PSO Based Hierarchical Task Scheduling with QoS Preference Awareness in Cloud Storage Environment

Juan Wang
School of Information Security Engineering, Chengdu University of Information Technology, Chengdu, China.
Email: wangjuan@cuit.edu.cn

Fei Li Luqiao Zhang and Yuanyuan Huang
School of Information Security Engineering, Chengdu University of Information Technology, Chengdu, China.
Email: {lifei, zhanglq, hy}@ cuit.edu.cn

***Abstract*—Most existing task scheduling algorithms in cloud storage fail to aware users ' QoS preference. In addition, these algorithms result in low user satisfaction rate for they do not consider the characteristics of cloud storage. In order to address these problems, the "optimal order comparison method" is used to aware users ' QoS preference, and also helps experts use their professional knowledge to decide the weight of QoS classes. We redefined the fitness function of the particle swarm optimization (PSO) algorithm by using these weights and proposed the "PSO based hierarchical task scheduling with QoS preference awareness: PSO-HQoSPA" algorithm. By consider both user and expert experience, the method can aware users' QoS preference and deal with multiple QoS requirements. The simulation results show our method achieved acceptable user satisfaction rate, and the same time maintains the efficiency as traditional PSO based method.**

***Index Terms*—multi-QoS constraints; QoS preference; task scheduling; cloud storage; particle swarm optimization**

I. INTRODUCTION

With the development of the Internet of things, the amount of data is continually increasing and thus it is difficulty to storage and share. To address this need, cloud storage services have been proposed. Therefore, cloud storage providers can offer lower cost by off-loading the burden of storage management and shielding enterprises from other costs as well, such as storage and network hardware changes. Cloud storage providers offer huge capacity cost reductions, the elimination of labor required for storage management and maintenance, and immediate provisioning of capacity at a very low cost per terabyte[1]. With these advantages, cloud storage becomes a key point in cloud computing area.

How to schedule mass tasks is a key challenge in both cloud computing and cloud storage systems. Task scheduling algorithm, to some extent, determines the efficiency of the cloud system. There are already many task scheduling algorithms in cloud computing system but few in cloud storage system. However, they don't reflect the characteristics of cloud storage and they are lack for QoS preference awareness ability which is important for users. In practice, we found users have various QoS requirements. Some of QoS requirements are asked to be satisfied at the same time which is called Multi-QoS constraints. Then some of these QoS requirements is important than others to users which is called the QoS preference.

Due to above shortcomings, the existing scheduling algorithms have to be improved to fit the characteristics of cloud storage system. Furthermore, task scheduling have to satisfy multi-QoS requirements and have QoS preference awareness ability, rather than only improve the throughput of system.

In this paper, we study the QoS preference awareness task scheduling algorithm in cloud storage environment. The rest of this paper is organized as follows: Section 2 introduces the related works especially the QoS guided task scheduling methods in cloud computing environment for there are few similar works in cloud storage environment. Section 3 describes the details of our method that introduces hierarchical weighted with optimal order comparison method into PSO algorithm. By these ways, our task scheduling algorithm can satisfy multiple QoS requirements and QoS preference. The simulations and comparisons analysis are presented in Section 4. Finally, Section 5 presents a short conclusion.

II. RELATED WORKS

In cloud system, whether cloud computing system or cloud storage system, million tasks are waiting for dispatching. Scheduling these mass tasks is a challenge to cloud environment. Many scheduling strategies are proposed in cloud computing. We review the scheduling algorithms in cloud computing firstly as follows.

In traditional cloud research area, task scheduling strategies aim to get the higher system throughput rate, namely get shorter makespan, such as the Min-Min and Max-Min algorithm[2] which are enumeration methods, an optimal solution can be selected if all the possible solutions are enumerated and compared one by one. When number of instances is large, exhaustive enumeration is not feasible for scheduling, and the heuristic is suggested algorithm to find reasonably solutions, such as ant colony based scheduling algorithm [3], Genetic algorithm (GA) based scheduling algorithms [4,5], Simulated annealing based scheduling algorithms [6] , Particle Swarm Optimization(PSO)[7], and so on.

Another aim of scheduling algorithms is making load balance, include Weighted Mean Time algorithm[8] and some of heuristic based scheduling algorithms[8-11].

In recent years,, the QoS received more and more attention in cloud systems for cloud itself is an idea to service users. In user view, task-scheduling method must satisfy their QoS requirements. There are already many QoS guided task-scheduling methods in cloud computing: QoS Guided Min-Min heuristic[12], based on the Min-Min heuristic, considers network bandwidth as QoS parameter. It divides the tasks in two groups: high and low QoS, and firstly scheduling the tasks from high QoS group on resources that can provide high QoS as required. Later, it schedules the tasks in low QoS group. This class division and hierarchical dispatching approach was then adopted by many algorithms[13-18] . These similar methods consider time or cost factors and divide them to four classes with different privilege. The difference in these methods is they use different scheduling algorithm inside privilege level. And the time and cost QoS parameters are the most used factors of existing method. However, these QoS guided methods can not deal with multiple QoS requirements. When the QoS factors increase, the class division is difficult and the privilege level is hard to decide. So they are not really multi-QoS guarantee.
The QWMTM Heuristic [19] integrate the QoS Min-Min heuristic and the Weighted Mean Time-min heuristic. By this way, it not only guarantees QoS but also guarantees load balance. Few multi-QoS guarantee algorithms [20, 23] use multiple workflows or multiple components to separate deal with single QoS. They are not the one can deal with multi-QoS in one component and at the same time.

Above all scheduling methods are used in cloud computing environment. These methods have following shortcomings when used into cloud storage system:

1) Task scheduling in cloud storage not only consider the technical QoS factors as in cloud computing but also take care the where the needed data are; We cannot ask one node offers the data it doesn't have.
2) Some important QoS factor in cloud storage is different from cloud computing, such as the CPU efficiency is a key factor in cloud computing, but it is not so important in cloud storage for the tasks in cloud storage are transmission tasks and the bandwidth instead of CPU efficiency plays an important role in cloud storage.
3) Existing multiple QoS guarantee methods almost consider from the aspect of system not from the aspect of user. Except cost and time, the factor they used is almost system parameter which user does not understand. In addition, most of their aims are system efficiency improvement. So we find these methods can not satisfy use's QoS requirements well.

In a word, there is not practical multiple QoS constraints and QoS preference awareness task scheduling method in cloud storage environment already. We propose a PSO based method to address this problem.

III. The PSO based Task Scheduling with QoS Preference Awareness in Cloud Storage Environment

The main difficult for multiple QoS guarantee and QoS preference awareness are: 1) there is contradiction among QoS factors, such as the transmission speed and transmission quality, because guarantee quality need some extra operation (timeout checking and retransmission mechanism for example) that cost more time. 2) It is difficult to synthetically consider various factors in one method. Existing methods use level division to guarantee multiple QoS. However, with the increase of factor number, the level division is more and more difficult. For example, with one QoS factor, it is easy to divide factor into the high and the low level. If there are two factors, they are usually divided into 4 levels by consider the factors are has high and low level for themselves. If the factors are more than 3, then the levels number is rapid growth. Obviously, in cloud storage there are far more than 3 factors and the levels maybe several dozen.

In order to address the first problem, we introduce "optimal order comparison method" into task scheduling to help user decide the important level of various QoS factors. Using important level to deal with contradiction requirement, namely if several incompatible factors have to be consider together, we use important level to decide which one is dominant and which one is secondary.

For the second problem, we use heuristic algorithm which uses fitness function to evaluate the scheduling scheme. We dispatch weights for multiple QoS factors respectively, and redefined fitness function to satisfy multiple QoS requirements. PSO (Particle Swarm Optimization) is a promising metaheuristic approach for solving diverse task scheduling problems as well as other application problems [24]. The simulations in reference [25] shown that the PSO algorithm minimize makespan and obtained higher performance than other compared techniques did. So we choose the PSO as the basic task scheduling algorithm.

A. *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:

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

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

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 scheduling is as follows:

1. Set iteration generation It_c=1. Initialize a population which including *m* particles. For the *i*th particle, it has random location X_i in specified space and for the *d*th 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.

B. Existing Matrix

However, cloud storage system is a special system different from cloud computing. The main difference between them is in cloud computing system, one task which asks host to computing, can be assigned to every node. But in cloud storage system, the task always asks remote node transfer 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. So we have to ensure the solution created by PSO satisfy the condition. We proposed a method named Existing Marix(EM) which we described in former work[24] to limit meaningless solution creation. The method also used in this paper, and here we will not go into details of them.

C. 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 $Q_i = \{q_1^i, q_2^i, \ldots, q_m^i\}$.

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

$$f_i = w_1 q_1^i + w_2 q_2^i + \ldots + w_m q_m^i \quad (3)$$

Where $\sum_i w_i = 1$. 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 chose a max value(or min depend on the definition, here we chose the max value) one.

Then the next problem is how to determine the weight for each factor to describe different importance of them.

D. Weight Dispatching Based on Optimal Order Comparison Method

We introduce the "optimal order comparison method (OOC)" to help uses to decide the weight. By this way we can reduce subjective judgment.

Firstly, the judgment scale is constructed. The 1, 2, 3, 4, 5 five numbers stand for five levels . The bigger value of number stands for the more importance. When comparing two factors, if user consider one' important level is 5, then another one's important is 0; if one' important level is 3, then another one's important is 2;and so on. In a word, the two factor's important levels sum to 5. By this way, we can create a judgment matrix which is an $n \times n$ square matrix where *n* is the number of factors, and the value of w_{ij} (*ith* row j*th* column) presents the relative important level of factor *i* and factor j. Following is an example: if w_{ij}=2, then the w_{ji}=5-2=3 which presents the relative importance of factor j compares to factor i. The sum of row values $\sum_i w_i$ describe the relative important of factor i in whole factor set. Take the whole relative important sum $\sum_j \sum_i w_{ij}$ as denominator, and the $\sum_i w_i$ as numerator, the quotient of them is the weight for factor i:

$$W_i = (\sum_i w_i) / (\sum_j \sum_i w_{ij}) \quad (4)$$

E. The QoS Factors and Their Utility Function

Following are usual technical factors in cloud storage system:

1) Bandwidth(Mbps): the bandwidth of backbone network is bigger than branch network obviously, denoted as *b*.
2) Cost (rmb/minute): the cost of a core server is far more than an ordinary PC, and even server themselves are has different cost, denoted as *c*.
3) Loads (percentage) include CPU load, storage load and bandwidth load, denoted as *l*.
4) Delay(second): network delay, the predict value from the historical data, including waiting time, preheating time and so on, denoted as *d*.
5) Delay variation(second): the variation of network delay, a mathematical measured value,denoted as *dv*.
6) Packet loss rate(percentage): the higher rate, the more worse communication quality, denoted as *plr*.
7) Mean Time Between Failures (MTBF for short,unit hours): the mean value of time between to failures

occur. The bigger the value, the better the network device quality.

8) Integrity, if the server offer integrity check, like MD5 check, to ensure the file is integrated and unchanged. This function may take more process time, denoted as *it*.
9) Encryption, if the server offer encryption function or not. Encryption can prevent files from reveal. This function also cost more time, denoted as *ep*.

We can see that these QoS factors have totally different physical significance and various units which prevent us to evaluate them in one function. Then the utility function is proposed to deal with this problem. A utility function can map a QoS factor value to a real value. We can evaluate these real values to choose the fit node. Our 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 independents of factors' unit and range.

If the factor q_i is efficient attribute which means the bigger value, the better quality of factor q_i, the utility function of factor q_i is[27,28]:

$$q_i = \begin{cases} (q_i - \min q_i)/(\max q_i - \min q_i), & \text{if } \max q_i \neq \min q_i \\ 1, & \text{if } \max q_i = \min q_i \end{cases} \quad (5)$$

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:

$$q_i = \begin{cases} 1, & \text{if } \max q_i = \min q_i \\ (\max q_i - q_i)/(\max q_i - \min q_i), & \text{if } \max q_i \neq \min q_i \end{cases} \quad (6)$$

where minq_i is the min value of q_i and maxq_i is the max value of q_i.

F. The Class Division of QoS Factors from the View of Users

We found that users are lack of knowledge of technical terms. For example, the term "delay variation" and 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 classify above mentioned QoS factors into these five classes as follows:

Cost class :{cost}

Efficiency class:{ Bandwidth, Loads, Delay, Delay variation }

Quality class:{ Packet loss rate , MTBF, Integrity }

Security class:{ Encryption }

These classes are denoted as C, E, Q and S.

In class, the multiple factors also use "optimal order comparison method" to decide the weight.

The weights for QoS classes are decided by users' experience and by this way our method can satisfy uses' QoS requirements. These weights named user weight, denoted as w_{user}={w^1,w^2,…,w^n}.

$$c_i = w^1 index_i^1 + w^2 index_i^2 + \ldots + w^n index_i^n \quad (7)$$

The weights for QoS factors inside class are decided by experts' knowledge and by this way can use experts' experience. These weights named expert weight, denoted as w_{expert}={w^1,w^2,…,w^n}. Then the fitness function formula(3) change to formula(8):

$$f_i = w^1 class_i^1 + w^2 class_i^2 + \ldots + w^n class_i^n \quad (8)$$

Above process is simply described in Fig. 1.

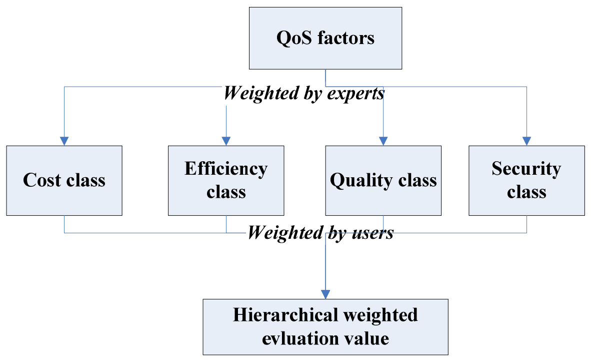

Figure 1. Hierarchical weighted by optimal order comparison

G. The Weights Definition and QoS Preference Divition

It is obviously different weights according to different environments. The expert weight for our simulation environment as follows.

TABLE I.
EXPERT WEIGHTS FOR EFFICIENCY CLASS

	b	l	d	dv	sum	weight
b	-	3	4	4	11	0.37
l	2	-	4	4	10	0.33
d	1	1	-	2	4	0.13
dv	1	1	3	-	5	0.17
sum					30	1

TABLE II.
EXPERT WEIGHTS FOR QUALITY CLASS

	plr	MTBF	it	sum	weight
plr	-	4	3	7	0.47
MTBF	1	-	1	2	0.13
it	2	4	-	6	0.4
sum				15	1

Expert weights are decided by experts, like network administrator and network researchers. Depending on the environment and the makers, the weights may have different value. Above values are fit for our environment.

Different from expert weight, the user weight for certain environment may have various values because the users' QoS requirements are different which called QoS preference. Some user think cost is the first factor take into consideration; some user prefers to the efficiency; some user takes care of quality and security. By investigation of our students (about 100 people), we found there are three main classes of users' QoS preference:

1) Cost class: in this class user only take care of cost, they enjoy free service.
2) Efficiency class: in this class user asks their task finished as quickly as possible, even this may cost more money.
3) Quality and security class: in this class users are willing to spend more money to ensure the quality and security. We found when quality is important to user, at the same time the security is also important. This is interesting that user takes care of their privacy of their important files.

Except above classes, few students need fast transmission and ensure quality and security at the same time; few students hope transfer file quick but the cost is low; few students hope ensure quality but the cost is low. These requirements are impossible or few students choose them. Here, we don't discuss these few task classes.

For the three classes, their user weight as Table III-V.

TABLE III.

USER WEIGHT FOR COST CLASS(W_C)

	C	E	Q	S	sum	weight
C	-	5	4	5	14	0.47
E	0	-	4	4	8	0.27
Q	1	1	-	3	5	0.17
S	0	1	2	-	3	0.1
sum					30	1

TABLE IV.

USER WEIGHT FOR EFFICIENCY CLASS(W_E)

	C	E	Q	S	sum	weight
C	-	1	3	4	8	0.27
E	4	-	5	5	14	0.47
Q	2	0	-	3	5	0.17
S	1	0	2	-	3	0.1
sum					30	1

TABLE V.

USER WEIGHT FOR QUALITY AND SECURITY CLASS(W_{QS})

	C	E	Q	S	sum	weight
C	-	1	0	0	1	0.03
E	4	-	0	0	4	0.13
Q	5	5	-	3	13	0.43
S	5	5	2	-	12	0.4
sum					30	1

Once again, above weights are suitable for our students, and the other people will have different weights.

H. PSO Based Hierarchical Task Scheduling with QoS Preference Awareness (PSO-HQoSPA)

In the case of expert and user weight are determined, when a batch tasks are arrive in unit time, we classify these tasks into above three QoS preference classes. Firstly we dispatching the "Quality and security class" tasks; secondly, the "Efficiency class" tasks, and finally the "Cost class" tasks are scheduled. When scheduling certain class tasks, the scheduling algorithm use according user weights. This is called hierarchical scheduling, as Fig. 2 shows.

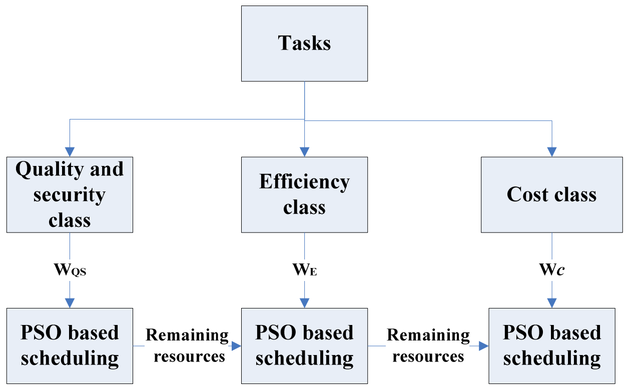

Figure 2. Hierarchical task scheduling with QoS preference awareness

The PSO based scheduling process as algorithm 1decribes.

```
-----------------------------------------------------------------
Algorithm 1: PSO based cloud storage task scheduling
-- --------------------------------------------------------------
   Input: number of particles $m$,
           inertia weight $w$,
          acceleration constants $c_1$ and $c_2$,
          maximum velocity $V_{max}$;
          task vector T ,
          Maximum iterating times MaxIt,
          QoS matrix Q,
          Expect Fitness Function Value $f_e$
          Exist matrix E;
  Output: task scheduling vector V

  Process:
  1.Set iterating times It=0, select a scheduling vector V
  with the limit E as the initialize particle;
  2.While It <= MaxIt
  3.Get $c_j$ by C, $b_{ij}$ by B;
  4.Computing $f_i$ as formula(8) and save $f_i$
  5.if $f <= f_e$ then break;
  6.It=It+1;
  7.update V by equation (2)、(3) ,get $V_{new}$;
  8.if $V_{new}$ is not in E then selection a V with the limit
  E as the $V_{new}$;
  9.End While.
  -----------------------------------------------------------------
```

IV.SIMULATIONS AND ANALYSIS

A. Simulation Environment

We developed a Cloud Storage Simulation System(CS^3) by Matlab7.0. This system includes three main modules: the Task Scheduling Module, the Update Module, and the Evaluation Module as the Fig.3 shows.

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).The order of QoS factor as the Section 3.E defined. 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($node_i$-$node_j$, $q_1,q_2,\dots,q_n$),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 $node_j$.

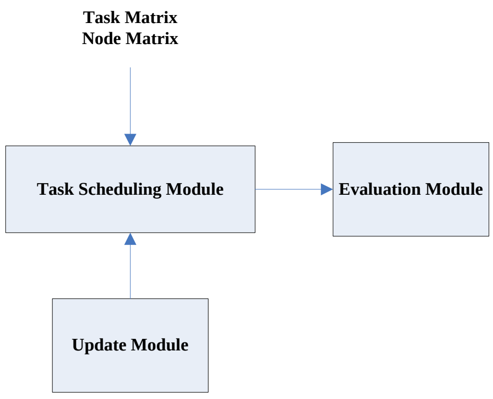

Figure 3. The simulation system CS^3

The system takes Task Matrix and Node Matrix as the input. Then the "Task Scheduling Module" dispatches tasks. The Update Module updates the QoS factor values of Nodes Matrix after one scheduling scheme is applied. Finally the Evaluation Module evaluates the scheme effect by user satisfaction rate. If task a was 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, we think the scheduling scheme satisfy the task. User satisfaction rate is defined as:

$$Usr(\%)=\frac{satisfied\ task\ number}{totall\ task\ number}\% \tag{9}$$

B. Simulation Result and Analysis

Simulation 1: The Effect of Existing Matrix

The effect of Existing Matrix is obvious. Using the same classical heuristic algorithm PSO algorithm, we compare the one with Existing Matrix(EM PSO) limited and the one without(PSO). We randomly created 100 tasks, 10 resource nodes and their Existing Matrix. The copy number of file is 3, and the three copies are storage in different nodes. The results as Table VI show.

TABLE VI.
THE EFFECT OF EXISTING MATRIX

	PSO	EM PSO
User satisfaction rate	13.40%	30.10%
Makespan(ms)	28.39	36.56

We can see the both algorithms have low user satisfaction rate. However, Existing Matrix improve user satisfaction rate from 13.40% to 30.10% and the same time the makespan decrease from 28.39ms to 36.56ms. Because traditional PSO algorithm only takes care of makespan and don't consider the QoS constraint. Existing Matrix is used to remove meaningless solution but also cannot offer QoS constraint. The more important effect of Existing Matrix is as shown in Fig.4 that it obviously decrease the iteration times of PSO to get the optimal solution.

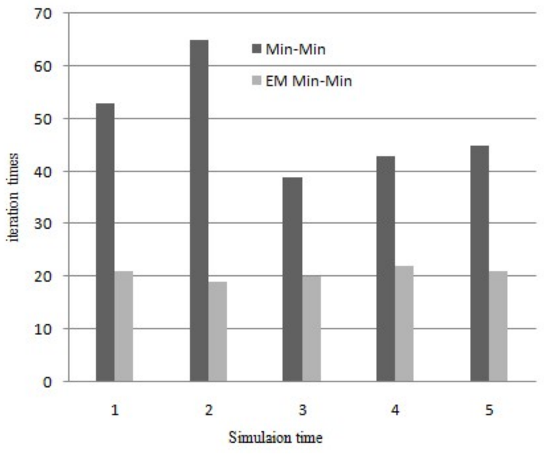

Figure 4. The iteration times comparison of Min-Min and EM Min-Min

Simulation 2: The Effect of Multi-QoS Guarantee

We randomly generated one hundred tasks (include their QoS requirements). The resource Nodes Matrix is the same as above and the Existing Matrix is updated. We compare our method PSO-HQoSPA to traditional PSO based scheduling method which use fix weights. The user satisfaction rate comparison as Fig. 5 shows. In order to avoid unforeseen interference, we repeat simulations 5 times.

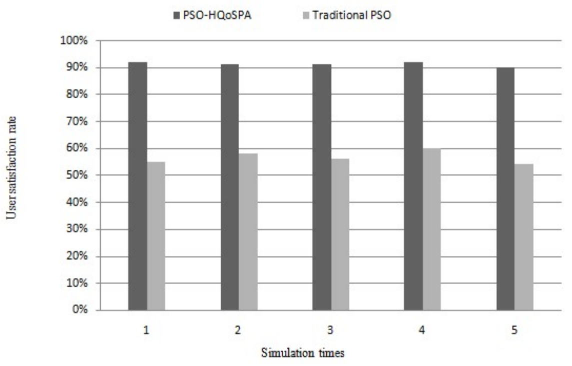

Figure 5. The user satisfaction rate comparison of PSO-HQoSPA and Traditional PSO

We can see PSO-HQoSPA have obviously higher user satisfaction rate than traditional PSO based one(former is about 90% and latter one is only 50%-60%). This demonstrates that our hierarchical weighted and self-adapting weights choose improvement offer QoS preference awareness ability to PSO based scheduling algorithm. By this way, when tasks have different QoS preference, our method can choose suitable weights by QoS preference class which is totally different from traditional PSO based scheduling which weights are fixed.

However, even PSO-HQoSPA add more operations into traditional PSO based scheduling method which indeed cost more time, but the makespan is not markedly increase as Fig.6 shows. This illustrate that the additional steps do not obstruct efficiency.

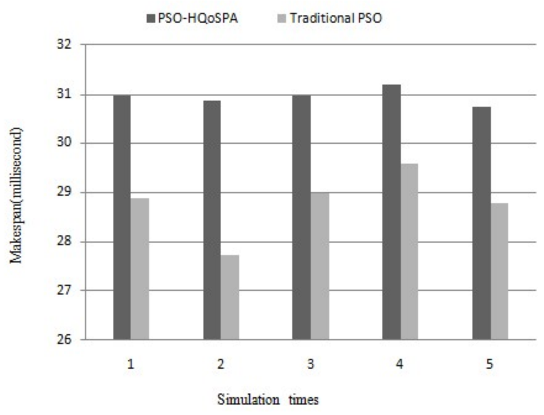

Figure 6. The makespan comparison of PSO-HQoSPA and Traditional PSO

In PSO based scheduling, the most time consuming step is its iterative step. Our method's iterative time compare to traditional PSO as Fig.7. We can see from Fig.7, the iterative time of PSO-HQoSPA is similar to traditional PSO which means our method maintain the efficiency.

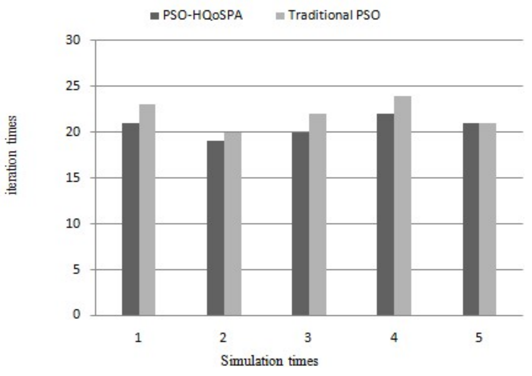

Figure 7. The iteration times comparison of PSO-HQoSPA and Traditional PSO

V. CONCLUSION

In this paper, we point out that existing task scheduling algorithms which come from cloud computing have following main shortcomings when they are used into cloud storage: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;3) These algorithms do not satisfy multi-QoS constraints very well.

For the first problem, we introduce "optimal order comparison method" to help users and experts decide the important level of various QoS factor. We do not consider the user experience but also the expert knowledge by hierarchical weighted. The expert level use experts' knowledge and the user level consider users' experience.

For the QoS preference awareness problem, we classify QoS preference and use different weights to describe them.

For the multi-QoS constraints problem, we refined the fitness function of the PSO.

By these changes, our PSO based hierarchical task scheduling algorithm creates acceptable user satisfaction rate solutions for cloud storage and maintains the efficiency at the same time.

ACKNOWLEDGMENT

This work was supported by the Application Basic Research Project of Sichuan Province(No. 2013JY0064, No. 2014JY0071), the Research Fund Young of Middle-Aged Academic Leaders in CUIT(J201107), the Sichuan Province Science and Technology Support Plan(No.2011GZ0195) and the Scientific Research Project of Education Department in Sichuan Province(No.14ZA0178, No.10ZB093).

REFERENCES

[1] J.Jiehui ,W. Jiyi ,F. Jianqing and L. Zhijie .A Survey on Cloud Storage,Joural of computers,vol.6,no.8,pp.1764-1771,August 2011.

[2] Maheswaran M, Ali S, Siegel H J, et al, Dynamic Mapping of a Class of Independent tasks onto Heterogeneous Computing Systems, 8th IEEE Heterogeneous Computing Workshop (HCW '99), Apr. 1999. pp. 30-44.

[3] Xiangqian S.,Lin G., Jieping W., "Job scheduling based on ant colony optimization in cloud computing," Computer Science and Service System (CSSS), 2011 International Conference on , vol., no., pp.3309,3312, 27-29 June 2011.

[4] Chenhong Z.; Shanshan Z.; Qingfeng L., Jian X., Jicheng H., "Independent Tasks Scheduling Based on Genetic Algorithm in Cloud Computing," *Wireless Communications, Networking and Mobile Computing, 2009. WiCom '09. 5th International Conference on* , vol., no., pp.1,4, 24-26 Sept. 2009

[5] Yujia G., Guiyi W., "GA-Based Task Scheduler for the Cloud Computing Systems," *Web Information Systems and Mining (WISM), 2010 International Conference on* , vol.2, no., pp.181,186, 23-24 Oct. 2010

[6] Guo-ning G.,Ting-lei H.,Shuai G., "Genetic simulated annealing algorithm for task scheduling based on cloud computing environment," *Intelligent Computing and Integrated Systems (ICISS), 2010 International Conference on* , vol., no., pp.60,63, 22-24 Oct. 2010

[7] Juan W.,, Fei L., Luqiao Z.,Task scheduling algorithm in cloud storage using PSO with limited solution domain, Applicaiton Research of Computers ,2013,30(1):127-129,154.

[8] Jinquan Z, Lina N, Changjun J, A Heuristic Scheduling Strategy for Independent Tasks on Grid, Proceedings of the Eighth International Conference on High-Performance Computing in Asia-Pacific Region (HPCASIA '05), November 2005.

[9] Xin L., Zilong G., "A load-adapative cloud resource scheduling model based on ant colony algorithm," *Cloud Computing and Intelligence Systems (CCIS), 2011 IEEE International Conference on* , vol., no., pp.296,300, 15-17 Sept. 2011

[10] Kun L., Gaochao X., Guangyu Z., Yushuang D.,Wang, D., "Cloud Task Scheduling Based on Load Balancing Ant Colony Optimization," *Chinagrid Conference (ChinaGrid), 2011 Sixth Annual* , vol., no., pp.3,9, 22-23 Aug. 2011

[11] Zhang B., Gao J., Jieqing A., "Cloud Loading Balance algorithm," *Information Science and Engineering (ICISE),*

2010 2nd International Conference on , vol., no., pp.5001,5004, 4-6 Dec. 2010
[12] Xiao-Shan H., Xian-He S., "QoS Guided Min-Min Heuristic for Grid Task Scheduling", Journal of Computer Science & Technology, 2003, (5): 442-451.
[13] Jeyarani, R.,Ram, R.V., Nagaveni, N., "Design and Implementation of an Efficient Two-Level Scheduler for Cloud Computing Environment", *Cluster, Cloud and Grid Computing (CCGrid), 2010 10th IEEE/ACM International Conference on* , vol., no., pp.585,586, 17-20 May 2010
[14] Shalmali A., Dipti B.i, Jaee K., Juhi B.,"An Optimistic Differentiated Job Scheduling System for Cloud Computing",International Journal of Engineering Research and Applications (IJERA) Vol. 2, Issue 2,Mar-Apr 2012, pp.1212-1214
[15] Qi-yi H., Ting-lei H., "An optimistic job scheduling strategy based on QoS for Cloud Computing," *Intelligent Computing and Integrated Systems (ICISS), 2010 International Conference on* , pp.673,675, 22-24 Oct. 2010
[16] Celaya,J.,Arronategui U., "A Highly Scalable Decentralized Scheduler of Tasks with Deadlines," *Grid Computing (GRID), 2011 12th IEEE/ACM International Conference on* , pp.58,65, 21-23 Sept. 2011
[17] Laiping Z., Yizhi R., Sakurai, K., "A Resource Minimizing Scheduling Algorithm with Ensuring the Deadline and Reliability in Heterogeneous Systems," *IEEE Third International Conference on Advanced Information Networking and Applications (AINA), 2011*,pp.275,282, 22-25 March 2011
[18] Van den Bossche, R., Vanmechelen, K.,Broeckhove, J., "Cost-Efficient Scheduling Heuristics for Deadline Constrained Workloads on Hybrid Clouds" *IEEE Third International Conference on Cloud Computing Technology and Science (CloudCom), 2011*,pp.320,327, Nov. 29 2011- Dec. 1 2011
[19] Sameer Singh Chauhan, R. C. Joshi. "QoS Guided Heuristic Algorithms for Grid Task Scheduling",International Journal of Computer Applications,Volume 2,No.9, pp.23-31,June 2010
[20] Meng X.,Lizhen C.,Haiyang W.,Yanbing B., "A Multiple QoS Constrained Scheduling Strategy of Multiple Workflows for Cloud Computing", *IEEE International Symposium on Parallel and Distributed Processing with Applications, 2009* , pp.629,634, 10-12 Aug. 2009
[21] Frincu, M.E.,Craciun, C., "Multi-objective Meta-heuristics for Scheduling Applications with High Availability Requirements and Cost Constraints in Multi-Cloud Environments" *IEEE International Conference on Utility and Cloud Computing (UCC), 2011*, pp.267,274, 5-8 Dec. 2011
[22] Baomin X., Ning W., Chunyan L., "A Cloud Computing Infrastructure on Heterogeneous Computing Resources",Journal of Computers, Vol.6, No.8, 1789-1796, Aug 2011
[23] Xiaoli W., Yuping W., Hai Z., "Energy-efficient Task Scheduling Model based on MapReduce for Cloud Computing using Genetic Algorithm", Journal of Computers, Vol .7, No.12,pp. 2962-2970, Dec 2012.
[24] Ruey-Maw Chen and Chuin-Mu Wang. "Project Scheduling Heuristics-Based Standard PSO for Task-Resource Assignment in Heterogeneous Grid,Hindawi Publishing Corporation Abstract and Applied Analysis",Vol.2011, pp.1-20.
[25] H. Izakian, Abraham A., andSnášel V.. "Metaheuristic based scheduling meta-tasks in distributed heterogeneous computing system", Sensors, vol. 9, pp. 5339-50, 2009.
[26] Juan W., Fei L., Aidong C., "An Improved PSO based Task Scheduling Algorithm for Cloud Storage System", AISS, Vol. 4, No. 18, pp. 465 ~ 471, 2012
[27] Alrifai M, Risse T. "Combining global optimization with local selection for efficient QoS-aware service composition". In: Proc. Of the 18th Int'l Conf. on World Wide Web. pp. 881-890, 2009.
[28] Yu T., Zhang Y., Lin K.J. "Efficient algorithms for Web services selection with end-to-end QoS constraints". ACM Transactions on the Web, Vol. 1, No. 1, Article 6, May 2007.

Juan Wang was born in China in 1981 and received her B.S. degree of computer science in 2003, and the M.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. Now, she is an assistant professor and her research interests include network security and cloud storage. Email: wangjuan@cuit.edu.cn.

Fei Li was born in China in 1966. He received his M.S. degree of computer science in 1993 from Chengdu University of Science and Technology. Now he is with Chengdu University of Information Technology(CUIT) as a full professor and dean of school of Information Security engineering in CUIT. His research interests include grid computing and the Internet of things technology. Email: lifei@cuit.edu.cn

Luqiao Zhang was born in China in 1983, received B.S. degree, M.S. degree and Ph.D degree in computer science from University of Electronics and Technology of China (UESTC) in 2003, 2006 and 2013 respectively. His research interests include wireless sensor network and mobile wireless P2P. Email: zhanglq@cuit.edu.cn

Yuanyuan Huang was born in China in 1982, he received his B.S. degree, M.S. degree and Ph.D. degree from University of Electronic Science and Technology of China (UESTC) in 2004, 2007 and 2013, respectively, all in computer science. From Nov. 2009 to Nov. 2011, he had been a visiting scholar in University of Washington (UW), Seattle, USA. Currently he is with Chengdu University of Information Technology (CUIT), China. His research interests include network science, multimedia security. Email: hy@cuit.edu.cn.