2016 17th International Conference on Parallel and Distributed Computing, Applications and Technologies

Optimal Scheduling Algorithm of MapReduce Tasks Based on QoS in the Hybrid Cloud

Xijun Mao, Chunlin Li, Wei Yan, Shumeng Du
Computer Science and Technology Wuhan University of Technology
mxj_cj@163.com, chunlin74@aliyun.com

***Abstract*—Research on MapReduce tasks scheduling method for the hybrid cloud environment to meet QoS is of great significance. Considering that traditional scheduling algorithms cannot fully maximize efficiency of the private cloud and minimize costs under the public cloud, this paper proposes a MapReduce task optimal scheduling algorithm named MROSA to meet deadline and cost constraints. Private cloud scheduling improves the Max-Min strategy, reducing job execution time. The algorithm improves the resource utilization of the private cloud and the QoS satisfaction. In order to minimize the public cloud cost, public cloud scheduling based on cost optimization selects the best public cloud resources according to the deadline. Experimental results show that the proposed algorithm in this paper has less job execution time, higher QoS satisfaction than the Fair scheduler and FIFO scheduler. It also has more cost savings and shorter job completion time than recent similar studies.**

***Keywords*—hybrid cloud; MapReduce; QoS satisfaction; optimal scheduling**

I. Introduction

Currently there are three cloud computing [1] models. The private cloud have good data security, high quality of service and low cost; the public cloud exists in the public network environment (Internet) with the form of the third party service provider, which has large available resources; hybrid cloud is made up of two or more clouds (private cloud and public cloud), where they are relatively independent but can work together again [2]. The main purpose of the hybrid cloud resource scheduling meet the demand of the user's Quality-of-Service(QoS) [3]. So when the private cloud can't satisfy the user's QoS requirements, some tasks need to be dispatched to public cloud. Public cloud has the characteristic of pay-per-use, so the optimal scheduling strategy based on the deadline and cost constraints in the hybrid cloud has important research significance. MapReduce plays an increasingly important role in the massive data analysis software architecture. Scheduling for MapReduce jobs especially in the hybrid cloud environment has become the focus of attention of scholars.

At present, most scheduling algorithms can not meet the QoS constraints in real time, and there is little consideration for the MapReduce job scheduling in hybrid cloud. So it is worth studying the task scheduling which satisfy the QoS demand of MapReduce in the hybrid cloud environment. Considering the deadline and cost constraints of MapReduce, this paper proposes a tasks scheduling algorithm based on QoS in a hybrid cloud. The main work is as follows: (1) In order to improve the utilization of private cloud, as much as possible tasks are executed in the private cloud. In the private cloud scheduling algorithm, this paper improves the Max-Min strategy, which assigns high priority tasks to the resource of the shortest completion time. The algorithm reduces the waste of private cloud resources, the task response time and job execution time. (2) Due to the lack of private cloud resources, tasks can not be completed within the deadline, it need to apply suitable public cloud resources. In this paper, according to cost optimization method, the public cloud scheduling algorithm solves the public cloud resources that have lowest cost and meet the deadline.

II. Related work

Three typical kinds of scheduling algorithms include FIFO Scheduler, Fair Scheduler and Capacity Scheduler [4][5]. But these schedulers do not consider QoS constraint, and do not apply to the demand of user real-time service, this kind of service is becoming more and more important and necessary in a hybrid cloud. Kc and Anyanwu [6] research scheduling algorithm based on Hadoop deadline constraint. Tang and Zhou [7] presented an algorithm of MapReduce task scheduling for deadline constraints, which is based on node classification to improve data locality in the heterogeneous cluster. Sandholm and Lai [8] proposed Dynamic Proportional Scheduler, which allows the user to adjust the priority based on the importance of the submitted job. Zaharia et al. [9] proposed Delay Scheduler that optimized data locality under different workloads. Ruben et al.

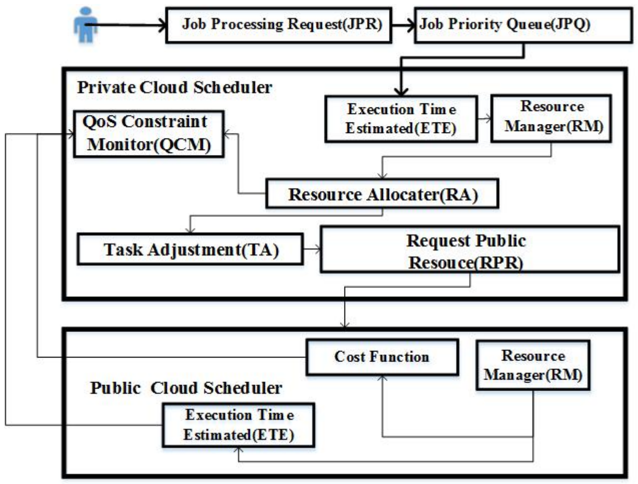

Fig. 1: Hybrid cloud scheduling model based on QoS

[10] researched Cost-Optimal scheduling in hybrid IaaS clouds. Bittencourt and Madeira [11] proposed a cost optimization algorithm for workflow scheduling in the hybrid cloud.

In this paper, we consider the deadline and cost constraints of the user submitting jobs in a hybrid cloud environment, and study a MapReduce job optimal scheduling algorithm based on QoS. The algorithm can maximize the utilization of private cloud and minimize the cost of public cloud.

III. MODEL AND METHOD OF THE PROPOSED ALGORITHM

A. Hybrid cloud scheduling model based on QoS

The scheduling model of hybrid cloud is illustrated in Fig. 1, mainly including Public Cloud Scheduler and Private Cloud Scheduler. Users submit jobs with QoS constraints to the Job Processing Request(JPR). JPR is based on the QoS constraints to calculate the priority of the job, and then submitted to the Job Priority Queue(JPQ). The execution time of a task on a resource is calculated by the Execution Time Estimated(ETE). If the completion time of all resources to meet the QoS constraints(QoS Constraint Monitor-QCM), the Resource Allocater(RA) allocate resources in the private cloud. Otherwise, the Task Adjustment(TA) need to adjust the task to the set of Request Public Resouce(RPR) to meet the deadline. When the task requires public cloud resources, public cloud scheduling solves the minimum cost(Cost Function) of resources to meet the deadline. The Resource Manager(RM) collects and manages cloud resource.

B. MapReduce task scheduling related definitions and problem formulations in the hybrid cloud

1) MapReduce job and tasks

$J_i = \{m_{i1}, m_{i2}, ..., m_{im}, r_{i1}, r_{i2}, ..., r_{ir}\}$: A MapReduce job($J_i$) consist of m Map tasks and r Reduce tasks. Submit jobs with additional user's QoS requirements, D_i and C_i are respectively deadline and cost constraint of the job. MapReduce tasks are defined as $m_{ik} = \{mW_{ik}, mD_{ik}\}$ and $r_{ik} = \{rW_{ik}, rD_{ik}\}$: mW_{ik} represents the workload size of the Map task. mD_{ik} represents the size of input data for the Map task. rW_{ik} represents the workload size of the Reduce task. rD_{ik} represents the size of the input data for the Reduce task. It is equal to the sum of the output data of w Map tasks.

$$rD_{ik} = f \times \sum_{u=1}^{w} mD_{iu} \quad 0 < f < 1 \tag{1}$$

Where f is the ratio of the size of output data of the map task to the input data. w represents the output value of a reduce task handling w Map tasks.

2) Hybrid cloud Resource

The different instances of public cloud resource may have different performance and different prices. In the following, we define resource in a formal way. Cloud provider's resource: $Container_j = \{Mips_j, Cost_j, Stg_j, Cin_j, Cout_j, Band_j, Rft_j\}$. Where $Mips_j$ represents the host's computing power, that is, the number of instructions that can be executed in a unit time. $Cost_j$: Computing cost of public resource. Stg_j: Storage cost. Cin_j: Input transmission cost. $Cout_j$: Output transmission cost. $Band_j$: Network bandwidth. Rft_j: Completion time of resource, namely the waiting time of resource reused.

3) The Problem formulation of MapReduce task scheduling

The transmission time of the Map task $mDtt$ and Reduce task $rDtt_{ik}$ are defined as follows respectively:

$$mDtt = \frac{mD_{ik}}{Band_j} \tag{2}$$

$$rDtt_{ik} = \frac{f \times \sum_{u=1}^{\gamma} mD_{iu}}{\gamma \times Band_j} \tag{3}$$

In the formula(3), γ represents the number of Map tasks processed by the Reduce task.

The following is the calculation formula for the execution time prediction of the Map task $mEEt\,[i, k, j]$ and Reduce task $rEEt\,[i, k, j]$:

$$mEEt\left[i,k,j\right]=\frac{mW_{ik}}{Mips_j}+\frac{mD_{ik}}{Band_j} \tag{4}$$

$$rEEt\left[i,k,j\right]=\frac{rW_{ik}}{Mips_j}+rDtt_{ik}+t_s \tag{5}$$

t_s represents sort phase time of the Reduce task.

The cost of private cloud is ignored; the public cloud cost includes the cost of computing, storage and transmission. The cost of Map task $mCostF\left[i,k,j\right]$ and Reduce task $rCostF\left[i,k,j\right]$ in the public cloud resource j are defined separately:

$$\begin{aligned}mCostF\left[i,k,j\right]=&Cost_j\times mW_{ik}+Stg_j\times mD_{ik}\\&+mD_{ik}\times(Cin_j+Cout_j)\end{aligned} \tag{6}$$

$$\begin{aligned}rCostF\left[i,k,j\right]=&Cost_j\times rW_{ik}+Stg_j\times rD_{ik}\\&+rD_{ik}\times(Cin_j+Cout_j)\end{aligned} \tag{7}$$

The priority of MapReduce job is defined:

$$priority_i=\left(\sum_{k=1}^{m}mD_{ik}+\sum_{k=1}^{r}rD_{ik}\right)\times(\frac{\varrho}{C_i}+\frac{1-\varrho}{D_i}) \tag{8}$$

Where ϱ represents the weight of the QoS factors. A greater the size of the input data, a smaller cost, and a shorter deadline will have higher the priority.

C. MapReduce task optimization scheduling based on QoS

In the literature[12] map the resource selection problem to 0-1 multi objective and multi choice knapsack problem, it allocates $m+r$ tasks to n resources, which will check $(m+r)^n$ kinds of cases. The public cloud is public service model that has a particularly large resource. So this method can not be used for hybrid cloud resource allocation. A heuristic algorithm is proposed to solve an optimal scheduling problem. In this paper, Private cloud scheduling improves the MAX-MIN strategy, which select the maximum priority task and the minimum completion time of the resource. And most researchers did not consider the difference between the Map and Reduce tasks and the links between them. The implementation code and input data of Map and Reduce tasks is very different, so this paper separates scheduling of the Map and Reduce tasks, and predicts that the execution time and cost calculation method is also different. And public cloud scheduling based on cost optimization to select the best public cloud resources.

IV. MapReduce task optimization scheduling based on QoS for the hybrid cloud

A. Private cloud scheduling

Algorithm I: Private Cloud Scheduling as is shown in TABLE I. Lines 1 initialize variables of the proposed algorithm. Lines 2-18 of the algorithm improve the Max-Min strategy. Lines 20 move tasks to RPR, which ensure tasks to meet deadline constraint. Lines 20 allocate task to private cloud resource.

TABLE I: Private cloud scheduling algorithm

Algorithm I: Private Cloud Scheduling

Input: J_i is the submitted MapReduce job. APR is the set of allocated private resources N containers.
Output: $\langle R_i, PRP_i\rangle$: where R_i record the mapping of task and resource. PRP_i is the set of tasks need the public cloud resource.

```
1.  $R_i \leftarrow \emptyset$; $PRP_i \leftarrow \emptyset$
2.  Sort tasks based on priority in ascending
3.  for each $Task_i \in J_i$
4.    $resourcej \leftarrow$ the first resource; $fitj \leftarrow \infty$; $tmp \leftarrow Rft_j$
5.    for each $j \in APR$
6.      if $Task_i$ is the map task, that is $m_{ik}$
7.        Calculate $mEEt\left[i,k,j\right]$
8.        if $mEEt\left[i,k,j\right] + Rft_j < fitj$
9.          $fitj \leftarrow mEEt\left[i,k,j\right] + Rft_j$; $resourcej \leftarrow j$
10.         endif
11.     else if $Task_i$ is the reduce task, that is $r_{ik}$
12.        Calculate $rEEt\left[i,k,j\right]$
13.        if $rEEt\left[i,k,j\right] + Rft_j < fitj$
14.          $fitj \leftarrow rEEt\left[i,k,j\right] + Rft_j$; $resourcej \leftarrow j$
15.        endif
16.     endif
17.   endfor
18.   $Rft_j \leftarrow fitj$
19.   if $Rft_j > D_i$
20.     Add $Task_i$ to $PRP_i$ and $Rft_j \leftarrow tmp$
21.   else
22.     Allocate $Task_i$ to $resourcej$ and record $R_i$
23. endfor
24. return $\langle R_i, PRP_i\rangle$
```

B. Public cloud scheduling

The output of Algorithm I is a collection of tasks that need public cloud resources. Algorithm II assigns these tasks to the public cloud resources, which first searches for the minimum cost of public cloud resources within the deadline of tasks, apply it, and then assigns tasks to those resource. Algorithm II: Public Cloud Scheduling as is shown in TABLE II. Lines 3-11 calculate the execution time and cost of tasks in public cloud. Lines 12-18 try to find the best public resource that have the lowest price and meet the deadline. If the best resource is found, the algorithm will create the resource instance and assign the task to the resource. Otherwise the job is rejected, because it is not satisfied with deadline. Lines 20-24 judge whether the cost constraint is met.

TABLE II: Public cloud scheduling algorithm

```
Algorithm II: Public Cloud Scheduling
Input: $PRP_i$ represents the tasks of $J_i$ need public cloud
  resource, $PR$ represents public resources, $T_{init}$
  represents the initialization time of public cloud resource.
Output: $\langle RP_i, R_i, RT_i \rangle$ $RP_i$ represent the set of public cloud
  resource, $R_i$ represent the tasks who are scheduled on the public
  cloud. $RT_i$ represents whether the deadline of $RT_i$ is met.
1.  $RP_i \leftarrow \emptyset$;$R_i \leftarrow \emptyset$; $sumcost \leftarrow \emptyset$
2.  for each task $Task_i$ from $PRP_i$
3.    for each resource type $rj$ in $PR$
4.      if the task is the map task,that is $m_{ik}$
5.        $SumCost \leftarrow SumCost + mCostF[i,k,rj]$
6.        $T_{task} \leftarrow mEEt[i,k,rj]$
7.      else if the task is the reduce task,that is $r_{ik}$
8.        $SumCost \leftarrow SumCost + rCostF[i,k,rj]$
9.        $T_{task} \leftarrow rEEt[i,k,rj]$
10.     endif
11.   endfor
12.   Find the public resource $rj_{best}$ that has the lowest cost
      subject to $T_{task} + T_{init} < D_i$
13.   if $rj_{best}$ exists
14.     The public resource $rj_{best}$ is created
15.     Assign $Task_i$ to resource $rj_{best}$ and record $R_i$
15.     Add $rj_{best}$ to $R_i$
16.   else
17.   return $\langle Null, Null, False \rangle$
18.   endif
19. endfor
20. if $SumCost < C_i$
21.   return $\langle RP_i, R_i, True \rangle$
22. else
23.   return $\langle RP_i, R_i, False \rangle$
24. endif
```

V. EXPERIMENTAL PROCESS AND RESULTS ANALYSIS

A. Experimental environment

In order to verify the feasibility and effectiveness of the algorithm, we build private cloud environment. The experimental configuration of the private cloud cluster is shown in TABLE III. Public cloud service providers such as Microsoft Azure, Amazon EC2, Ali cloud have instance type, CPU type and quantitative cost, as shown in TABLE IV. Considering the situation that the experimental data is smaller than the real data, the cost is quantified as $\$/s$, which means the expense per second. According to resource requirements, public cloud resources are added to the private cloud cluster dynamically which forms the hybrid cloud. The Virtual Private Network(VPN) access device is used to integrate public cloud and private cloud resources. The experimental environment architecture in Hybrid cloud is shown in Fig. 2. In the private cloud environment we use the open source Hadoop distributed platform, which consists of a namenode and 10 datanode. The operating system uses the Ubuntu 14.04. Hadoop version is 2.7.1 and the Java

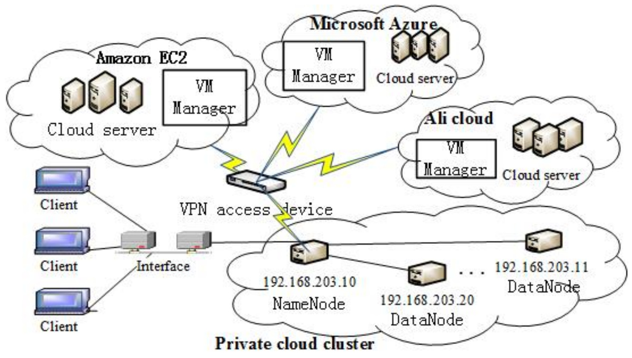

Fig. 2: The experimental environment architecture in Hybrid cloud

Platform of Java SE Development Kit(JDK) is jdk-7u79-linux-x64. The VM manager storages instance type and price information.

TABLE III: The experimental configuration of the private cloud clusters

HostName	Cluster Configuration	IP
Master (NameNode)	Intel(R)Core(TM) i5-4590CPU@3.30GHz RAM:4G	192.168.203.10
Slave1-slave10 (DataNode)	Intel(R)Core(TM) i5-4590CPU@3.30GHz RAM:4G	192.168.203.11∼ 192.168.203.20

TABLE IV: Instance type, CPU type of public cloud service provider

Public Cloud	Instance Type	CPU Type	Cost ($\times 10^{-6}\$/s$)
Microsoft Azure	Core:2 RAM:3.5G Bandwidth:10Mbps	Intel Xeon(R) E5-2650 v2 2.60GHZ	28
	Core:4 RAM:7G Bandwidth:20Mbps		73
	Core:8 RAM:14G Bandwidth:30Mbps		228
Amazon EC2	Core:2 RAM: 8G Bandwidth: 10Mbps	Intel Xeon(R) E5-2676 v3 2.4 GHz	27
	Core:4 RAM:16G Bandwidth:30Mbps		65
	Core:8 RAM:32G Bandwidth:40Mbps		248
Ali Cloud	Core:1 RAM:1G Bandwidth :10Mbps	Intel Xeon(R) E5-2650 v2 2.60GHZ	29
	Core:4 RAM:8G Bandwidth:20Mbps		89
	Core:8 RAM:16G Bandwidth:50Mbps		235

1) Test cases

In the experiment, the number of Reduce tasks for these jobs is set to 25 percent of the number of Map tasks, and the size of the input file for each Map task is set to 128MB. We submit a series of jobs, including 16, 48, 80, 96, 160 Map tasks.

2) Evaluation index

In the first experiment, we evaluated the job execution time in a private cloud by contrasting Fair, FIFO and the proposed MROSA. Job execution time represents the time from start to finish. We evaluated the QoS satisfaction in a private cloud by contrasting Fair, FIFO and the proposed MROSA in Experiment 2. QoS satisfaction is defined as the percentage that job submission can be completed within the deadline. Next in experiment 3 and experiment 4, we evaluate cost and completion time of the job considing public cloud resources. Cost represents the total cost of tasks performed on resources; job completion time represents the time from job arrives to being finished.

3) Benchmark algorithm

In experiment 1 and experiment 2, We evaluated the job execution time in a private cloud by contrasting Fair, FIFO and the proposed MROSA. Literature [11] proposed a algorithm based on binary integer programming, namely COSHIC, which can maximize the private cloud utilization and minimize the public cloud cost. In order to compare the performance of MROSA and COSHIC, we did some experiments to evaluate their cost and completion time in experiment 3 and experiment 4.

B. Experimental results and analysis

In order to evaluate the job execution time of the proposed algorithm, we submit a series of jobs, including 16, 48, 80, 96, 180 Map tasks. The deadline for each job is set to infinity without considering public cloud resources in the first experiment. The results of experiment 1 are shown in Fig. 3.

Fig. 3 shows that the average execution time of the scheduling of multiple jobs containing 180 Map tasks by using MROSA is 10% smaller than Fair, 33% than FIFO. All results show that the performance of the algorithm is better as the number of tasks increases. Taking into account the difference between the Map and Reduce tasks, the MROSA has better resource allocation, which reduces the response time of the tasks and the total execution time of tasks. So the average execution time of a job is shorter than the Fair and FIFO, thus improving the resource utilization.

In Experiment 2 we evaluate QoS satisfaction, a series of jobs were submitted which include 96 Map Tasks, and the number of Reduce tasks were set to 19. The deadline of those jobs was set 90s-190s, and step size 10. The experimental results are shown in Fig. 4.

Fig. 4 shows that the longer the job deadline, the greater the QoS satisfaction. When the deadline for the jobs was 148s, QoS satisfaction of MROSA can reach 100%, while the Fair and the FIFO were 78% and 73%. The main reason for the higher QoS satisfaction of MROSA is that the average execution time of MROSA is smaller than the Fair and FIFO, and the resource utilization is maximized. At the same time, the job's priority is defined according to the QoS constraint in MROSA, which reduces the waiting time of the job. MROSA can handle more tasks so that it has a higher QoS satisfaction than the other two Schedulers.

In the experiment 3 and experiment 4, we consider public cloud resources. In the experiment 3, we submit a series of jobs, including 48, 96, 180 map tasks. The deadline for each job was set to 60s. The experiment evaluate cost by contrasting COSHIC and the proposed MROSA. The experimental results are shown in Fig. 5.

Fig. 5 shows that when the number of Map jobs is 96, the average cost of the MROSA is 24% less than COSHIC; while the number is 180, the average cost of the MROSA is 90% smaller. In the condition of the same deadline, the larger the size of input data is, the greater the public cloud resources for a job needs, and the cost is higher. However, due to that the job execution time

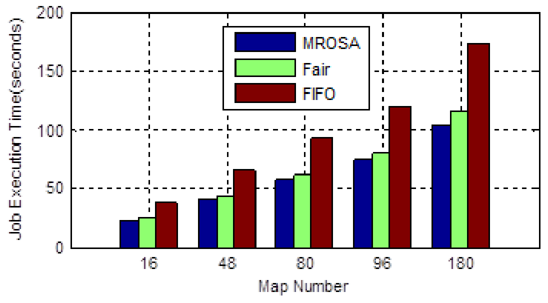

Fig. 3: Job execution time for the MROSA, the fair scheduler, and the FIFO scheduler

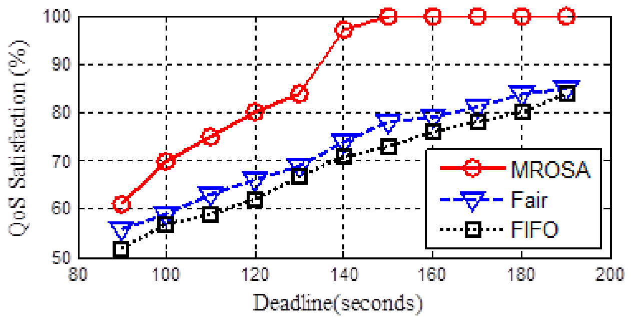

Fig. 4: QoS stisfaction for the MROSA, the Fair scheduler, and the FIFO scheduler

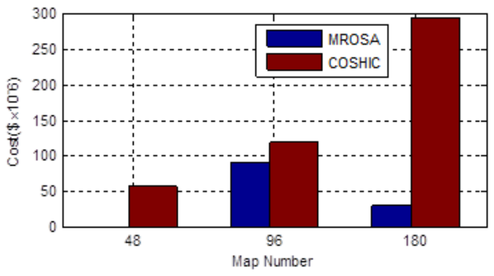

Fig. 5: Cost comparison between the MROSA and the COSHIC in the same deadline

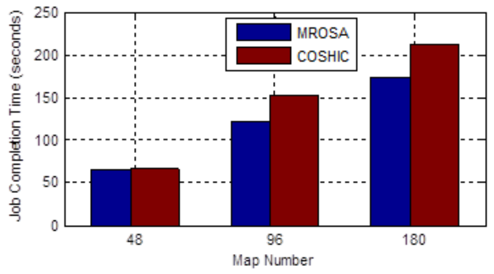

Fig. 6: The job completion time for the MROSA and the COSHIC in the same cost

of MROSA is less than the COSHIC, the requirement of public cloud resources is relatively small so that the saving of cost is obvious.

Next, we set the cost constraint to 56×10^{-6}\$ in the experiment 4. we evaluate completion time by contrasting COSHIC and the proposed MROSA. The experimental results are shown in Fig. 6. As shown in Fig. 6, the job completion time of MROSA is 1%, 21%, 17% smaller than COSHIC, respectively. When the Map task number is 48, the jobs can be executed in a private cloud by using MROSA, and public cloud resources do not need to be applied. However, because of using public cloud resources, the efficiency of the COSHIC is improved, resulting that MROSA job completion time is only 1% smaller than COSHIC. When the number of Map is 96 and 180 respectively, the job's completion time of MROSA is significantly smaller than COSHIC.

VI. CONCLUSION AND PROSPECT

In this paper, we consider a MapReduce job scheduling model in hybrid cloud. This model submit jobs to the priority queue according to the weight of deadline and cost constraints. And we propose a MapReduce job optimal scheduling algorithm based on the QoS in the hybrid cloud environment, which can solve the problem of resources selection of MapReduce tasks. By optimizing the MapReduce task scheduling model and combining with the improved MAX-MIN strategy, MROSA can get the optimal MapReduce task resource allocation scheme. The algorithm optimize utilization of private cloud, public cloud costs and the completion time of the job. Comparing with other similar algorithms in the experiment, MROSA has better performance. The results show that MROSA performs better than Fair and FIFO , having shorter job execution time and better QoS satisfaction. MROSA can save more cost and have shorter job completion time than COSHIC.

The next research work mainly includes two aspects: first, the time of the MapReduce tasks is predicted by the small sample. Second, we will consider the operation cost of public cloud and energy consumption of private cloud in the future.

ACKNOWLEDGMENT

The work was supported by the National Natural Science Foundation (NSF) under grants (No.61472294, No.61672397), Key Natural Science Foundation of Hubei Province (No.2014CFA050).

REFERENCES

[1] Zhang S, Zhang S, Chen X, et al. Analysis and Research of Cloud Computing System Instance[C]. Future Networks, 2010. ICFN '10. Second International Conference on. IEEE, 2010:88-92.

[2] Goudarzi H, Ghasemazar M, Pedram M. SLA-based Optimization of Power and Migration Cost in Cloud Computing[C]. Cluster, Cloud and Grid Computing (CCGrid), 2012 12th IEEE/ACM International Symposium on. IEEE, 2012:172-179.

[3] Chen X, Zheng Z, Liu X, et al. Personalized QoS-Aware Web Service Recommendation and Visualization[J]. IEEE Transactions on Services Computing, 2011, 99(1):35-47.

[4] Armstrong P et al. Cloud Scheduler: a resource manager for distributed compute clouds[J]. Computer Science, 2010.

[5] Pakize S R. A Comprehensive View of Hadoop MapReduce Scheduling Algorithms[J]. International Journal of Computer Networks & Communications Secu, 2014.

[6] Kc K, Anyanwu K. Scheduling Hadoop Jobs to Meet Deadlines[C]. IEEE Second International Conference on Cloud Computing Technology and Science. IEEE, 2010:388-392.

[7] Tang Z, Zhou J, et al. A MapReduce task scheduling algorithm for deadline constraints[J]. Cluster Computing, 2013, 16(4):651-662.

[8] Sandholm T, Lai K. Dynamic Proportional Share Scheduling in Hadoop[M]. Job Scheduling Strategies for Parallel Processing. Springer Berlin Heidelberg, 2010:110-131.

[9] Zaharia M, Borthakur D, et al. Delay scheduling: a simple technique for achieving locality and fairness in cluster scheduling[C]. European Conference on Computer Systems. ACM, 2010:265-278.

[10] Ruben V D B, Vanmechelen K, Broeckhove J. Cost-Optimal Scheduling in Hybrid IaaS Clouds for Deadline Constrained Workloads[C]. IEEE International Conference on Cloud Computing, Cloud 2010, Miami, Fl, Usa, 5-10 July. 2010:228-235.

[11] Bittencourt L F, Madeira E R M. HCOC: a cost optimization algorithm for workflow scheduling in hybrid clouds[J]. Journal of Internet Services & Applications, 2011, 2(3):207-227.

[12] Wang W J, Chang Y S, Lo W T, et al. Adaptive scheduling for parallel tasks with QoS satisfaction for hybrid cloud environments[J]. Journal of Supercomputing, 2013, 66(2):783-811.