Estimating the Optimal Configuration of a Heterogeneous Cluster: the Case of NAS Parallel Benchmarks

Shuichi Ichikawa*, Masayoshi Asakura, Yusuke Sakumoto
Department of Knowledge-based Information Engineering,
Toyohashi University of Technology
*ichikawa@ieee.org**

Abstract[1]

Invoking multiple processes on fast processing elements is a simple way of alleviating load-imbalance in heterogeneous clusters, while the optimal process allocation is not obvious. It is also important to select the optimal subset of PEs, though it is difficult to determine. Ichikawa et al. presented a scheme to estimate the optimal configuration of a heterogeneous cluster, based on the execution-time estimation models of four parallel applications (CFD, FEM, HPL, and FFT). This study examines their scheme with four new applications, which were taken from NAS parallel benchmarks. For CG and LU, the estimation errors are less than 30% in most cases, while the errors remain larger than 100% in some cases of MG and FT. More improvements are desired for communication-oriented applications such as MG and FT.

Keyword(s)*: PC cluster, high-performance computing (HPC), performance evaluation, optimization*

1. Introduction

PC clusters have been widely adopted for scientific computation, from small-scale parallel computing up to supercomputing. Since the advances in microprocessor technology are drastic, it is reasonable to enhance the performance of an existing PC cluster by adding the latest high-performance processors. The resulting cluster becomes heterogeneous, consisting of various processing elements (PEs) from fast to slow.

Since many existing applications distribute computational workloads equally among PEs, their performances are degraded on a heterogeneous cluster by load-imbalance. Invoking multiple processes on fast PEs (multiprocessing scheme), load-imbalance on heterogeneous clusters may be alleviated.

The communication time and multiprocessing overhead makes things more complicated. It is not always preferable to use all available PEs, because superfluous communications can prolong the total execution time. It is thus very important to select the optimal subset of PEs for a given problem size, together with finding the optimal number of processes on each PEs.

Kishimoto and Ichikawa [1] presented a scheme to estimate the optimal configuration of a heterogeneous cluster, i.e., the optimal subset of PEs and the optimal process allocation. They constructed the execution-time estimation models from the measurement results of HPL (High-performance Linpack) [4] , and showed that the optimal or sub-optimal configurations were actually estimated for various sizes of HPL. Ichikawa et al. [2] then proposed a new execution-time estimation model (NP-T model), which was examined with four benchmark programs. Ichikawa and Kawai [3] also discussed a method of constructing models from diverse processing elements of a heterogeneous cluster.

Although the preceding studies [1] [2] [3] presented some promising results, only four applications have been examined to date. More parallel applications have to be examined to verify the applicability of this scheme. This study examines four new programs that were selected from NAS parallel benchmarks (ver. 2.4) [5] , and presents their evaluation results.

[1] Acknowledgement: This study was partially supported by a Grant-in-Aid for Scientific Research from the Japan Society for the Promotion of Science (JSPS).

2. Methodology

The NAS parallel benchmark (NPB) [5] is a small set of programs designed to evaluate the performance of parallel supercomputers. NPB consists of five kernels and three pseudo-applications, which are derived from computational fluid dynamics (CFD) applications. Although NPB supports four problem sizes (class A, B, C, and D), it is necessary for this study to examine arbitrary problem sizes. Thus, the following four programs are specifically selected from NPB for this study.

CG is a kernel of a conjugate gradient method, which is used to compute an approximation to the smallest eigenvalue of a large, sparse, symmetric positive definite matrix. This kernel is typical of unstructured grid computations in that it tests irregular long-distance communication and employs sparse matrix-vector multiplication.

LU is named after the lower-upper diagonal (LU) benchmark. Actually, it does not perform a LU factorization, but instead employs a symmetric successive over-relaxation (SSOR) to solve a regular-sparse, block 5x5 lower and upper triangular system. This problem represents the computations associated with a newer class of implicit CFD algorithms.

MG is a simplified multi-grid kernel, which solves a 3-D Poisson PDE (partial differential equations). It requires highly structured long distance communication and tests both short and long distance data communication.

FT is a kernel to solve a 3-D PDE using FFT (fast Fourier transform). This kernel performs the essence of many spectral methods. It is a good test of long-distance communication performance.

By close inspection of these four programs, the execution time (T) of each application is summarized as shown below. As in the previous studies, T is represented as a function of the problem size (N) and the total number of processes (P).

$$\textbf{CG}: \quad T(N,P) = P^{-1/2} \cdot O(N^2) + \log P \cdot O(1)$$

$$\textbf{LU}: \quad T(N,P) = P^{-1} \cdot O(N^3) + \log P \cdot O(1)$$

$$\textbf{MG}: \quad T(N,P) = P^{-1} \cdot O(N^3) + \log P \cdot O(1)$$

$$\textbf{FT}: \quad T(N,P) = P^{-1} \cdot O(N^3 \log N) + \log P \cdot O(1) + O(N^2)$$

Though the details are omitted, the term of P^{-1} includes the computation time of each process, and the term of log P represents the time for reduction operation or synchronization among processes. It should also be noted that P is restricted to a power of two in these benchmarks.

The NP-T models [2] are easily derived from the above equations. The coefficients of models are extracted from the measurement results of various N and P, using non-negative least squares method. The models are constructed for each sub-cluster, and the execution time of a heterogeneous cluster is estimated by the maximum of the estimated execution time of each sub-cluster, which is calculated with the corresponding NP-T model for a given configuration. More details on this subject are found in the preceding studies [1] [2] .

Once the models of an application are constructed for every possible configuration, it is easy to estimate the optimal configuration for a given size. The quality of estimation is quantified by the estimation error $\varepsilon = \hat{\tau}/\hat{T} - 1$. Here, $\hat{\tau}$ is the actual execution time of the estimated optimal configuration, while $\hat{T}$ is the actual execution time of the actual optimal configuration.

Table 1 summarizes the specifications of our evaluation platform. This heterogeneous cluster consists of three sub-clusters (G_1, G_2, and G_3), each of which consists of eight equivalent nodes. All nodes are connected by Gigabit Ethernet with a wire-speed switch. All programs were compiled with gcc-4.1.2 with MPI library (mpich-1.2.7p1).

Table 1. Specifications of our heterogeneous cluster. (RH9: Red Hat Linux9, FC5: Fedora Core 5.)

	CPU	Clock	Mem.	OS	#node
G_1	Pentium 4	3.6 GHz	1 GB	RH9	8
G_2	Xeon	2.8 GHz	1 GB	FC5	8
G_3	Celeron M	1.5 GHz	1 GB	RH9	8

Let P_i be the number of PEs actually used in sub-cluster G_i (i = 1, 2, 3). To examine various mixtures of PEs, three subsets of our cluster are adopted in the following evaluation.

(8,8,8): $0 \le P_1 \le 8,\ 0 \le P_2 \le 8,\ 0 \le P_3 \le 8$

(4,8,8): $0 \le P_1 \le 4,\ 0 \le P_2 \le 8,\ 0 \le P_3 \le 8$

(0,8,8): $P_1 = 0,\ 0 \le P_2 \le 8,\ 0 \le P_3 \le 8$

M_i represents the number of processes on each PE of G_i, where $1 \le M_1 \le 2$ and $M_2 = M_3 = 1$ are assumed. A *configuration* of our heterogeneous cluster is thus defined by a sextuplet (P_1, M_1, P_2, M_2, P_3, M_3). P of a configuration is calculated by $P = P_1M_1+P_2M_2+P_3M_3$.

3. Evaluation Results

Figure 1 summarizes the error ε of CG for three subsets (0,8,8), (4,8,8), and (8,8,8). As readily seen, the estimations are quite satisfactory, and the average errors are quite small for (4,8,8) and (8,8,8). Table 2

lists the estimated optimal configurations and the actual optimal configurations for various sizes of (4,8,8), where the optimal configurations are successfully estimated in most cases.

In case of (0,8,8), slight errors are observed overall; the actual optimal configurations involved 8 processes (e.g., (0,0,4,1,4,1)) for all sizes, while the estimated optimal configurations involved 16 processes for all sizes (i.e., (0,0,8,1,8,1)).

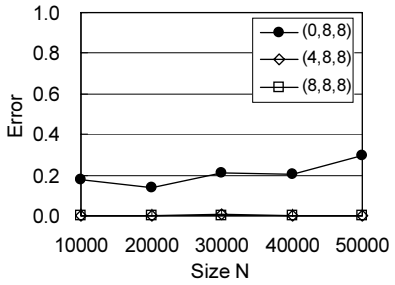

Figure 1. The error (ε) of CG.

Table 2. The estimated optimal configuration and the actual optimal configuration of CG (4,8,8).

N	Estimated Opt.	Actual Opt.
10000	4,1,0,0,0,0	4,1,0,0,0,0
20000	4,1,0,0,0,0	4,1,0,0,0,0
30000	4,1,0,0,0,0	1,1,8,1,7,1
40000	4,1,0,0,0,0	4,1,0,0,0,0
50000	4,1,0,0,0,0	4,1,0,0,0,0

Figure 2 summarizes the error ε of LU. Though the errors are larger than that of CG, they are still less than 30% in most cases. Table 3 lists the estimated and the actual optimal configurations of (4,8,8), where the estimations seem reasonable for $N \geq 80$. In case N=60 of (4,8,8), the estimated configuration involves 4 processes, while the actual optimal configuration involves 16 processors. The error for N=140 of (8,8,8) is also large; the estimation was (8,1,0,0,0,0), while the actual optimal configuration was (8,2,8,1,8,1).

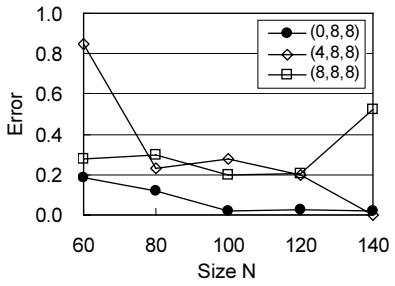

Figure 2. The error (ε) of LU.

Table 3. The estimated optimal configuration and the actual optimal configuration of LU (4,8,8).

N	Estimated Opt.	Actual Opt.
60	4,1,0,0,0,0	4,1,8,1,4,1
80	4,2,0,0,8,1	4,1,8,1,4,1
100	4,2,0,0,8,1	4,1,8,1,4,1
120	4,2,0,0,8,1	4,1,8,1,4,1
140	4,2,8,1,0,0	4,2,8,1,0,0

Figure 3 summarizes the error ε of MG, which is much larger than that of CG and LU. Table 4 lists the estimated and the actual optimal configurations of (4,8,8), where the errors are greater than 100% for $N \leq 128$. As seen in Table 4, (4,1,0,0,0,0) is estimated as optimal for N=64 and 128, while the actual optimal configuration is (2,1,0,0,0,0). Though the errors are large, the estimations seem rational in themselves, since more processors are allocated for larger sizes.

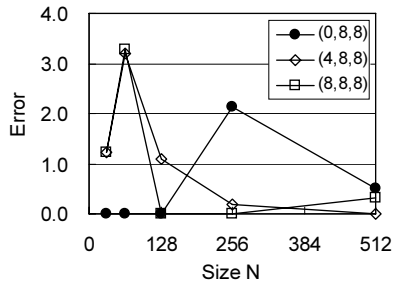

Figure 3. The error (ε) of MG.

Table 4. The estimated optimal configuration and the actual optimal configuration of MG (4,8,8).

N	Estimated Opt.	Actual Opt.
32	1,1,0,0,0,0	2,1,0,0,0,0
64	4,1,0,0,0,0	2,1,0,0,0,0
128	4,1,0,0,0,0	2,1,0,0,0,0
256	4,1,0,0,0,0	3,1,5,1,0,0
512	4,1,8,1,4,1	4,1,8,1,4,1

Figure 4 plots the measured execution times of 1, 2, 4, and 8 Pentium 4 nodes for MG. As readily seen, the behavior of 4 nodes is peculiar. For each N, the execution time decreases as P increases for P=1, 2, and 8. Compared to these, the execution times of P=4 are exceptionally large. Since *one* NP-T model is constructed from all these data, the derived model estimates much smaller execution times for P=4. This is the reason why (4,1,0,0,0,0) is erroneously estimated for N=64 and 128. Since a NP-T model is a smooth function, it cannot reflect such anomalous behaviors exactly.

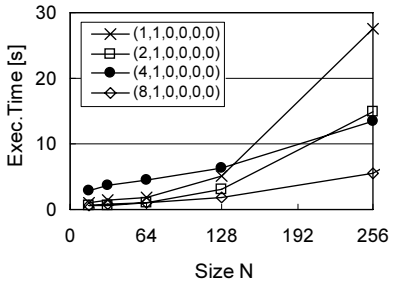

Figure 4. The measured execution times of MG with 1, 2, 4, and 8 Pentium 4 nodes.

Figure 5 summarizes the error ε of FT, which is larger than our expectations. However, it should also be noted that the errors are quite small in 10 out of 15 points of Figure 5. Table 5 lists the estimated and the actual optimal configurations of (4,8,8), where the errors are greater than 100% for N=32 and 128. The estimations seem rational in themselves, since more processors are allocated for larger sizes. The errors are supposed to be caused by anomalous behavior (as shown in MG) or the errors in model parameters.

As shown in Section 2, the model equation of FT includes the largest number of parameters; it is thus the most difficult target for precise model construction. Though the analysis of errors is not yet completed, the errors might be alleviated by using a larger number of PEs for model construction.

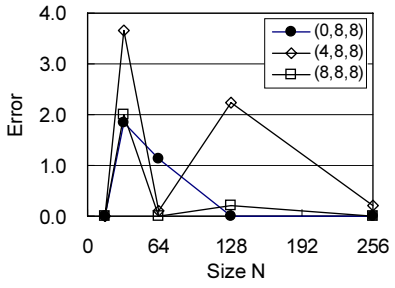

Figure 5. The error (ε) of FT.

Table 5. The estimated optimal configuration and the actual optimal configuration of FT (4,8,8).

N	Estimated Opt.	Actual Opt.
16	1,1,0,0,0,0	1,1,0,0,0,0
32	1,1,0,0,0,0	4,1,7,1,5,1
64	4,1,0,0,0,0	4,1,5,1,7,1
128	4,2,0,0,8,1	4,1,8,1,4,1
256	4,2,0,0,8,1	4,1,8,1,4,1

4. Conclusion

This study examined four benchmark programs (CG, LU, MG, and FT) from NAS parallel benchmarks, and presented the preliminary evaluation results of the estimated optimal configurations for three subsets of our heterogeneous cluster. The derived estimation errors of CG and LU are less than 30% in most cases, while the errors of MG and FT remain much larger. Although further improvements are required, the feasibility of our scheme was verified to a certain extent for these four programs.

In this study, we pointed out that the performance anomaly is one of the important factors of estimation errors in MG. Though the other reasons of large errors in MG and FT are not yet fully elucidated, further investigations and improvements are currently attempted.

MG and FT are both communication-oriented applications, which are difficult target for execution-time estimation models. It is difficult to estimate communication time precisely, because many external factors are involved; e.g., network congestion is beyond the scope of our current models. Precise modeling of communication time might be required to improve the estimation quality of MG and FT. All these things are left for future studies.

References

[1] Y. Kishimoto, S. Ichikawa: "Optimizing the Configuration of a Heterogeneous Cluster with Multiprocessing and Execution-Time Estimation," *Parallel Computing*, vol. 31, no. 7, pp. 691-710 (2005). (http://dx.doi.org/10.1016/j.parco.2005.04.004)

[2] S. Ichikawa, S. Takahashi, Y. Kawai, "Optimizing Process Allocation of Parallel Programs for Heterogeneous Clusters," *Concurrency and Computation: Practice and Experience*, (to appear). (http://dx.doi.org/10.1002/cpe.1349).

[3] S. Ichikawa, Y. Kawai, "Constructing Execution-Time Estimation Models from Diverse Processing Elements of Heterogeneous Clusters," Proc. IEEE TENCON 2008 (2008).

[4] A. Petitet et al., "HPL—A Portable Implementation of the High-performance Linpack Benchmark for Distributed-Memory Computers," http://www.netlib.org/benchmark/hpl/ (2008).

[5] NASA Advanced Supercomputing Division, "NAS Parallel Benchmarks," http://www.nas.nasa.gov/Resources/Software/npb.html (2008).