THE PERFORMANCE EVALUATION OF NETWORK OF COMPUTERS

HIROKO MIDORIKAWA

Department of Information Sciences, Seikei University

3-3-1 Kichijoji-kitamachi, Musashino-shi, Tokyo 180-8633, Japan, midori@is.seikei.ac.jp

Abstract

A performance of a network of computers is evaluated by executing several benchmark programs with two programming models, PVM as a message passing model and TreadMarks as a shared memory model. Moreover, a communication/calculation characteristic in a application of low performance is investigated and compared with different parallel systems, shared-memory system Luna88k2 and distributed-memory system nCUBE2.

1. Introduction

In parallel processing, dedicated parallel computers with special architectures dominated for a long time, but today, a conventional computer soon catch up with these dedicated systems in a performance and a cost, because of the high speed development of conventional CPU chips. With these backgrounds, a cluster of cheap and high performance conventional computers connected by high speed network is one of the main streams in parallel processing systems.

In this paper, a performance of such a computer cluster system is evaluated by executing several benchmark programs with two representative programming models, message passing model and shared memory model.

Moreover, communication/calculation characteristics in a application of low performance is investigated and compared with other parallel systems, shared-memory system Luna88k2 and distributed-memory system nCUBE2.

2. Experimental Environment

A computer system used for a experiment is shown in TABLE 2. The PC cluster used here has eight PCs connected by a 100bps ether network.

TABLE 1 shows thebenchmark programs used for the evaluation. They are EP(Embarrassingly Parallel), 3D-FFT, IS(Integer Sort) from the NAS benchmarks, Water from the SPLASH benchmark, Successively Over-Relaxation, Traveling Salesman, Quick Sort, and Jacobian.

The programming models used here are PVM as a message passing model and TreadMarks[1] as a shared memory model. TreadMarks is a page-based software distributed shared memory system developed by Rice University. It uses a lazy release memory consistency.

3. Performance Evaluation

FIG.1-a and FIG.1-b show sequential execution times of the benchmark programs. These values are used for speedup calculation of each program. Some benchmark programs require much calculation time. FIG.2 shows the amount of communication data between processors during 8-processor execution. The TreadMarks requires more messages than PVM because a memory consistency maintenance is done in page base. FIG.3.1 to FIG.3.6 show the speedup curves of several benchmark programs in TABLE 1.

3.1 IS

It requires ranking an unsorted sequence of keys. The program divides up the keys among processors and each processor counts its keys by reading a whole integer array. Each processor counts list is summed up at the final stage. It accesses large range of memory so it requires much consistency maintenance messages compared to a light calculation in TreadMarks. So PVM and TreadMarks speedup are 2.6 and 3.0 when using 8 processors.

3.2 WATER

Although TreadMarks requires much message exchange, a calculation of power between molecules requires more computation. TreadMarks and PVM achieve 7.1 and 7.6 speedup at 8 processors.

3.3 EP

In EP communication is negligible small and much floating calculation is needed, so TreadMarks and PVM achieve a speedup of 7.2 and 7.6 using 8 processors.

3.4 3D-FFT

In 3D-FFT, 2 dimensional arrays are allocated to each processor, so in the first and the second step of 1D-FFT, it needs no communication between processors. Before the third step, communication for whole 3D-array transposition is required. A speedup of 4.1 and 4.0 obtained by TreadMarks and PVM respectively shown in FIG.3.4.

3.5 Quick Sort

In QSORT, divided sublists are queued in a work pool, each processor continuously get a unprocessed sublist. When

Programs	Problem Size
EP	2**28
IS.medium	N=2**23, max=2**10,R=10
IS.large	N=2**23, max=2**15,R=10
3D-FFT	64 x 64 x 64, R=6
Water-288	288 molecules,R=5
Water-1728	1728 molecules,R=5
SOR-z, SOR-nz	1024 x 3072,R=50
TSP	19 Cities
Qsort.qsort, bubble	N/minblock=1000K/1000(PC)
Jacobi	1536 x 1536,R=20

Table 1 Programs

the sublist is sufficiently small, the integers are sorted in the processor by localsort. If the sublist is further divided, the processor puts the newly generated sublist back on the work pool. The PVM version uses a master-slave style and the master maintains the work pool. In our experiment, integer arrays size was 1000K and the localsort threshold was 1000.

The speedup curves are shown in FIG.3.5 and FIG.3.6 when using a quick sort and a bubble sort as a localsort respectively. Low computation/communication ratio causes a low speedup. Although each processor accesses a independent area of array, many memory consistency maintenance messages are generated as shown in FIG.2. It is caused by a false sharing and a work pool maintenance. When using a quick sort as a localsort, a parallel execution time became longer than a sequential time.

3.6 Summary

From the experiment, SOR, EP and Water gained good performances and 3D-FFT, IS.MEDIUM are not so good. They are depend on a communication/calculation ratio of application. The difference between TreadMarks and PVM is not so large in most of the applications, although PVM programs remain a possibility that a programmer can tune to get a better performance.

Generally, TreadMarks gives us a better programmability than the PVM when writing a program which is complicated to grasp when and which data are accessed, 3D-FFT for example.

Though, even if using a shared memory model, a network of computers has a nature of distributed memory system, so considering how to reduce communication cost is important whenever you write a program .

4. Communication/Calculation characteristics

In order to process data by multiple processors on a distributed memory system, the data may be divided or distributed among the processors so data exchange is inevitable. If data transmitting costs becomes high compared to local data processing costs, a parallel processing becomes meaningless.

To clear the characteristics of network of computers, local data processing costs and data communication costs are measured in case of sort program as a material. Moreover we compared the PC cluster to other different parallel systems. One is a shared memory parallel system, Luna88k2, and the other is a distributed memory parallel system nCUBE2. The specifications of both systems are shown in TABLE 2.

In this experiment, the quick sort program and a parallel merge sort program are used. The localsort in both programs is quick sort, which gains no speedup in PC cluster as described in section 3.5.

4.1 Experimental Environment

Luna88k2 has four processors connected by bus and the memory is shared by the processors. LunaOS is based on Mach 2.5 and a thread is allocated to one of four processors arbitrarily by the OS. Programs are written by using a cthread library.

nCUBE2 has 32 processors connected by hypercube links. The link employs a wormwhole routing, so link communication time between two processors is constant independent of the distance between them. A official link speed is 2.22MB/sec. It provides a ncube message passing library as a programming environment.

4.2 Execution time in three different parallel systems

FIG.4 to FIG.6 show the execution times of these three systems for qsort and merge sort. FIG.6 shows that the execution times for 1000K integers become slower than the time at one processor in PC cluster. The quick sort time grows more rapidly than the merge sort.

On the other hand, a parallel processing in Luna88k2 and nCUBE obtain a speedups. FIG.4 shows a execution time for 3000K integers when using 1 to 4 processors in Luna88k2. The qsort and mergesort obtains 2.6 and 2.8 speedup at 4 processors in Luna88k2. FIG.5 shows a execution time of merge sort for 10K integers when using 1 to 32 processors in nCUBE2. nCUBE2 achieves a speedup of 2.6 at 4 processors.

4.3 Local processing cost

To evaluate a local processing cost in sort program, a sequential quick sort execution time was measured in each system. As shown in FIG.9, a quick sort time of nCUBE2 and Luna88k2 for 1000K integers is 6.6 and 4.2 times longer than PC cluster, and 11.8 and 4.1 times longer for 2000k integers.

4.4 Data transmitting cost

Communication cost in data transmitting through a message passing library was measured in nCUBE2 and PC cluster. From our experiment[2], nCUBE2 communication speed saturates 1.59MB/sec when transmitting larger data than 32KB. A local processing cost and a communication cost for the same size of integers in nCUBE2 is shown in FIG.7. In PC cluster, a message passing time by PVM was measured, and local processing and a communication cost is shown in FIG.8.

These results show that a gap between local processing cost and data exchange cost becomes smaller in PC cluster. A ratio of data transmitting cost/local processing cost is shown in FIG.10. In nCUBE2, the ratio is about 10% to 20% , but in PC cluster it grows 35% to 40%. It means that for applications consits of light comoputation, obtaining a high performance by parallel processing becomes more difficult in the PC cluster.

5. Conclusion

A network of computers, even if cheap PCs connected by ether have good performance for many applications compared with the former dedicated parallel systems. There is a little difference in performance between a message passing and a shared memory programming model.

Recent CPU speed becomes more faster than network speed, so a gap between CPU and communication speed grows larger than before. It is caused by not only a slow hardware speed of the communication link but also by the software overhead in a library and OS. To reduce a communication overhead becomes more important on a recent network of computers.

References

[1]P.Keleher, S.Dwarkadas, A.L.Cox and W.Zwaenepoel, TreadMarks: Distributed shared memory on standard workstations and operating system, Proceedings of the 1994 Winter Usenix Conference, January 1994, 115-131

[2]Okamoto, Midorikawa, Iizuka,Evaluation of nCUBE2 system, Technology Reports of Seikei University No.52, 1991, 3515-3526 (In Japanese)

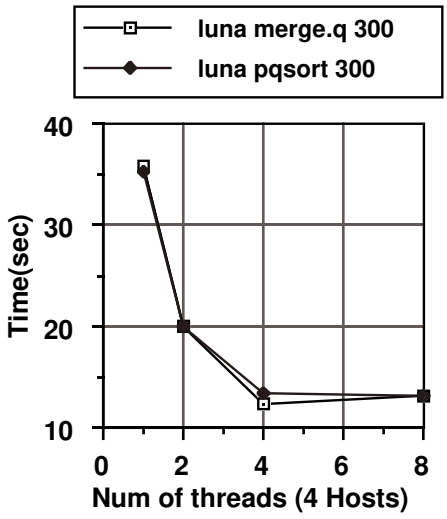

FIG.4 Luna88k2

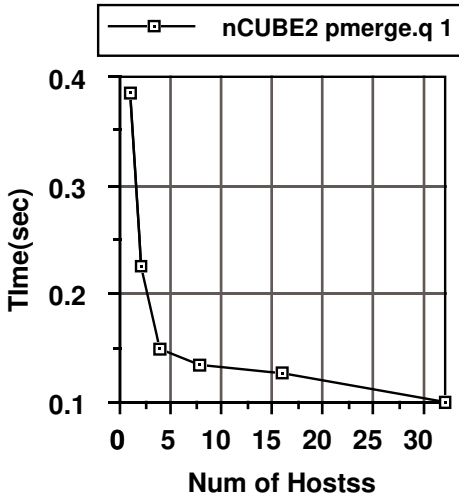

FIG.5 nCUBE2

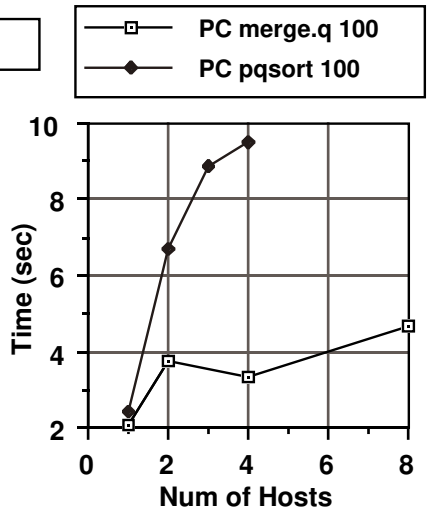

FIG.6 PC cluster(Tmk)

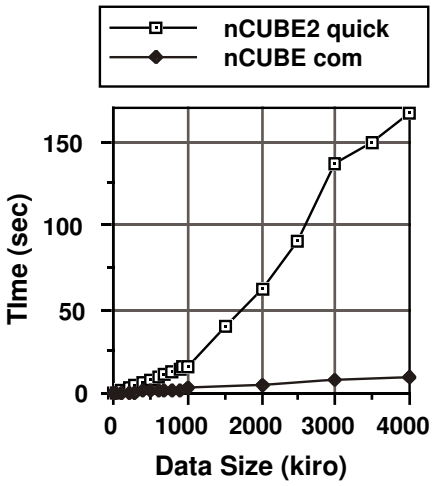

FIG.7 com & cal on nCUBE2

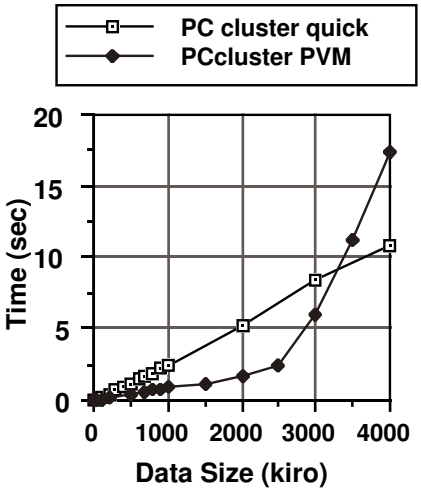

FIG.8 com & cal on PC cluster

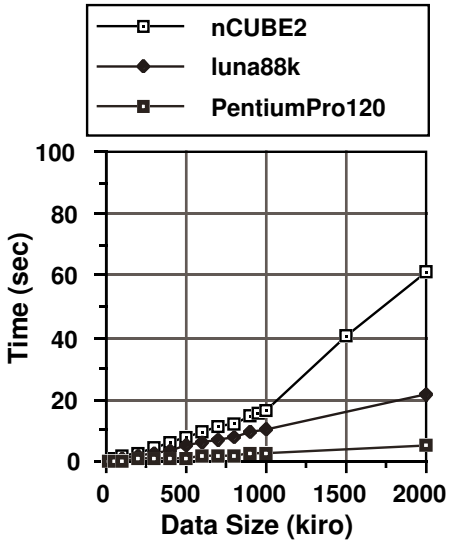

FIG.9 Sequential quick sort Time

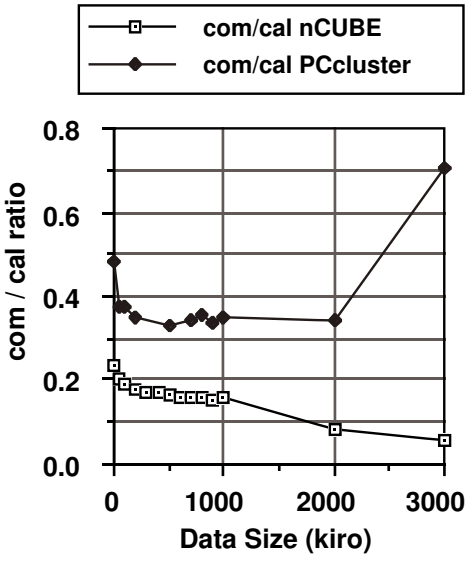

FIG.10 com / cal Ratio

System	PC cluster	Luna- 88k2	nCUBE2
Type	DM or DSM	SM	DM
CPU	Cyrix 6X86-P120-GP	M88100+ 88200(33MHz)	custum 64bit (20MHz)
memory	64MB/CPU	48MB/4CPU	4MB/CPU
Connection	100Mbps Ether Hub	bus	2.22MB/s hypercube link
OS	FreeBSD 2.2.2	LunaOS 2.00 (Mach2.5)	Vertex & SunOS
Num of CPU	8	4	32

TABLE 2 Systems

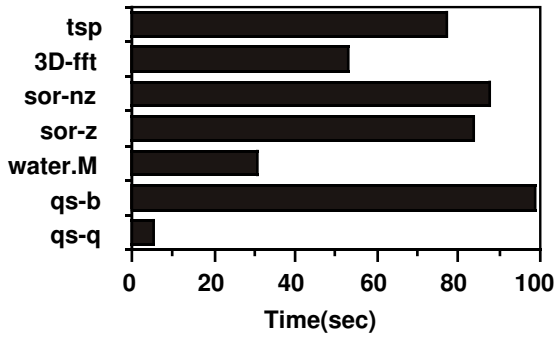

FIG.1-a SEQUENTIAL TIME

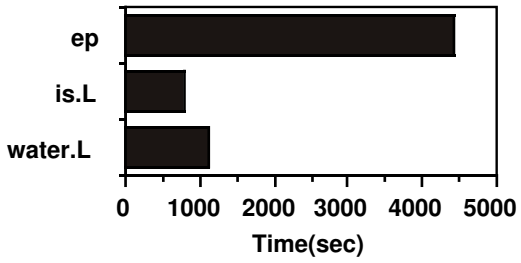

FIG.1-b SEQUENTIAL TIME

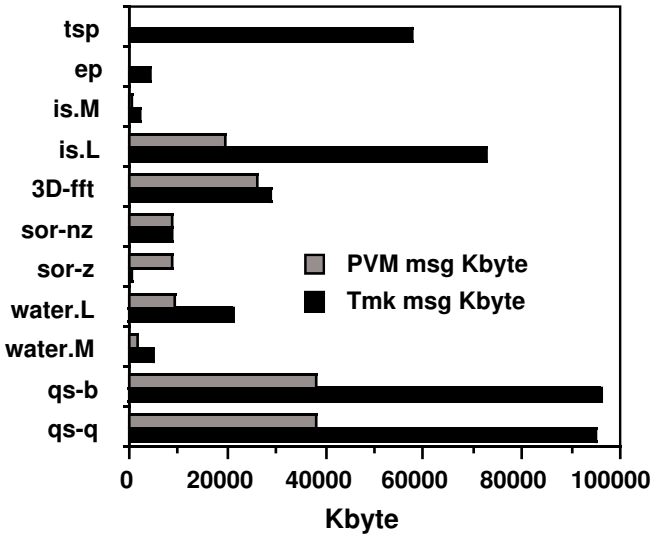

FIG.2 Messages (Kbytes,8hosts)

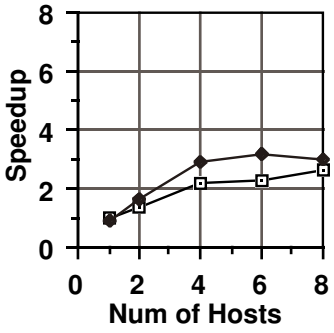

FIG.3.1 IS.MEDIUM

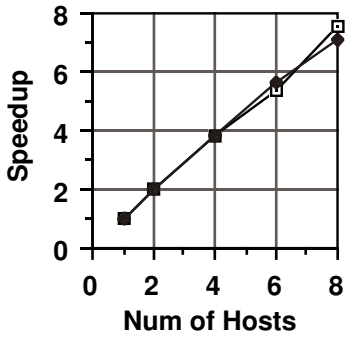

FIG.3.2 WATER.LARGE

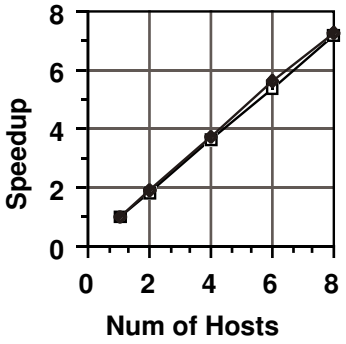

FIG.3.3 EP

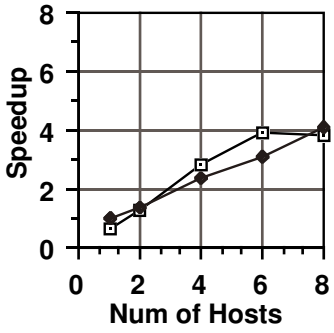

FIG.3.4 3D-FFT

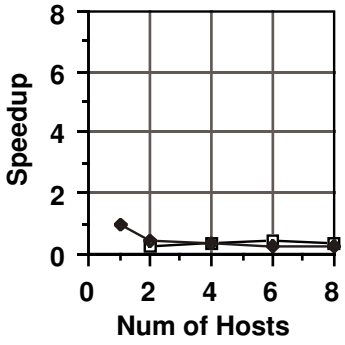

FIG.3.5 QSORT.QUICK

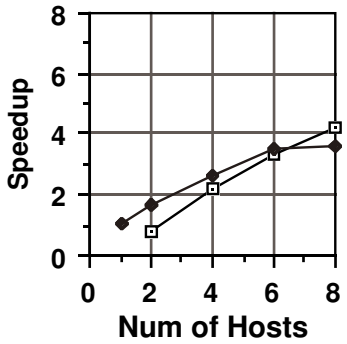

FIG.3.6 QSORT.BUBBLE