PAPER Special Section on Parallel/Distributed Computing and Networking

Influence of Inaccurate Performance Prediction on Task Scheduling in a Grid Environment*

Yuanyuan ZHANG[†a)], *Nonmember and* **Yasushi INOGUCHI**[††], *Member*

SUMMARY Efficient task scheduling is critical for achieving high performance in grid computing systems. Existing task scheduling algorithms for grid environments usually assume that the performance prediction for both tasks and resources is perfectly accurate. In practice, however, it is very difficult to achieve such an accurate prediction in a heterogeneous and dynamic grid environment. Therefore, the performance of a task scheduling algorithm may be significantly influenced by prediction inaccuracy. In this paper, we study the influence of inaccurate predictions on task scheduling in the contexts of task selection and processor selection, which are two critical phases in task scheduling algorithms. We develop formulas for the misprediction degree, which is defined as the probability that the predicted values for the performances of tasks and processors reveal different orders from their real values. Based on these formulas, we also investigate the effect of several key parameters on the misprediction degree. Finally, we conduct extensive simulation for the sensitivities of some existing task scheduling algorithms to the prediction errors.

key words: *grid computing, task scheduling, performance prediction, task selection, processor selection*

1. Introduction

Grid computing [1], the internet-based infrastructure that aggregates geographically distributed and heterogeneous resources to solve large-scale problems, is becoming increasingly popular because it provides us with the ability to dynamically link resources together as an ensemble to support the execution of large-scale, resource-intensive, and distributed applications. Many projects have been developed to implement the pervasive applications of Grid, such as Globus [2], GrADS [3], DataGrid [4] and Legion [5].

Resource management is the central part of a grid environment that makes the grid work properly. The process of resource management in grid environment involves, in general, the discovery of available resources, the selection of an application-appropriate subset of those resources, and the mapping of tasks onto the selected resources. The mapping of tasks onto resources, which is called task scheduling, is a critical component for achieving high performance in a grid computing environment. The objective of task scheduling problem is to minimize some metric. A typical goal is to minimize the application makespan, the time elapsed between the first application task starts executing and the last task completes. Since this problem is NP-complete in most cases [6], different heuristics have been used to reach a near-optimal solution.

Because of its key importance, the task scheduling problem has been extensively studied and various algorithms have been proposed in the literature [7]–[10]. Usually, the scheduling process of these algorithms involves two phases: task selection and processor selection. In the task selection phase, the tasks are sorted in a list; while in the processor selection phase, the first task in the list is selected and allocated to a processor. Scheduling algorithms often use prediction values for the performances of tasks and processors to perform the assignment of tasks to resources, therefore, accurate performance prediction is critical for the achievement of satisfactory performance of a grid task scheduling system. Usually such predictions are executed by some performance prediction tool and can be distinguished into two categories: application run time prediction [11], [12] and processor load prediction [13]–[15], each of which has been widely studied.

It is almost always assumed that a scheduling algorithm is able to obtain perfect prediction knowledge about the performance of both tasks and processors, and thus don't take prediction error into consideration. However, although modern performance prediction tools are increasingly accurate, it is still impossible to obtain exact predictions since the grid, in which many applications share resources, is a highly dynamic environment. The implementations of the scheduling algorithms will use inaccurate predictions, accordingly the performance of these algorithms is influenced by such inaccuracies, and different algorithms reveal different degrees of sensitivity to them. Therefore it is necessary to evaluate the scheduling algorithms not only under the traditional assumption of perfectly accurate prediction, but also under a variety of prediction error scenarios. In this paper, we focus on the influence of inaccurate performance prediction on task scheduling in detail from the perspectives of task selection and processor selection in a grid environment.

The remainder of this paper is organized as follows: in the next section, we analyze the influence of inaccurate prediction on task scheduling, introduce the concept of misprediction degree, and establish related formulas. The impact of some key parameters in the formulas is evaluated and presented in Sect. 3. Section 4 presents the simulation results of the sensitivity of several existing scheduling algorithms

Manuscript received April 4, 2005.
Manuscript revised August 16, 2005.

[†]The author is with the School of Information Science, JAIST, Nomi-shi, 923–1292 Japan
[††]The author is with the Center for Information Science, JAIST, Nomi-shi, 923–1292 Japan

*This research is conducted as a program for the "21st Century COE Program" by Ministry of Education, Culture, Sports, Science and Technology, Japan.

a) E-mail: yuanyuan@jaist.ac.jp

DOI: 10.1093/ietisy/e89–d.2.479

to the inaccurate performance prediction. Finally, we conclude the paper in Sect. 5.

2. Analysis of the Influence of Prediction Error on Task Scheduling

Grid computing is a highly heterogeneous and dynamic environment. The heterogeneity comes from the fact that the resources in grid can cover many different types of resources, such as supercomputers, storage systems and databases, scientific instruments, and visualization devices; even resources of same type may have different configurations. The dynamic characteristic comes from the non-dedicated nature of grid resources, that is, grid applications can only use the spare capability of the grid resources. Therefore the task scheduling problem in a grid environment is much different from that in a homogeneous system and is much more difficult. In this section, we analyze the effect of inaccurate prediction on task scheduling from task selection phase and processor selection phase.

2.1 Notations

Table 1 describes the notations used in the paper.

For two tasks T_1 and T_2, let at_1 and at_2 be respectively the actual run times of T_1 and T_2. et_1 and et_2, which denote prediction errors of at_1 and at_2, are independent random variables. In a grid scheduling system, when we use performance prediction tools to predict the performance of the tasks, the predicted errors usually lie in an interval of the actual run times according to some probability distribution [16], so we assume et_1 and et_2 follow some probability distribution with the probability density function g(et)in the ranges of $[-\alpha at_1, \beta at_1]$ and $[-\alpha at_2, \beta at_2]$.

pt_1 and pt_2 are the predicted run times of T_1 and T_2. For pt_1 and pt_2, we have the following equations:

$$pt_1 = at_1 + et_1; \qquad pt_2 = at_2 + et_2. \tag{1}$$

Since et_i(i=1,2) is in $[-\alpha at_i, \beta at_i]$, pt_i will fall in $[(1-\alpha)at_i, (1+\beta)at_i]$. Because $pt_i>0$, the possible value ranges of α and β are [0, 1) and [0, ∞) respectively.

For the relationship between actual and predicted speeds of processors, we also have similar equations.

Table 1 Notations.

Phase	Notation	Description
Task Selection	T_i	A task in the application
	at_i	Actual run time. $at_i>0$
	et_i	Prediction error of at_i
	pt_i	Predicted run time. $pt_i>0$
	f(at)	p.d.f. of actual run time at
	g(et)	p.d.f. of et
Processor Selection	m	Number of processors
	P_i	A processor in the grid
	as_i	Actual speed. $as_i>0$
	es_i	Prediction error of as_i
	ps_i	Predicted speed. $ps_i>0$
	h(es)	p.d.f. of es

2.2 Task Selection

Since it is impossible to obtain accurate predictions for the run times of the tasks because of the dynamic nature of the grid, the actual run times of the tasks will be different from the predicted values, and this will affect the performance of task scheduling algorithms in the grid environment. For example, if the actual run time of task T_i is less than that of task T_j, while because of prediction error, the predicted run time of T_i is more than that of T_j, then we will make an incorrect scheduling decision if we schedule the tasks based on their predicted run times. Furthermore, different algorithms will have different degrees of sensitivity to such an error.

In this paper we focus on a grid application which is composed of a set of independent tasks. The actual run times of these tasks are independent identical distribution (i.i.d.) random variables. Usually task scheduling algorithms schedule tasks based on their predicted run times. When, for example, an algorithm schedules the task with the longest run time first or with the shortest run time first, inaccurate prediction can have a remarkable influence on its performance if the actual run time of task T_1 is smaller than that of task T_2 while the predicted value of T_1 is greater than that of T_2. We call such a situation *misprediction.* Because different performance prediction tools have different degrees of prediction inaccuracy, they can create different degrees of misprediction. Therefore, what we are interested in is the probability that misprediction will occur, i.e., $P(pt_1 > pt_2|at_1 < at_2)$, which is called *misprediction degree for two tasks* here.

The above probability can be converted into:

$$\begin{aligned} & P\ (pt_1 > pt_2|at_1 < at_2) \\ &= P(at_1 + et_1 > at_2 + et_2|at_1 < at_2) \\ &= P(et_1 > et_2 + at_2 - at_1|at_1 < at_2), \end{aligned} \tag{2}$$

where $et_1 \in [-\alpha at_1, \beta at_1]$, and $et_2 \in [-\alpha at_2, \beta at_2]$.

In the coordinate system of et_1 and et_2, the inequality $et_1 > et_2 + at_2 - at_1$ is the area above the line L: $et_1 = et_2 + at_2 - at_1$, and the probability of $P(et_1 > et_2 + at_2 - at_1|at_1 < at_2)$ can be expressed by the area of the overlapping region between L and the rectangle surrounded by the lines $et_1 =$

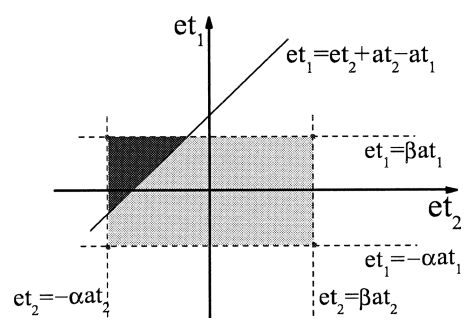

Fig. 1 The probability $P(et_1 > et_2 + at_2 - at_1|at_1 < at_2)$ can be expressed as the overlapping area between L: $et_1 = et_2 + at_2 - at_1$ and the rectangle.

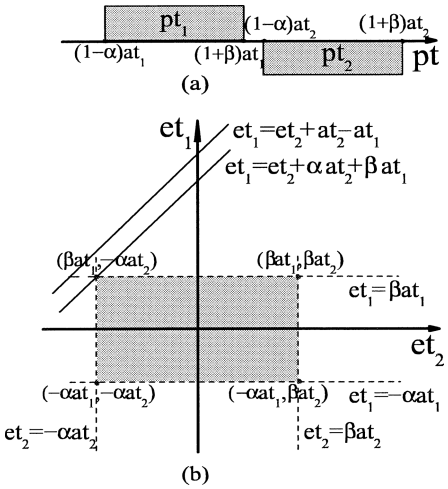

Fig. 2 The case when there is no misprediction. (a) value ranges of predicted run times which don't overlap; (b) situation of the misprediction degree.

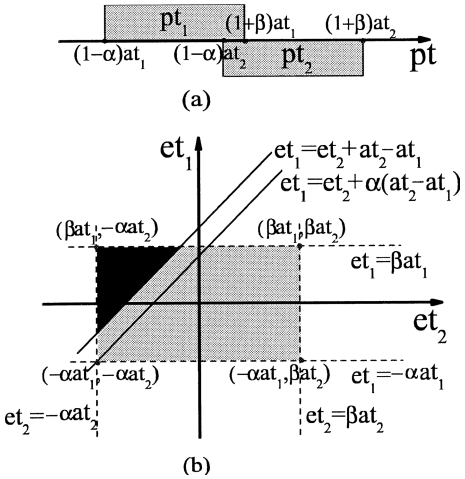

Fig. 3 The case in which misprediction happens. (a)the value ranges of the predicted run times which overlap; (b)situation of the misprediction degree. The overlapping region is a triangle.

$-\alpha at_1$, $et_1 = \beta at_1$, $et_2 = -\alpha at_2$ and $et_2 = \beta at_2$, as shown in Fig. 1.

About the overlapping region, we have the following conclusion:

Proposition 1. There are only two cases for the overlapping region between L and the rectangle: either they don't overlap or they overlap in a triangle region.

Proof. (1) The line which is parallel to L and passes the point $(\beta at_1, -\alpha at_2)$ is $et_1 = et_2 + \alpha at_2 + \beta at_1$. If L is above this line, then we can see intuitionally that there is no overlapping between L and the rectangle. That is to say, if $at_2 - at_1 \geq \alpha at_2 + \beta at_1$, i.e., if $(1-\alpha)at_2 \geq (1+\beta)at_1$, the probability $P(et_1 > et_2 + at_2 - at_1 | at_1 < at_2)$ equals to 0. This is very easy to understand because in this case the predicted run time of T_2 is always greater than that of T_1, it is impossible for the misprediction to happen. This case is shown in Fig. 2. It can be expressed mathematically as:

$$P(et_1 > et_2 + at_2 - at_1 | at_1 < at_2) = 0 \quad if (1-\alpha)at_2 \geq (1+\beta)at_1, at_2 > at_1 \tag{3}$$

(2) If L is under the line $et_1 = et_2 + \alpha at_2 + \beta at_1$, then it will overlap with the rectangle. The probability for misprediction to happen is equal to the area of the overlapping region. The line which is parallel to L and passes the point $(-\alpha at_1, -\alpha at_2)$ is: $et_1 = et_2 + \alpha(at_2 - at_1)$. Since $0 \leq \alpha < 1$, $\alpha(at_2 - at_1) < at_2 - at_1$. So L is always above the line $et_1 = et_2 + \alpha(at_2 - at_1)$. From Fig. 3 we can see that the overlapping region can only be a triangle.

So the proof is completed.

In the case in which L and the rectangle overlap in a triangle, the area of the overlapping region, i.e., the probability $P(et_1 > et_2 + at_2 - at_1 | at_1 < at_2)$, can be expressed by the double integral of the probability density functions $g(et_1)$ and $g(et_2)$. So we have the following equation:

$$P(et_1 > et_2 + at_2 - at_1 | et_1 < et_2) = \int_{-\alpha at_2}^{(1+\beta)at_1 - at_2} \int_{et_2 + at_2 - at_1}^{\beta at_1} g(et_1) g(et_2) det_1 det_2 \quad if (1-\alpha)at_2 < (1+\beta)at_1 \tag{4}$$

If the probability density function of the actual run times of the tasks in the grid application is f(at) and the value range is [tl, tu], then the *misprediction degree for the grid application,* which is denoted by *DM,* is defined as the average of the misprediction degree between any two tasks in the application. By virtue of the equation for the misprediction degree between two tasks as shown before, *DM* can be expressed as:

$$DM = \int_{tl}^{tu} \int_{tl}^{\frac{1+\beta}{1-\alpha}at_1} f(at_1) f(at_2) \int_{-\alpha at_2}^{(1+\beta)at_1 - at_2} \int_{et_2 + at_2 - at_1}^{\beta at_1} g(et_1) g(et_2) det_1 det_2 dat_2 dat_1 \tag{5}$$

This result is independent of the specific probability density functions for $f(at)$ and $g(et)$.

2.3 Processor Selection

In processor selection phase of a task scheduling algorithm, the algorithm usually selects a processor to execute the selected task according to the execution times of the task on the processors. Given the computational size, the running time of a computational-bound task on a processor is decided by the computational speed of the processor which is perceived by this task, that is, the speed dedicated to execute this task, which is in turn decided by the load on the processor. Many researchers have worked on the processor load prediction, but as a grid is a highly dynamic environment, the prediction for the processor loads usually can't be perfectly accurate, so that the predicted speed to execute the

task on the processor will be different from the actual speed, and such inaccurate prediction will affect the performance of the task scheduling algorithm.

Let as_i and as_j be the actual speeds of processor P_i and P_j respectively($as_i > 0$, $as_j > 0$). The prediction errors for them are es_i and es_j, which are independent random variables. es_i and es_j lies in $[-\gamma as_i, \theta as_i]$ and $[-\gamma as_j, \theta as_j]$ respectively with the same probability density function h(es). Similar as the constraint for the values of α and β in task selection phase, we have $0 \le \gamma < 1$, $\theta \ge 0$. The predicted speeds of P_i and P_j are ps_i and ps_j($ps_i > 0$, $ps_j > 0$). We have the following equations:

$$ps_i = as_i + es_i; \qquad ps_j = as_j + es_j. \tag{6}$$

The misprediction degree for two processors is defined as the probability of the event that the actual computational speed as_i is smaller than as_j, while because of prediction errors, the predicted speed ps_i is greater than ps_j, that is, the probability $P(ps_i > ps_j | as_i < as_j)$. This can be further transformed to:

$$\begin{aligned} &P(ps_i > ps_j | as_i < as_j) \\ &= P(as_i + es_i > as_j + es_j | as_i < as_j) \\ &= P(es_i > es_j + as_j - as_i | as_i < as_j) \end{aligned} \tag{7}$$

Following a similar method as in task selection, we can derive the equation for processor selection:

$$\begin{aligned} &P\ (es_i > es_j + as_j - as_i | as_i < as_j) = 0 \\ &\quad if (1-\gamma) as_j \ge (1+\theta) as_i \end{aligned} \tag{8}$$

and

$$\begin{aligned} &P\ (es_i > es_j + as_j - as_i | as_i < as_j) \\ &= \int_{-\gamma as_j}^{(1+\theta)as_i - as_j} \int_{es_j + as_j - as_i}^{\theta as_i} h(es_i) h(es_j) des_i des_j \\ &\quad if (1-\gamma) as_j < (1+\theta) as_i \end{aligned} \tag{9}$$

The misprediction degree for the scheduling of a task in a grid system with m processors, which is denoted by DM_p, is defined as the average of the misprediction degree between any two processors. In virtue of the equation for the misprediction degree between two processors as shown before, DM_p can be expressed as:

$$DM_p = \frac{1}{C_m^2} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} P(es_i > es_j + as_j - as_i | as_i < as_j) \tag{10}$$

3. Study of Evaluation Results

The formulas established in Sect. 2 do not depend on the specific distribution of the probability density functions f(at), g(et) and h(as). In this section, we present the results of extensive evaluations which assess the impact of the parameters in formula (5) and formula (10), which represent the influence of inaccurate prediction on task selection and processor selection respectively.

3.1 Task Selection Phase

For task selection phase, in formula (5) the parameters are α, β, tl, tu, f(at) and g(et). For our evaluation, we choose uniform distribution for both f(at) and g(et). The evaluation results are shown in Figs. 4 and 5.

Figure 4 shows the influence of parameters α and β on the misprediction degree for a grid application. In Fig. 4 (a), (α, β) increases from (0.1, 0.1) to (0.9, 0.9) with the range of prediction error increases from $(-0.1at_i, 0.1at_i)$ to $(-0.9at_i, 0.9at_i)$, while the average prediction error remains constantly to be 0. From Fig. 4 (a) it's shown that under same average prediction error, DM increases as range of prediction error increases. This is because as the enlargement of the range of prediction errors, the range of predicted run times also increases, so that the probability of misprediction is increased. In Fig. 4 (b), the range of (α, β) shifts gradually from (0.1, 0.9) to (0.9, 0.1), so that the predicted error shifts gradually from $(-0.1at_i, 0.9at_i)$ to $(-0.9at_i, 0.1at_i)$, that is, the range size of prediction error remains, while its value shifts gradually from overestimate to underestimate. From Fig. 4 (b) we can see that with the same range size of prediction error, DM increases as the prediction shifts from overestimate to underestimate. This can be explained as follows: for any two tasks T_1 and T_2 with $at_1 < at_2$, pt_i is in $[(1-\alpha)at_i, (1+\beta)at_i]$(i=1,2). As the prediction changes from overestimate to underestimate, that is, α is increased and β is decreased, both of the average values of pt_1 and pt_2 decreases, but the decrease of mean of pt_2 is larger than that of

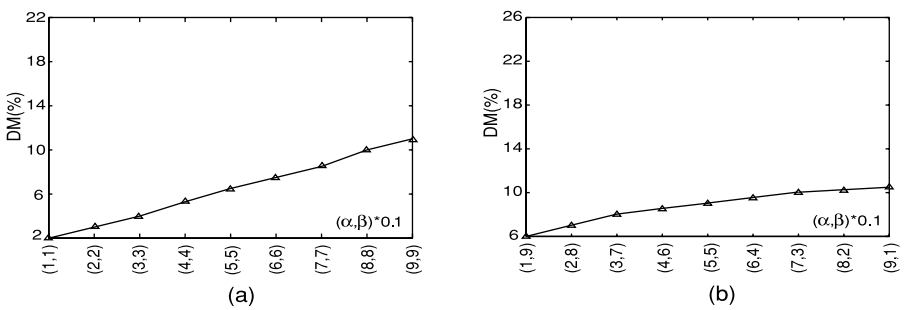

Fig. 4 Influence of parameters α and β on DM:(a) influence of range size of prediction errors. (b)influence of range location of prediction errors.

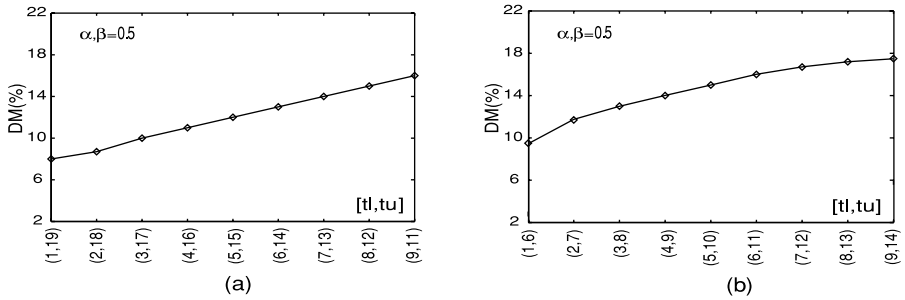

Fig. 5 Influence of actual run time on DM:(a)influence of range size of actual run time. (b)influence of range location of actual run time.

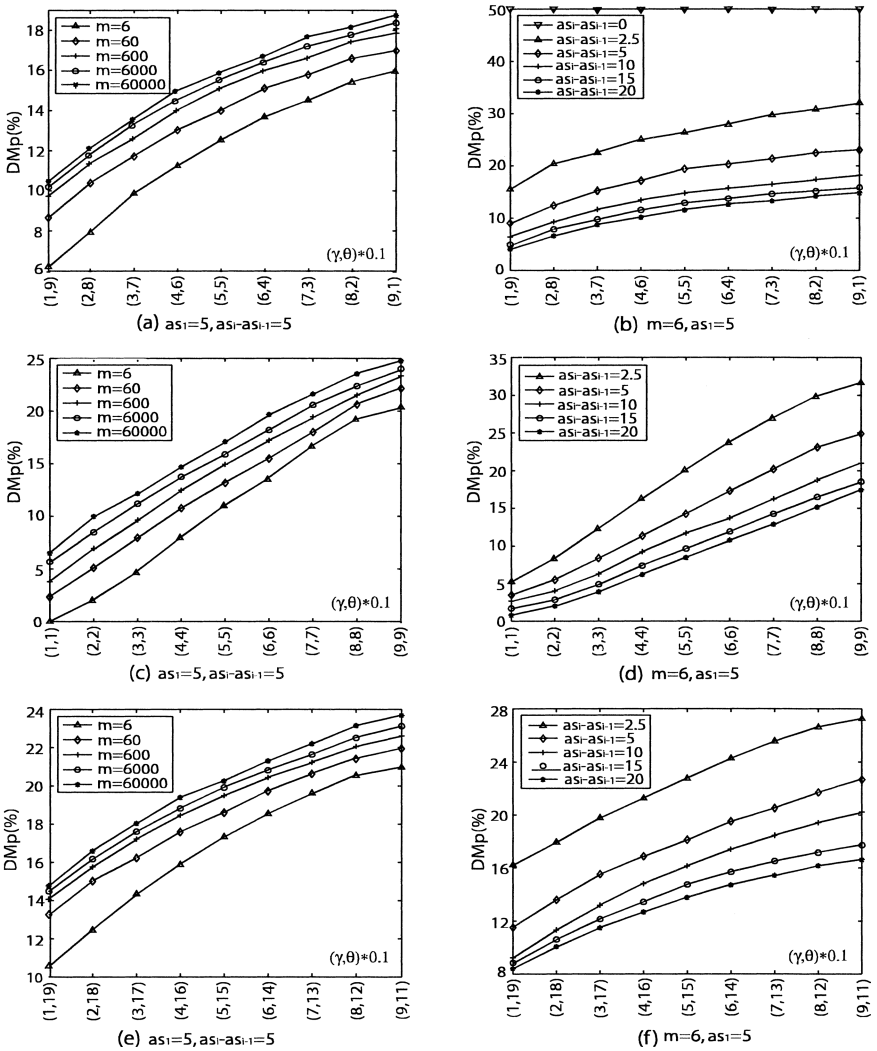

Fig. 6 Influence of parameters γ, θ, m and processor heterogeneity on the misprediction degree.

pt_1 since $at_2 > at_1$, therefore the probability $P(pt_1 > pt_2 | at_1 < at_2)$ increases.

Figure 5 shows the influence of actual run time on DM. In Fig. 5 (a), the average value of actual run times remains (at 10), while the range size decreases. Figure 5 (a) shows that with same average value of actual run time, DM increases as the range size of actual run time decreases. This is because as the range size of actual run times decreases, the range size of predicted run times also decreases, so that increases the probability of misprediction. In Fig. 5 (b), the range size of actual run time remains constant, while the average actual run time increases. Figure 5 (b) shows that DM increases in this case.

3.2 Processor Selection Phase

The parameters in formula (10) are γ, θ, m, as_i ($1 \leq i \leq m$) and h(es). In our investigations, uniform distribution for h(es) is assumed. The evaluation result is shown in Fig. 6 (a)–(f), where the horizontal axis in every figure is the combination of the values of γ and θ, and the vertical axis is DM_p, the misprediction degree for processor selection in the grid task scheduling system. We assume the processors are sorted by increasing actual speeds here, that is, we have $as_i \leq as_j$ if $i < j$.

Figure 6 (a), (c) and (e) show the influence of grid scale on the misprediction degree, while (b), (d) and (f) show the impact of heterogeneity. From Fig. 6 (a), (c) and (e) we can see that the influence of prediction error on the misprediction degree increases as the number of processors increases under same processor heterogeneity, while from Fig. 6 (b), (d) and (f) we draw the conclusion that the influence of prediction error on the misprediction degree decreases as the heterogeneity of grid system increases under same grid scale(number of processors remains). The misprediction degree is maximum(0.5) when the grid becomes a homogeneous environment.

In Fig. 6 (a) and (b) the range of (γ,θ) shifts gradually from (0.1, 0.9) to (0.9, 0.1), so that the predicted error shifts gradually from overestimate to underestimate. From Fig. 6 (a) and (b) we draw the conclusion that the misprediction degree increases gradually as the inaccurate prediction changes from overestimate to underestimate. This result is consistent with that in task selection phase. Moreover, such an increase is slow in highly heterogeneous grid systems, while in grid systems with low heterogeneity, the misprediction degree increases faster with the shift of (γ,θ).

In Fig. 6 (c) and (d), the range size of prediction error increases with the increase of (γ,θ), while the average prediction error remains constantly to be 0. In Fig. 6 (c) and (d), the misprediction degree increases as the range of prediction error increases. This can be explained similarly as in Fig. 4 (a).

In Fig. 6 (e) and (f), the predicted error shifts from $[-0.1as_i, 1.9as_i]$ to $[-0.9as_i, 1.1as_i]$. Comparing Fig. 6 (e) with (a) and (f) with (b), we draw the conclusion that the misprediction degree increases when both of the range size and average of prediction error increase.

4. Evaluation of Performance Fluctuation of Scheduling Algorithms under Prediction Error

As we have said in Introduction, different scheduling algorithms have different sensitivities to the prediction inaccuracy. We have simulated some existing scheduling algorithms to evaluate the influence of inaccurate predictions on their performances.

4.1 Simulation Environment

To effectively evaluate and compare the efficacy of the scheduling algorithms over inaccurate prediction for increasingly complex distributed, dynamic, heterogeneous environments in order to determine which ones are practical in computational grid settings, we have developed a simulated grid environment to simulate these algorithms based on Simgrid toolkit [17] which provides core functionalities for the simulation of distributed applications in heterogeneous distributed environments.

In the simulated grid environment, there are 100 processors. Taking into account the characteristic of dynamicity of grid environment, the background load of every processor are simulated by a vector of time-stamped values, or traces, obtained from NWS [18] measurements on real resources. NWS provides dynamic performance prediction for network and resources over a given time interval. The total number of tasks is 4000 and tasks arrive in a Poisson process with the average task arrival interval is 5s. The actual run times of the tasks are also modelled by traces to simulate the application in the real world. The scheduling interval is 1000s.

The task scheduling algorithms we choose here are Fast Greedy algorithm, Min-Min algorithm, Max-Min algorithm and Sufferage heuristic [19]. For every condition, we simulate every algorithm for 1000 runs to get an unbiased result. The performance metric we use to evaluate the algorithms is makespan.

4.2 Experimental Results

In our simulation, we assume uniform distribution for the prediction error of both application run time and processor speed.

In Fig. 7, every algorithm is executed with accurate prediction, ±5%, ±10%, ±15%, ±20%, ±25%, and ±30% prediction errors respectively for both of the task run time and processor speed. Figure 7 shows the average, minimum and maximum values of makespan of every algorithm under the selected prediction errors. From the figure we can see that all of the average value, maximum and minimum value of makespan of every algorithm increase as the increase of the prediction error. This means that the influence of prediction error on all of the scheduling algorithms increases as the range size of prediction error increases, which is consistent with our expectation.

From Fig. 7 it can be seen that under the given grid environment, for accurate prediction, among the four selected scheduling algorithms the makespan of Min-Min is largest while Max-Min algorithm has the best scheduling performance. However, under same prediction error, the increase of makespan over the actual values is fastest for the Sufferage algorithm, while it is slowest for Min-Min algorithm, this means that the influence of inaccurate prediction is smallest for Min-Min algorithm although it reveals worst

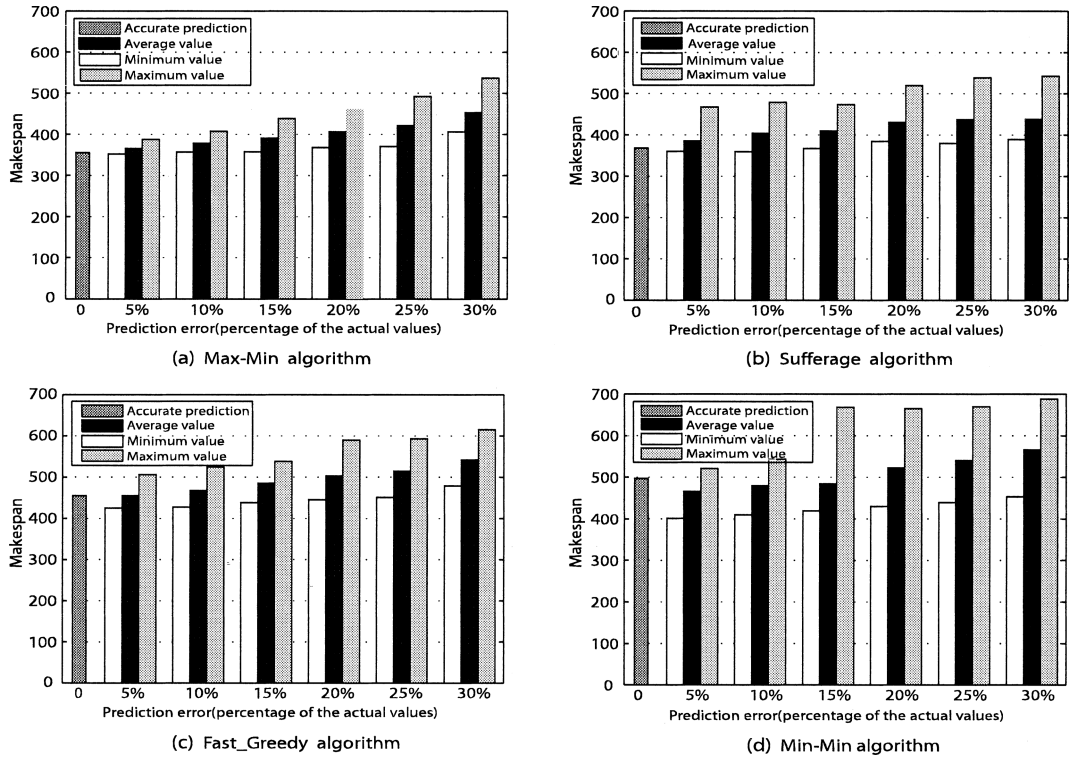

Fig. 7 Influence of prediction error to different scheduling algorithms.

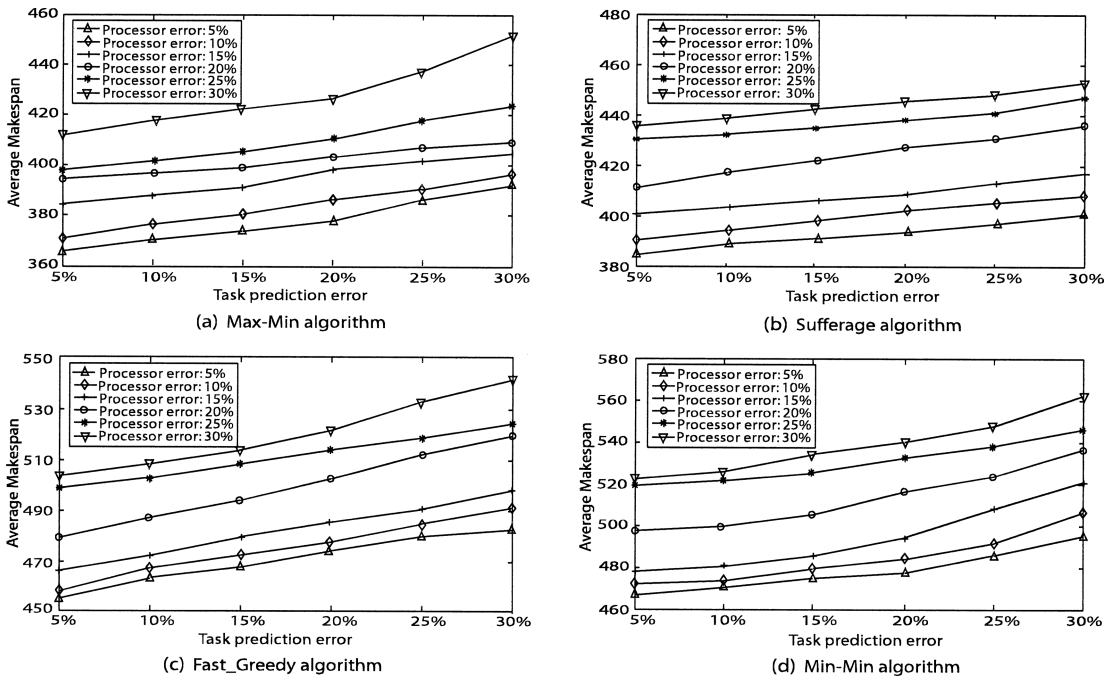

Fig. 8 Performance of different scheduling algorithms under different prediction errors for tasks and processors.

performance under accurate prediction.

In Fig. 7 it should also be noticed that in most cases, scheduling results with prediction errors are worse than without errors. However, in some cases, scheduling with prediction errors results in better performance than with accurate prediction. This is because these algorithms are only heuristics; although they can provide good scheduling performance in most cases, it can not be ensured that what they

provide is the best result every time. Therefore in some cases the scheduling result with inaccurate predictions is better than with accurate ones.

Figure 8 shows the simulation results for the scheduling algorithms with different prediction errors for tasks and processors. It is shown that for same task(or processor) prediction error, the average makespan of every algorithm increases as the increase of the prediction error of processor(tasks).

5. Conclusion

Task scheduling algorithms usually include two phases: task selection and processor selection. Prediction inaccuracy for the performances of tasks and processors usually exists, so influencing the performance of task scheduling algorithms. This paper studies the influence of inaccurate prediction on task scheduling from the perspectives of task selection and processor selection respectively. Evaluation results show that in general, underestimating performance can have a greater influence on task scheduling compared with an overestimate of performance, while higher heterogeneity in a grid computing system results in a smaller influence. Since different task scheduling algorithms have different degrees of sensitivity to the prediction error, the sensitivity of four selected scheduling algorithms to inaccurate performance prediction is also evaluated by extensive simulation. Our results are of significant benifit for task scheduling using predicted data in heterogeneous and dynamic grid environments.

References

[1] I. Foster and C. Kesselman, The Grid: Blueprint for a New Computing Infrastructure, Morgan Kaufmann Publishers, San Fransisco, CA, 1999.
[2] http://www.globus.org/
[3] http://grads.iges.org/grads/grads.html
[4] http://eu-datagrid.web.cern.ch/eu-datagrid/
[5] http://www.cs.virginia.edu/ legion/
[6] J.D. Ullman, "NP-complete scheduling problems," J. Comput. Syst. Sci., vol.10, pp.384–393, 1975.
[7] A. Abraham, R. Buyya, and B. Nath, "Nature's heuristics for scheduling jobs on computational grids," Int'l Conf. Advanced Computing and Commun., pp.45–52, 2000.
[8] C. Banino, O. Beaumont, A. Legrand, and Y. Robert, "Scheduling strategies for master-slave tasking on heterogeneous processor grids," Applied Parallel Computing: Advanced Scientific Computing: 6th Int'l Conf., pp.423–432, 2002.
[9] C. Weng and X. Lu, "Heuritics scheduling for bag-of-tasks applications in combination with QoS in the computational grid," Future Generation Computer Systems, vol.21, no.2, pp.271–280, Feb. 2005.
[10] N. Fujimoto and K. Hagihara, "Near-optimal dynamic task scheduling of precedence constrained coarse-grained tasks onto a computational grid," Int'l Symp. on Parallel and Distributed Computing, pp.80–87, Oct. 2003.
[11] W. Smith, I. Foster, and V. Taylor, "Predicting application run times with historical information," J. Parallel Distrib. Comput., vol.64, no.9, pp.1007–1016, Sept. 2004.
[12] P.A. Dinda, "Online prediction of the running time of tasks," Cluster Computing, vol.5, no.3, pp.225–236, July 2002.
[13] R. Wolski, N. Spring, and J. Hayes, "Predicting the CPU availability of time-shared Unix systems on the computational grid," Proc. IEEE Int'l Symp. on HPDC, pp.1–12, Aug. 1999.
[14] P.A. Dinda and D.R. O'Hallaron, "Host load prediction using linear models," Cluster Computing, vol.3, no.4, pp.265–280, 2000.
[15] L. Yang, I. Foster, and J.M. Schopf, "Homeostatic and tendency-based CPU load predictions," Int'l Parallel and Distributed Processing Symposium (IPDPS'03), pp.1–9, 2003.
[16] G.R. Nudd, D.J. Kerbyson, E. Papaefstathiou, S.C. Perry, J.S. Harper, and D.V. Wilcox, "PACE: A toolset for the performance prediction of parallel and distributed systems," Int'l J. High Performance Computing Applications, vol.14, no.3, pp.228–251, 2000.
[17] H. Casanova, A. Legrand, D. Zagorodnov, and F. Berman, "Heuristics for scheduling parameter sweep applications in grid environments," IEEE Proc. Ninth Heterogeneous Computing Workshop, pp.349–363, May 2000.
[18] R. Wolski, N. Spring, and J. Hayes, "The network weather service: A distributed resource performance forecasting service for metacomputing," J. Future Generation Computing Systems, vol.15, pp.757–768, 1999.
[19] T.D. Braun, H.J. Siegely, N. Becky, L.L. Boloni, M. Maheswaranx, A.I. Reuthery, J.P. Robertson, M.D. Theysy, B. Yao, D. Hensgen, and R.F. Freund, "A comparison study of static mapping heuristics for a class of meta tasks on heterogeneous computing systems," Proc. Eighth Heterogeneous Computing Workshop, pp.15–29, April 1999.

Yuanyuan Zhang received the B.S. degree in School of Mechano-Electronic Engineering, and M.S. degree in School of Computer Science and Technology from Xidian University, China, in 2000 and 2003, respectively. She is now a doctoral student in the Graduate School of Information Science, JAIST(Japan Advanced Institute of Science and Technology). Her current research interest is about resource management in grid computing.

Yasushi Inoguchi received his B.E. degree from Department of Mechanical Engineering, Tohoku University in 1991, and received MS degree and Ph.D from JAIST(Japan Advanced Institute of Science and Technology) in 1994 and 1997, respectively. He is currently an Associate Professor of Center for Information Science at JAIST. He was a research fellow of the Japan Society for the Promotion of Science from 1994 to 1997. He is also a researcher of PRESTO program of Japan Science and Technology Agency since 2002. His research interest has been mainly concerned with parallel computer architecture, interconnection networks, GRID architecture, and high performance computing on parallel machines. Dr. Inoguchi is a member of IEEE and IPS of Japan.