Jestr

Journal of Engineering Science and Technology Review 11 (5) (2018) 161 - 169

JOURNAL OF
Engineering Science and Technology Review

Research Article

www.jestr.org

Task Scheduling Algorithm Based on Virtual Machine Availability Awareness in Cloud Platform

Zhixin Li[1,2,*], Lei Liu[1] and Zeyu Tong[3]

[1]*College of Computer Science and Technology, Jilin University, Changchun 130012, China*
[2]*School of Computer Technology and Engineering, Changchun Institute Of Technology, Changchun 130012, China*
[3]*Department of Applied Mathematics and Statistics, Johns Hopkins University, Baltimore, 21218, United States*

Received 29 April 2018; Accepted 11 November 2018

Abstract

Virtual machine (VM) availability awareness is a key task scheduling technology in a cloud platform. However, the dynamic change and uncertainty of VM availability constitute difficulties for task scheduling, and quality requirements of task services cannot be satisfied and thus seriously affect the task scheduling capacity of the cloud platform. A task scheduling algorithm based on VM availability awareness was proposed to solve the unmatching problem between VM availability and quality of service (QoS) to improve task scheduling capacity of VMs. This algorithm combined available task processing capacities of VMs and task requirement features to establish a differential entropy model between VM availability and task availability requirement. Task availability matching and scheduling was realized through the principle of maximum entropy, and task scheduling was optimized from the aspect of balance of server workloads. Finally, a comparative verification between the task scheduling algorithm and Random and Minmin algorithms was implemented. Results demonstrate that through the task scheduling algorithm based on VM availability awareness, the task execution speed of VMs is higher than those of Random and Minmin algorithms by 28% and 6%, respectively. Therefore, task execution speed of the cloud platform is significantly elevated, and server workloads are more balanced than those in Random and Minmin algorithms. Within the same time, QoS satisfaction rate is higher than those of Random and Minmin algorithms by 10% and 2%, respectively. QoS task requirement is satisfied while task completion time is reduced. This study concludes that the VM availability awareness method satisfies the task requirements and improves task processing performance of the VM in the cloud platform. Relevant conclusions can provide technical support for task scheduling of the VM in the cloud platform.

Keywords: task scheduling, availability, cloud platform, cloud computing

1. Introduction

With development and application of cloud computing, increasingly more computing tasks have operated on virtual machine (VM) resources in the cloud platform [1]. The availability of VM resources directly affects the quality of service (QoS) [2,3]. VM availability awareness, an important task scheduling algorithm [4], is an important means of improving task processing performance of the VM in the cloud platform and satisfying QoS requirement. The objective of VM availability awareness is to calculate whether availability of VM resources can satisfy task QoS requirement. Based on this evaluation, task scheduling will be executed to improve performances of application programs operating in the VM system. However, owing to the dynamics and complexity of availability of VM resources in the cloud platform, the traditional evaluation of computer system availability and resource workload cannot satisfy the dynamic environment of the cloud platform. Therefore, how to combine availability evaluation of VM resources and dynamic server workloads is a problem that needs urgent solution in the scheduling process.

Task scheduling algorithm based on VM availability awareness in the cloud platform emerged under the circumstances. In terms of the development of the present task scheduling algorithms based on VM availability awareness, VMs meeting conditions were filtered according to requirements of computing tasks for availability of VM resources for task scheduling, but the matching problem between availability evaluation of VM resources and task QoS requirement as well as workload balancing problem between server resources in the task scheduling process has not been handled very well. Specific to the matching problem between availability evaluation of traditional VM resources and task QoS requirement, availability modeling of computing power of each server node was implemented through task execution time and availability requirement analysis, task response time was shortened, and requirement for task scheduling services was satisfied [5]. However, availability modeling in this method could not adapt to availability evaluation of VMs in the cloud platform. In addition, task scheduling method gave an availability quantitative criterion under cloud computing environment to realize task availability matching according to availability requirement of computing tasks and availability of computing resources [6]. Its deficiency was that it did not consider server workload balancing problem, which affected task execution speed. For server workload balancing

*E-mail address: 52868081@qq.com
ISSN: 1791-2377
doi:10.25103/jestr.115.20

problem in task scheduling, availability factor was calculated for each available resource through the relationship between availability and workload balancing, scheduling requests were processed within the minimum span time, and a certain effect was generated on workload balancing [7]. Despite evaluating the availability of cloud VMs, this method lacked matching with task QoS requirement. These methods can neither guarantee server workload balancing nor match VM availability with task QoS requirement.

Therefore, a task scheduling algorithm was established in this study through the relationship between availability of VM resources and task QoS requirement, and server workload balancing was ensured using a server workload balancing strategy. This solved the availability matching problem of VM resources and server workload balancing problem to some degree and effectively improved task execution efficiency.

2. State of the art

Regarding task scheduling matching problem caused by dynamic availability change among task scheduling algorithms in VMs, domestic and foreign theoretical and practical circles carried out many studies on VM availability modeling in the cloud platform and task scheduling under availability constraints [8]; the general belief is that VM availability awareness not only can guarantee availability requirement of tasks for VM resources but also can elevate task processing speed. Now these algorithms include heuristic scheduling algorithms such as ant colony algorithm [9], genetic algorithm [10], and classical algorithms such as Minmin [11]; these algorithms have elevated task processing speed, but they have scarcely involved availability evaluation of VM resources and cannot satisfy task QoS requirement. In the aspect of availability evaluation of resources in the cloud platform, Kim D. S. et al. [12] utilized a stratification method of the two-stage virtualized system model, where fault tree was used at the upper layer, and uniform continuous time Markov chain was used at the lower layer to analyze availability of the virtualization system. This method realized and verified a virtualized system availability model. Ghosh R. et al. [13] conducted an extended analysis of availability of infrastructure cloud, realized extensible and random model driving method, and verified that this method could improve availability of the cloud platform through the interactive method of Markov chain. These methods certified the availability of VMs in the extensible cloud platform from a theoretical level, but they have not realized task scheduling based on VM availability awareness.

In the aspect of realizing matching of task QoS requirement based on VM availability awareness, Zuo L. Y. et al. [14] used maximum entropy and principle of entropy increase to satisfy resources satisfying user QoS requirement for scheduling. This method realized entropy optimization of virtual resources and a dynamic weighting evaluation model, improving the system availability. Jammal M. et al. [15] used a scheduling method based on high availability awareness of components to maximize availability of application programs while not violating the user Service level agreement and realized run scheduling in OpsnStack environment. Mosong Z. et al. [16] evaluated and optimized surplus capacity during the operation of computing resources, improved the matching degree between resource supply on the cloud platform, and optimized performance of the cloud platform. Shi X. L. et al. [17] distributed virtual resources through the utility maximization model by taking maximum utility as the scheduling objective. These methods have realized resource evaluation and screening from different angles, guaranteed the matching between resource availability and task QoS, shortened task response time, and optimized performance of the cloud platform. However, these methods were deficient in guaranteeing server workload balancing during task scheduling, which will affect task processing capacity of VMs. Regarding task scheduling algorithms realizing server workload balancing, Liu C. et al. [18] realized a gaming method of workload balancing through a request migration strategy based on server availability, and this method could converge into Nash balancing very rapidly. However, what this method considered more was dynamic server workload balancing to compensate for the insufficient availability of individual servers with a lack of availability evaluation and analysis of virtualized resources. Chang J. H. et al. [19] realized an extensible and workload balancing method of VM clusters by balancing server workloads through the difference between physical machine and VM, but this method lacked availability evaluation of VM resources, which affected task execution efficiency. Nowadays, these task scheduling algorithms based on these availability awareness have problems such as: the gap between available task processing capacity of VM resources in the cloud platform and task availability requirement is large, and accurately evaluating availability of VM resource is impossible; workload balancing of resources cannot be guaranteed under availability constraints; when task workload pressure is large among VMs, task completion time will be affected.

Therefore, starting from task availability requirement of VMs in the cloud platform, the differential entropy model between task requirement for availability and available task processing capacities of VMs was established in this study through evaluation of availability processing capacities of VMs in the cloud platform and VM workloads. Through this model and taking maximum entropy as the goal, a task scheduling algorithm based on availability awareness, which elevated task processing speed, reduced workloads of VMs, solved the matching problem between availability of VM resources and task requirement, and guaranteed server workload balancing and improved task execution efficiency in VMs, was established.

The rest part of this study is organized as follows: Section 3 describes cloud platform models, task requirement model in the cloud platform, server workload evaluation, and task scheduling based on VM availability awareness. Section 4 presents experimental results and result analysis. In the final section, the entire study was summarized, and relevant conclusions were drawn.

3. Methodology

3.1 Cloud platform model

In the cloud platform system, a resource pool consists of a series of servers, each server is expressed by S_i, and server resource cluster of the cloud environment is $S=\{S_1,S_2...S_n\}$. Under initial state, k VMs are assumed distributed on the server S_i at each node and expressed as $S_i=(vm_{i,1}...vm_{i,k})$, where $vm_{i,k}$ is a VM on the server S_i. Each server S_i

contains multiple hardware resources such as CPU, memory and network.

3.2 Task requirement model in the cloud platform

The user provides a server request $\{T_i, i = 1...n\}$ for a service, and each type of service has QoS requirement. The requests for the same type of the service may have different requirements for time of arrival (TOA), completion time, and completion rate. Before VMs execute a task, the task QoS requirement must be defined so that VMs can be distributed according to QoS requirement. $Q_{i,j} = \{d_{i,j}, r_{i,j}\}$ is set, where $d_{i,j}$ is the requirement of the j (th) VM for task completion time, and $r_{i,j}$ is task completion rate of the VM within the stipulated time limit $d_{i,j}$.

Task request rate of a single VM resource is $\theta_{i,j} = p_{i,j}\lambda$, which represents the quantity of user requests received by the VM in the cloud platform within unit time, $p_{i,j}$ is the probability for the j (th) VM in the i (th) server to receive a task, and λ is task arrival rate in the cloud platform. The sum of tasks of VM resources is $\Lambda = \sum_{i=1}^{n}\sum_{j=1}^{k}\theta_{i,j}$. The user submits the task $\{T_i, i = 1...n\}$ to VMs in the cloud platform, and VM clusters will optimize task scheduling according to task requirement and availability evaluation of VM resources.

3.3 Availability evaluation of VMs in the cloud platform

VM availability directly influences task scheduling in the cloud platform, so evaluation and analysis of VM availability constitute the precondition for VM availability awareness.

Definition 1: Task availability requirement. For task service request in the cloud platform, availability of VM resources expresses the satisfaction degree of task QoS requirement. Availability analysis of VM resources is analyzed in this study so that they can satisfy task availability requirement.

Definition 2: Availability of VM resources. Availability of VM resources is the capacity of providing functional services within stipulated time after the task is submitted to VMs. A is used to express availability of VM resources. In this study, when task arrival rate is fixed, availability of VMs can be described through available task processing capacity of VMs. Available task processing capacity of VMs can be measured using task arrival rate, completion time, and completion rate.

Definition 3: Available task processing capacity $\mu_{i,j}$ of VMs. The greater the $\mu_{i,j}$ of VM resources within unit time, the stronger the resource service capability. $\mu_{i,j}$ expresses processing capacity of the j (th) VM in the i (th) server after receiving the task $\{T_i, i = 1...n\}$.

$$\mu_{i,j} = p_{i,j}\lambda / t_q \tag{1}$$

Task completion time t_q refers to the difference between task time of arrival t_r and task completion time t_c, namely $t_q = t_r - t_c$.

When arrival rate of the same type of services is fixed, the shorter the task completion time, and the stronger the available task processing capacity of VMs.

Definition 4: Available task processing capacity of each server is the sum of processing capacities of all VMs in the server,

$$\mu_i = \sum_{j=1}^{k} p_{i,j}\lambda / t_q . \tag{2}$$

According to Literature [20], task completion time of VMs complies with exponential distribution, $f(t) = ae^{-at}, a > 0$, and a is a constant. $a = \mu - \lambda$, where λ is task arrival rate, and μ is task processing rate. Available task processing capacity of VMs with QoS requirement is as follows:

$$\mu_{i,j}^{*} = \ln[\frac{1}{1 - r_{i,j}}] / d_{i,j} + p_{i,j}\lambda . \tag{3}$$

The relationship between VM resources and task QoS requirement should be considered in the task scheduling process of VM resources, such as considering available task processing capacities of VMs and resource balancing, and distributing the task that does not satisfy QoS requirement to the VM should be avoided. Theoretically, if the VM satisfies task QoS requirement, available task processing capacity of the VM is greater than task request rate $\mu_{i,j} \geq \theta_{i,j}$. However, when a VM in the cloud platform processes a task request, it cannot satisfy task QoS requirement; consequently, the task cannot be completed, or completion time will be lengthened.

Definition 5: The difference of the available task processing capacity of the VM resource means the large gap between task requirement for resource availability and available task processing capacities of actual VM resources.

$$b_{i,j} = \begin{cases} 0, & \text{if } \mu_{i,j}^{*} \leq \mu_{i,j} \\ \mu_{i,j}^{*} - \mu_{i,j} & \text{if } \mu_{i,j}^{*} > \mu_{i,j} \end{cases}. \tag{4}$$

$b_{i,j}$ indicates the difference of available task processing capacity of the j (th) VM in the i (th) server.

The difference between available task processing capacity of a single virtual machne and task processing capacity satisfying QoS requirement is calculated as below:

$$\delta_{i,j} = \theta_{i,j} b_{i,j} = p_{i,j}\lambda b_{i,j} . \tag{5}$$

The sum of actual task processing capacities of VMs distributed with tasks in the whole platform and task processing capacity satisfying QoS requirement is:

$$\delta = \sum_{i=1}^{n}\sum_{j=1}^{k}\delta_{i,j} = \sum_{i=1}^{n}\sum_{j=1}^{k} p_{i,j}\lambda b_{i,j} . \tag{6}$$

Definition 6: Differential entropy of VM availability. Under VM environment in the cloud platform, the relative difference of available task processing capacities of VMs is $B_{i,j} = \delta_{i,j} / \sum_{i=1}^{n}\sum_{j=1}^{k}\delta_{i,j}$ at time t. The differential entropy of VMs in the cloud platform at time t is as follows:

$$A(t) = \sum_{i=1}^{n}\sum_{j=1}^{k}[B_{i,j} \times \ln(1/B_{i,j})], B_{i,j} \neq 0 . \quad (7)$$

The entropy concept is used to express uncertainty degree of available information state of a VM. $A(t)$ is used to express the fitness between availability of VM resources and task QoS requirement. The greater the entropy value, the more the task QoS requirement coincides with available task processing capacity provided by VMs and the higher the fitness.

3.4 Evaluation of server workload

Server workload in the cloud platform is an important factor influencing available task processing capacity of VMs, so server workload balancing should be considered in the evaluation of task processing capacities of VMs.

Definition 7: The workload $L(S_i)$ at server node. resource (CPU, memory, network, and so on) utilization ratio on the server is a direct index of system workload.

$$L(S_i)=\alpha_1 w_c + \alpha_2 w_m + \alpha_3 w_n \quad (8)$$

where α_1 is a group of objective weights, and the triple (w_c, w_m, w_n) represents utilization ratio of CPU, memory, and network.

If overworkload happens at the server where the VM is located, the system performance of the entire server will clearly decline even if available task processing capacity satisfies QoS requirement. This will affect task execution of the VM. Therefore, an upper workload limit is set for each server in the cloud platform,

$$L(S_i) \leq Z . \quad (9)$$

Z is upper workload limit of the server.

Formula (9) expresses that server workload balancing in the cloud platform system is guaranteed by limiting server over workload. Measurement of workload balancing is the sum of absolute values of average workloads at each physical node which deviate from the system. The goal of workload balancing is to minimize the sum of absolute values of average workloads which deviate from the system, as shown in Formula (10):

$$\text{Minimize} \sum_{i=1}^{n}\left|L(S_i) - \bar{l}\right| . \quad (10)$$

where $\bar{l}$ is average workload of the server as shown in Formula (11),

$$\bar{l} = \frac{\sum_{i=1}^{n} L(S_i)}{n} . \quad (11)$$

Definition 8: Relative workload of the server. The initial workload of the server S_i is $L_0(S_i)$, and this is no task in the VM. During the task scheduling process, the workload of the server where the VM is located is $L(S_i)$, and relative workload at the time is:

$$L(S_i)=\frac{L(S_i)}{L_0(S_i)} . \quad (12)$$

The relative workload $L(S_i)$ of the server S_i reflects the average proportion of tasks of VMs in the server resources. The sum of tasks of VMs in the server S_i is $\sum_{j=1}^{k} p_{i,j}\lambda$. The greater the sum, the greater the relative workload $L(S_i)$ of the server, and the smaller the relative workload $L(S_i)$.

3.5 Task scheduling algorithm

3.5.1 Basic principles of task scheduling based on availability awareness

The relationship between availability awareness of VM resources and task QoS requirement is a problem of evaluation of VM availability and optimal task combination-type scheduling problem. Under VM availability constraint conditions, task QoS is maximized to guarantee task completion rate.

Problem definition: A service request $\{T_i, i = 1...n\}$ is given in the cloud platform, each of the servers $S=\{S_1, S_2 ... S_n\}$ includes VMs $S_i = (vm_{i,1} ... vm_{i,k})$. The maximization of differential entropy $A(t)$ of VM availaibilities ensures that VM availability satisfies task QoS requirement. Meanwhile, server workloads in the cloud platform are balanced.

According to conceptual models and problem definition in 3.3 and 3.4, the task scheduling algorithm based on VM availability is decided by the following factors: 1) working intensity and computing power of VMs embodied by available task processing capacities of VMs and 2) workload of the server where the VM is located.

Theorem 1: If the upper workload limit Z of a server node is small and approximate to average workload $\bar{l}$, then the server workloads in the cloud platform will be more balanced.

Proof: If upper workload limit Z of each server is equal to average workload $\bar{l}$, then server workloads are completely balanced, that is, workloads of all servers are equal to average workload $\bar{l}$, and the value of Formula (10) is 0.

It's assumed that $L(S_i)=Z_i$ and $L(S_j)=Z_j$, namely, upper workload limits of servers S_i and S_j are Z_i and Z_j, respectively. If upper workload limit of the server S_i is greater than $\bar{l}$, that is, $Z_i > \bar{l}$, then a server satisfies the workload being smaller than $\bar{l}$, namely, $Z_i > \bar{l} > Z_j$. Therefore, the value calculated through Formula (10) is greater than 0. The more greatly the workload deviates from the upper workload limit, the less balanced the system will be.

Theorem 1 indicates that if server workloads are balanced, then server workload value should be $\bar{l}$. Satisfying the strict condition—workload balancing—during task scheduling process of VMs is difficult. However, the server workload that is as close as to $\bar{l}$ is feasible in this process. If upper workload limit of each server is restricted and the upper limit Z is made as close as to $\bar{l}$, then server workloads will be more balanced.

Theorem 2: For two given VMs vm_1 and vm_2, their configurations are the same and so are the time for task request $\{T_i, i=1...n\}$ to reach VMs. Available task processing capacities of the two VMs which satisfy QoS requirement are assumed to satisfy $\mu_1^* > \mu_2^*$, and then $p_1\lambda > p_2\lambda$.

Proof: Formula (3) shows that $\mu_1^* = \ln[\frac{1}{1-r_1}]/d_1 + p_1\lambda$ and $\mu_2^* = \ln[\frac{1}{1-r_2}]/d_2 + p_2\lambda$. When task completion rate is 1 and TOA is close to 0, that is, tasks arriving at VMs are immediately all completed, then $\lim(\ln[\frac{1}{1-r_1}]/d_1) = \lim(\ln[\frac{1}{1-r_2}]/d_2) = 0$. $\mu_1^* > \mu_2^*$ achieves the result $p_1\lambda > p_2\lambda$.

The greater the available task processing capacity of the VM, the more the tasks can be processed by the VM, and it can satisfy task QoS requirement.

Theorems 1 and 2 indicate that the task scheduling method, which sets upper workload limit of servers and takes advantages of available processing capacities of VMs, can realize workload balancing of servers and elevate task processing speed.

Theorem 3: For the given service requests $\{T_i, i=1...n\}$ and servers $S=\{S_1, S_2...S_n\}$ in the cloud platform, each server includes VMs $S_i = (vm_{i,1}...vm_{i,k})$. When and only when available differential entropy of VMs in the cloud platform system reaches maximum Max $(A(t))$ within unit time, available task processing capacities of VM resources and task QoS requirement will be more balanced.

Proof: According to Definition 6, available differential entropy is $A(t) = \sum_{i=1}^{n}\sum_{j=1}^{k}[B_{i,j} \times \ln(1/B_{i,j})]$, where $B_{i,j} = \delta_{i,j} / \sum_{i=1}^{n}\sum_{j=1}^{k}\delta_{i,j}$. The relative difference $B_{i,j}$ of available task processing capacities of VMs has limited values, namely, $B_{1,1},...B_{i,j}$, and the constraint condition is $s.t. \sum_{i=1}^{n}\sum_{j=1}^{k}B_{i,j} = 1$. $B_{i,j}$ with maximum entropy value can be solved according to Lagrange multiplication method. $G(B_{1,1},...B_{i,j}) = \sum_{i=1}^{n}\sum_{j=1}^{k}[B_{i,j} \times \ln(1/B_{i,j})] + \gamma(\sum_{i=1}^{n}\sum_{j=1}^{k}(B_{i,j}) - 1)$, G is used to solve partial derivative of $B_{i,j}$, the derivative is set as 0, and the equation set ($\partial G / \partial B_{i,j} = \ln(1/B_{i,j}) - 1 + \gamma = 0$ and $B_{i,j} = \exp(\gamma - 1)$) is obtained. As $\sum_{i=1}^{n}\sum_{j=1}^{k}B_{i,j} = 1$, $B_{1,1} = B_{1,2} = ... = B_{i,j} = 1/n+k$. The available differential entropy is the maximum. According to principle of maximum entropy, available task processing capacity of a VM is related to task QoS requirement. When task processing capacities of VMs are uniformly distributed, the entropy value $A(t)$ will reach the maximum.

According to Theorem 3, when available task processing capacities of VMs in the cloud platform present uniform distribution with task QoS requirement, the entropy value is the maximum. Therefore, available differential entropy of VM resources can ensure balance between VMs in available task processing capacities and satisfy task QoS requirement.

Theorem 4: Given servers S_1 and S_2 in the cloud platform have the same configurations of VMs, and the task sum of VMs in the server S_i is $\Omega(S_i) = \sum_{j=1}^{k} p_{i,j}\lambda$. The two servers have the same initial workload, namely, $L_0(S_1)=L_0(S_2)$. For given task requests $\{T_i, i=1...n\}$, when available differential entropy of VMs reaches the maximum value and total sums of tasks in the two servers satisfy $\Omega(S_1) > \Omega(S_2)$, their relative workloads will satisfy $L(S_1)>L(S_2)$.

Proof: Under Max $(A(t))$, namely, $B_{1,1} = B_{1,2} = ... = B_{i,j}$, available differential entropy between servers S_1 and S_2 is k/n, indicating that their available task processing capacities are equal. The number of tasks processed by the server S_1 is greater than that processed by the server S_2, namely, $\Omega(S_1) > \Omega(S_2)$ and initial workloads satisfy $L_0(S_1)=L_0(S_2)$, so their workloads satisfy $L(S_1)>L(S_2)$. According to Definition 8, relative workloads of the two servers satisfy $\tilde{L}(S_1)>\tilde{L}(S_2)$.

Theorem 4 indicates that more tasks are processed by the server S_i under maximization of available differential entropy of VMs, and relative workload $\tilde{L}(S_i)$ of the server is greater.

Theorems 3 and 4 explain the relationship between available task processing capacities VMs and server workloads. During the task scheduling process which aims to maximize available differential entropy of VMs, the task QoS requirement is satisfied. This ensures balanced distribution of available task processing capacities of VMs. In the meantime, server workloads in the whole cloud platform can be more balanced.

3.5.2 Task scheduling algorithm based on availability awareness

Based on dynamic evaluation of availabilities of VM resources, a task scheduling algorithm for VMs is proposed by directing at availabilities of VM resources.

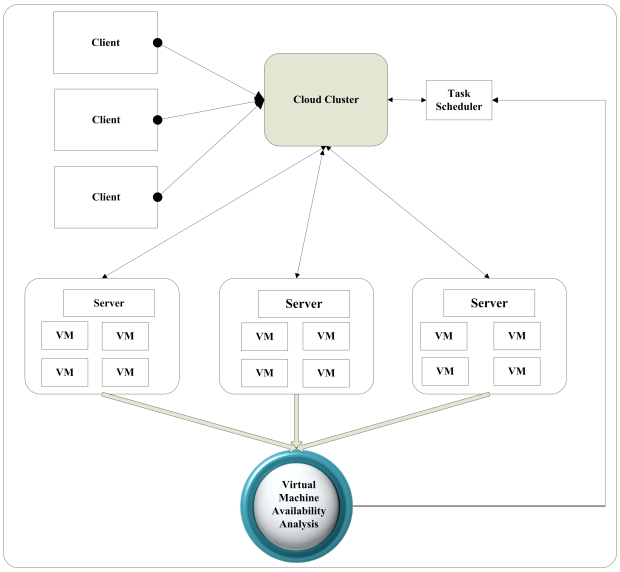

Fig. 1. Architecture of VM availability-aware scheduling

Task scheduling architecture based on availability awareness is shown in Figure 1. The "first arriving, first

served" mode is adopted, but available task processing capacities of VM resources should be considered in the task allocation process. Selection is conducted among resources with great availabilities, which avoids slowing down task processing.

For given task resource requests $\{T_i, i = 1...n\}$, task scheduling is managed through task scheduler module. According to descriptions given in Theorem 3, task scheduling aims to maximize available differential entropy of VMs, and VMs with relatively small workloads on the servers where they are located are selected, which can improve task processing speed as shown in Algorithm 1:

Algorithm 1. Task scheduling algorithm of maximization of available differential entropy of virtual machines

Input: A set of tasks $\{T_i, i = 1...n\}$, corresponding λ, $Q_{i,j} = \{d_{i,j}, r_{i,j}\}$. Cloud resource server cluster $S=\{S_1, S_2 ... S_n\}$ and virtual machine $S_i = (vm_{i,1} ... vm_{i,k})$.

Ouput: The request tasks allocation

Step 1) Calculate virtual machine available task processing capabilities to satisfy QoS requirements based on formula(3).
Step 2) Initialize virtual machine available task processing capabilities $\mu_{i,j} \leftarrow \infty$
Step 3) for all tasks $\{T_i, i = 1...n\}$ do
Step 4) for all VMs $vm_{i,1} ... vm_{i,k}$ do
Step 5) Assigned tasks T_i into $vm_{i,k}$
Step 6) Calculate $vm_{i,k}$ vailable task processing capabilities $\mu_{i,j}$ based formula (1)
Step 7) end for
Step 8) for all tasks $\{T_i, i = n+1.....\}$ do
Step 9) if $B_{1,1} \neq B_{1,2} \neq ... \neq B_{i,j}$ then
Step 10) Assigned tasks T_i into $vm_{i,k}$ based on theorem 3.
Step 11) else add a new VM
Step 12) Assigned tasks T_i into $vm_{i,k}$ based on algorithm 2
Step 13) End if
Step 14) End for

The ratio of initial workload of each server to the current workload is used to conduct periodic and dynamic evaluation of server workloads so that task scheduling reaches the goal of workload balancing from two aspects—available task processing capacities of VMs and server workloads, Moreover, task QoS requirement is satisfied.

Algorithm 2. Server workload balancing algorithm of maximization of available differential entropy of virtual machines.

Input: A set of tasks $\{T_i, i = 1...n\}$, Cloud resource server cluster $S=\{S_1, S_2 ... S_n\}$ and virtual machine $S_i = (vm_{i,1} ... vm_{i,k})$.

Ouput: Server workload balancing
Step 1) Calculate initial server workload based on formula(8)
Step 2) Initialize the server workload upper threshold Z
Step 3) for all tasks $\{T_i, i = n+1.....\}$ do
Step 4) if $B_{1,1} = B_{1,2} = ... = B_{i,j}$ then
Step 5) Calculate current server workload based on formula (8)
Step 6) if $(L(S_i) \leq Z)$ and $(\min L(S_i))$ then
Step 7) Assigned tasks T_i into S_i based on theorem 4

4. Results analysis and discussion

The effectiveness of the algorithm will be verified in this section from three aspects: (1) server workload balancing analysis of the task scheduling algorithm based on availability awareness, (2) evaluation and analysis of available task processing capacities of VMs, and (3) three task scheduling algorithms are compared in the experiment to verify the effectiveness of the algorithm proposed in this study. The three experimental comparison methods are:

(1) Random algorithm: express random allocation of task resource requests so that a task can be randomly allocated to a VM.

(2) Minmin algorithm: A scheduling algorithm featuring completing tasks at the highest speed, it means allocating task resources to the VM with the highest operation speed so tasks can be completed as soon as possible. This algorithm attaches an importance to task execution efficiency of VMs.

(3) Virtual machine availability awareness task scheduling algorithm (VMAATSA): The algorithm proposed in this study conducts task scheduling based on available task processing capacities of VMs to realize maximization of available differential entropy of VMs. This algorithm means equally allocating tasks to different VMs on different servers to keep workload balancing according to available task processing capacities of VMs.

4.1 Deployment of the experimental environment

Open-source Cloudsim 3.0 was used in this study to carry out simulation realization of this algorithm. Cloudsim provided modeling and simulation of VM infrastructure in the cloud platform as well as task generation simulation and realization simulation of the scheduling algorithm [21].

Computer configuration: CPU: Intel i5-3450, quad-core, memory: 12 GB, solid-state disk: Samsung SSD 850, 120 GB.

Deployment of software environment: Windows 7 flagship version, Java 1.8.0_144.

Experimental parameters of Cloudsim 3.0 are set as shown in Table 1.

Table 1. Simulation experiment environment configuration

Number of servers	2 servers 4 servers
Number of VMs	20 VMs 40 VMs
Number of tasks	{40,80,160,320}
Task arrival rate λ	{0.2,0.4,0.8,1.0}

Task QoS requirement satisfies $Q_{i,j} = \{d_{i,j}, r_{i,j}\}$. Task completion time requirements were randomly distributed within $200 - 600ms$ s, and task completion rate $r_{i,j}$ was required to be 95%.

4.2 Analysis of experimental

4.2.1 Comparison of workload balancing degrees among servers

In the experiment, server workload balancing analysis of the task scheduling algorithm based on availability awareness was first implemented. Two servers (20 VMs) and four servers (40 VMs) were simulated through Cloudsim, and task workloads were uniformly 0.2. When task arrival rate λ was very low, the effectiveness of the algorithm on server workload balancing could not be embodied. Therefore, task

arrival rate in this experiment was taken as 0.8, and quantities of simulated tasks were 40, 80, 160, and 320. Workload balancing degrees of servers were measured through Formulas (8)–(10), and the mean values of relative workload quantities of servers are shown in Table 2. Experimental results show that among the three algorithms, workload balancing degree of VMAATSA algorithm proposed in this study is the highest, followed by Random and Minmin in succession. Balancing degree of Minmin algorithm is higher than those of Random and Minmin algorithms by about twice and four times, respectively. Therefore, the results verify the effectiveness of the algorithm proposed in this study on server workload balancing as well as Theorem 4.

Table 2. Workload balance comparison

Number of tasks	Algorithm	2 servers (20 VMs)	4 servers (40 VMs)
40	Random	0.745341615	0.779220779
	Minmin	1.012658228	1.699346405
	VMAATSA	0.125786164	0.263157895
80	Random	0.591715976	1.278538813
	Minmin	0.968858131	2.765957447
	VMAATSA	0.280701754	0.674157303
160	Random	0.683760684	1.507246377
	Minmin	1.265822785	3.018867925
	VMAATSA	0.449438202	0.806451613
320	Random	0.742115028	1.732851986
	Minmin	1.365187713	5.34562212
	VMAATSA	0.502512563	0.904522613

4.2.2 Evaluation analysis of available task processing capacities of VMs

Evaluation analysis of available task processing capacities of VMs was carried out under the following conditions: arrival rate $\lambda = 0.8$ and quantity of tasks is 320. Available differential entropies of VM resources among three algorithms—Random, Minmin, and VMAATSA—are compared as shown in Formula (7). Experimental results are shown in Figure 2. Within task execution time of 1–4 s, available differential entropy of VM resources in VMAATSA algorithm is greater than those in Random and Minmin algorithms, indicating that within the time scope, available task processing capacities of VMs are more balanced. After 4 s, as all tasks are approximately completed, available differential entropy of VMAATSA declines significantly.

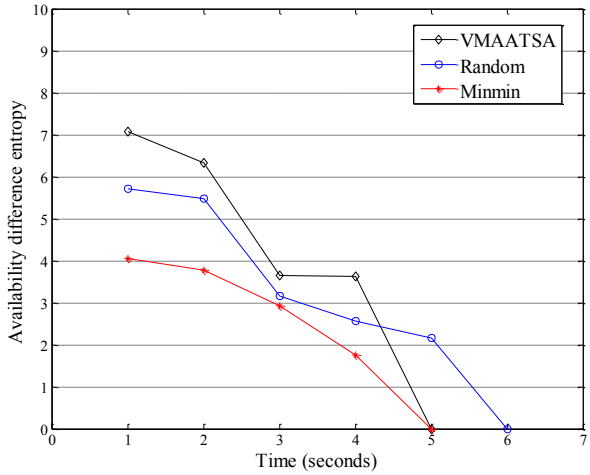

Fig. 2. Comparison of available differential entropy of VMs

In the evaluation experiment of available task processing capacities of VMs, task QoS satisfaction rates of VMs in the three algorithms were analyzed when arrival rate was $\lambda = \{0.2, 0.4, 0.8, 1.0\}$ and quantity of tasks was 320. Experimental results are shown in Figure 3. When $\lambda = 0.2$, task QoS satisfaction rates of three algorithms were 100. When $\lambda = 1.0$, task QoS satisfaction rates were 90.1%, 88.1%, and 80.8% respectively. Experimental results indicate that the algorithm proposed in this study can better satisfy task QoS requirement of VMs in the cloud platform.

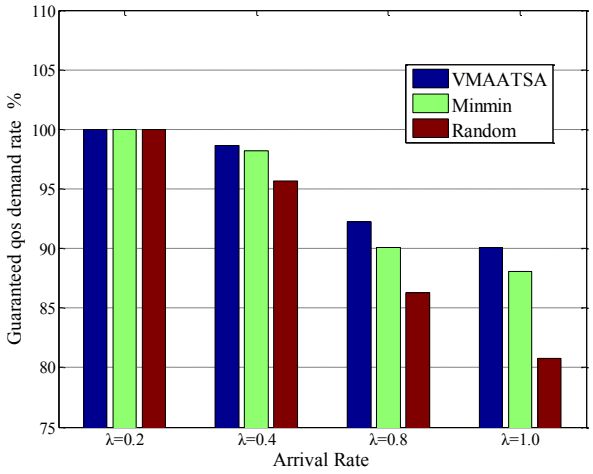

Fig. 3. Comparison of QoS satisfaction rates of VM tasks

4.2.3 Efficiency analysis of task scheduling algorithm based on availability awareness

Task completion time of three algorithms—Random, Minmin, and VMAATSA—were compared in the experiment through 40 VMs to analyze the execution efficiency of the algorithm proposed in this study. Experimental results are shown in Figure 4. According to Figures 4 (a), (b), (c), and (d), quantity of tasks are 40, 80, 160, and 320, respectively, when task arrival rates are 0.2, 0.4, 0.8, and 1.0, respectively. When task arrival rate is $\lambda = 0.8$ and quantity of tasks is 160 and 320, respectively, task completion time of VMAATSA is ahead of those of Random and Minmin algorithms by 28% and 6% and by 28% and 8%, respectively. Experimental results in Figure 4 show that the more the quantity of tasks, the more rapidly the tasks are completed using VMAATSA algorithm.

Workload balancing of the task scheduling algorithm based on VM availability awareness was verified in the experiment, followed by availability evaluation and a comparative analysis of task completion time. During the

experimental period, three task scheduling algorithms were compared. The results show that VMAATSA algorithm not only satisfies task QoS requirement but also contributes to greater balance of workloads in the VM platform system. Moreover, it can improve task execution efficiency. The results show that VMAATSA algorithm has more significant comprehensive performance.

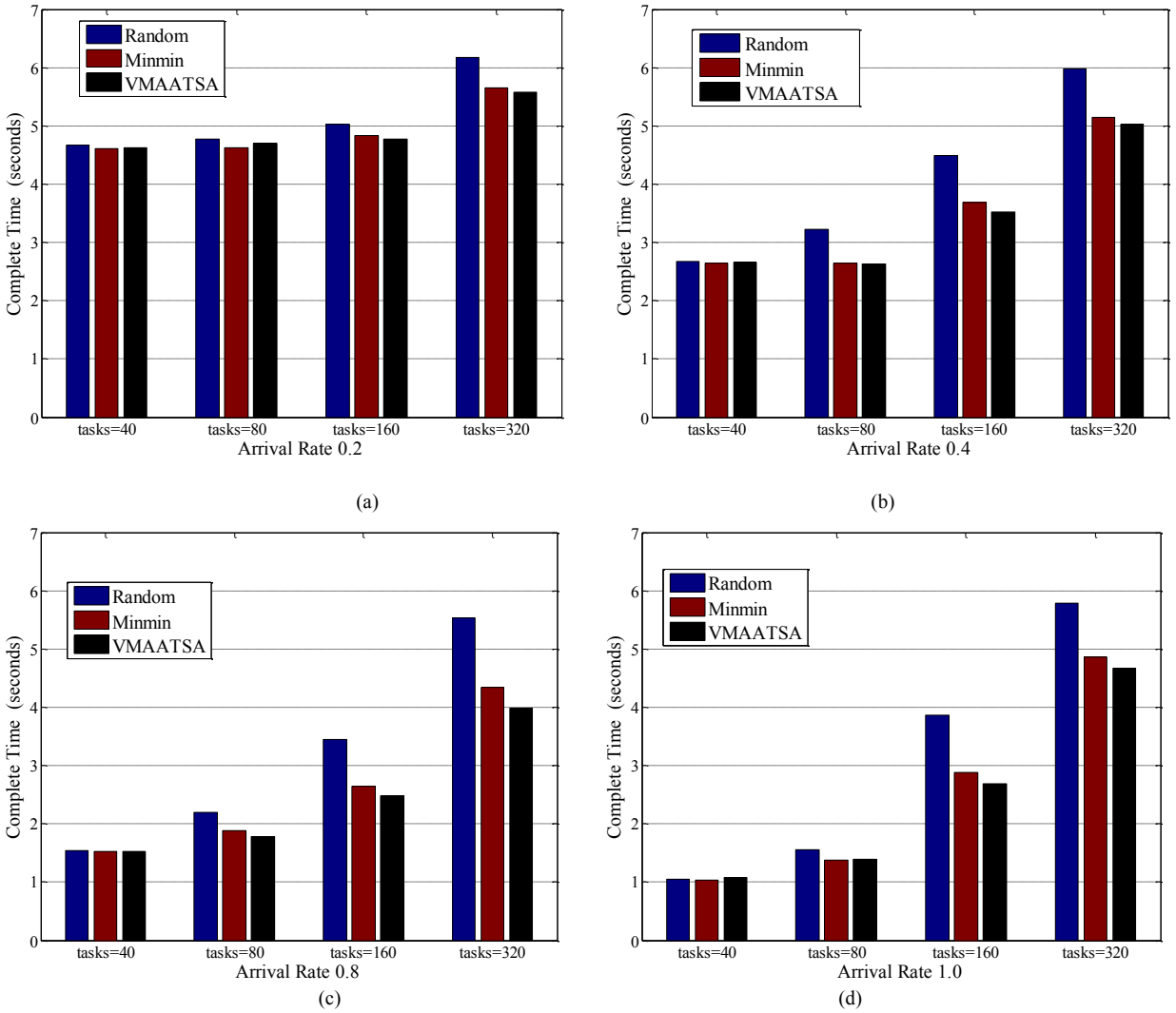

Fig. 4. Comparison of task completion time of VM

5. Conclusions

In the cloud platform, dynamic change of VM availabilities makes satisfying the task QoS requirement difficult. To improve task scheduling capacities of VMs in the cloud platform and satisfy task QoS requirement, a task scheduling algorithm based on VM availability awareness was proposed in this study to solve the matching problem between available task processing capacities of VMs and task QoS requirement and realized workload balancing of servers in the cloud platform. The following conclusions could be drawn:

(1) Task scheduling based on VM availability awareness can guarantee satisfying task QoS requirement. The valuation method of VM availability can be used to evaluate available task processing capacities of VMs and realize the matching between available task processing capacities of VMs and task QoS requirement.

(2) The relationship between available task processing capacities of VMs and task QoS requirement is used to establish an available differential entropy model, which can accurately allocate tasks and prevent excessive use of VM resources.

(3) The task scheduling algorithm based on VM availability awareness improved task execution speed of VMs, shortened task completion time, and guaranteed workload balancing of servers in the cloud platform.

The algorithm proposed in this study realized task scheduling among VMs given the relationship between available task processing capacities of VMs in the cloud platform and task QoS requirement. This algorithm, which is suitable for the dynamic environment of VMs in the cloud platform, can improve task scheduling efficiency and provide a more accurate technical support for task scheduling among VMs in the cloud platform. However, unified and fixed workload method was adopted in this study during the task scheduling process of VMs without consideration of actual diversity of tasks. This problem will be further studied to improve extensive applicability of the task scheduling algorithm based on availability awareness.

Acknowledgements

This work was supported by following projects: the Key Program for Science and Technology Development of Jilin

Province of China (Grant No. 20130206052GX). Project supported by the National Natural Science Foundation of China (Nos. 61602057), the Science and Technology Department of Jilin Province, China (No. 20170520059JH)

References

1. Sharma, B., Chudnovsky, V., Hellerstein, J. L., Rifaat, R., Das, C. R., "Modeling and synthesizing task placement constraints in Google compute clusters". In: *Proceedings of the 2nd ACM Symposium on Cloud Computing*, New York, USA: ACM, 2011, pp.1-14.
2. Gulati, A., Shanmuganathan, G., Holler, A. M., Ahmad, I., "Cloud scale resource management:challenges and techniques". In: *Proceedings of the 3rd USENIX Conference on Hot Topics in Cloud Computing,* Portland, USA: USENIX Association Berkeley, 2011, pp.3-3.
3. Liu, J., Wang, S., Zhou, A., Yang, F., Buyya, R., "Availability-aware virtual cluster allocation in bandwidth-constrained datacenters". *IEEE Transactions on Services Computing*, 2017, doi: 10.1109/TSC.2017.2694838.
4. Javadi, B., Kondo, D., Vincent, J. M., Anderson, D. P., "Discovering statistical models of availability in large distributed systems: An empirical study of seti@ home". *IEEE Transactions on Parallel and Distributed Systems*, 22(11), 2011, PP.1896-1903.
5. Qin, X., Xie, T., "An availability-aware task scheduling strategy for heterogeneous systems". *IEEE Transactions on Computers*, 57(2), 2007, pp. 188-199.
6. Cao, J., Zeng, G., Niu, J., Xu, J., "Availability-Aware scheduling method for parallel task in cloud environment". *Journal of Computer Research and Development*, 50(7), 2013, PP. 1563-1572.
7. Jeyakrishnan, V., Sengottuvelan, P., "Efficient on demand dynamic availability-distribution-span scheduling and load balancing scheme for cloud computing". *Journal of Computational and Theoretical Nanoscience*, 13(10), 2016, pp. 7655-7660.
8. Sun, J., Zhang, X., Dong, X., "A real-time task availability improving fault-tolerant scheduling algorithm on heterogeneous platform". *Journal of Computer Research and Development*, 52(12), 2015, pp. 2669-2683.
9. Jin, G., Zhang, P., Yu, M., "Cost constrain load balanced ant colony scheduling of cloud environment". *Journal of Information &Computational Science*, 12(3), 2015, pp. 1045-1054.
10. Wang, T., Liu, Z., Chen, Y., Xu, Y., Dai, X., "Load balancing task scheduling based on genetic algorithm in cloud computing". In: *2014 IEEE 12th International Conference on Dependable, Autonomic and Secure Computing*, Dalian, China: IEEE, 2014, pp.146-152.
11. Chen, H., Wang, F., Helian, N., Akanmu, G., . "User-priority guided min-min scheduling algorithm for load balancing in cloud computing". In: *2013 National Conference on Parallel computing technologies*, Bangalore, India: IEEE, 2013, pp.1-8.
12. Kim, D. S., Machida, F., Trivedi, K. S., "Availability modeling and analysis of a virtualized system". In: *2009 15th IEEE Pacific Rim International Symposium on Dependable Computing*, Shanghai, China: IEEE, 2009, pp. 365-371.
13. Ghosh, R., Longo, F., Frattini, F., Russo, S., Trivedi, K. S., "Scalable analytics for IaaS cloud availability". *IEEE Transactions on Cloud Computing*, 2(1), 2014, pp. 57-70.
14. Zuo, L. Y., Cao, Z. B., Dong, S. B., "Virtual resource evaluation model based on entropy optimized and dynamic weighted in cloud computing". *Journal of Software*, 24(8), 2013, pp. 1937-1946.
15. Jammal, M., Kanso, A., Shami, A., "CHASE: Component high availability-aware scheduler in cloud computing environment". In: *2015 IEEE 8th International Conference on Cloud Computing*, New York, USA: IEEE, 2015, pp. 477-484.
16. Mosong, Z., Xiaoshe, D., Heng, C., Xingjun, Z., "Improving cloud platform based on the runtime resource capacity evaluation". *Journal of Computer Research and Development*, 54(11), 2017, pp. 2516-2533.
17. Shi, X. L., Xu, K., "Utility maximization model of virtual machine scheduling in cloud environment". *Chinese Journal of Computers,* 36(2), 2013, pp. 252-262.
18. Liu, C., Li, K., Li, K., "A game approach to multi-servers load balancing with load-dependent server availability consideration". *IEEE Transactions on Cloud Computing*, 2018, doi: 10.1109/TCC.2018.2790404.
19. Chang, J. H., Cheng, H. S., Chiang, M. L., "Design and implementation of scalable and load-balanced virtual machine clusters". In: *2017 IEEE 7th International Symposium on Cloud and Service Computing (SC2)*, Kanazawa, Japan: IEEE, 2017, pp.40-47.
20. Robertazzi, T. G., "*Computer networks and systems: queueing theory and performance evaluation*". Springer Science & Business Media, Germany, 2012, pp.19-37.
21. Calheiros, R. N., Ranjan, R., Beloglazov, A., De Rose, C. A., Buyya, R., "CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms". *Software: Practice and Experience*, 41(1), 2010, pp.23–50.