International Journal of Control and Automation
Vol.7, No.4 (2014), pp.137-152
http://dx.doi.org/10.14257/ijca.2014.7.4.13

QoS PreferenceAwareness Task Scheduling Based on PSO and AHP Methods

Juan Wang*, Fei Li and Luqiao Zhang

School of Network Engineering, Chengdu University of Information Technology, Chengdu, China

{wangjuan,lifei, zhanglq}@ cuit.edu.cn*

Abstract

Most existing task scheduling algorithmsfail to aware users' QoS preference and result in low user satisfaction rate for they do not reflect users'QoS requirements. We classify QoS factors into four main QoS class which users understand well and can describe their important level, andintroduce AHP method to help user decide the class weight and avoid judgment logical error.Then, we improve existing standard PSO scheduling by use above AHP based different weights for different QoS classes to make PSO haveQoS preference awareness ability.The simulations show our method gets obviously higher user satisfaction rate and maintains the efficiency at the same time.Finally, the simulations also point out that the hierarchical scheduling is necessary to avoid the common tasks take over the special resource.

Keywords: *QoS Preference; Analytic Hierarchy Process; Task Scheduling; Cloud System; PSO*

1. Introduction

With the development of the internet technology,the data obviouslyincrease and make it is difficultfor individualto store, manage and sharethese mass data. Cloud storage service is proposed to deal with these problems.Cloud storage systemlets users easily store, access, and protect their datawithout worrying about maintenance, scaling up or down or hardware and firmware upgrades.With these advantages, cloud storage becomes a hot point in cloud computing area.

Task schedulingalgorithms play a key role in cloud storage system. However, existing algorithms ignorethe different QoS requirements of task and result in low user satisfaction rate. In order to address the problem, task schedule have to satisfy multi-QoS requirements and have QoS preference awareness ability, rather than only improve the throughputof whole system.

In this paper, we study the QoS preference awareness task scheduling algorithm in cloud storageenvironment.We found that the reasonwhy heuristicalgorithms, such as particle swarm optimization (PSO), failed toaware QoS preference is that they use the fixed weighted fitnessfunction. It is nature way to improve them is making their weights of fitnessfunctionadapt fortasks QoS requirements change.But the QoS factors are too professional to users to understand and give weights for them. And the weights decided by researchers or technicians do not reflect the users' real demand. So the improve difficulty is how to make the weights of fitnessfunction reflect the users'requirements. We introduce the analytic hierarchy process (AHP) method to help user decide the weights for its consistency check avoid subjective logic errors.

ISSN: 2005-4297 IJCA
Copyright ⓒ 2014 SERSC

The remainderis organized as follows: Section 2 introduces the related works especially the QoS guided heuristic algorithms task scheduling method. Section 3 simply reviews the existing PSO scheduling method and point out its shortcomings. Section 4 describes the details of our methodthat introduces AHPinto PSO algorithm to help it adapt for tasks QoS requirements change.The simulations and comparisons analysis are presented inSection 5. Finally, a short conclusionis given in Section 6.

2. Related Works

In cloud system, whether cloud computing system or cloud storage system, million tasks waiting for dispatching. Scheduling these mass tasks is a challenge to cloud environment.Theheuristic algorithmsis suggested to find reasonably solutions, such as ant colony based scheduling algorithm [1], Genetic algorithm (GA) based scheduling algorithms [2, 3], Simulated annealing based scheduling algorithms[4], particle swarm optimization(PSO) [5], andso on. They evaluate multi-QoS factors by fitness function by taking QoS factors as fitness functionvariable. However, the aimsof thesescheduling algorithmsare getting higher throughput and making load balance which make the weights of fitness functions are defined for these aims too. And once the weights are decided, they will not change through the all scheduling process.These scheduling methods consider from the aspect of system not from the aspect of user, namely the factor they used is almost system parameter. In addition, most of their aims are system efficiency improvement too.

We find these methods can not satisfy users' QoS requirementsand aware QoS preference. In order to address these problems, following we improve existing PSO based algorithm by integrating AHP method.

3. The PSO Basedtask Scheduling Introduction

Particle Swarm Optimization (PSO) is a promising metaheuristic approachfor solving diverse task scheduling problems as well as other application problems [6]. The simulations in reference [7] shown that PSO algorithmminimize makespan and obtained higher performance than other compared techniques did. So we choose the PSO as the basic task scheduling algorithm.

3.1.Standard PSO introduction

In PSO, a swarm of particles spread in the search space and the position of a particle presents a solution, namely a task scheduling scheme in cloud storage system. Every particle may move to a new position depends on the local experience and the global experience heading toward the global optimum. The standard PSO is initialized with a population of random positioned particles and searches for the best position with best fitness. The details as follows:

The location of the *i*th particle is represented as $X_i = (x_{i1},\ldots,x_{id}, \ldots, x_{iD})$. The best previous position (which giving the best fitness value) of the *i*th particle is recorded and represented as $P_i=(p_{i1},\ldots, p_{id}, \ldots, p_{iD})$, which is also called *pbest*. The index of the best *pbest* among all the particles is represented by the symbol *g*. The location P_g is named*gbest*. The velocity for the *i*th particle is represented as $V_i = (v_{i1},\ldots, v_{id}, \ldots, v_{iD})$. The particle swarm optimization concept consists of, at each time step, changing the velocity and location of each particle toward its *pbest* and *gbest* locations according to the equations (1) and (2), respectively:

$$v_{id} = w * v_{id} + c_1 * rand() * (p_{id} - x_{id}) + c_2 * rand() * (p_{gd} - x_{id}) \quad (1)$$

 Copyright ⓒ 2014 SERSC

$x_{id} = x_{id} + v_{id}$ (2)

Where w is inertia weight, c_1 and c_2 are acceleration constants, and *rand()* is a random function in the range [0, 1].

The process for implementing PSO in cloud storage task schedule is as follows:

1. Set iteration generation It_c=1. Initialize a population which including m particles. For the ith particle, it has random location X_i in specified space and for the dth dimension of V_i, v_{id} = *Rand2*()* $v_{max,d}$, where *Rand2*() is a random value in the range [-1, 1];

2. Evaluate the fitness for each particle;

3. Compare the evaluated fitness value of each particle with its *pbest*. If current value is better than *pbest*, then set the current location as the *pbest* location. Furthermore, if current value is better than *gbest*, then reset *gbest* to the current index in particle array;

4. Change the velocity and location of the particle according to the equations (1) and (2);

5. It_c=It_c+1, loop to step 2 until fitness is met Expect Fitness Function Value f_e or It_c is achieve Max value.

3.2. The existing matrix of PSO

For the cloud storage is a special cloud system different from cloud computing ones.In cloud computing system, one task which ask some host to computing, can be assigned to every node and the only problem is the execution time, namely the low efficiency nodes cost longer time and the high efficiency nodes cost shorter time. But in cloud storage system, the task always asks remote node transfer it data, and we can't ask a node offer the data it doesn't have! In another words, for certain task of cloud storage system can only be assigned to some of node instead of every nodes.

As our former study, the traditional PSO of cloud computing using in cloud storage create many meaningless solutions. So **Exist Matrix** [5] has to be used to create meaningful solutions. A improved PSO method is given in references [5] which is our foundation of this paper. We create Exist Matrix by the file distribution table which storage in core nodeindicates which node has which data. And then in PSO scheduling, the initialize scheduling vector and the newscheduling created vector have to follow the Exist Matrix. By this way, the meaningless solutions are limited.

3.3. Fitness function definition for multi-QoS

Assume there are m QoS factors which users are interest in.Let Q be the QoS vector,$Q=\{q_1,q_2,\ldots,q_m\}$. For every node in cloud storage system has its own QoS vector at certain moment, denote the QoS vector of node $i$as $Q_i = \{q_1^i, q_2^i, ..., q_m^i\}$.

For different QoS factor has different importance for use. We use different weight to describe their important level. Let W be the weight vector for Q, denote as $W=\{w_1,w_2,\ldots,w_m\}$. And then the fitness function definition as

$f_i = w_1 q_1^i + w_2 q_2^i + ... + w_m q_m^i, \sum_i W_i = 1$ (3)

By this way, we integrate multi-QoS factor into one fitness function. For every scheduling scheme created by PSO, we evaluate their fitness value, and choose a max valueor min

Copyright ⓒ 2014 SERSC

depends on the definition, here we choose the max valueone. So the QoS factors $q_i$is the bigger the better.

As mentioned in Section 2, once the weights are decided, they will not change through the all scheduling process which is the reason why existing heuristicalgorithmfail to aware task's QoS preference. In our opinion, the weights have to adapt to QoS changes of different tasks.

Then the next problem is how to determine the weight for each factor to describe different importance of them. Existing methods mostly set weights by expert which cannot really reflect the user's preferences. However, let users themselves to decide the weights is also difficult for users are not familiar with technical QoS factors.

Following we introduce AHP method to deal with these problems.

4. AHP Integrated to aware QoS preference

We found that users are lack of knowledge of technical terms. For example, the term "delay variation" user may understand "delay" but confuse"variation". Then they cannot compare the importance between "delay variation" and "Mean Time Between Failures (MTBF)" for they confuse these meanings. So let users themselves directly to compare technical terms is not reasonable in our opinion. In order to address this problem, we classify technical terms into QoS classes which users totally understood.

Past experience has taught us that user consider only four things: the cost, the efficiency, the quality and the security which receive more attentions in recent years. We class various QoS factors into these four main classes which users understand well.In order to avoid logical error, we introduce AHP method to help describing the QoS classes' importance.

4.1. The analytic hierarchy processintroduction

The analytic hierarchy process (AHP) [8] is a structured technique for organizing and analyzing complex decisions. Based on mathematics and psychology, it was developed by Thomas L. Saaty in the 1970s and has been extensively studied and refined since then.

Using AHP, firstly decompose the decision problem into a hierarchy of more easily comprehended sub-problems, each of which can be analyzed independently.For weights decide application, it is decompose the decision problem into a hierarchy of independent variables.Once the hierarchy is built, the decision makers systematically evaluate elements couple by couple. The AHP converts these evaluations to numerical values that can be processed and compared over the entire range of the problem. A numerical weight is derived for each element of the hierarchy which we need, allowing diverse and often incommensurable elements to be compared to one another in a rational and consistent way. This capability distinguishes the AHP from other decision making techniques.

The main steps as follows:

1. *For certain objective, build judgment matrix A, AHP compares every two factors at a time. The element x_{ij} (i row and j column)indicate evaluated ranking of $x_i$compare tox_j.*

The evaluated ranking is reflected by the rationality of ranking scale. A ranking scale is a scale of numbers that indicates how many times more important or dominant one element is over another with respect to the criterion or property with respect to which they are compared. We use the standard Saaty [8] 9 level scales as table 1 shows.

 Copyright ⓒ 2014 SERSC

Table 1.Saatyscale of absolute numbers

Intensity ofImportance	Definition（Compare factor itoj）
1	Equal Importance
3	Moderate importance
5	Strong importance
7	Very strong or demonstrated importance
9	Extreme importance
2, 4, 6, 8	The intermediate value of above two adjacent judgment
The reciprocal of 1 ~ 9	$a_{ji}=1/a_{ij}$， $a_{ii}=1$

2. Does consistency check for the judgment matrixA.

The consistency of judgment matrix is important and it asks ratings should be transitive that means: If A is better than B, and B is better than C, then A must be better than C.Further,ratings should be numerically consistent: if for factor P, M and G, where P = 3M and P = 5G, then we have 3M = 5G, namelyM = (5/3) G. The lattercondition is strong constraint, and in practice we ask the matrix A have to satisfy the former one at least.If the matrix is inconsistent that means user consider A is better than B, and B is better than C, but the A isn't better than C! It is anabsolute logic error which we cannot accept, and then we have to rebuild the matrix A.

If the matrix A is consistency, assume λ is the largest eigenvalue of A, then the normalizationvector$\{w_1, w_2, ..., w_n\}$, $\sum_{i=1}^{n} w = 1$of λ is our needed weights.

The consistency checks process as follows, let *n*be the factor number:

Define ConsistencyIndex(CI): $CI = \frac{\lambda - n}{n-1}$ (4)

if CI=0， matrix A has complete consistency； if CI → 0， matrix A has acceptedconsistency；the bigger CI the more inconsistency。

Define ConsistencyRatio (CR)： $CR = \frac{CI}{RI}$ (5)

If $CR = \frac{CI}{RI} < 0.1$,then we think the consistency is in accepted range. The values of RI as Table 2 shows.

Table 2. Random consistency index (RI) [8]

n	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
RI	0	0	0.58	0.90	1.12	1.24	1.32	1.41	1.45	1.49	1.51

4.2. The AHP based weight defined for different class tasks

As mentioned above, there are four QoS class: Cost, Time,Quality and Security. Then we classify tasks into various task classes by different QoS requirements.We divided the tasks into the following six classes:

(1) Cost class(C1): in this class, tasks only take care of cost.

Copyright ⓒ 2014 SERSC

(2) Efficiencyclass(C2): in this class, tasks only take care of time.

(3) Quality class: in this class(C3), tasks only take care of transmission quality.

(4) Efficiency and Qualityclass(C4): in this class, tasks take care of time and quality at the same time. Such as the transmission of important and emergency files.

(5) Quality and Securityclass(C5): in this class, not only have to ensure the transmission quality, but also cannot reveal any information.

(6) Quality, Security and Efficiencyclass(C6):Based on C5,in this class ask efficiency and the same time.

In this application, the QoS requirementsof task class are the AHP target, and the four QoS classes are the evaluation elements.

Here, we use C1 as the example to show how the weights define. As C1 description, users of C1 tasks only care the cost, so the Cost factor C is extreme important than other three factors, so the number of C compare to other factors is 9 as the Table 1 given. Then the other three factors themselves are equal for users do not care them and the number 1 is used to indicate this situation. So we get following judgmentmatrix for C1.

$$c1 = \begin{bmatrix} & C & E & Q & S \\ C & 1 & 9 & 9 & 9 \\ T & 1/9 & 1 & 1 & 1 \\ Q & 1/9 & 1 & 1 & 1 \\ S & 1/9 & 1 & 1 & 1 \end{bmatrix}$$

Next we have to check the consistency of C1 judgmentmatrix to avoid logical error. The consistency check results are: CI=-0.00025, CR=-0.00028<1. So the c1matrix satisfy the consistency check. And the λ is 3.9992,weight vector is [0.7500 0.0833 0.0833 0.0833] which means for class C1 the weights of PSO fitnessfunction are 0.7500 for Cost factor, 0.0833 for other three factors. The formula (3) is redefined as:

$$f_i = w_c Cost + w_e \text{Efficiency} + w_q \text{Quality} + w_s \text{Security} \quad (6)$$

The weight vector is defined as [W_c, W_e, W_q, W_s].

Classes C2 to C6 use similar process, we give the judgmentmatrixs, and consistency check results and weight vectors are summarized as Table 3.

$$c2 = \begin{bmatrix} & C & E & Q & S \\ C & 1 & 1 & 0.111 & 1 \\ E & 1 & 1 & 0.111 & 1 \\ Q & 9 & 9 & 1 & 9 \\ S & 0.111 & 0.111 & 0.111 & 1 \end{bmatrix}$$

 Copyright ⓒ 2014 SERSC

$$c3=\begin{bmatrix} & C & E & Q & S \\ C & 1 & 0.111 & 1 & 1 \\ E & 9 & 1 & 9 & 9 \\ Q & 1 & 0.111 & 1 & 1 \\ S & 1 & 0.111 & 1 & 1 \end{bmatrix}$$

$$c4=\begin{bmatrix} & C & E & Q & S \\ C & 1 & 0.143 & 0.143 & 1 \\ T & 7 & 1 & 1 & 7 \\ Q & 7 & 1 & 1 & 7 \\ S & 1 & 0.143 & 0.143 & 1 \end{bmatrix}$$

$$c5=\begin{bmatrix} & C & E & Q & S \\ C & 1 & 1 & 0.111 & 0.111 \\ E & 1 & 1 & 0.111 & 0.111 \\ Q & 9 & 9 & 1 & 1 \\ S & 9 & 9 & 1 & 1 \end{bmatrix}$$

$$c6=\begin{bmatrix} & C & E & Q & S \\ C & 1 & 1/9 & 1/9 & 1/9 \\ T & 9 & 1 & 1 & 1 \\ Q & 9 & 1 & 1 & 1 \\ S & 9 & 1 & 1 & 1 \end{bmatrix}$$

Table3. Consistency check results and weight vectors

	λ	CI	CR
c1	3.9992	-2.50E-04	-2.81E-04
c2	3.4898	-0.1701	-0.1911
c3	3.9992	-2.50E-04	-2.81E-04
c4	4.001	3.33E-04	3.74E-04
c5	3.999	-3.33E-04	-3.75E-04
c6	3.9992	-2.50E-04	-2.81E-04

4.3. The weight definition of inner class

The final problem is how to weight the "technical QoS factors(TQFs)" in the QoS classes(QCs).We transfer TQFs into QCs as following operations.

4.3.1. Normalization:

We can see these QoS factors have totally different physical significance and also various units which prevent us to evaluate them in one function. The utility function is proposed to deal with this problem. A utility function can map a QoS factor value to a real number. Our

Copyright ⓒ 2014 SERSC

utility function is normalization function which compares to the max and min value and gets a real number between 0 and 1. The real number is independentof factors' unit and range.

If the factor $q_i$is efficientattribute which means the bigger value, the better quality of factor q_i, the utility function of factor q_i is[9,10]:

$$q_i = \begin{cases} (q_i - \min q_i)/(\max q_i - \min q_i), & \text{if } \max q_i \neq \min q_i \\ 1, & if \ \max q_i = \min q_i \end{cases} \quad (7)$$

If the factor q_i is cost attribute which means the smaller value, the better quality of factor q_i, the utility function of factor q_i is:

$$q_i = \begin{cases} 1, & \text{if } \max q_i = \min q_i \\ (\max q_i - q_i)/(\max q_i - \min q_i), & \text{if } \max q_i \neq \min q_i \end{cases} \quad (8)$$

where$\min q_i$is the min value of $q_i$and max$q_i$is the max value of q_i.

4.3.2.Set weights of TQFs:

The weights for QoS factors inside classes are decided by experts'knowledge and by this way can use experts'experience.

The $w_i$is decided by "Optimal Sequence Method" to reduce subjective judgment as follows:

Firstly, determine scale is described by 1，2，3，4，5 levels, the bigger number indicate the more importance. Then compare factors couple by couple, if one factor's importance level is set to 5, then the another one's importance level is 0; if one is 3, then another is 2. By this way, the Judgement Matrix is built which is a $n \times n$squarematrix, n is the number of factors. And the value w_{ij} (row i, column j) is the importance of factor i compare to j, such as w_{ij}=3, then opposite w_{ji}=5-3=2 which indicate the importance of factor j compare to i, $i \neq j$. The sum of rows $\sum_i w_i$ indicate the importance of factor i in all factors. Take the sum of all rows and columns $\sum_j \sum_i w_{ij}$ as the denominator which indicate the importance level of factor i in all factors, and the $\sum_i w_i$ as numerator, then the quotient is the weight of the factor i:

$$W_i = (\sum_i w_i)/(\sum_j \sum_i w_{ij}) \quad (9)$$

4.4. AHP-PSO task scheduling algorithm

On the basis of the above works, we improved existing PSO based task scheduling algorithmby adjusting the weights of fitness function according to different class tasks. For example, for a Quality class task, then the PSO fitness function uses Quality weights as formal (6). The class weights defined process as Figure 1 shows.

 Copyright ⓒ 2014 SERSC

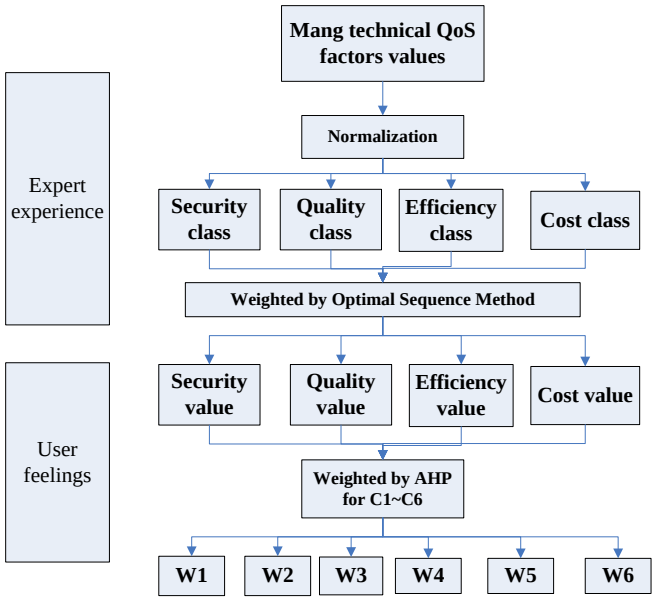

Figure 1. Two levels weights definition process

Then the AHP-PSO task scheduling process as the Figure 2 shows. When a batch tasks are arrive in unit time, firstly classify these tasks into four QoS classes. Secondly, for the different QoS class tasks use according weights for fitness function. Finally, do the standard PSO scheduling process as references [5] does.

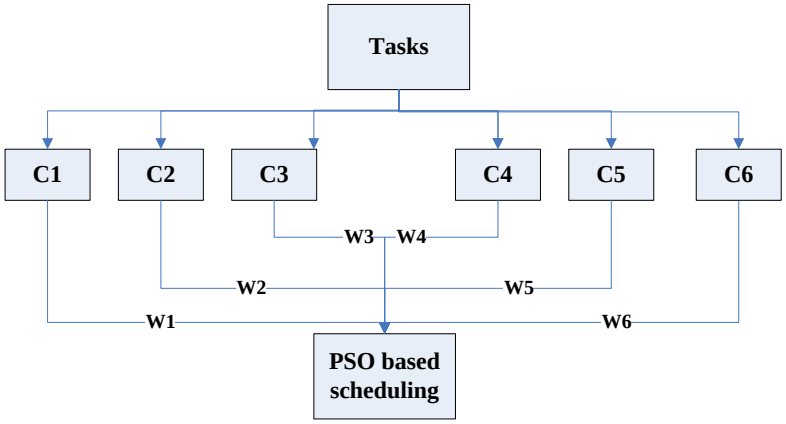

Figure 2. AHP-PSO task scheduling process

5. Simulations and Analysis

5.1. Simulation environment

We developed a Cloud Storage Simulation System (CS3) by Matlab7.0. This system includes three main modules: the Task Scheduling Module, theUpdate Module, and theEvaluation Module as the Figure 3 shows.

Copyright ⓒ 2014 SERSC

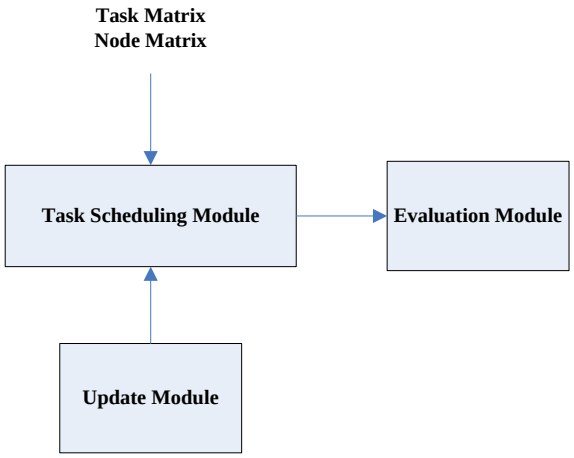

Figure 3. The simulation system CS^3

The row of Task Matrix task vector which contains task size and task' QoS requirements, denoted as task(Tsize,q_1,q_2,…,q_n).If task does not require certain QoS factor, the according value set to be NULL.

The Nodes Matrix contains the information of nodes, such as the node QoS similar to task vector and the nodes relation describe weather two nodes are connected or not, denoted as node vector(node$_i$-node$_j$, q_1,q_2,…,q_n),where if i=j, the q describe the node$_i$'QoS factor, and if i≠j, the q describe the connect QoS factor between node$_i$ and node$_j$.

The system takes Task Matrix and Node Matrix as the input. Then the "Task Scheduling Module"dispatch task.

The Update Module update the QoS factor value of Nodes Matrix after one scheduling scheme is applied.

Finally the Evaluation Module evaluates the scheme effect by user satisfaction rate. If task a dispatched to node b, then we compare the task vector and node vector, only all the QoS factor of the task are satisfied by node, the scheduling scheme is thought satisfy the task.User satisfaction rate is defined as:

$$USR(\%) = \frac{satisfied\ \ task\ \ number}{totall\ \ task\ \ number}\% \qquad (10)$$

Tasks group created by CS^3 have 1000 tasks, and the number of C1 tasks>>C2 tasks>C3 tasks >C4 tasks>C5 tasks>C6 tasks. In practice, common tasks(C1) far more than special task is a common phenomenon. There are 20 resource nodes of the simulation system, and 10 of them is cheap cost, 5 of them can ensure quality,3 of them can ensure both quality and security(have encryption function). Following simulations, if not addition explains, using the same task group.

5.2.Simulation result and analysis

In this section, we compare the USR, task completion time,execution time of standard PSO and our AHP-PSO to see the effect of real-time adjust the weights of the fitness function.In order to avoid some random interference, every simulation is repeated three times.

 Copyright ⓒ 2014 SERSC

Simultiaon1: the USR comparison

The USR comparison result as the Figure 4 shows.

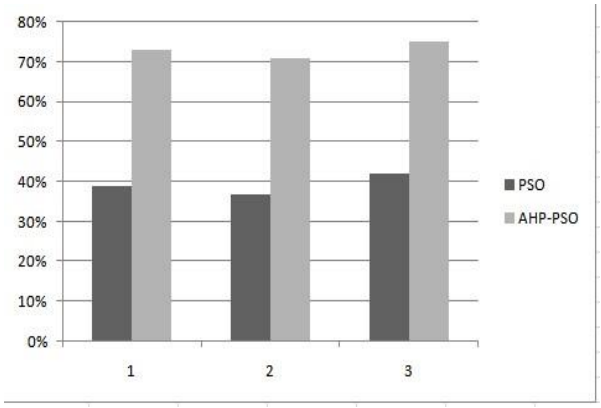

Figure 4. The USR comparison of standard PSO and AHP-PSO

We can see AHP-PSO algorithm has higher USR than standard PSO, from about 40% up to about 70%. This demonstratesthe weights of fitness function change with different class tasks can make the standard PSO has QoS preference awareness ability.

However, the USR is not as high as our expected. We try to improve USR by change the AHP weights, however, the effect is not obvious. Then we check the schedulingresults find that many special tasks (like C5,C6 tasks) are dispatched to nodes which cannot satisfy their QoS requirement. Because many C1 tasks arrived firstly and are dispatched to all nodes make there aren't suitable nodes dispatching to special tasks.

In order to address this problem, a nature method is hierarchical scheduling. Let privilege levels of tasks are:C6>C5>C4>C3>C2>C1. We schedule the higher level tasks firstly and lower level tasks share the remainder resources as the Figure 5 shows.

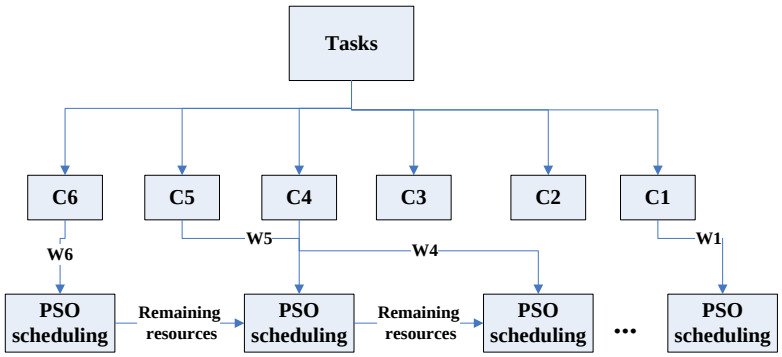

Figure 5. Hierarchical task scheduling

The hierarchical schedulingeffect as the Figure 6 shows. By this way, we get satisfyingUSR about 90% demonstrate the hierarchical scheduling is necessary. So our AHP-PSO method becomes to hierarchical AHP-PSO.

Copyright ⓒ 2014 SERSC

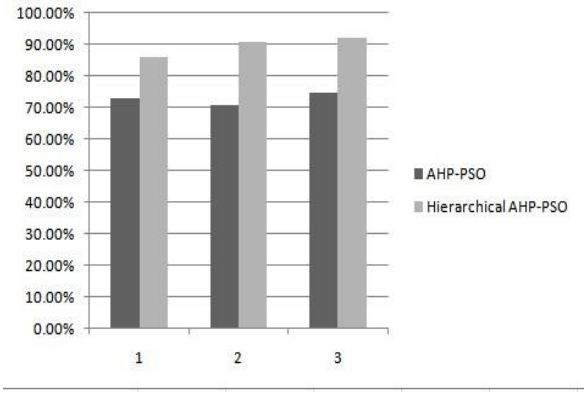

Figure 6. The USR comparison of AHP-PSO and hierarchical AHP-PSO

Simulation 2: The execution time comparison

Then the execution time of standard PSO and our hierarchicalAHP-PSO is as Figure 7 shows. The execution time means the time between the algorithms receive the input and give the scheduling solution.

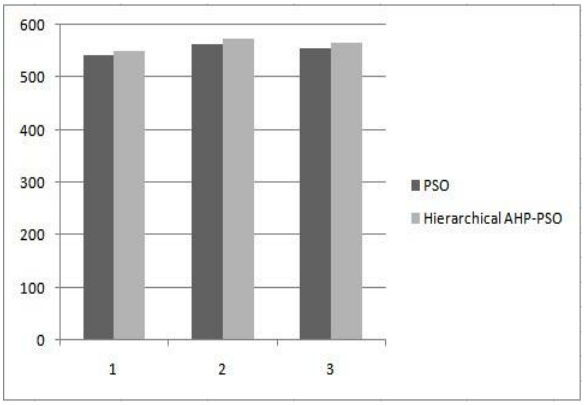

Figure 7. The execution time comparison（unit：millisecond）

We can see from Figure 7, the hierarchicalAHP-PSO cost more time than standard PSO. The reasonis our hierarchicalAHP-PSO indeed does more work than standard PSO that classify tasks into different classes and adjustsfitnessfunctionweights for classes.

However, addition operation dose not cost too much time and with the scale of tasks increase, the weight adjustment' impact is became less. Here we use the proportion of weight adjustment time in all execution time as the evaluationvalue to compare the efficiency of hierarchicalAHP-PSO and PSO.

Let the hierarchical AHP-PSO execution time is T_a; the PSO execution time is T_p; then the classification process and weight adjustment time T_w can be approximatecalculated as $T_w = T_a - T_p$, and the impact evaluation ratio is defined as:ratio= T_w / T_p.The relation of task scale and the ratio as the Table 4 shows. We can see when we schedule 1000 tasks at one time, the ratio only 1.55%.

 Copyright ⓒ 2014 SERSC

Table 4. The effect of classification and weight adjustment

task scale	10	100	1000
PSO	4	56	645
Hierarchical AHP-PSO	5	58	655
ratio	25.00%	3.57%	1.55%

Finally, we compare the task completion time of the two algorithms as figure 8 shows.

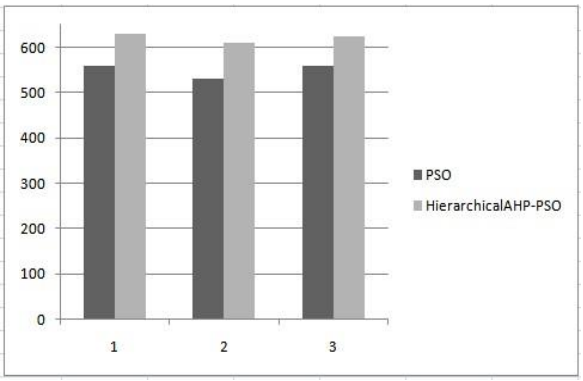

Figure 8. The comparison of task completion time（Unit：millisecond）

We find out that the completion time of hierarchicalAHP-PSO is longer than PSO, but also not so much, only increase about 10%. The addition time is because the hierarchicalAHP algorithm is satisfy users' QoS requirementfirstly, and then choose the least completion time scheduling schema. However, the standard PSO is direct to find the least completion time scheduling schema. As the idea of SaaS (Software-as-a-service), every application functions in cloud system is given to uses as service. So sacrifice some efficiency for USR is worth in our opinion.

6. Conclusion

In this paper, we point out that existing task scheduling algorithms have following main shortcomings:1) These algorithms almost consider from the aspect of system not from the aspect of user which result in low user satisfaction rate;2) These algorithms lack the QoS preference awareness ability;

The first problem is because users don't understand the QoS technicalterms and cannot describe their QoS requirements oftechnicalterms. We classify QoS factors into four main QoS class which users understand well. By this way, users can give judgment of QoS class important. In order to help users avoid logical error, the AHPis suggested to be the weights decide method.

Then we improve existing standard PSO scheduling by use different weights for different QoS classes to make PSO have QoS preference awareness ability.

The simulations show our method gets obviously higher user satisfaction rate and maintains the efficiency at the same time.

Finally,the simulations also point out that the hierarchical scheduling is necessary toavoid the common tasks take over the special resource.

Copyright ⓒ 2014 SERSC

Acknowledgments

This work was supported in part by grants from:

1) The Application basic research project of Sichuan Province (2013JY0064);
2) The Scientific Research Foundation of CUIT (Chengdu University of Information Technology), No.KYTZ201121);
3) The Research Fund Young of Middle-Aged Academic Leaders in CUIT(J201107);
4) The Sichuan Province Science and Technology Support Plan(No.2011GZ0195);
5) The Scientific Research Project of Education Department in Sichuan Province(No.10ZB093).

References

[1] X. Song, L. Gao and J. Wang, "Job scheduling based on ant colony optimization in cloud computing", IEEE, (**2011**).
[2] C. Zhao, S. Zhang, Q. Liu, J. Xie and J. Hu, "Independent Tasks Scheduling Based on Genetic Algorithm in Cloud Computing", IEEE, **(2009).**
[3] Y. Ge and G. Wei, "GA-Based Task Scheduler for the Cloud Computing Systems", IEEE, **(2010).**
G. Guo-ning, H. Ting-lei and G. Shuai, "Genetic simulated annealing algorithm for task scheduling based on cloud computing environment", IEEE, **(2010).**
[4] J. Wang, F. Li and L. Zhang, "Task scheduling algorithm in cloud storage using PSO with limited solution domain", Applicaiton Research of Computers , vol. 30, no. 1, **(2013)**, pp. 127-129,154.
[5] R. –M. Chen and C. –M. Wang, "Project Scheduling Heuristics-Based Standard PSO for Task-Resource Assignment in Heterogeneous Grid", Hindawi Publishing Corporation Abstract and Applied Analysis, **(2011)**, Article ID 589862, 20 pages,doi:10.1155/2011/58986.
[6] H. Izakian, A. Abraham and V. Snášel, "Metaheuristic based scheduling meta-tasks in distributed heterogeneous computing system", Sensors, vol. 9, **(2009),** pp. 5339-50.
[7] T. L.Saaty, "Decision making with the analytic hierarchy process", Int.J.Services Sciences, vol. 1, no. 1, **(2008)**, pp. 83-98.
[8] M. Alrifai and T. Risse, "Combining global optimization with local selection for efficient QoS-aware service composition", WWW '09 Proceedings of the 18th international conference on World wide web, ACM New York, NY, US, pp. 881-890.
[9] T. Yu, Y. Zhang and K. J. Lin, "Efficient algorithms for Web services selection with end-to-end QoS constraints", ACM Transactions on the Web (TWEB), ACM New York, NY, USA, vol. 1, is. 1, May **(2007)**.

Authors

Juan Wang

Juan Wang was born in China in 1981 and received her B.S. degreeof computer sciencein 2003, and theM.S. degree of Computer Architecture and Ph.D degree of Information Security from University of Electronics and Technology of China (UESTC) in 2006 and 2010.And being a visiting scholar at University of North Carolina at Charlotte(UNCC) from 2007.9 to 2008.9.Her research interests include network security and cloud storage. Email: wangjuan@cuit.edu.cn.

 Copyright ⓒ 2014 SERSC

Fei Li

Fei Li wasborn in China in 1966. He received his M.S. degree of computer science in 1993.7 from Chengdu University of Science and Technology. Now he is the dean of thenetwork engineering school of Chengdu University of Information Technology. His research interests include grid computing and the Internet of things technology. Email: lifei@cuit.edu.cn

Luqiao Zhang

Luqiao Zhang was born in China in 1983,received B.S. degreeand M.S. degree in computer science from University of Electronics andTechnology of China (UESTC) in 2003 and 2006, respectively. He is now a Ph.D.candidate in University of Electronics and Technology of China (UESTC). Hisresearch interests include wireless sensor network and mobile wireless P2P.Email:zhanglq@cuit.edu.cn

Copyright ⓒ 2014 SERSC

Copyright ⓒ 2014 SERSC