 Digital Object Identifier : 10.1109/TENCON.2008.4766708

Constructing Execution-Time Estimation Models from Diverse Processing Elements of Heterogeneous Clusters

Shuichi Ichikawa and Yuu Kawai
Department of Knowledge-based Information Engineering,
Toyohashi University of Technology
1-1 Hibarigaoka, Tempaku, Toyohashi 441-8580, Japan
E-mail: ichikawa@ieee.org

***Abstract*—Heterogeneous cluster is a reasonable extension of conventional PC clusters, while it is a difficult target for optimization. Although load imbalance can be alleviated by invoking multiple processes on fast nodes (without modification of source code), the optimal process allocation is not obvious. The preceding studies reported that practical estimation is possible by constructing the execution-time estimation models from homogeneous sub-clusters. In this study, we propose a method to construct the models from diverse processing elements of a heterogeneous cluster, and present some preliminary evaluation results. The derived models were accurate enough to find optimal or sub-optimal allocations, while requiring less nodes for model construction.**

I. Introduction

PC clusters have been widely adopted for scientific and technological 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.

Many existing applications are written for PC clusters, which consist of homogeneous processing elements. Since these 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*) is a simple and straightforward way to alleviate load-imbalance of parallel applications on heterogeneous clusters. The multiprocessing scheme basically requires no modification of source codes, while providing a reasonable speedup by configuring cluster middleware appropriately.

Despite all these advantages, it is not an easy task to find the optimal process allocation for a multiprocessing scheme. The first problem is that the performance ratio between PEs is not always an integer, while the number of processes is always an integer. The communication time and multiprocessing overhead makes things even 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][2] 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) [3], and showed that the optimal or sub-optimal configurations were actually estimated for various sizes of HPL. Ichikawa et al. [4] improved the above scheme by proposing a new execution-time estimation model (NP-T model), and reported that the quality of estimation is drastically improved by using non-negative least squares method to extract parameters for NP-T models.

Although these preceding studies [1][2][4] presented some promising results, they premised that a heterogeneous cluster is comprised of several homogeneous sub-clusters. Such situation is reasonable in case users assemble a large (heterogeneous) cluster from several homogeneous sub-clusters to solve a large computational problem. Their models were thus constructed from 8 (or more) equivalent PEs of each homogeneous sub-cluster.

Meanwhile, the preceding studies presented no reliable means to construct the execution-time estimation models from diverse processing elements of heterogeneous clusters. This study aims (1) to construct the execution-time estimation models of multiprocessing scheme from diverse PEs of heterogeneous clusters, and (2) to evaluate the quality of these models quantitatively.

The rest of this paper is organized as follows. Section II outlines the background and related studies of this work. Then, Section III outlines the execution-time estimation models and the original contribution of our work. The evaluation results are summarized in Section IV.

II. Background and Related Studies

There have been many attempts to rewrite existing parallel applications for heterogeneous clusters. Typically, the applications for heterogeneous clusters are designed to distribute one process to each PE, where each process handles heterogeneous

computational workload according to PE performance. This strategy is called *HoHe* (Homogeneous distribution of processes of parallel program over processors with Heterogeneous distribution of data over processes) in Kalinov's terminology [5].

A serious problem of HoHe strategy is that it is very costly to redesign the existing application. Though the HoHe strategy might be suited to derive the maximum performance possible from a heterogeneous cluster, it is sometimes very difficult to optimize the application. Much effort is required to establish high performance and reliability, and such effort must be repeated for each application.

Meanwhile, a multiprocessing scheme basically requires no modification of the application program; just by modifying the configuration file of cluster/communication middleware, we can derive a reasonable speedup. The multiprocessing scheme does not aim to extract the maximum performance from a heterogeneous cluster, but seeks rather to provide an easy and simple way to accelerate a wide range of conventional parallel applications in heterogeneous clusters.

The multiprocessing scheme is called *HeHo* strategy in Kalinov's terminology [5], which stands for Heterogeneous distribution of processes of parallel program over processors with Homogeneous distribution of data over processes. Besides the author's works [1][2][4], there are some recent studies on this strategy [5][6]. Despite these preceding studies, the advantages and limitations of HeHo strategy are not yet fully explored.

III. Methodology

A. Kishimoto's scheme

This section outlines the original Kishimoto's scheme. More details may be found in previous papers [1][2].

Let N be the size of the problem. A sub-cluster G_i is a group of PEs comprised of equivalent PEs in a heterogeneous cluster, while Γ represents the number of sub-clusters that compose the whole heterogeneous cluster. We construct models from the measurement results of each G_i.

Let P_i be the number of PEs actually used for the job in G_i ($0 \leq P_i \leq |G_i|$). M_i is the number of processes on each PE in G_i, if $P_i \neq 0$; M_i is zero, if $P_i = 0$. A *configuration* of a heterogeneous cluster is defined by the set of P_i and M_i for all sub-clusters: i.e., $(P_1, M_1, ..., P_\Gamma, M_\Gamma)$. P represents the total number of processes in the cluster. When a configuration is given, P is calculated by $P = \sum_i P_i M_i$.

Our goal is to build the execution time estimation model for each possible configuration. In other words, it is to build models that estimate the execution time T_i of G_i from parameters N, P, and M_i. The overall execution time T is estimated by $T = \max_i T_i$.

Each T_i can be estimated as a function of P and N. In case of HPL [3], the estimation function of T_i is given by the following equation [1][2]:

$$T_i = \frac{1}{P} \cdot O(N^3) + P \cdot O(N^2) + O(N^2) \qquad (1)$$

T_i is thus represented by a cubic function of N for specific P and M_i:

$$T_i(N)|_{P,M_i} = k_0 N^3 + k_1 N^2 + k_2 N + k_3 \qquad (2)$$

This kind of model is designated as *N-T model*. The constant factors $k_0, ..., k_3$ are determined from the measurement results by the least squares method. Though the above discussions were made for HPL, the estimation function of N-T model is dependent on the algorithm of target application. The estimation functions for other benchmarks are found in Table III.

Once we have constructed the models for every possible configuration, we can estimate the optimal configuration for a given N by solving a combinatorial optimization problem that minimizes the estimated execution time. Although any pruning techniques for combinatorial optimization might be applied to reduce the search time, we adopt a simple exhaustive search in this study, since the search space is small enough.

This is no more than an estimation based on models; therefore, the estimated optimal configuration is not necessarily an actual optimal configuration. The quality of the estimated optimal configuration may be evaluated by the estimation error ε, which is defined by $\varepsilon = (\hat{\tau} - \hat{T})/\hat{T}$. Here, $\hat{\tau}$ is the actual execution time of the estimated optimal configuration, and $\hat{T}$ is the actual execution time of the actual optimal configuration.

B. NP-T model

Although each N-T model is constructed specific to a configuration (P and M_i), it is not practical to manage many N-T models for every combination of parameters. Therefore, it is desirable to construct more generic models that include both N and P as parameters.

Directly inducing from Eq. (1), the estimation equation of T_i for parameters N and P would be as follows:

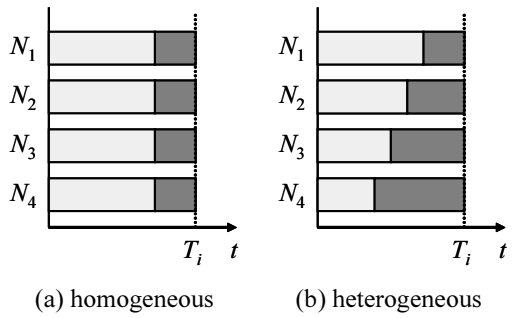
$$\begin{aligned} T_i(N,P)|_{Mi} &= \frac{1}{P}(k_0 N^3 + k_1 N^2 + k_2 N + k_3) \\ &+ P(k_4 N^2 + k_5 N + k_6) \\ &+ k_7 N^2 + k_8 N + k_9 \end{aligned} \qquad (3)$$

This model is designated as *NP-T model* [4]. The above equation includes 10 coefficients ($k_0, ..., k_9$), which can be determined from 10 or more measurements using the least squares method (*parameter extraction*). In particular, the non-negative least squares method is reported to be preferable for NP-T models.

NP-T models are adopted only for the case $P > M_i$, which also follows [4]. The case $P = \exists M_i$ is special in that the application is executed with a single processor, where no communication over a network is involved. Therefore, N-T model is adopted when $P = \exists M_i$ holds.

C. Model construction from heterogeneous sub-clusters

As stated in Section I, the estimation models of the preceding studies were constructed using eight or more equivalent processors of each homogeneous cluster. Figure 1 (a) illustrates the measurement of the execution time T_i of a homogeneous sub-cluster G_i. Since four nodes $(N_1, ..., N_4)$



(a) homogeneous (b) heterogeneous

Fig. 1. Model construction with (a) homogeneous sub-cluster and (b) heterogeneous sub-cluster.

TABLE I
PROBLEM SIZES N FOR MODEL CONSTRUCTION AND MODEL EVALUATION.

Benchmark	Model construction	Model evaluation
FEM	$60 \leq N \leq 442$ (7 pts.)	$60 \leq N \leq 600$ (12 pts.)
HPL	$400 \leq N \leq 6400$ (9 pts.)	$400 \leq N \leq 9600$ (11 pts.)
FFTE	$2^{12} \leq N \leq 2^{20}$ (9 pts.)	$2^{12} \leq N \leq 2^{23}$ (12 pts.)
Himeno	$32 \leq N \leq 192$ (9 pts.)	$32 \leq N \leq 256$ (11 pts.)

of G_i are equivalent, the computation, communication, and synchronization of each node are expected to finish (almost) simultaneously.

Meanwhile, in a heterogeneous sub-cluster, the execution times of each PE are different as illustrated in Fig. 1 (b). Though the faster nodes (N_2, N_3, N_4) finish their computation earlier, they have to wait for the slowest node (N_1) to finish the communication and synchronization. Thus, T_i observed in this case represents the execution time of the slowest node N_1. This fact enables us to construct the model of N_1, even if there are no sufficient number of equivalent nodes.

The problem is the quality of the models derived from heterogeneous sub-clusters. Although the T_i of (b) is mainly dominated by N_1, it might be somewhat affected by other nodes. Namely, T_i might be affected by the arrangement of sub-cluster, or the mixture of heterogeneous nodes. It is thus necessary to compare the models constructed from homogeneous with various heterogeneous sub-clusters.

Though T_i of (a) and (b) might disagree to some extent, it does not necessarily mean that the models derived by (b) are inferior to that by (a). Since we are constructing the models to estimate the optimal configurations of heterogeneous clusters, the situation (b) might be more realistic for our purpose. This is another point that has to be examined experimentally in this study.

IV. EVALUATION

The present study examines the quality of estimation models with four conventional parallel applications as in the preceding study [4]: a CFD (computational fluid dynamics) benchmark, an FEM (finite element method) benchmark, a HPL (linear algebraic system) benchmark, and an FFT (fast Fourier transform) benchmark. The problem sizes N for model construction and evaluation are summarized in Table I for each benchmark. The configurations examined are summarized in Table II. In the following experiments, we adopt the N-T and NP-T models [4], whose parameters are extracted with the non-negative least squares method. The N-T and NP-T models of four benchmarks are summarized in Table III.

TABLE II
CONFIGURATIONS EXAMINED FOR FOUR BENCHMARKS. THE TOTAL NUMBER OF CONFIGURATIONS IS 53 FOR FFTE, AND 188 FOR HIMENO, FEM, AND HPL.

G_1	G_2	G_3
$0 \leq P_1 \leq 2$ $0 \leq M_1 \leq 3$	$0 \leq P_2 \leq 4$ $0 \leq M_2 \leq 2$	$0 \leq P_3 \leq 2$ $0 \leq M_3 \leq 1$

TABLE IV
SPECIFICATIONS OF OUR HETEROGENEOUS CLUSTER.

Sub-cluster	G_1	G_2	G_3
Processor	Pentium 4	Xeon	Celeron M
Clock Freq.	3.6 GHz	2.8 GHz	1.5 GHz
Memory	1 GB	1 GB	1 GB
$\|G_i\|$	8	16	8
OS	FedoraCore 4	Redhat Linux 9	FedoraCore 5
Network	1000BASE-TX		

Table IV lists the specifications of the simple heterogeneous cluster used for the following evaluations. For compilation, Intel C/C++ Compiler 9.0 and Intel Fortran Compiler 9.0 were adopted with MPICH 1.2.7-p1 (buffer size 8 KB) as the MPI library. In this cluster, all processing elements are connected by a single wire-speed switch to exclude the effects of network topology and contention.

Table V lists four arrangements of our heterogeneous cluster for model construction. In the arrangement *homo*, models are constructed with three homogeneous sub-clusters (G_1, G_2, G_3). Meanwhile, in *hetero7*, seven Pentium 4 nodes are appropriated to construct the models of Xeon (G_2) and Celeron M (G_3). It should be noted that only 10 nodes (eight Pentium 4, one Xeon, and one Celeron M) are required for hetero7 arrangement, while 24 nodes are indispensable for homo arrangement.

A. FEM benchmark

The FEM benchmark [7] was originally developed to measure the performance of 3D linear elastic finite-element application. The benchmark kernel is based on a parallel preconditioned Conjugate Gradient (CG) iterative solver with incomplete Cholesky (IC) factorization. The original benchmark deals with a three-dimensional domain of $N \times N \times N$, where N must be a multiple of P. A two-dimensional domain, $N \times N \times 1$, is measured in this study for the restrictions of execution time and memory space.

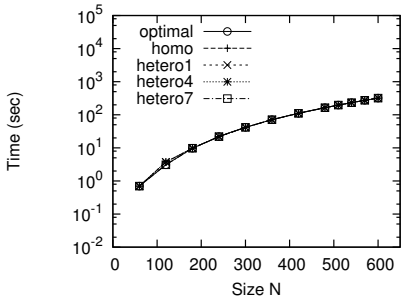
Figure 2 summarizes the evaluation results of the FEM benchmark. Left graph (Fig. 2(a)) plots $\hat{\tau}$, i.e., the actual execution times of the estimated optimal configurations, estimated with four sets of models constructed with the corresponding

TABLE III
N-T AND NP-T MODELS FOR FOUR BENCHMARK PROGRAMS [4].

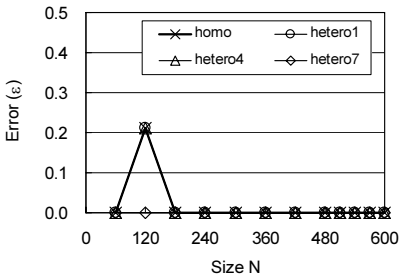
Benchmark	Model	Estimation equation		
FEM	N-T	$T_i(N)\|_{P,Mi}$	$=$	$k_0N^3 + k_1N^2 + k_2N + k_3$
	NP-T	$T_i(N,P)\|_{Mi}$	$=$	$\frac{1}{P}(k_0N^3 + k_1N^2 + k_2N + k_3) + k_4N^2 + k_5N + k_6 + k_7 \log P$
HPL	N-T	$T_i(N)\|_{P,Mi}$	$=$	$k_0N^3 + k_1N^2 + k_2N + k_3$
	NP-T	$T_i(N,P)\|_{Mi}$	$=$	$\frac{1}{P}(k_0N^3 + k_1N^2 + k_2N + k_3) + P(k_4N^2 + k_5N + k_6) + k_7N^2 + k_8N + k_9$
FFTE	N-T	$T_i(N)\|_{P,Mi}$	$=$	$k_0N \log N + k_1N + k_2N^{\frac{1}{3}} + k_3$
	NP-T	$T_i(N,P)\|_{Mi}$	$=$	$\frac{1}{P}(k_0N \log N + k_1N + k_2) + k_3P + k_4N + k_5N^{\frac{1}{3}} + k_6$
Himeno	N-T	$T_i(N)\|_{P,Mi}$	$=$	$k_0N^3 + k_1N^2 + k_2N + k_3$
	NP-T	$T_i(N,P)\|_{Mi}$	$=$	$\frac{1}{P}(k_0N^3 + k_1N^2 + k_2N + k_3) + k_4N^2 + k_5N + k_6 + k_7 \log P$

TABLE V
FOUR ARRANGEMENTS TO CONSTRUCT THE ESTIMATION MODELS FOR EACH SUB-CLUSTER.

	G_1	G_2	G_3
homo	Pentium4×8	Xeon×8	CeleronM×8
hetero1	Pentium4×8	Xeon×7, Pentium4×1	CeleronM×7, Pentium4×1
hetero4	Pentium4×8	Xeon×4, Pentium4×4	CeleronM×4, Pentium4×4
hetero7	Pentium4×8	Xeon×1, Pentium4×7	CeleronM×1, Pentium4×7



(a) The actual execution times of the actual and estimated optimal configurations ($\hat{T}$ and $\hat{\tau}$).



(b) The estimation error (ε).

Fig. 2. Evaluation results of FEM benchmark.

arrangements (**homo**, **hetero1**, **hetero4**, and **hetero7**). The line **optimal** represents $\hat{T}$, i.e., the actual execution times of the actual optimal configurations. Right graph (Fig. 2(b)) plots the corresponding estimation errors (ε).

As readily seen, the optimal configurations were estimated in most cases, with any set of models. The only exception appeared at $N = 120$, where the homo, hetero1, and hetero4 models incurred 21% estimation error, while the hetero7 models found the optimal configuration. This difference is insignificant, and all four sets of models are regarded practically comparable for FEM benchmark.

B. HPL

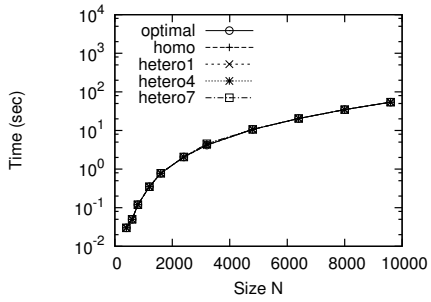
HPL (High Performance Linpack) [3] is a portable implementation of the Linpack benchmark for distributed-memory computers. HPL solves a random dense linear system of equations in double precision floating-point arithmetic. In this study, ATLAS [8] was adopted as the BLAS library, and the process grid was fixed to one-dimensional block cyclic as in the previous study [2].

Figure 3 plots the evaluation results of the HPL benchmark. The results of HPL are similar to that of FEM; the optimal configurations were successfully estimated in most cases, with any of four sets of models. The only exception appeared at $N = 3200$, where the hetero4 and hetero7 models incurred 8% error, while the homo and hetero1 models successfully found the optimal configuration. This difference is regarded insignificant, and all four sets of models are practically comparable for HPL benchmark.

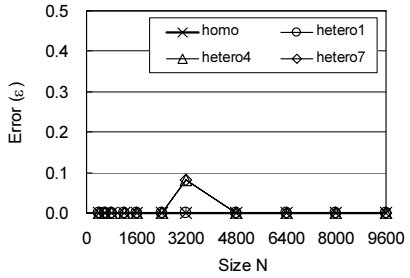
C. FFTE benchmark

FFTE benchmark [9] measures the execution time of double precision complex one-dimensional DFT of size N, where N must be a multiple of P^2. In this study, we examine the cases where P is a power of 2, because FFTE is written on this implicit assumption.

Figure 4 plots the evaluation results of the FFTE benchmark. The results of FFTE are also similar to that of FEM and HPL; the optimal configurations were successfully estimated in most cases, with any of four sets of models. The first

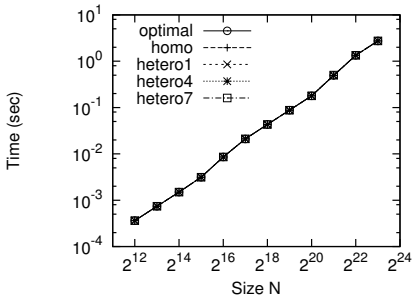


(a) The actual execution times of the actual and estimated optimal configurations ($\hat{T}$ and $\hat{\tau}$).

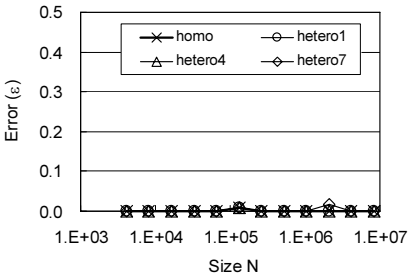


(b) The estimation error (ε).

Fig. 3. Evaluation results of HPL benchmark.



(a) The actual execution times of the actual and estimated optimal configurations ($\hat{T}$ and $\hat{\tau}$).



(b) The estimation error (ε).

Fig. 4. Evaluation results of FFTE benchmark.

difference appeared at $N = 2^{17}$, where all four sets estimated a sub-optimal configuration whose error is 1%. The second difference appeared at $N = 2^{21}$, where the hetero7 models incurred 2% error, while other three successfully found the optimal configuration. Needless to say, these differences are insignificant, and all four sets of models are regarded practically comparable for FFTE benchmark.

D. Himeno benchmark

The Himeno benchmark [10] was originally developed to evaluate the performance of incompressible Navier-Stokes solver in fluid analysis code. It measures the performance of a kernel, which solves a pressure Poisson equation with Jacobi iteration. Recently, the Himeno benchmark has been widely used as an HPC benchmark for supercomputers and clusters.

Figure 5 displays the evaluation results of Himeno benchmark. The results of Himeno benchmark are much different from that of other three benchmarks. It is readily seen that the estimation errors ε of Himeno benchmark are larger than in other benchmarks. The homo and hetero1 models yielded the identical results in this experiment. Though they incur slight errors in $48 \leq N \leq 96$, the optimal configurations were successfully found for other sizes ($N = 32$ and $112 \leq N \leq 256$). The results of the hetero4 models are similar to that of the homo models, except that they additionally incur 28% error in $N = 128$ and 17% error in $N = 160$. The errors of the hetero7 models are still larger than that of the hetero4 models.

In summary, the heterogeneity in sub-clusters gradually degrades the quality of the derived models in Himeno benchmark.

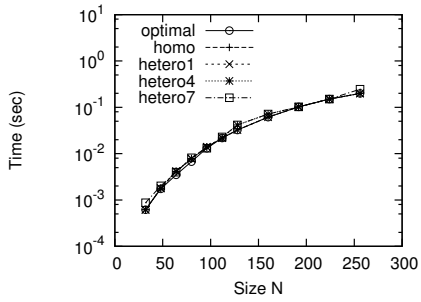
E. The effect of heterogeneity on execution time

As stated in Section III-C, the execution time T_i of sub-cluster G_i might be affected by the mixture of nodes in G_i. In fact, it sounds reasonable that T_i is dependent on the total performance of G_i. If this hypothesis is correct, the execution time for a specific configuration should increase in the following order: hetero7, hetero4, hetero1, homo. This issue is examined in this section.

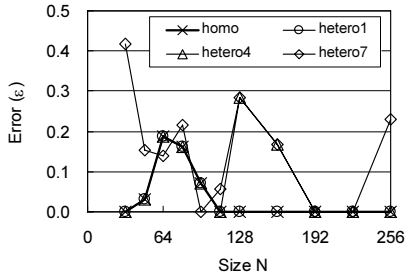
Figure 6 summarizes the execution times of Xeon sub-cluster of four arrangements. In HPL benchmark (Fig. 6(a)), the execution times increase in due order as expected. Meanwhile, the behavior of Himeno benchmark (Fig. 6(b)) does not meet our expectations. The execution times of four arrangements are not simply dependent on the total performance; the differences of execution times are too large considering the total performance of each arrangement. The relationships between T_i and N are also irregular. All these behaviors might have affected the accuracy of models and the quality of estimation.

V. Conclusion

This present study examined three simple heterogeneous arrangements to construct models from heterogeneous sub-

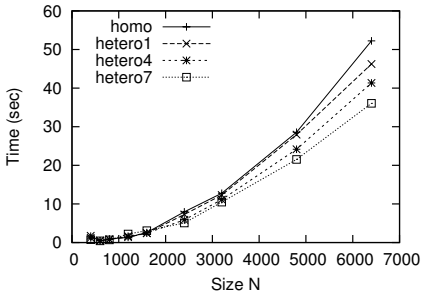


(a) The actual execution times of the actual and estimated optimal configurations ($\hat{T}$ and $\hat{\tau}$).

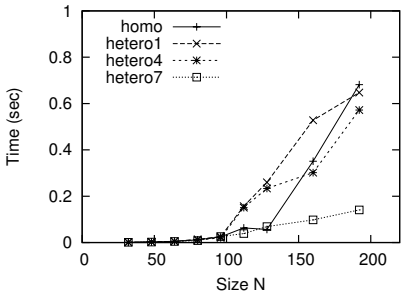


(b) The estimation error (ε).

Fig. 5. Evaluation results of Himeno benchmark.



(a) HPL benchmark



(b) Himeno benchmark

Fig. 6. The measured execution-times of a Xeon node, where 2 processes are invoked on a Xeon ($M_i = 2$) and the total number of processes is 16 ($P = 16$).

clusters. The models constructed with heterogeneous sub-clusters were comparable to that of homogeneous sub-clusters in FEM, HPL, and FFTE benchmarks. Meanwhile, in case of Himeno benchmark, the estimation quality gradually degrades as the heterogeneity increases. After the quantitative evaluation of the estimation errors, we concluded that the modest heterogeneity in sub-clusters is well acceptable for model construction.

Although our preliminary evaluation results suggest that our scheme is promising, further investigation is desired for more heterogeneous sub-clusters, each of which consists of three or more different kinds of PEs. In the following studies, a larger heterogeneous cluster, which consists of more PEs, should be also examined to verify the quality of our scheme with a larger number of configurations. All these issues are left for future studies.

ACKNOWLEDGMENT

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

REFERENCES

[1] Y. Kishimoto and S. Ichikawa, "An execution-time estimation model for heterogeneous clusters," in *Proc. 18th International Parallel and Distributed Processing Symposium (IPDPS 2004)*. IEEE Computer Society, 2004, (CD-ROM).

[2] ——, "Optimizing the configuration of a heterogeneous cluster with multiprocessing and execution-time estimation," *Parallel Computing*, vol. 31, no. 7, pp. 691–710, 2005.

[3] A. Petitet, R. C. Whaley, J. Dongarra, and A. Cleary, "HPL – a portable implementation of the high-performance Linpack benchmark for distributed-memory computers," http://www.netlib.org/benchmark/hpl/.

[4] S. Ichikawa, S. Takahashi, and Y. Kawai, "Optimizing process allocation of parallel programs for heterogeneous clusters," *Concurrency and Computation: Practice and Experience*, (accepted).

[5] A. Kalinov and S. Klimov, "Optimal mapping of a parallel application processes onto heterogeneous platform," in *Proc. 19th IEEE International Parallel and Distributed Processing Symposium (IPDPS 2005)*. IEEE Computer Society, 2005, (CD-ROM).

[6] J. Cuenca, D. Gimenez, and J.-P. Martinez, "Heuristics for work distribution of a homogeneous parallel dynamic programming scheme on heterogeneous systems," *Parallel Computing*, vol. 31, no. 5, pp. 711–735, 2005.

[7] K. Nakajima, "Test programs for parallel iterative methods on various types of architectures (hpc-mw-solver-test ver.1.0)," http://www.fsis.iis.u-tokyo.ac.jp/en/result/software/, 2003.

[8] "Automatically tuned linear algebra software (ATLAS)," http://math-atlas.sourceforge.net/, 2006.

[9] D. Takahashi, "FFTE: A fast Fourier transform package," http://www.ffte.jp/, 2005.

[10] R. Himeno, "Himeno benchmark," http://accc.riken.jp/E/HPC_e/HimenoBMT_e/index_e.html, 2005.