# The Study of Genetic Algorithm-based Task Scheduling for Cloud Computing

Sung Ho Jang, Tae Young Kim, Jae Kwon Kim and Jong Sik Lee

*School of Information Engineering*
*Inha University #253, YongHyun-Dong, Nam-Ku*
*Incheon 402-751, Republic of Korea*
*ho7809@hanmail.net, silverwild@gmail.com, jaekwonkorea@naver.com, jslee@inha.ac.kr*

***Abstract***

*Task scheduling is an important and challenging issue of Cloud computing. Existing solutions to task scheduling problems are unsuitable for Cloud computing because they only focus on a specific purpose like the minimization of execution time or workload and do not use characteristics of Cloud computing for task scheduling. A task scheduler in Cloud computing has to satisfy cloud users with the agreed QoS and improve profits of cloud providers. In order to solve task scheduling problems in Cloud computing, this paper proposes a task scheduling model based on the genetic algorithm. In the proposed model, the task scheduler calls the GA scheduling function every task scheduling cycle. This function creates a set of task schedules and evaluates the quality of each task schedule with user satisfaction and virtual machine availability. The function iterates genetic operations to make an optimal task schedule. Experimental results show effectiveness and efficiency of the genetic algorithm-based task scheduling model in comparison with existing task scheduling models, which are the round-robin task scheduling model, the load index-based task scheduling model, and the ABC based task scheduling model.*

***Keywords:*** *Task Scheduling, Genetic Algorithm, Cloud Computing*

## 1. Introduction

With the development of system virtualization and Internet technologies, Cloud computing has emerged as a new computing platform. Cloud computing is to provide virtualized IT resources as cloud services by using the Internet technology [1]. In Cloud computing, a cloud user commits to an agreement called Service Level Agreement (SLA) [2] with a cloud provider. The cloud user utilizes IT resources like storage and server as a service and pays for the service. A cloud provider constructs a computing system called cloud, which consists of several virtual machines interconnected, and makes profits by processing tasks from users on the computing system. Therefore, how to allocate tasks to virtual machines efficiently is an important and challenging issue in Cloud computing.

The main goal of Cloud computing is to satisfy cloud users with the agreed QoS and improve profits of cloud providers [3]. A task scheduling model is necessary to achieve the main goal. There are various task scheduling models [4, 5, 6] developed by a lot of researchers to utilize computing resource in distributed computing environments, but the existing models are unsuitable for Cloud computing. Because, the existing models mainly focus on improvement of system performance. Most of the task scheduling models for Cluster computing tried to minimize the completion time of a batch of tasks. Task scheduling models

for Grid computing, the forerunner of Cloud computing, aimed at improvement of specific performance metrics like computing speed and storage availability. Besides, user satisfaction and provider's profit must be considered to solve task scheduling problems in Cloud computing. In other words, a scheduler makes task schedules, which meet the QoS agreed on a SLA between cloud users and cloud providers and augment profits of cloud providers.

Rest of this paper is as follows: In Section 2, we propose the GA-based task scheduling model for Cloud computing and describes how to design elements of the GA task scheduling function. Section 3 shows the configuration of simulation and brief results. Finally, conclusion is given in Section 4.

## 2. GA-based Task Scheduling for Cloud Computing

In this section, we propose the GA-based task scheduling model and illustrate its design. We also explain how to make an optimal task schedule and compose elements of the GA scheduling function. As mentioned in the previous section, a cloud user reaches a SLA with a cloud provider to process a task. A SLA document includes user requirements like time and budgetary constraints of the task, which indicate acceptable deadline and payable budget of the cloud user. QoS attributes like response time and throughput can be comprised in a SLA document besides time and budgetary constraints [2]. A cloud provider has to consider user requirements and virtual machine information before allocating tasks from to virtual machines.

### 2.1. Encoding and Initiation

A chromosome $ch_k$ indicates the task allocation information, i.e. a task schedule. $k$ is from 1 to $z$, which denotes the number of chromosomes in a population. In consideration of user satisfaction and provider's profit, the task scheduler determines where to allocate each task every scheduling cycle. A chromosome $ch_k$ consists of $\alpha[i]$ and $\beta[i]$, which indicate the information of task processing and virtual machine allocation. The length of a chromosome is the same as the number of inputted tasks.

The encoding operation to express a task schedule as a chromosome is as follows. A set of inputted tasks is sorted before the creation of chromosomes. Unlike existing distributed computing environments, cloud users pay for computing services in person. Accordingly, tasks received from cloud users with high cost have to be allocated to virtual machines faster than others. In this paper, we classify takes into four groups by time and budgetary constraints as shown in Figure 1.

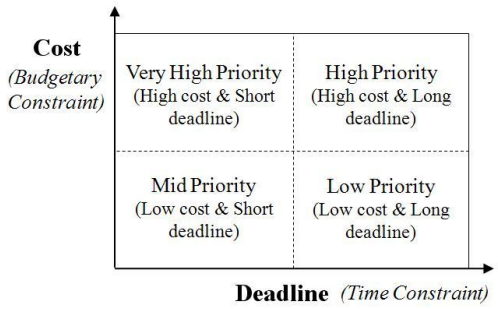


**Figure 1. Classification of Tasks by Time and Budgetary Constraints**

### 2.2. Fitness Function and Selection

The fitness function generates a fitness value of each chromosome. The value indicates how suitable for solving a task scheduling problem a chromosome is. The fitness function for general task scheduling problems is based on execution time of tasks. But, task scheduling problems in Cloud computing are different from general task scheduling problems because computing services in Cloud computing are offered through a SLA between cloud users and providers. The minimization of execution time is a goal of task scheduling in Cloud computing, but the main goal is to improve user satisfaction and increase provider's profit. Therefore, the fitness function in our genetic algorithm is defined by these elements.

The fitness function generates a fitness value of each chromosome. The value indicates how suitable for solving a task scheduling problem a chromosome is. The fitness function for general task scheduling problems is based on execution time of tasks. But, task scheduling problems in Cloud computing are different from general task scheduling problems because computing services in Cloud computing are offered through a SLA between cloud users and providers. The minimization of execution time is a goal of task scheduling in Cloud computing, but the main goal is to improve user satisfaction and increase provider's profit. Therefore, the fitness function in our genetic algorithm is defined by these elements.

User satisfaction can be guaranteed with the QoS designated in a SLA document. In this paper, response time and processing cost, which are typical QoS attributes in a SLA [2], are used to define user satisfaction. In our scheduling algorithm, the roulette wheel selection [11], a stochastic selection method, is used to create the basis of the next generation. We assume that there is a roulette wheel, size of which is the sum of fitness values in a population.

### 2.3. Crossover and Mutation

The crossover operation is a simple mechanism to swap one part of a chromosome for that of another chromosome. In this paper, the two-point crossover [12] is used for transferring genetic material of parent to children.

The mutation operation is to expand the search space by changing one part of a chromosome. In the beginning of the genetic algorithm, the quality of generated chromosomes is not particularly good. As time goes by, the quality of chromosomes becomes more improved. There is the potential for improvement in quality by the mutation operation in the early part of the genetic algorithm. But, it is hard to improve the quality of chromosomes by the mutation operation after the quality attains a certain standard. In this case, the mutation operation increases execution time of the genetic algorithm. We therefore the non-uniform mutation [13], which gradually decreases the mutation rate by the number of reproductions.

### 2.4. Restart & Stop Condition

Figure 2 shows the restart operation. If the best fitness value of the current population does not reach the minimum fitness threshold, chromosomes with high fitness values in the current population fill the half of the next population. And, the rest of the next population is filled with chromosomes generated randomly. This restart operation helps the GA scheduling function to prevent local optimum and explore various search places.

```
Restart () //Restart operation
Input: a population po, a set of n tasks T, and a set of m
processors P
Output: new population po'
{
1. find the chromosome b_ch with the best fitness value in po;
2. if (b_ch.fitness < f_t ) { // f_t is the minimum fitness threshold
3.   sort po by fitness value in descending order;
4.   for (i=1; i≤|po|/2; i++) {
5.     copy po[i] to po'[i]; }
6.   for (j=(|po|/2)+1; j≤|po|; j++) {
7.     po'[j] = encoding(T,P); }
   }
8. else {
     Selection(po); }
}
```

**Figure 2. Pseudocode of Restart Operation**

And, the stop condition of the GA scheduling function is as Eq. 1 where $bf_c$ is the best fitness value in the current population, $bf_p$ is the best fitness value in the past population, $\theta$ is a boundary variation, and $f_t$ is the minimum fitness threshold.

$$\left|bf_c - bf_p\right| \le \theta, \qquad (f_t \le bf_c) \tag{1}$$

If, the above stop condition is met, the GA scheduling function stops and outputs a chromosome with the best fitness value, i.e., the best task schedule. Otherwise, the GA scheduling function continues restarting or reproducing a population.

## 3. Simulation Results

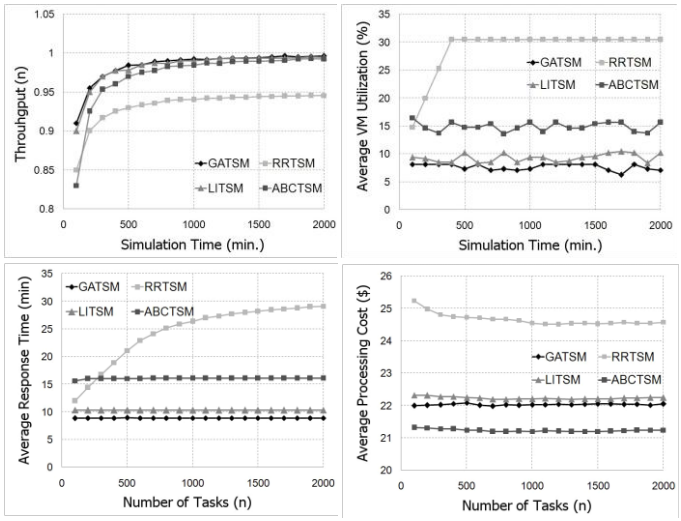


**Figure 3. Simulation Results**

In this section, we simulated the GA-based Task Scheduling Model (GATSM) on the discrete event system modeling and simulation environment [14] and conducted various experiments to compare performances of GATSM with those of the Round Robin Task

Scheduling Model (RRTSM) [7], the Load Index-based Task Scheduling Model (LITSM) [8], and the Activity Based Costing-based Task Scheduling Model (ABCTSM) [9]. The RRTSM allocates takes to virtual machines in sequence regardless of information on tasks or virtual machines. The LITSM uses the load index of virtual machines for task scheduling. In the LITSM, a scheduler allocates tasks to lightly loaded virtual machines. And, the ABCTSM uses the cost required to process tasks on each virtual machine and allocates tasks to virtual machines as per the cost. Performance comparisons are based on throughput, response time, virtual machine utilization, processing cost, and user satisfaction. In experiments, 12 virtual machines with different computing abilities were simulated. A total of 2000 tasks were generated from cloud users each simulation. A cloud user generates a task with requirements including time and budgetary constraints. We classified tasks into four types as shown in Figure 1. And Figure 3 shows brief results of simulation.

## 4. Conclusion

In this paper, we presented scheduling problems in Cloud computing and proposed a task scheduling model to solve the scheduling problems. In the proposed task scheduling model, the task scheduler calls the GA scheduling function to make task schedules based on information of tasks and virtual machines. The GA scheduling function creates a population, a set of task schedules, and evaluates the population by using the fitness function considering user satisfaction and virtual machine availability. The function iterates reproducing populations to output the best task schedule. The restart operation is also applied to the GA scheduling function for the improvement in quality of task schedules. For performance evaluation, we simulated the proposed task scheduling model and conducted diverse experiments. Empirical results prove that the proposed task scheduling model outperforms existing task scheduling models, which are the round-robin task scheduling model [7], the load index-based task scheduling model [8], and the activity based costing based task scheduling model [9].

## Acknowledgements

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (2012002751).

## References

[1] B. Hayes, “Cloud computing”, Communications of the ACM, vol. 51, no. 7, (2008), pp. 9-11.
[2] P. Patel, A. Ranabahu and A. Sheth, “Service level agreement in cloud computing”, http://knoesis.wright.edu/library/download/OOPSLA_cloud_wsla_v3.pdf, **(2009)**.
[3] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg and I. Brandic, "Cloud computing and emerging IT platforms: vision, hype, and reality for delivering computing as the 5th utility", Future Generation Computer Systems, vol. 25, issue 6, (2009), pp. 599-616.
[4] W. Sun, Y. Zhang and Y. Inoguchi, "Dynamic task flow scheduling for heterogeneous distributed computing: algorithm and strategy", IEICE Trans. on Inf. & Sys., vol. E90-D, no. 4, **(2007)**, pp. 736-744.
[5] I. K. Savvas and M. T. Kechadi, "Dynamic task scheduling in computing cluster environments", Proc. 3rd Int'l. Workshop on Parallel and Distributed Computing, **(2004)**, pp. 372-379.
[6] R. Moreno, "Job scheduling and resource management techniques in dynamic grid environments", Lecture Notes in Computer Science, vol. 2970, **(2004)**, pp. 25-32.
[7] K. Li, "Job scheduling and processor allocation for grid computing on metacomputers", Journal of Parallel and Distributed Computing, vol. 65, no. 11, **(2005)**, pp. 1406-1418.

[8] G. Kakarontzas and I. K. Savvas, "Agent-based resource discovery and selection for dynamic grids", Proc. 15th IEEE Int'l Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, **(2006)**, pp. 195-200.
[9] Q. Cao, Z. B. Wei and W. M. Gong, "An optimized algorithm for task scheduling based on activity based costing in cloud computing", Proc. 3rd Int'l Conf. on Bioinformatics and Biomedical Engineering, (2009), pp. 1-3.
[10] E. S. H. Hou, R. Hong and N. Ansari, "Efficient multi processor scheduling based on genetic algorithms", Proc. IEEE Conf. on Industrial Electronics Society, vol. 2, **(1990)**, pp. 1239-1243.
[11] G. Syswerda, "Uniform crossover in genetic algorithms", Proc. 3rd Int'l Conf. on Genetic Algorithms, **(1989)**, pp. 2-9.
[12] A. Neubauer, "Adaptive non-uniform mutation for genetic algorithms", Lecture Notes in Computer Science, vol. 1226, **(1997)**, pp. 24-34.
[13] B. P. Zeigler, H. S. Sarjoughian, S. W. Park, J. S. Lee, Y. K. Cho and J. J. Nutaro, "DEVS modeling and simulation: a new layer of middleware", Proc. of 3rd Annual Int'l Workshop on Active Middleware Services, **(2001)**, pp. 22-31.

## Authors

**Sung Ho Jang**

He received the Ph.D degree at Information Engineering from Inha University in 2011. His research interests include Cloud computing and Ubiquitous computing

**Tae Young Kim**

He received the MS degree at Computer Science and Information Engineering from Inha University in 2009. His research interests include Cloud computing, software modeling & simulation.

**Jae Kwon Kim**

He received the BS degree at Computer Science and Information Engineering from Gachon University Medical and Science in 2011. His research interests include Cloud computing and Ubiquitous computing

**Jong Sik Lee**

He is a professor in the School of Computer Science and Information Technology. He received the Ph.D degree at Electronics and Computer Engineering from University of Arizona in 2001. Current research interests include Cloud computing, software modeling and simulation.