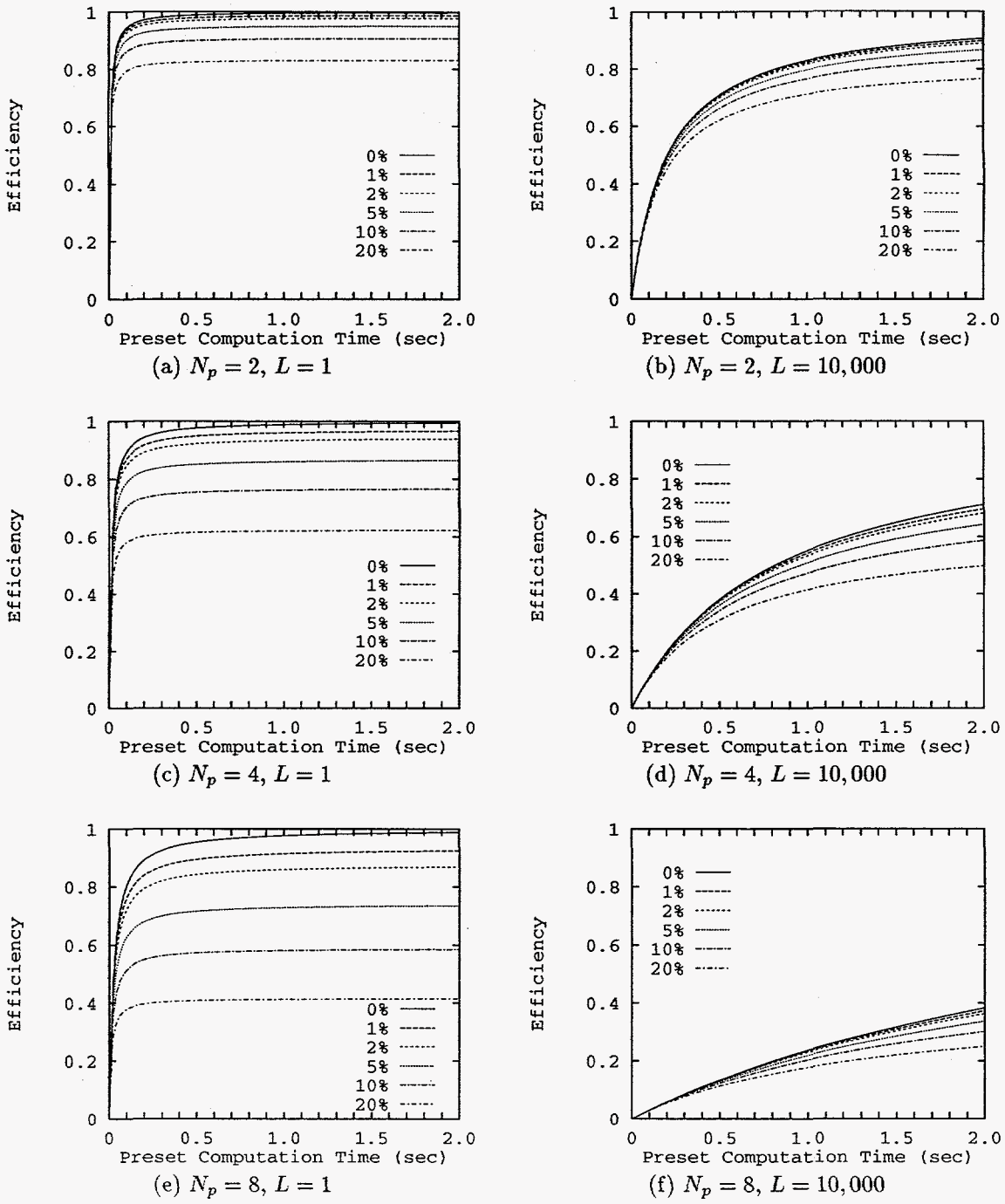

Figure 4: Theoretical Efficiency for the non–perfect load balanced case. Results are given for 2, 4 and 8 processors (N_p) and Message Sizes (L) of 1 and 10,000. The serial fraction is varied between 0% and 20%.

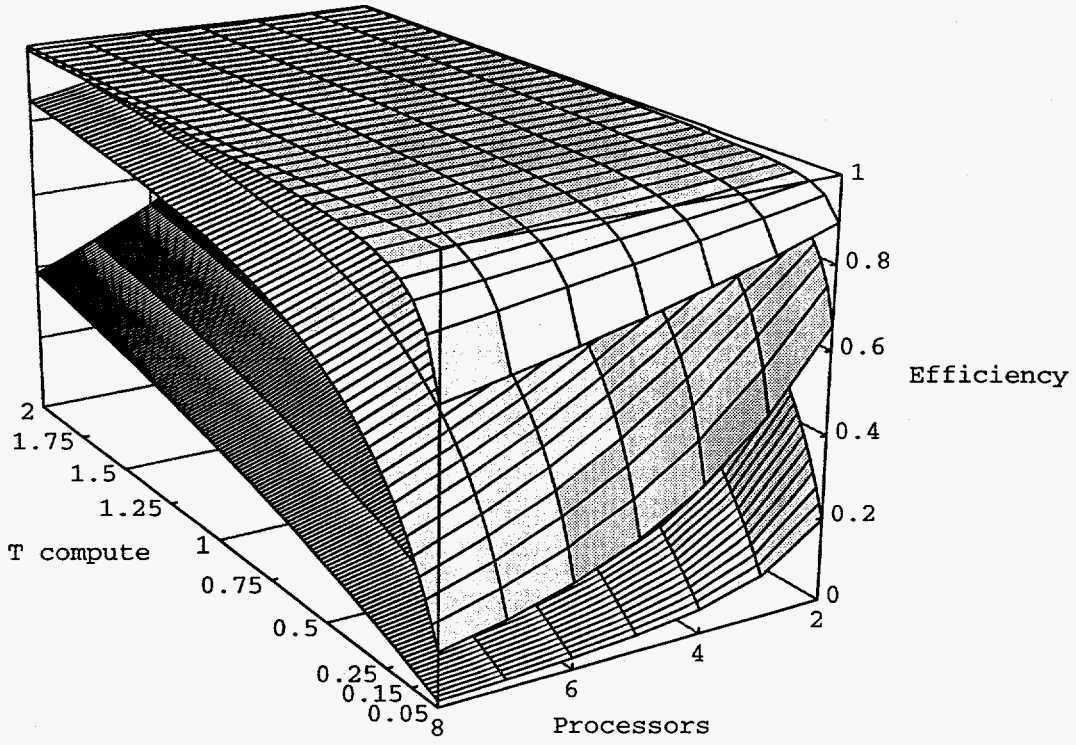

Figure 5: Efficiency Planes (Eq.7) for perfect load balance (s = 0%); Top: Msg. Size = 1; Middle: Msg. Size = 1,000; Bottom: Msg. Size = 10,000

[3] Amdahl, G. (1967). *Validity of the single - processor approach to achieving large scale computing capabilities.* Proceedings of the AFIPS Conference, 483–485.

[4] Foster, I. (1994). Designing and Building Parallel Programs. Addison-Wesley, Reading, MA, 1994. (on-line book available on the world wide web at http://www.mcs.anl.gov/dbpp)

[5] Dillon, E., Gamboa Dos Santos, C., Guyarde, J. (1995). *Homogeneous and Heterogeneous Networks of Workstations: Message Passing Overhead.* MPI Developers Conference, University of Notre Dame, South Bend, IN, June 22-23, 1995. (Proceedings available on the world wide web at http://www.cse.nd.edu/mpidc95)

[6] Georgitsis, V., Sobolewski, J.S. (1995). *Performance of MPL and MPI of SP2 System.* MPI Developers Conference, University of Notre Dame, South Bend, IN, June 22-23, 1995. (Proceedings available on the world wide web at http://www.cse.nd.edu/mpidc95)

[7] Stoica, I., Sultan, F., Keyes, D. (1994). *A Simple Hyperbolic Model for Communication in Parallel Processing Environments.* ICASE Report 94-78, Institute for Computer Applications in Science and Engineering, NASA Langley Research Center, Hampton, VA, September 1994.

[8] Leutenegger, S.T., Sun, X.-H. (1993). *Distributed Computing Feasibility in a Non-Dedicated Homogeneous Distributed Systems.* ICASE Report 93-65, Institute for Computer Applications in Science and Engineering, NASA Langley Research Center, Hampton, VA, September 1993.