如何测量 SMP 机群可扩放性*

何家华+, 陈国良, 单久龙

(中国科学技术大学 计算机科学技术系 国家高性能计算中心(合肥),安徽 合肥 230027)

How to Measure Scalability of SMP Cluster

HE Jia-Hua+, CHEN Guo-Liang, SHAN Jiu-Long

(National High Performance Computing Center (Hefei), Department of Computer Science and Technology, University of Science and Technology of China, Hefei 230027, China)

+ Corresponding author: Phn: +86-551-3601548, E-mail: jiahua@mail.ustc.edu.cn, http://www.ustc.edu.cn

Received 2003-01-09; Accepted 2003-05-27

He JH, Chen GL, Shan JL. How to measure scalability of SMP cluster. ***Journal of Software*****, 2004,15(7): 977~986.**

http://www.jos.org.cn/1000-9825/15/977.htm

Abstract: Scalability is an important performance criterion of parallel computing. However the conventional scalability metrics are not suitable for SMP(symmetric multiprocessor) cluster. How to measure scalability of SMP cluster? This paper proposes a solution to the problem. It first finds out and verifies the reason, nonequivalence of the processor sets and then it adopts the viewpoint of processor set to observe the behaviors of the system correctly and comprehensively instead of using only the number of processors to describe the parallel system. By introducing the concept of performance reference factor, it extends the conventional metrics to fit the SMP cluster architecture. As the experiments indicate, the extended metrics are applicable to the SMP cluster and have high precision.

Key words: parallel computing; performance evaluation; SMP cluster; scalability

摘 要: 可扩放性是并行计算的一个重要性能标准,但是传统的可扩放性准则并不适用于 SMP 机群.如何测量 SMP 机群的可扩放性?试图提出该问题的一个解决方案.首先找出并验证问题的根源——处理器集合不等价性.然后,采用处理器集合的观点来全面、正确地观察系统的行为,而并非像传统的做法那样仅仅使用处理器数来描述并行系统.通过引入性能参考因子的概念,扩展了传统的准则以适应 SMP 机群体系结构.实验结果显示,扩展后的度量准则适用于 SMP 机群,且具有较高的准确性.

关键词: 并行计算;性能评测;SMP 机群;可扩放性

∗ Supported by the National High-Tech Research and Development Plan of China under Grant No.2001AA111041 (国家高技术研究发展计划(863))

HE Jia-Hua was born in 1976. He is a master candidate at the Department of Computer Science and Technology, the University of Science and Technology of China. His current research areas include parallel and distributed computing. **CHEN Guo-Liang** was born in 1938. He is a professor and doctoral supervisor at the Department of Computer Science and Technology, the University of Science and Technology of China and is an academician of Chinese Academy of Sciences. His research areas are parallel and distributed computing. **SHAN Jiu-Long** was born in 1977. He is a Ph.D. candidate at the Department of Computer Science and Technology, the University of Science and Technology of China. His current research interests include parallel and distributed computing.

中图法分类号: TP338 文献标识码: A

1 Introduction

There are many different kinds of architectures in high performance computing. SMP(symmetric multiprocessor) and cluster are two important ones among them. The shared-memory programming model of SMP derived from its symmetric and shared memory makes it easy to program and exploit high DoP(degree of parallelism), and its shared resources bring it high performance-cost ratio. However, it is the symmetry that limits the scale of the SMP. In contrast, cluster is famous for its excellent scalability. There seems to be an inherent complementarity and potential advantages in combining the two architectures into a SMP cluster, that is, a cluster with SMP nodes. Johnston[1] had predicted its success in the last century and now we can see many examples of the architecture on the list of Top500[2] including the second fastest computer, HP ASCI Q.

Scalability analysis plays an important role in performance evaluation of large parallel systems. One key issue in these systems is how to make an effective use of the increasing processors because their average performance will usually drop while more processors are used. Scalability is just such a metric that measures the capability of the system, including hardware and software, to utilize effectively the scaling-up processors. Conventional approaches to scalability analysis include Isoefficiency metric[3], Isospeed metric[4], Latency metric[5], and Time-scale metric[6], etc.

However, we cannot directly apply the conventional approaches to the SMP cluster because of the nonequivalence of processor sets. That is, a particular parallel program will take different times on different processor sets with the same number of processors. We will solve this problem by two concepts, "processor-set viewpoint" and "performance reference factor", and give a solution for scalability measurement of the SMP cluster.

Before beginning our discussion, we must make the following two assumptions, based on the consideration that most of the SMP cluster systems used practically in large-scale scientific computing will adopt homogeneous or almost homogeneous processors and nodes in order to achieve high performance.

(1) Homogeneity of processors: all the processors are identified in frequency, cache and architecture etc.;

(2) Homogeneity of nodes: the features of all the nodes, such as number of processors, capability of main memory and bandwidth of system bus, are the same.

The rest of the paper is arranged as follows. In Section 2, conventional scalability metrics are reviewed briefly, and then we verify the existence of nonequivalence of processor sets and introduce the concept "processor-set viewpoint" in Section 3. On the base of the concepts "performance reference program" and "performance reference factor" defined in Section 4, we extend the conventional scalability metrics in Section 5 and illustrate them by the example algorithm of parallel matrix multiplication in Section 6. Some conclusions and suggestions for future work are given in the last section.

2 Conventional Scalability Metrics

In this section, we will give a brief review of the conventional approaches to scalability analysis, including Isoefficiency metric, Isospeed metric, Latency metric, and Time-scale metric.

Isoefficiency metric: Let T_{calc} be the computation time, namely, the sum of the time spent by all the processors in useful computation, T_{comm} be the communication time, namely, the sum of the time spent by all processors in communicating with neighboring processors, waiting for messages, time in starvation, etc., and T_n be the parallel execution time on n processors, then we can infer that $nT_n=T_{calc}+T_{comm}$ and the efficiency E is given as:

$$E=\frac{T_{calc}}{nT_n}=\frac{T_{calc}}{T_{calc}+T_{comm}}=\frac{1}{1+\frac{T_{comm}}{T_{calc}}}.$$

According to the above formula, Kumar *et al*. proposed in 1987 that the problem size W needs to grow as a function of n, the number of the increasing processors, to maintain the efficiency E. They named the function "Isoefficiency Function" and use the function curve, "Isoefficiency Curve", to reflect the algorithm's scalability.

Isospeed metric: Sun *et al*. presented the metric in 1994. They first defined the average speed $\overline{V}$ as the achieved speed of the given computing system divided by the number of processors, and then gave a definition: "Let W be the amount of work of an algorithm when n processors are employed in a machine, and let W' be the amount of work of the algorithm when $n'>n$ processors are employed to maintain the average speed, then we define the scalability form system size n to the system size n' of the algorithm-machine combination as follows."

$$\psi(n,n')=\frac{n'W}{nW'}$$

Latency metric: Zhang *et al*. also gave a definition in 1994. "For a given algorithm implementation on a given machine, let $L_e(W,n)$ be the average latency when the algorithm for solving a problem of size W on n processors, and $L_e(W',n')$ be the average latency when the algorithm for solving the problem of size of W' on $n'>n$ processors. If the system size changes from n to n', and the efficiency is kept to a constant $E\in[0,1]$, the scalability latency metric is defined as "

$$scale(E,(n,n'))=\frac{L_e(W,n)}{L_e(W',n')}.$$

Time-scale metric: This is our previous work in 2000. For a given algorithm implementation on a given machine, let $T_p(W,n)$ be the parallel execution time with the problem workload W on n processors, and $T_p(W',n')$ the parallel execution time with the problem workload W' on $n'(n'>n)$ processors. If the number of processors increases from n to n', and the efficiency E keeps conserved, then the Time-scale scalability metric is defined as:

$$time\_scale(E,(n,n'))=\frac{T_p(W,n)}{T_p(W',n')}.$$

3 Nonequivalence of Processor Set and Processor-Set Viewpoint

We have surveyed the conventional metrics of scalability. They can do a good job in most of the parallel and distributed computing architectures, but in fact they do not work unless all the n-processor sets are equivalent because they use only one parameter n to describe the parallel program. It is a pity that the SMP cluster does not satisfy the assumption. Namely, different n-processor sets for a given n may have different computing abilities in the SMP cluster. We call that "nonequivalence of n-processors sets".

To verify the existence of the nonequivalence, we implemented the algorithms of parallel matrix multiplication and parallel longest common subsequence on Dawning2000-II[7] at NHPCC (Hefei)[8]. The execution times of different 4-processor sets with the same problem size are listed in Table 1, where "1*4" means the set includes 4 nodes with 1 processor per node, "1*2+2*1" means the set includes 2 nodes with 1 processor per node and 1 node with 2 processors, and so on. What we must explain is the sets "4*1"and "1*1+3*1" do not exist because the nodes used in the experiment are 2-way nodes. The experiment data demonstrate the existence of the "nonequivalence".

One question is what causes the nonequivalence? By analyzing the architecture features of the SMP cluster carefully, we can see that the nonequivalence derives from the following two heterogeneities:

(1) Communication heterogeneity: there are two communication methods in the SMP cluster; shared memory inside one node and message passing between nodes.

(2) Resource heterogeneity: the processors inside one node share resources, and the ones in respective nodes have their dedicated resources.

These two factors affect the things in opposite directions. The communication heterogeneity prefers the processors in one node to those in different nodes. In contrast, the resource heterogeneity will make the performance of latter condition better than that of the former.

Table 1 Parallel execution times on different 4-processor sets

Processor sets	1*4	1*2+2*1	2*2
Execution time of matrix multiplication (s)	305.50	347.60	360.10
Execution time of longest common subsequence (s)	55.10	55.36	55.61

Since the way using only one parameter n to describe the parallel program is invalid for the SMP cluster, we need a new standing point to observe the behaviors of the system. We will adopt the "processor-set viewpoint" in the rest of the paper. That is, no matter when and where we mention the size of the machine, we are always referring to a certain processor set, which implies not only the number but also the allocation of the processors.

4 Performance Reference Program and Performance Reference Factor

To fulfill the "processor-set viewpoint" is not easy because no matter which scalability metric we adopt, Isoefficiency metric, Isospeed metric, Latency metric or Time-scale metric, the parameter n will appear in the definition of efficiency E or average speed $\overline{V}$. We have to find a new parameter according to "processor-set viewpoint" to replace n.

In the last section, we have discussed that the reason for the invalidation of the parameter n is that different n-processor sets for a given n have different computing abilities. It implies that the original purpose of n is to describe the computing ability of the processor set, so the new parameter must be able to finish this task. The interesting issue is how to find the new one. The current way is using the amount of resources to describe the computing ability indirectly. Now that the indirect way is not feasible in the SMP cluster, the new parameter may describe the computing ability directly. Speed V is one of the candidates.

What program should be used to measure the speed? Just like the way we select a "frame of reference" in mechanics, we need to choose a program as reference for indicating their relative performance to those of others, and therefore name it "performance reference program". Such a relative concept gives us a flexibility to choose different performance reference programs according to demands and conditions, especially in different application fields.

However, the performance reference program should satisfy the following requirements for practicability and high precision.

(1) The workload should be in proportion to the number of the processors. We set the requirement instead of the fixed workload to avoid too heavy load on single processor and too light load on n-processor set when n is large. Let $W_{ref}(1)$ be the workload on single processor and $W_{ref}(n\_set)$ be the workload on a given n-processor set, then $W_{ref}(n\_set)=nW_{ref}(1)$.

(2) The computation to communication ratio should be constant when the workload increases. Or the performance reference factor will vary with the size of workload. In practical operation, this requirement will not be satisfied until the size of workload is large enough because of the initialization affects.

As for the performance reference program used in this paper, we adopt the simplest way to construct it, that is, let each processor proceed a certain amount of computational and communicational operations round and round. In addition to the advantage of simpleness, such a performance reference program makes it possible for us to adjust the computation to communication ratio as needed.

Since we have finished the discussion of performance reference program, it is time to come back to the new parameter we are looking for. Speed (not average speed) V is the very thing to describe the computing ability of the machine, but we cannot use it to replace n directly because the latter is a nondimensional parameter, and we have to introduce a new concept “performance reference factor”. Let $V_{ref}(1)$ be the speed on single processor and $V_{ref}(n\_set)$ be the speed on a given n-processor set, then we define the performance reference factor of the n-processor set as:

$$rf(n\_set)=\frac{V_{ref}(n\_set)}{V_{ref}(1)}.$$

In practical operation, speed is not easy to measure directly. We will measure execution time instead and calculate out the result of performance reference factor. Let $T_{ref}(1)$ be the execution time on single processor and $T_{ref}(n\_set)$ be the time on n-processor set, then

$$rf(n\_set)=\frac{V_{ref}(n\_set)}{V_{ref}(1)}=\frac{\dfrac{W_{ref}(n\_set)}{T_{ref}(n\_set)}}{\dfrac{W_{ref}(1)}{T_{ref}(1)}}=\frac{\dfrac{nW_{ref}(1)}{T_{ref}(n\_set)}}{\dfrac{W_{ref}(1)}{T_{ref}(1)}}=\frac{nT_{ref}(1)}{T_{ref}(n\_set)}.$$

5 Scalability Metrics of SMP Cluster

On the base of the discussion in the last section, let’s see how to extend the scalability metrics of the SMP cluster replacing n with performance reference factor.

Extended Isoefficiency metric: When the parameter n is replaced by the performance reference factor, the efficiency E looks like:

$$E=\frac{T_{calc}}{rf(n\_set)T_p(n\_set)}=\frac{T_{calc}}{\dfrac{nT_{ref}(1)}{T_{ref}(n\_set)}\cdot\dfrac{T_{calc}+T_{comm}}{n}}=\frac{1}{\dfrac{T_{ref}(1)}{T_{ref}(n\_set)}\left(1+\dfrac{T_{comm}}{T_{calc}}\right)}.$$

From the formula, we can see that there is still a functional relation between W and the processor set (not n) when E is fixed. But to work out the relation between $T_{ref}(1)$ and $T_{ref}(n\_set)$ is not easy. We will leave the problem as a future work.

Extended Isospeed metric: Let $T_p(W,n\_set)$ be the parallel execution time with problem workload W on the n-processor set, the average speed is defined as:

$$\overline{V}=\frac{W}{rf(n\_set)T_p(W,n\_set)}.$$

Let W be the amount of work of an algorithm when an n-processor set is employed in a machine, and W' be the amount of work of the algorithm when an n'-processor set is employed to maintain the average speed, then we define the scalability from n-processor set to n'-processor set of the algorithm-machine combination as follows.

$$\psi(n\_set,n'\_set)=\frac{rf(n'\_set)W}{rf(n\_set)W'}=\frac{T_p(W,n\_set)}{T'_p(W',n'\_set)}.$$

Extended Latency metric: Let $T_p(W,1)$ be the parallel execution time with problem workload W on single processor, and let $T_p(W,n\_set)$ be the parallel execution time with problem workload W on the n-processor set, the efficiency is defined as:

$$E=\frac{T_p(W,1)}{rf(n\_set)\cdot T_p(W,n\_set)}.$$

For a given algorithm implementation on a given machine, let $L_e(W,n\_set)$ be the average latency when the algorithm for solving a problem of size W on an n-processor set, and $L_e(W',n'\_set)$ be the average latency when the algorithm for solving the problem of size of W' on an n'-processor set. If the system size changes from n-processor set to n'-processor set, and the efficiency is kept to a constant, the scalability latency metric is defined as:

$$scale(E,(n\_set,n'\_set))=\frac{L_e(W,n\_set)}{L_e(W',n'\_set)}.$$

Extended Time-scale metric: The definition of E is the same as that in Latency metric. For a given algorithm implementation on a given machine, let $T_p(W,n\_set)$ be the parallel execution time with problem workload W on an n-processor set, and $T_p(W',n'\_set)$ the parallel execution time with problem workload W' on an n'-processors set. If the system size changes from n-processor set to n'-processors set, and the efficiency E keeps conserved, then the Time-scale scalability metric is defined as:

$$time\_scale(E,(n\_set,n'\_set))=\frac{T_p(W,n\_set)}{T_p(W',n'\_set)}.$$

6 Case Study

In this section, let’s see how the two extended metrics, Isospeed metric and Time-scale metric, work in the SMP cluster. Before implementing the metrics, we run the performance reference program, measure the execution times and then calculate the performance reference factors as follows.

Table 2 Execution times of performance reference program and performance reference factors

Processor set	Time (s)	*Rf*(*n_set*)
1×1	33.07	1.00
1×2	48.59	1.36
1×3	45.01	2.20
1×4	44.39	2.98
1×5	44.28	3.73
1×6	44.47	4.46
1×7	44.72	5.18
1×8	44.81	5.90
1×9	44.74	6.65
1×10	45.33	7.30
2×1	48.35	1.37
2×2	47.69	2.77
2×3	48.11	4.12
2×4	48.05	5.50
2×5	48.40	6.83
2×6	48.91	8.11
2×7	48.13	9.62
2×8	48.93	10.81
2×9	49.48	12.03
2×10	49.07	13.48

Extended Isospeed Metric: By the approach 2 described in Ref.[4], we get the time variation and the computed scalability of the matrix multiplication algorithm as follows.

Table 3 Time variation under extended Isospeed metric

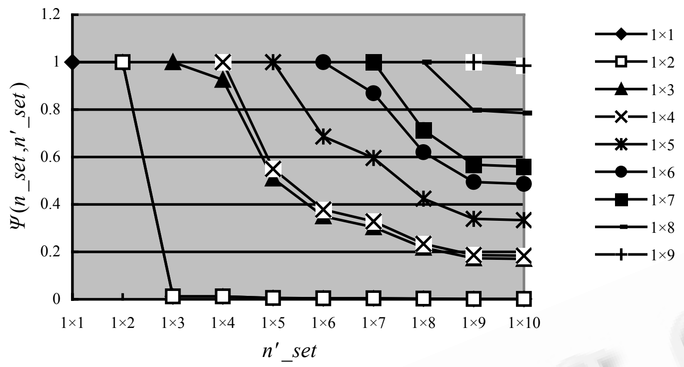
Processor set	Time (s)	Processor set	Time (s)
1×1	0.004	2×1	0.002
1×2	0.004	2×2	0.918
1×3	0.314	2×3	1.240
1×4	0.339	2×4	1.728
1×5	0.616	2×5	2.355
1×6	0.898	2×6	2.867
1×7	1.033	2×7	4.829
1×8	1.449	2×8	7.046
1×9	1.819	2×9	17.797
1×10	1.846	2×10	19.125

Table 4 Computed scalability of extended Isospeed metric

$\Psi(n\_set, n'\_set)$	1×1	1×2	1×3	1×4	1×5	1×6	1×7	1×8	1×9	1×10
1×1	1.000	1.000	0.012	0.012	0.006	0.004	0.004	0.003	0.002	0.002
1×2		1.000	0.012	0.012	0.006	0.004	0.004	0.003	0.002	0.002
1×3			1.000	0.926	0.510	0.350	0.304	0.217	0.173	0.170
1×4				1.000	0.550	0.378	0.328	0.234	0.186	0.184
1×5					1.000	0.686	0.596	0.425	0.339	0.334
1×6						1.000	0.869	0.620	0.494	0.486
1×7							1.000	0.713	0.568	0.559
1×8								1.000	0.797	0.785
1×9									1.000	0.985
1×10										1.000

$\Psi(n\_set, n'\_set)$	2×1	2×2	2×3	2×4	2×5	2×6	2×7	2×8	2×9	2×10
2×1	1.000	0.002	0.002	0.001	0.001	0.001	0.000	0.000	0.000	0.000
2×2		1.000	0.740	0.531	0.390	0.320	0.190	0.130	0.052	0.048
2×3			1.000	0.718	0.527	0.433	0.257	0.176	0.070	0.065
2×4				1.000	0.734	0.603	0.358	0.245	0.097	0.090
2×5					1.000	0.821	0.488	0.334	0.132	0.123
2×6						1.000	0.594	0.407	0.161	0.150
2×7							1.000	0.685	0.271	0.252
2×8								1.000	0.396	0.368
2×9									1.000	0.931
2×10										1.000

In fact, the dependant variable (Ψ) and the two independent variables (*n_set* and *n′_set*) form a three dimensional space. The two sub-tables of Table 4 reflect only two cross sections of the space. These tabular data can also be represented by the following pictorial diagrams.



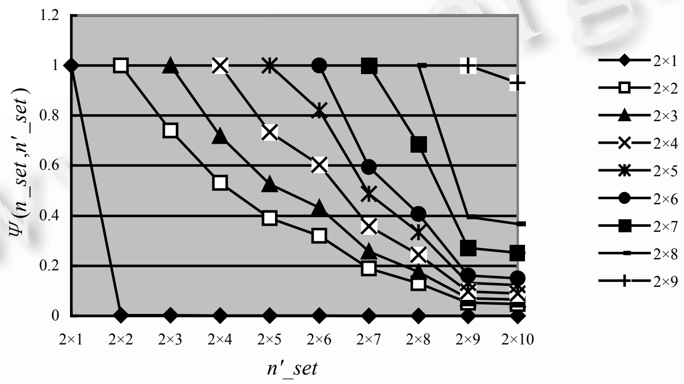


Fig.1 Scalability curves of extended Isospeed metric

Extended Time-scale Metric: By the approach described in Ref.[6], we get the time variation and the computed scalability of the matrix multiplication algorithm as follows.

Table 5 Time variation under extended Time-scale metric

Processor set	Time (s)	Processor set	Time (s)
1×1	0.007	2×1	0.004
1×2	0.007	2×2	1.120
1×3	0.419	2×3	1.464
1×4	0.467	2×4	2.011
1×5	0.793	2×5	2.663
1×6	1.106	2×6	3.124
1×7	1.295	2×7	5.153
1×8	1.726	2×8	7.432
1×9	2.069	2×9	16.936
1×10	2.168	2×10	19.161

As you may have noticed, most of the processes are very similar to that of the conventional metrics. The only difference is that the performance reference factor must be measured and calculated first, and then replace "*n*" through the entire process of computing scalability. The solution is simple and easy to operate, but it does solve the problem how to measure the scalability of the SMP cluster. In addition, the fact that the two extended metrics present the very similar scalability trends of the same problem indicates that they have high precision.

Table 6 Computed scalability of extended Time-scale metric

time_scale(E_{ref}, (*n_set*, *n′_set*))	1×1	1×2	1×3	1×4	1×5	1×6	1×7	1×8	1×9	1×10
1×1	1.000	1.000	0.016	0.014	0.008	0.006	0.005	0.004	0.003	0.003
1×2		1.000	0.016	0.014	0.008	0.006	0.005	0.004	0.003	0.003
1×3			1.000	0.896	0.528	0.378	0.323	0.243	0.202	0.193
1×4				1.000	0.590	0.422	0.361	0.271	0.226	0.215
1×5					1.000	0.716	0.612	0.459	0.383	0.365
1×6						1.000	0.854	0.641	0.535	0.510
1×7							1.000	0.751	0.626	0.597
1×8								1.000	0.834	0.796
1×9									1.000	0.954
1×10										1.000

time_scale(E_{ref}, (*n_set*, *n′_set*))	2×1	2×2	2×3	2×4	2×5	2×6	2×7	2×8	2×9	2×10
2×1	1.000	0.003	0.003	0.002	0.001	0.001	0.001	0.001	0.000	0.000
2×2		1.000	0.765	0.557	0.421	0.359	0.217	0.151	0.066	0.058
2×3			1.000	0.728	0.550	0.469	0.284	0.197	0.086	0.076
2×4				1.000	0.755	0.644	0.390	0.271	0.119	0.105
2×5					1.000	0.852	0.517	0.358	0.157	0.139
2×6						1.000	0.606	0.420	0.184	0.163
2×7							1.000	0.693	0.304	0.269
2×8								1.000	0.439	0.388
2×9									1.000	0.884
2×10										1.000

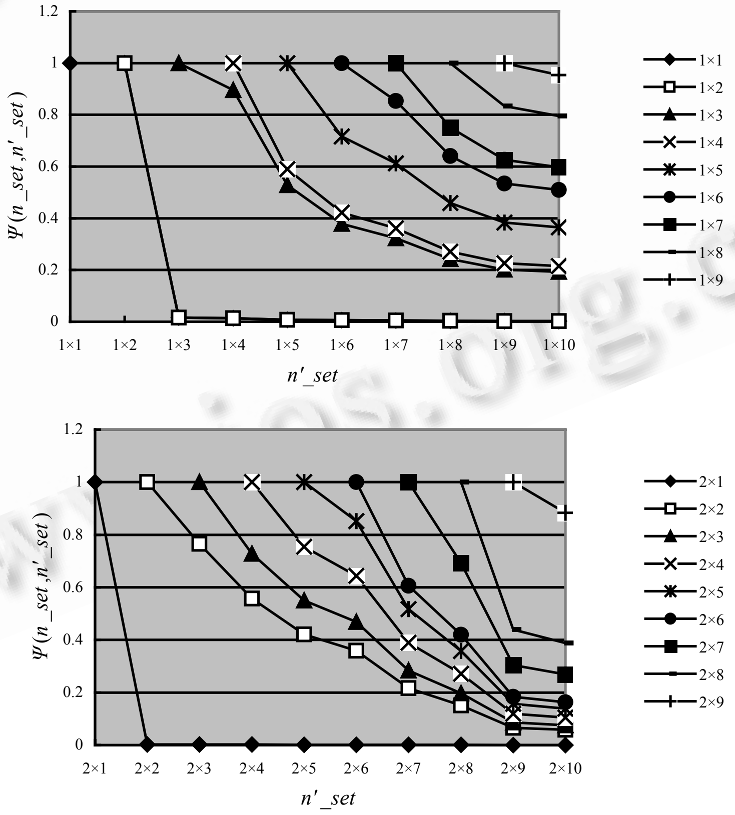


Fig.2 Scalability curves of extended Time-scale metric

7 Conclusions and Future Directions

In this paper, we review the conventional scalability metrics, including Isoefficiency metric, Isospeed metric, Latency metric and Time-scale metric, present their unsuitability for the SMP cluster and verify the reason, nonequivalence of the processor sets. To solve the problem, we adopt the viewpoint of processor-set to observe the behaviors of the system, introduce the concepts of 'performance reference program' and 'performance reference factor' and then extend the scalability metrics to fit the architecture of SMP cluster. With the example of matrix multiplication algorithm, we illustrate the validity of the extended metrics.

As for the future work, we will put an emphasis on how to construct "performance reference program". Now we adopt the simplest way to construct and believe that it is the best way to begin. We will try other real programs such as a benchmark to compare their effects in future. At the same time, how to use the extended Isoefficiency metric to analyze the scalability of SMP cluster in practice is another problem.

References:

[1] Johnston WE. The case for using commercial symmetric multiprocessors as supercomputers. http://www-itg.lbl.gov/~johnston/Scientific.Comp.Arch/Scientific.Comp.Arch.fm.html

[2] Top500. http://www.top500.org

[3] Kumar V, Rao VN. Parallel depth first search, Part Ⅱ: Analysis. International Journal of Parallel Programming, 1987,16(6): 501~519.

[4] Sun XH, Rover DT. Scalability of parallel algorithm-machine combinations. IEEE Trans. on Parallel Distributed Systems, 1994, 5(6):599~613.

[5] Zhang XD, Yan Y, He K. Latency metric: An experimental method for measuring and evaluating parallel program and architecture scalability. Journal of Parallel and Distributed Computing, 1994,22(3):392~410.

[6] Ji YC, An H, Ding WQ, Chen GL. A scalability metric for algorithm-machine on NOW and MPP. In: Proc. of the 4th Int'l Conf. on High-Performance Computing in Asia-Pacific Region. 2000. 405~407.

[7] Dawning-2000. http://202.38.76.202/dawn2000

[8] NHPCC(Hefei). http://nhpcc.ustc.edu.cn