Cloud Simulation Resource Scheduling Algorithm Based on Multi-dimension Quality of Service

Wuqi Gao and Fengju Kang
College of Marine Engineering, Northwestern Polytechnical University, Xi'an 710032,
People’s Republic of China

Abstract: Aiming at the shortcomings of cloud simulation task scheduling algorithm, this paper put forward cloud simulation scheduling algorithm based on multi-dimension QoS (Quality of Service). Firstly, analytic hierarchy process in economic field was introduced into the Resource scheduling algorithm to compute every dimensional parameters weight, then the tasks was allocated to appropriate resource according to customer satisfaction, QoS distance and loading equilibrium, etc. Finally, scheduling algorithm was analyzed by theory example and was simulated with CloudSim tool package. The experiment shows that this cloud simulation scheduling algorithm not only meets customer needs for multi-dimension QoS, but shortens the simulation finishing time and greatly improves simulation resource utility rate.

Key words: Cloud computing, simulation, quality of service, resource scheduling, efficiency coefficient method, CloudSim

INTRODUCTION

In order to apply cloud computing to simulation field, scholar Li Bohu proposed the Concept of Cloud Simulation Platform in 2009 (Li et al., 2009). The resource service scheduling technology of the cloud simulation platform is different from other common computing and it possesses the following characteristics: (1) Multi-machine collaboration which means that cloud simulation virtual machines are distributive in geographical places, but they belong to different owners logistically. (2) Dynamic isomerism which means that net virtual nodes can be dynamically added into and exited from cloud simulation system. (3) Different QoS goals (Yan-Bing et al., 2010; Salim et al., 2007; Du et al., 2006; Akhtar, 2007; Nehra et al., 2007). Cloud simulation often involves different entities, such as managers and users of cloud simulation and virtual machines. In terms of managerial mechanism, safety strategy, cost and etc, different goals are set for different entities. Some users take the shortest finishing time as priority, some take the lowest cost as priority while what the others emphasize most is whether virtual machines keep higher stability or not (Yan-Bing et al., 2009). Therefore, the hottest issue of present study is to achieve optimal matching between simulation tasks and virtual machines and to satisfy the applied multi-dimension QoS needs.

The traditional resource scheduling algorithms include FCFS (First Come First served) and Round-Robin algorithm. These methods can easily cause problems of task starvation, unbalanced resource and etc. (Zhongxu, 2003). Heuristic algorithm and QoS Guided Min-Min algorithm cause resources idle and tasks extruded, not integrating into the diversity of QoS constrains. Thus, it is difficult for them to reflect dynamical heterogeneous (Li et al., 2008; Shi et al., 2011; Yan-Ping and Zeng-Zhi, 2007; Xu and Wang, 2007; Shen et al., 2011; Yan et al., 2006; Yan-Bing et al., 2009; Gong et al., 2009). A typical scheduling algorithm with the multi-dimension QoS guiding is QDDN (Wu et al., 2007) which is only based on the distance between the task and resource. It does not consider user’s satisfaction and loading equilibrium, etc which results in unsatisfactory scheduling effect. For the above problems to be solved, Cloud Simulation Resource Scheduling algorithm based on QoS, S-CSRISA was put forward.

QUALITY OF SERVICE PARAMETER

Assuming there is no any dependent relationship among the cloud simulation tasks and the length of the submitted task is known, the m submitted tasks can be expressed as \( T = \{ t_1, t_2, ..., t_m \} \), n resource can be indicated as \( R = \{ r_1, r_2, ..., r_n \} \).
Cloud Simulation can use mass simulation resource based on the Internet, so every simulation service function need be monitored in order to select proper resource to finish simulation tasks. Also, the loading equilibrium modeling of cloud simulation platform requires a simulation member to move dynamically, if the problem of function overloaded or breakdown of one simulation member arises and the loading equilibrium modeling needs to monitor simulation resource. This paper presents cloud simulation QoS index system (Fig. 1) consisting of 18 parameters in four aspects of computer, Internet, statistics and simulation member.

**MULTI-DIMENSION QUALITY OF SERVICE EVALUATION**

QoS Standardization: Based on k-dimension QoS capability of resource provisioning, a n x k matrix made of QoS modeling by n resources can be indicated as Qn, k:

\[
Q_{n,k} = \begin{bmatrix}
q_{1,1} & q_{1,2} & \cdots & q_{1,k} \\
q_{2,1} & q_{2,2} & \cdots & q_{2,k} \\
\vdots & \vdots & \ddots & \vdots \\
q_{n,1} & q_{n,2} & \cdots & q_{n,k}
\end{bmatrix}
\]  

(1)

In the same way, based on each dimension QoS parameter’s difference of simulation tasks, a m x k matrix made of QoS modeling by m tasks can be indicated as Qm, k.

Due to the big difference of each dimension parameter values for resource, tasks and for the purpose of a systemic evaluation to simulation QoS need and to resource service capability, standardizing the both matrixes is needed. This study adopts efficiency coefficient method to finish the standardization, the following are the specific rules: the bigger and better index value is defined as maximum variables; the smaller and better index value as minimum variables; the better index value at a point as stability variables; the best index value at a range as range variables.

To design efficiency coefficient method for the above four variables, respectively, Table 1 is the efficiency coefficient method formula.

**Identifying user’s QoS synthetic value with synthetic single efficiency coefficient method:** First of all, set up hierarchy structural modeling. Analytic hierarchy process need the problems to be categorized at the very beginning to set up hierarchy structural modeling. The hierarchy can be divided into three classes:

- The highest tier, also called objective tier, is very often to analyze the expected objectives or ideal outcomes of the problems
- The medium tier, namely principle one, includes intermediary involved in achieving the objectives which consists of a lot of tiers
- The lowest tier, measure or solution one, includes various available measures and solutions for achieving the objectives

Then, through 9 classifications judgment matrix \( C_i \) can be set up to get maximum character value and eigenvector. The analytical hierarchy process takes character vector of the judgment matrix as weight vector of every index and uses summation to compute eigenvector of the said judgment matrix \( C \). The followings are the specifics:

- Normalize every column of judgment matrix:
Table 1: Efficiency coefficient method formula

<table>
<thead>
<tr>
<th>Type</th>
<th>Formula</th>
<th>Condition</th>
<th>Type</th>
<th>Formula</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max</td>
<td>(A-D)(B-D) + 40 + 60</td>
<td>A&gt;B</td>
<td>Min</td>
<td>(A-C)(B-C) + 40 + 60</td>
<td>A&gt;B</td>
</tr>
<tr>
<td>Stability</td>
<td>[C(A-D)]/[C(B-D)] + 40 + 60</td>
<td>A&gt;B</td>
<td>Range</td>
<td>[C(A-D)]/[C(B-D)] + 40 + 60</td>
<td>A&gt;B</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>A=B</td>
<td></td>
<td>100</td>
<td>A=B</td>
</tr>
<tr>
<td></td>
<td>[A(D)-(B-D)] + 40 + 60</td>
<td>A&gt;B</td>
<td></td>
<td>[A(D)-(B-D)] + 40 + 60</td>
<td>A&gt;F</td>
</tr>
</tbody>
</table>

A: Actural value, B: Satisfaction, C: Upper unallowable value, D: Lower unallowable value, E: Upper value, F: Lower value

\[
\bar{a}_{ij} = \frac{a_{ij}}{\sum_{i=1}^{n} a_{i j}} (i,j=1,2,...,n) \quad (2)
\]

- Calculate the sums of all elements in every line:
\[
W_i = \frac{1}{\sum_{j=1}^{n} W_{ij}} (i=1,2,...,n) \quad (3)
\]

- Normalize them:
\[
\bar{W}_{ij} = \frac{1}{\sum_{i=1}^{n} \bar{W}_{ij}} (i=1,2,...,n) \quad (4)
\]

\(W_i\) means relative weight vector of every element in this tier to an element in a previous tier. After identifying every QoS dimensional metrics, formula 5 can be used to compute synthetic value of QoS need:

\[U_q(\omega) \sum_{i=1}^{n} \bar{W}_{ij} \quad (5)\]

**User’s satisfaction:** Cloud simulation task scheduling algorithm should be used to try to meet user’s QoS needs. On the condition that the bigger QoS parameter value in a dimension is, the higher the QoS is, formula (6) is adopted to compute simulation user’s obtained satisfaction in a dimension. Vice versa, on the condition that the bigger the QoS parameter value is, the lower the QoS is, formula (7) is adopted. In formula (7) \(j \in [1, k]\), \(r_{Q_{ij}}\) stands for service ability provided by QoS parameter at Dimension J in Resource Ri, \(t_{Q_{ij}}\) stands for need volume by QoS in Dimension J for Task I.

\[
\begin{align*}
\text{Satisfy}_{ij} &= [r_{Q_{ij}} - r_{Q_{ij}} < t_{Q_{ij}}] \quad (6) \\
\text{Satisfy}_{ij} &= [t_{Q_{ij}} - r_{Q_{ij}} < t_{Q_{ij}}] \quad (7)
\end{align*}
\]

Thus, synthetic satisfaction in all operation resource dimensions for the user Task I can be defined as:

**QoS distance measurement:** QoS needs of various tasks are different, so are the QoS provided by various tasks. In order not to make higher QoS service capability resource occupied by lower QoS need tasks which affects the operation of other tasks and leads to increasing total simulation operating time, tasks should be allotted to its matched resource to be operated. So weight distance measurement method can be used to compute the QoS distance between simulation tasks and the resource:

\[\text{Dis} = \sqrt{\sum_{j=1}^{n} w_j (r_{Q_{ij}} - r_{Q_{ij}})^2; w_j \geq 0; j=1,2,...,k; \sum_{j=1}^{n} w_j = 1} \quad (9)\]

**S-CSRSA ALGORITHM**

Based on multi-dimensional QoS cloud simulation resource scheduling algorithm, strategy of this algorithm is: to arrange the order and allot virtual machines according to synthetic efficiency coefficient method of tasks. At first, select virtual machines with higher satisfaction to allot. If there is only one virtual machine that obtains the highest satisfaction, allot the task to it; if there are many of them with the highest satisfaction, select unused virtual machines to achieve loading equilibrium; if there is no unused virtual machine, allot the task to the virtual machine with shortest distance among the virtual machines and the task. The specific flowchart is showed in the following Fig. 2:

The algorithm of flow chart is as:

**Step 1:** Set up Matrix \(Q_{Q_{t,h}}\) with all task QoS needs of the to be scheduled task set T

**Step 2:** Set up Matrix \(Q_{Q_{a}}\) with QoS capabilities of all virtual machines in available resource set R

**Step 3:** Emerge \(Q_{t,h}\) and \(Q_{Q_{a}}\) to set up a new Matrix \(Q_{m_{t,h}}\)

**Step 4:** Get \(S_{m_{t,h}}\) by standardising the Matrix \(Q_{m_{t,h}}\) by adopting of the Table 1 method

**Step 5:** Isolate \(S_{m_{t,h}}\) to get the standardised task QoS Matrix \(S_{T_{m_{t,h}}}\) and the virtual machine QoS Matrix \(S_{R_{m_{t,h}}}\)
Fig. 2: S-CSRSA algorithm flowchart

**Step 6:** Use analytic hierarchy process to compute the QoS weights

**Step 7:** Compute the synthetic QoS need volume of every task, arrange in the big-to-small order in accordance with the task synthetic QoS need volume and allot firstly the bigger synthetic QoS need volume to operating lines of the proper virtual machines

**Step 8:** Take out the first task $t_0$ in the task lines, compute the satisfaction of this task at every resource, then find out the resource which gives highest satisfaction to this task and store it into resource line $R$

**Step 9:** Compute the distance between Task $T[i]$ and Line $R$ and form $Rs$ with the increasing arrangement

**Step 10:** If $Rs[0]$ is unused, allot the task $T[i]$ to virtual machine $Ra[0]$ and turn to Step 13.

**Step 11:** Compute the biggest differential time $DT$ of operating task of all virtual machines in Line $R$ with the highest satisfaction, mark the virtual machine $M_j$ getting the shortest operating time

**Step 12:** If the operating time $DT$ is longer than operating time of $T[i]$, allot $T[i]$ to virtual machine $M_j$

**Step 13:** Delete $T[i]$ from $T$

**Step 14:** Decide whether $T$ is blank, turn to Step 8 if not

**Step 15:** Finish

**S-CSRSA algorithm theory test:** Given that at present there are 3 cloud simulation tasks, $t_1$, $t_2$, $t_3$, their QoS need dimensional value is 4, they are CPU operation speed, Internet broadband width, stability and simulation member task length, respectively. Of them, the first three elements are positive vector, the last one negative vector. Task dimensional QoS need and the available QoS index value of virtual machines are indicated in the following Table 2. For easy comparison, the index value is the relative value from 0 to 1 on average.

Get the standardized virtual machine Matrix $vS_{t_4}$ and task Matrix $oS_{t_4}$ by using Table 1 to standardize the positive and negative vectors, respectively:
By formula (8) the satisfactions of virtual machines from r1 to r5 in t1 can be computed, respectively they are 0.86, 1.0, 1.0, 1.0, 0.60. The virtual machine v2, v3, v4 have task t1 get the highest satisfaction and make themselves marked with the highest satisfaction for task t1. Thus, task t1 is allotted to the virtual machines with the shortest distance between it and the virtual machines v2, v3, v4 to be operated. By formula (9) the distances between Task t1 and v2, v3, v4 can be computed, respectively they are 0.31, 0.23 and 0.25. The distance between Task t1 and the virtual machine v1 is the smallest and v3 is free at the time, so allotting Task t1 to v3 to be operated.

SIMULATION RESULT AND ANALYSIS

CloudSim (Calheiros et al., 2009, 2010) is a cloud computing simulator device which is developed under the team led by professor Rajkumar Buyya from Melbourne University, Australia. In it, the method of bind Cloudlet To VM (int cloudletID, int VMID) provisioning by Data center Broker binds every single task to single fixed VM (virtual machine) to be operated which materializes the resource research need for the reasonable and special-tasked VMs. The type of Data center Broker of expansion CloudSim platform is to self-define bind Cloudlet To VM Method to achieve its own modeling strategy. After the expansion, the CloudSim platform need to be retranslated and set up.

At first, add relative QoS index in cloudlet and VM of CloudSim. For easy testing, this paper chooses four parameters of CPU operation speed, Internet broadband width, stability and simulation member task length as example to test.

Through simulation platform this section is mainly to model task finishing time, resource utility rate, loading equilibrium, user satisfaction and etc of algorithm of QDNN and S-CSRSA proposed by Wu et al. (2007). Net environment of 10 VMs and only one PE for every resource were designed, the processing capability of every resource is 515 (MIPS). Resource description is indicated in Table 3.

Computing time comparison: Figure 3 and 4 are the computing time comparison of QDNN and S-CSRSA algorithms. From the aspects of operating result, for the same task and same virtual environment, computing time of QDNN algorithm is 751.63 while that of S-CSRSA algorithm is 377.57, the latter saves time by a half of the former.

Comparison of satisfaction and loading equilibrium: Establish the following regulations to the scheduling VMs.
and user tasks. All QoS value of user tasks should use rand() to get a random number ranged 0-1. Look at Table 3 for VM QoS parameters.

Table 3: VM QoS parameters

<table>
<thead>
<tr>
<th>VM No</th>
<th>CPU operation speed</th>
<th>Internet broadband speed</th>
<th>Stability</th>
<th>Simulation member task length</th>
</tr>
</thead>
<tbody>
<tr>
<td>VM1</td>
<td>0.7</td>
<td>0.6</td>
<td>0.3</td>
<td>0.6</td>
</tr>
<tr>
<td>VM2</td>
<td>0.1</td>
<td>0.1</td>
<td>0.9</td>
<td>0.5</td>
</tr>
<tr>
<td>VM3</td>
<td>0.7</td>
<td>0.7</td>
<td>0.3</td>
<td>0.6</td>
</tr>
<tr>
<td>VM4</td>
<td>0.9</td>
<td>0.9</td>
<td>0.2</td>
<td>0.6</td>
</tr>
<tr>
<td>VM5</td>
<td>0.7</td>
<td>0.6</td>
<td>0.3</td>
<td>0.5</td>
</tr>
<tr>
<td>VM6</td>
<td>1.0</td>
<td>1.0</td>
<td>0.2</td>
<td>0.6</td>
</tr>
<tr>
<td>VM7</td>
<td>0.5</td>
<td>0.4</td>
<td>0.6</td>
<td>0.5</td>
</tr>
<tr>
<td>VM8</td>
<td>0.1</td>
<td>0.1</td>
<td>1.0</td>
<td>0.5</td>
</tr>
<tr>
<td>VM9</td>
<td>0.5</td>
<td>0.5</td>
<td>0.6</td>
<td>0.5</td>
</tr>
<tr>
<td>VM10</td>
<td>1.0</td>
<td>1.0</td>
<td>0.1</td>
<td>0.6</td>
</tr>
</tbody>
</table>

...and user tasks. All QoS value of user tasks should use rand() to get a random number ranged 0-1. Look at Table 3 for VM QoS parameters.

Table 4 is the simulation result of algorithms of QDDN and S-CSRSA. The above experiment result shows that user satisfaction apparently improves with the algorithm S-CSRSA than that of QDDN. This is because, when alloting tasks, algorithm of QDDN only takes distance difference between user tasks and resource into account rather than other elements. According to S-CSRSA algorithm, VMs can get the highest satisfaction to user tasks. In order not to make QoS required tasks occupy resources with the higher QoS capability, S-CSRSA algorithm allots user tasks to VMs with smallest distance to itself and with the highest satisfaction. When computing distance, S-CSRSA algorithm fully considers every dimension’s service capability difference of the resource and gives different metrics to different parameters to show the better match between user tasks and VMs.

Figure 5 shows that VM utility rate of S-CSRSA algorithm keeps around 54% on average while that of QDDN algorithm only remains around 32% on average, because a lot of resources in net blocks can make a task get the highest satisfaction at the same time. When resource is unused, S-CSRSA algorithm moves the Not
Table 4: Comparison of task user satisfaction

<table>
<thead>
<tr>
<th>Task No.</th>
<th>QDDN</th>
<th>S-CSRSA</th>
<th>Improved</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>0.845</td>
<td>0.97</td>
<td>14.793</td>
</tr>
<tr>
<td>20</td>
<td>0.823</td>
<td>0.963</td>
<td>0.965</td>
</tr>
<tr>
<td>30</td>
<td>0.844</td>
<td>0.9695</td>
<td>14.870</td>
</tr>
<tr>
<td>40</td>
<td>0.8985</td>
<td>0.686</td>
<td>15.564</td>
</tr>
<tr>
<td>50</td>
<td>0.894</td>
<td>0.972</td>
<td>13.871</td>
</tr>
<tr>
<td>60</td>
<td>0.8635</td>
<td>0.973</td>
<td>12.681</td>
</tr>
<tr>
<td>80</td>
<td>0.8595</td>
<td>0.973</td>
<td>13.205</td>
</tr>
<tr>
<td>100</td>
<td>0.8705</td>
<td>0.9745</td>
<td>11.947</td>
</tr>
</tbody>
</table>

Fig. 5: VM Utility Rate Comparison between S-CSRSA and QDDN

Fig. 6: Loading Equilibrium Comparison between S-CSRSA and QDDN

implemented tasks to VMs with the guaranteed satisfaction to be operated. While QDDN algorithm does not take this into account, so its resource utility rate is obviously lower than that of S-CSRSA algorithm. Due to the said reason, the total task finishing time of S-CSRSA algorithm is shorter than that of QDDN algorithm.

With the increase of tasks numbers, the difference of total task finishing time is bigger and bigger. The resource loading equilibrium from the both algorithms is indicated in Fig. 6. When allotting tasks by S-CSRSA algorithm, resource capability is fully used and operating difference of various resources is small, therefore, loading equilibrium of S-CSRSA is clearly higher than that of QDDN. When allotting tasks by QDDN algorithm, it causes many tasks allotted to a single resource to be operated, so its loading equilibrium is lower.

CONCLUSION

This study proposes a cloud simulation task scheduling model supporting QoS and introduces S-CSRSA algorithm in details on the bases of this model. Through algorithm example and cloud simulation platform testing based on CloudSim, S-CSRSA algorithm guarantees satisfaction of user tasks and good performance in resource utility rate and loading equilibrium.

REFERENCES


