Journal of information and
communication convergence engineering
http://jicce.org

J. Inf. Commun. Converg. Eng. 11(4): 235-240, Dec. 2013 Regular paper

A Two-Step Job Scheduling Algorithm Based on Priority for Cloud Computing

Jeongwon Kim[*], ***Member*, *KIICE***

Department of Computer Engineering, Silla University, Busan 617-736, Korea

Abstract

Cloud systems are popular computing environment because they can provide easy access to computing resources for users as well as efficient use of resources for companies. The resources of cloud computing are heterogeneous and jobs have various characteristics. One such issue is effective job scheduling. Scheduling in the cloud system may be defined as a multiple criteria decision model. To address this issue, this paper proposes a priority-based two-step job scheduling algorithm. On the first level, jobs are classified based on preference. Resources are dedicated to a job if a deadline failure would cause severe results or critical business losses. In case of only minor discomfort or slight functional impairment, the job is scheduled using a best effort approach. On the second level, jobs are allocated to adequate resources through their priorities that are calculated by the analytic hierarchic process model. We then analyze the proposed algorithm and make a scheduling example to confirm its efficiency.

Index Terms: Analytic hierarchic process, Cloud computing, Job Scheduling, Priority

I. INTRODUCTION

Under a cloud computing environment, which is Internet-based computing, users can utilize resources such as computing software, hardware performance monitoring, and information systems for public utilities like gas, electricity, and water. As a result, because of the cloud system's centralized storage, memory processing, hardware, and bandwidth, this system can provide an efficient and low-cost computing environment [1].

On the other hand, handling jobs efficiently in cloud services is a problem that remains to be solved. In general, the job scheduling problem has been a major research topic of grid and cloud computing. Static or dynamic algorithms have been proposed in a single workflow in order to achieve optimum performance. The schemes are mainly focused on scheduling completion time guarantees and latency reduction techniques.

These techniques are mainly best-effort scheduling algorithms. However, the scheduling in cloud computing algorithms has different requirements because of the importance of how quickly a certain level of resources can be allocated and how much of the resource is dedicated to a job [2]. In addition, as cloud computing users have a variety of levels of the use of services, a wide range of scalability, and this is dynamically set through virtualization, jobs should be scheduled in different ways.

A scheduling algorithm in the cloud environment must take into special consideration that the storage and data transfer costs of data-oriented jobs continue to grow exponentially as time passes. Therefore, this study proposes a new scheduling algorithm that takes into consideration the

Received 07 March 2013, **Revised** 30 June 2013, **Accepted** 17 July 2013
***Corresponding Author** Jeongwon Kim (E-mail: jwkim@silla.ac.kr, Tel: +82-51-999-5749)
Department of Computer Engineering, Silla University, 140 Baegyang-daero 700beon-gil, Sasang-gu, Busan 617-736, Korea.

Open Access **http://dx.doi.org/10.6109/jicce.2013.11.4.235** print ISSN: 2234-8255 online ISSN: 2234-8883

characteristics of each job.

The proposed scheme is divided into two steps: the first step is to classify jobs by their degree of importance, and the second step is to assign resources to each job. In the first step, if missing a deadline may bring about severe results or critical business losses, resources are dedicated to that job (resource provisioning). If a deadline failure would cause only minor discomfort or slight functional impairment, the job is scheduled using a best effort approach (best effort). In the second step, jobs are assigned to a virtual machine (VM) to catch the user's quality of service (QoS) and maximize the efficiency of cloud computing.

In a cloud environment, many nodes exist, and the reliabilities of all heterogeneous nodes are inevitably low. This allocation issue can be defined as a kind of multi-criteria decision making (MCDM) because responsiveness, cost, and loads of jobs have variability in terms of QoS. Therefore, our work uses the analytic hierarchic process (AHP) method to solve this allocation issue and our scheduling algorithm chooses a VM to run the job by determining the priorities of the alternatives for the various decision criteria.

This paper is organized as follows: The research related to the proposed algorithm is described in Section II. In Section III, our new algorithm is introduced in detail. In Section IV, the effectiveness of our algorithm is analyzed and verified. We then discuss conclusions and future work in Section V.

II. RELATED WORKS

Many studies have addressed job scheduling in grid and cloud environments. There are online and offline schemes in the case of batch job scheduling which are known to be suitable for scientific applications. This paper is focused on the scheduling of real-time requests.

Yu et al. [3] proposed a cost-based workflow scheduling algorithm considering the minimization of the execution cost in which it satisfies the deadline. Yu and Buyya [4] proposed a scheduling technique to minimize the execution time while satisfying workflow execution cost. In this scheduling approach, a genetic algorithm was applied to solve the optimization problem and experimental results were presented for a grid environment. Padala et al. [5] proposed an algorithm that satisfies the QoS of workflows and can improve resource utilization between applications by adjusting resource sharing. Yu and Shi [6] proposed a plan-based algorithm for multiple workflows that determines the execution order with rankings of workflows. Xu et al. [7] proposed a scheduling algorithm to support multiple workflows and nested multiple QoS requirements. They further showed that the scheme can improve scheduling accessibility. Kosinska et al. [8] proposed a variety of phases to improve the reliability and scalability of the applications that run on cloud resources. Ghanbaria and Othman [9] reported on a technique similar to this study. Their scheme used the AHP technique to determine a job's priority, the attribute level of cloud resources, and the alternative to a job. On the other hand, our study obeys the basic principle of the AHP model. In our custom model, the attribute level defines decision criteria. This level determines which criteria, such as responsiveness, cost, and system load, are preferable for choosing alternatives. The alternative level is defined as a resource allocation unit. This unit is meant as a VM in our scheme to determine which VM is more adequate for executing a job.

III. TWO-STEP JOB SCHEDULING SCHEME BASED ON PRIORITY

The proposed scheduling scheme in this paper is composed of two steps. In the first step, the priority of a job is calculated based on the degree of importance of each job. In the second step, the algorithm selects a VM on which to run the job. Every candidate VM is weighted using the AHP model.

A. First Step: Job Classification Based on Its Degree of Importance

As all resources in cloud computing are dedicated to jobs, it is advantageous for a cloud system to secure the resources needed in advance. In the case of data-oriented jobs, this resource provisioning will be an especially important factor in performance improvement because the required storage as well as bandwidth increases with time passing. Thus, our scheme classifies all jobs into one of two categories: resource provisioning or best effort.

In this step, the jobs for which the user pays a high cost or that are of higher importance belong to the resource provisioning case. If scheduling failures of jobs may bring about severe business loss, the jobs are classified to the first level of priority or the highest level. If the failures may cause considerable after-effects, these jobs belong to the second level of priority. Jobs for which the user pays regular or low costs or are of medium importance belong to the best effort case. If scheduling failures of jobs may bring about a trivial inconvenience to a small number of users, the jobs are classified into the third level of priority. If the failures may cause minor functional impairments, these jobs belong to the fourth level or the lowest priority. The classified jobs are queued in order in the cloud system scheduler. The sequences are fixed, and the jobs are executed in a non-preemptive way. The jobs with the same priority are served in a round-robin manner.

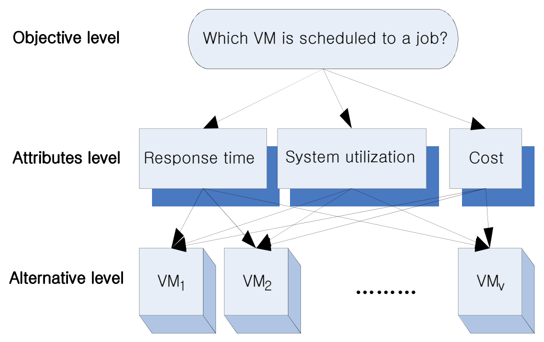

Fig. 1. The analytic hierarchic process model to place jobs on the virtual machine (VM).

B. Second Step: Job Allocation to VM Based on the AHP Model

The AHP is a kind of MCDM that helps to choose one of the alternatives. Each selection has a related attribute attached to it, and the weights of each attribute are set. Therefore, the AHP model can select the best choice out of the list of alternatives. The merit of the AHP is that it considers variable parameters for many alternatives and generates the result that best matches the parameters [10]. The proposed scheme uses an AHP model for the VM selection.

As Fig. 1 shows, the objective level in the proposed algorithm is to place a job in a VM that it best matches under many different parameters. The attribute level in our algorithm is composed of user requirements or decision criteria such as response time, system utilization, and cost, and the alternative level represents all VMs in a cloud system.

As each parameter in the AHP model has a preference, the proposed scheme also has a preference from 1 to 9. The preference 9 is the highest one. The values 2, 4, 6, and 8 are intermediate values for the preference, and the inverse number represents the counterpart preference. Therefore, these preference values are used in representing and calculating the requirements of the job as well as parameters of each VM, such as the response time, system utilization, and cost.

Suppose that a set of jobs, which are scheduling objects in cloud environment, is $\zeta = \{J_1, J_2, ..., J_m\}$, and the criteria set is $\psi = \{C_1, C_2, ..., C_n\}$, and set of VMs is $\xi = \{VM_1, VM_2, ..., VM_v\}$.

The VM allocation is done in two steps. In the first step, the algorithm calculates the weight vector by pairwise comparison and does a consistency check for both of the levels. In the second step, the best VM is selected by multiplying the two weight vectors of each level. The process is explained in detail below.

Sub-step 1: Calculation of the weight matrix and consistency index between the objective and attribute level.

$$PCM^{i,j} = \left\{\begin{matrix} \frac{1}{PCM^{j,i}}, & i \neq j \\ 1, & i = j \end{matrix}\right\} \gg$$

$$\begin{bmatrix} \frac{w_1}{w_1} & \cdots & \frac{w_1}{w_n} \\ \vdots & \ddots & \vdots \\ \frac{w_n}{w_1} & \cdots & \frac{w_n}{w_n} \end{bmatrix} \begin{bmatrix} w_1 \\ \vdots \\ w_n \end{bmatrix} = \lambda_{max} \begin{bmatrix} w_1 \\ \vdots \\ w_n \end{bmatrix}. \tag{1}$$

Suppose that the pairwise comparison matrix between the objective level and attribute level is named "pairwise comparison matrix (PCM)" in this paper. This PCM represents the preference for a job under all criteria such as responsiveness, system utilization, and cost. The PCM is an n by n matrix like Eq. (1). If the element of PCM, (i,j) is 5, the preference of PCM(j,i) will be 1/5. There are n pairwise comparison matrices for all criteria, which are created according to the priority of the decision criteria. For each of the comparison matrices, the scheme should compute a priority vector (vector of weights). The priority vector can be obtained by solving Eq. (2). The λ_{max} is the principal eigenvalue of PCM and is denoted by the corresponding eigenvector $\omega^{criteria}$. With any arbitrary comparison of PCM, the model can produce a vector of weights such as $\omega^{criteria} = \{\omega_1, \omega_2, .., \omega_n\}$. An essential step in this model is to obtain a vector of weights. The vector of weights can be computed through Eq. (2). A positive n by n matrix has the ratio form PCM = (w_i/w_j), $i,j = 1, ..., n$, if, and only if, it is consistent. The matrix of ratios PCM = (w_i/w_j) is consistent if and only if n is its principal eigenvalue and PCM$\cdot\omega^{criteria} = \lambda_{max}\cdot\omega^{criteria}$. Further, $\omega^{criteria} > 0$ is unique to within a multiplicative constant.

$$\text{PCM} \times \omega^{criteria} = \lambda_{max} \times \omega^{criteria}. \tag{2}$$

Saaty [10] has defined the consistency ratio (CR) as Eq. (3).

$$CR = \frac{CI}{RI}, where\ CI = \frac{\lambda_{max} - n}{n}. \tag{3}$$

In Eq. (3), RI is the random index, which is randomly calculated based on the rank of the comparison matrix. Eq. (3) also uses the RI values, which were calculated by Saaty [10]. If CR < 0.1, then the PCM should be considered consistent.

Sub-step 2: Calculation of the weight matrix and consistency index between the attribute and alternative level.

The next step is also to calculate the weight matrix and consistency index for each decision criterion to all VMs. Like Eqs. (1)–(3), this scheme determines the pairwise comparison matrix and calculates the weight matrix and consistency ratios for each of the decision criteria to each VM.

Suppose that the weight vector for each VM is $\omega^{vm}=\{\omega_1,\omega_2,\dots,\omega_v\}$, and ω^{vm} is a v by n matrix. The v is the number of the VMs and the n is the number of the decision criteria. We then obtain the final score for each VM by multiplying the weight vector of each sub-step 1 by the weight matrix of the sub-step 2. The number of elements of the score vector in Eq. (4) is v. The index of maximum value means a VM that will execute the job.

$$\text{score} = \omega^{vm} \times \omega^{criteria}, \tag{4}$$

where $\{\omega^{vm}\ is\ v\ by\ n, \omega^{criteria}\ is\ n\ by\ 1\}$.

IV. ANALYSIS AND EXAMPLE OF THE PROPOSED SCHEDULING ALGORITHM

First, we analyze the proposed scheme by time complexity. As the priority classification of the first step is determined by job characteristics or requirements, the time complexity may be trivial. The computation of the pairwise comparison matrix and CR occupies a huge portion of complexity in the proposed scheme, such as Eq. (5).

$$O = n^{2.38} + n \times v^{2.38} + d \times n^{2.38} \ . \tag{5}$$

In Eq. (6), n is the number of the decision criteria and v is the number of the VMs, and d denotes the number of the eliminated matrices because of inconsistency while checking CR. In Eq. (6), $n^{2.38}$ are the additions and the multiplications in the calculation of the weight vector from the objective level to the attribute level. The $n \times v^{2.38}$ is the arithmetic operations in the computation of the weight matrix from the attributes level to the alternative level. Furthermore, if the candidate pairwise matrix is rejected in the consistency check, the matrix must be recalculated in $d \times n^{2.38}$ times. Thus, the final time complexity may be determined by Eq. (6).

$$O = [\max(v,n)]^{2.38} * [\, v > n\,?\,n{:}\,d]. \tag{6}$$

The time complexity for multiplying n by n matrices requires $O(n^3)$ multiplications that are of worst case complexity. The fastest known algorithm, devised by Don Coppersmith and Shmuel Winograd, runs in $O(n^{2.38})$ time [11]. Most researchers believe that an optimal algorithm will run in essentially $O(n^2)$ time, yet until recently, no further progress has been made in finding one [12]. Therefore, the best case complexity is $O(n^{2.38})$, and the worst case complexity of the proposed scheme is $O(n^3)$, and a general case of time complexity is $n^{2.38}$ or $v^{2.38}$. However, as the number of VMs is generally greater than the decision criteria, the complexity of the proposed algorithm should be described as $n \times v^{2.38}$.

The following is an example of the proposed scheduling algorithm. Table 1 is a sample preference matrix of a job.

The following Eq. (7) is a PCM and weight vector.

$$\text{PCM} = \begin{bmatrix} 0.222 & 0.250 & 0.217 \\ 0.111 & 0.125 & 0.130 \\ 0.667 & 0.625 & 0.652 \end{bmatrix},$$

$$weight\ vector = \begin{bmatrix} 0.230 \\ 0.122 \\ 0.648 \end{bmatrix}. \tag{7}$$

For obtaining the CR, we first multiply the preference matrix by the eigenvalue of the weight vector as shown below Eq. (8):

$$0.230 \times \begin{bmatrix} 1 \\ 0.5 \\ 3 \end{bmatrix} + 0.122 \times \begin{bmatrix} 2 \\ 1 \\ 5 \end{bmatrix} + 0.648 \times \begin{bmatrix} 0.333 \\ 0.2 \\ 1 \end{bmatrix}$$
$$= \begin{bmatrix} 0.230 & 0.244 & 0.216 \\ 0.115 & 0.122 & 0.130 \\ 0.690 & 0.611 & 0.648 \end{bmatrix}. \tag{8}$$

From Eqs. (2) and (3), we now can get the $\lambda_{max} = 3.004$ and CR is 0.002. As the CR is smaller than 0.1, this preference matrix must be consistent.

Table 1. Preference matrix example of a job

	Response time	System utilization	Cost
Response time	1	2	1/3
System utilization	1/2	1	1/5
Cost	3	5	1

Table 2. λ_{max} and consistency ratio (CR) for response time

	VM₁	*VM₂*	*VM₃*	*VM₄*	*VM₅*	*Weight*
VM_1	1	2	1/3	4	1/2	0.166
VM_2	1/2	1	1/5	2	1/5	0.086
VM_3	3	5	1	6	2	0.433
VM_4	1/5	1/2	1/6	1	1/3	0.059
VM_5	2	4	1/2	3	1	0.255
λ_{max} = 5.122, CR = 0.022 (<0.1)						

Table 3. λ_{max} and consistency ratio (CR) for system utilization

	VM₁	*VM₂*	*VM₃*	*VM₄*	*VM₅*	*Weight*
VM_1	1	3	1	2	1/3	0.214
VM_2	1/3	1	1/2	2	1/2	0.129
VM_3	1	2	1	3	1	0.251
VM_4	1/2	0.5	1/3	1	1/3	0.086
VM_5	3	2	1	3	1	0.320
λ_{max} = 5.236, CR = 0.042 (<0.1)						

Table 4. λ_{max} and consistency ratio (CR) for cost

	VM₁	*VM₂*	*VM₃*	*VM₄*	*VM₅*	*Weight*
VM_1	1	2	1/4	4	1/2	0.174
VM_2	1/2	1	1/3	2	1/3	0.108
VM_3	4	3	1	3	2	0.391
VM_4	1/4	0.5	1/3	1	1/3	0.076
VM_5	2	3	1/2	3	1	0.250
λ_{max} = 5.281, CR = 0.050 (<0.1)						

Finally, the score of each VM is calculated by multiplying the weight matrix (ω^{vm}) in the alternative level by the weight vector ($\omega^{criteria}$) in the attribute level through Eqs. (4)–(9), which follows, is the score vector for each VM. Because VM_3 recorded the maximum score (0.375), the proposed scheduler should select VM3 to execute the job.

$$\text{score} = \omega^{vm} \times \omega^{criteria}$$
$$= \begin{bmatrix} 0.166 & 0.214 & 0.174 \\ 0.086 & 0.129 & 0.108 \\ 0.433 & 0.251 & 0.391 \\ 0.059 & 0.086 & 0.076 \\ 0.255 & 0.320 & 0.250 \end{bmatrix} \times \begin{bmatrix} 0.268 \\ 0.195 \\ 0.537 \end{bmatrix} = \begin{bmatrix} 0.180 \\ 0.106 \\ 0.375 \\ 0.073 \\ 0.265 \end{bmatrix}. \quad (9)$$

Next, we calculate the weight vector and CR of each VM for the response time, system utilization, and cost. Tables 2–4 show these weight vectors and CRs. As all CRs are less than 0.1, all preference matrices must be consistent.

V. CONCLUSION

Job scheduling is an important problem in cloud computing, which is naturally composed of heterogeneous resources. We defined this issue as MCDM. To settle this issue, this paper introduced a two-step job scheduling scheme based on priority, with consideration of both the job characteristics and the MCDM. In the first step, the priority of a job was classified in 1 of 4 levels, which are determined by importance attributes. In the second step, we applied the AHP process in job allocations to the VM to tackle the MCDM issue. We adopted responsibility, system utilization, and cost of jobs as decision criteria. We then analyzed the time complexity of the proposed scheme and confirmed that the proposed algorithm showed acceptable complexity. In the future, work will be carried out aiming to minimize the complexity and to implement a real scheduler in a sample cloud system.

REFERENCES

[1] R. Baraglia, G. Capannini, P. Dazzi, and G. Pagano, "A multi-criteria job scheduling framework for large computing farms," *Journal of Computer and System Sciences*, vol. 79, no. 2, pp. 230-244, 2013.

[2] E. Deelman, "Grids and clouds: making workflow applications work in heterogeneous distributed environments," *International Journal of High Performance Computing Applications*, vol. 24, no. 3, pp. 284-298, 2010.

[3] J. Yu, R. Buyya, and C. K. Tham, "Cost-based scheduling of scientific workflow applications on utility grids," in *Proceedings of the 1st IEEE International Conference on e-Science and Grid Computing*, Melbourne, Australia, pp. 5-8, 2005.

[4] J. Yu and R. Buyya, "A budget constrained scheduling of workflow applications on utility grids using genetic algorithms," in *Proceedings of the 15th IEEE International Symposium on High Performance Distributed Computing*, Paris, France, pp. 19-23, 2006.

[5] P. Padala, K. G. Shin, X. Zhu, M. Uysal, Z. Wang, S. Singhal, A. Merchant, and K. Salem, "Adaptive control of virtualized resources in utility computing environments," in *Proceedings of the 2nd ACM SIGOPS/EuroSys European Conference on Computer Systems*, Lisbon, Portugal, pp. 289-302, 2007.

[6] Z. Yu and W. Shi, "A planner-guided scheduling strategy for multiple workflow applications," in *Proceedings of the 37th International Conference on Parallel Processing Workshops*, Portland, OR, pp. 1-8, 2008.

[7] M. Xu, L. Cui, H. Wang, and Y. Bi, "A multiple QoS constrained scheduling strategy of multiple workflows for cloud computing," in *Proceedings of IEEE International Symposium on Parallel and Distributed Processing with Applications*, Chengdu, China, pp. 629-634, 2009.

[8] J. Kosinska, J. Kosinski, and K. Zielinski, "The concept of application clustering in cloud computing environments: the need for extending the capabilities of virtual networks," in *Proceedings of the 5th International Multi-Conference on Computing in the Global Information Technology*, Valencia, Spain, pp. 139-145, 2010.

[9] S. Ghanbaria and M. Othman, "A priority based job scheduling algorithm in cloud computing," in *Proceedings of the International Conference on Advances Science and Contemporary Engineering 2012*, Jakarta, Indonesia, pp. 778-785, 2012.

[10] T. L. Saaty, *Decision Making for Leaders: The Analytical Hierarchy Process for Decisions in a Complex World.* Pittsburgh,

PA: RWS Publications, 2012.
[11] V. Strassen, “Gaussian elimination is not optimal,” *Numerische Mathematik*, vol. 13, no. 4, pp. 354-356, 1969.
[12] S. Robinson, “Toward an optimal algorithm for matrix multiplication,” *SIAM News*, vol. 38, no. 9, 2005.

Jeongwon Kim
studied computer science at Pusan National University in Pusan, Korea and earned a doctoral degree from the university in 2000. He is an associate professor in the Computer Engineering Department at Silla University in South Korea. Prior to joining Silla University, he worked at the Korea Technology Finance Corporation. His current interests include embedded systems, ubiquitous health care, cloud computing, and pervasive computing.