Power Consumption Optimization Strategy of Cloud Workflow Scheduling Based on SLA

YONGHONG LUO, SHUREN ZHOU
School of Computer and Communication Engineering
Changsha University of Science and Technology
960, 2nd Section, Wanjiali South RD, Changsha, Hunan
CHINA
luoyonghong_km@163.com, zsr_hn@163.com

Abstract: -Cloud computing, as a new model of service provision in distributed computing environment, faces the great challenge of energy consumption because of its large demand for computing resources. Choosing improper scheduling method to execute cloud workflow tends to result in the waste of power consumption. In order to lower the higher power consumption for cloud workflow executing, we propose a power consumption optimization algorithm for cloud workflow scheduling based on SLA (Service Level Agreement), which can reduce power consumption while meeting the performance-based constraints of time and cost. The algorithm first searches for all feasible scheduling solutions of cloud workflow application with critical path, then the optimal scheduling solution can be found out through calculating total power consumption for each feasible scheduling solution. The experimental results show that compared with traditional workflow scheduling algorithms based on QoS, the optimization algorithm proposed in this paper not only meets the constraints of time and cost defined in SLA, but also reduces the average power consumption by around 10%.

Key-Words: - Cloud computing, Cloud workflow, SLA, Critical path, Scheduling solutions, Power consumption optimization

1 Introduction

Due to integrating a large number of computing resources and storage resources in cloud data center, cloud computing system needs to solve various problems to implement a high effective, low-cost and safe distributed computing platform [1]. The high power consumption [2, 3] is one of the most serious problems for cloud computing system. According to statistics, the power consumption of cloud data centers [4] has risen by 56 percent from 2005 to 2010, and in 2010 accounted to be between 1.1 and 1.5 percent of the global electricity use [5]. There exists power consumption waste caused by improper scheduling method in addition to the necessary power consumption for executing user tasks in cloud computing system [6]. Cloud computing system usually contains a lot of computers with different performance which may need different response time and power consumption to execute the same tasks. With regard to power consumption, the mismatched scheduling solution [7] usually spends higher power consumption to finish user required task which can be executed with lower power consumption. Therefore, how to realize cloud computing system with low power consumption through scheduling resources appropriately have been widely concerned.

Cloud workflow [8, 9] is a new application mode for workflow management system in cloud

computing environment, which can provide optimization solutions for cloud computing system to reduce operation cost and improve the quality of cloud services [10]. The scheduling of cloud workflow which is the same as that of grid workflow [11] is the problem of mapping each task to a suitable resource and of ordering the tasks on each resource to satisfy some performance criterion [12,13]. The scheduling algorithm of workflow can be divided into two categories: scheduling algorithms based on best-effort service and scheduling algorithms based on OoS constraints [14]. Yu et al. [15] proposed economy-based methods to handle large-scale grid workflow scheduling under deadline constraints, budget allocation, and QoS. Dogan and Özgüner [16] developed a matching and scheduling algorithm for both the execution time and the failure probability that can trade off them to get an optimal selection. Moretti et al. [17] suggested all of the pairs to improve usability, performance, and efficiency of a campus grid.

At present, there are a few works addressing cloud workflow scheduling. Juve in literature [18] compared the performance of running some scientific workflows on the NCSA's Abe cluster, against the Amazon EC2. Both use Pegasus [19] as the workflow management system to execute the workflows. Prodan et al. [20] proposed a bi-criteria scheduling algorithm that follows a different approach to the optimization problem of two arbitrary independent criteria, e.g. execution time and cost. "RC2" algorithm [21] for scheduling tasks in hybrid cloud was proposed by Lee and Zomaya to achieve reliable completion. An initial schedule is first calculated based on private cloud (or locally owned resources) to minimize cloud resource usage. In 2011, Bittencourt and Madeira proposed the "HCOC" algorithm [22] to schedule cloud workflows within deadline while minimizing compute cost. In addition to task execution time and compute cost that are used in the techniques described so far, the "PBTS" algorithm proposed by Byun et al. [23] begins to consider other aspects in the cloud.

So far, there are few works solving energy-aware cloud workflow scheduling. Regarding the existing energy consumption models [24, 25, 26], they all consider only two levels of energy consumption in a machine corresponding to its idle and full-load states. These models, however, do not properly reflect the current energy-aware multi-core architectures. Authors in literature [27] proposed an energy-aware heuristic scheduling for data-intensive workflows in virtualized datacenters, which introduces a novel heuristic called Minimal Data-Accessing Energy Path for scheduling data-intensive workflows aiming to reduce the energy consumption of intensive data accessing. Pareto-based multi-objective workflow scheduling algorithm was proposed in literature [28, 29], which captures the real behavior of energy consumption in heterogeneous parallel systems based on empirical models.

We can know from the aforementioned scheduling algorithms of workflow applications that regardless of time optimization algorithms, cost optimization algorithms scheduling or energy-aware scheduling algorithms for workflow applications, all of them have not effectively solved the power consumption optimization problem faced in the cloud computing environment, which likely result in power waste phenomenon as mismatch scheduling of cloud workflows. So, we propose a power consumption optimization algorithm of cloud workflow scheduling based on service level agreement which tries to match each task of workflow application to the reasonable service provided by server in cloud computing system. The key of optimization algorithm proposed in this paper is to find out all feasible scheduling solutions for candidate cloud workflow applications.

2 Cloud workflow model

2.1 DAG model of cloud workflow

The DAG (Directed Acyclic Graph) is a well-known model for describing workflow applications in

different computing environments. So, a cloud workflow application also can be represented by the DAG G=(T,E) (as shown in fig.1), where T is a set of tasks t_i (i=1,2,…,n), and E is a set of edges $e_{i,j}(t_i \neq t_j)$ that describe the dependencies between tasks. In given DAG model for a cloud workflow, if $t_a \in T$ and $e_{i,a} \notin E$ for all $t_i \in T$, then the task t_a is called an entry task of the cloud workflow; if $t_z \in T$ and $e_{z,i} \notin E$ for all $t_i \in T$, then the task t_z is called a exit task of the cloud workflow. In order to better understand the DAG model, we always add two dummy tasks of t_{entry} and t_{exit} to the beginning and end of the cloud workflow, respectively.

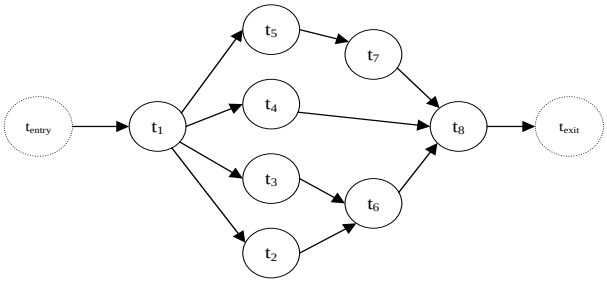

Fig.1 DAG model of a cloud workflow

The service providers offer several services with different QoS for each task of every cloud workflow. We assume that each task t_i of cloud workflow can be executed by k services with different QoS attributes, $S_i = \{s_{i,1}, s_{i,2},\ldots, s_{i,k}\}$. There are many QoS attributes for services in cloud computing system, including execution time, cost, reliability, power energy efficiency, and so on. In this paper, we consider the most important three factors: execution time, cost and power energy efficiency for our scheduling model. $ET(t_i, s_{i,j})$ and $EC(t_i, s_{i,j})$ are defined as the execution time and the execution cost of executing task t_i on service $s_{i,j}$, respectively. The data transfer time of a dependency $e_{i,j}$ only depends on the amount of data to be transferred between corresponding tasks, and it is independent of the services which execute them [30]. Therefore, $TT(e_{i,j})$ is defined as the data transfer time of a dependency $e_{i,j}$, and independent of the selected services for t_i and t_j.

2.2 Critical path for cloud workflow

A schedule of cloud workflow application is defined as an assignment of services to the cloud workflow tasks. If $SS(t_i)$ denotes the selected service for task t_i, then a schedule of cloud workflow w(T , E) can be defined as:

$$\mathrm{Sched}(w) = \{SS(t_i) \mid \forall t_i \in T\ SS(t_i) = s_{i,j} \in S_i\} \quad (1)$$

Definition 1.Service Graph: SG=(S,D), where $S=\{s_i|\ \mathrm{mapping}(s_i,t_i)\}$ is a set of services which include all selected services for each task in cloud workflow application, $D=\{e_{i,j}|(s_i,s_j) \Leftrightarrow (t_i,t_j)\}$ is a set of edges between services (each edge describes a dependency between services in Service Graph). In this paper, a feasible scheduling solution for cloud workflow application is represented as the corresponding service graph. For example, cloud workflow application in figure 1 is scheduled to form mappings between tasks and services as follows: $t_1 \rightarrow s_{1,3}$, $t_2 \rightarrow s_{2,2}$,$t_3 \rightarrow s_{3,4}$, $t_4 \rightarrow s_{4,6}$,$t_5 \rightarrow s_{5,6}$, $t_6 \rightarrow s_{6,8}$, $t_7 \rightarrow s_{7,8}$, $t_8 \rightarrow s_{8,1}$, and its corresponding service graph can be obtained(as shown in fig.2).

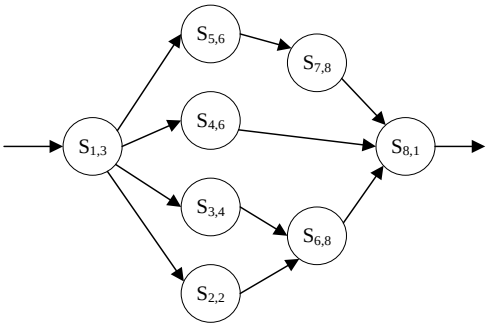

Fig.2 Service graph of a cloud workflow application

Definition 2.Critical path: For any cloud workflow application established workflow model with directed acyclic graph w(T, E), each task is matched to a suitable service during the scheduling. All of mappings between services and tasks can generate a service graph in accordance with w(T,E) of the cloud workflow application, which includes a corresponding critical path denoted as WCP in this paper.

With the WCP of cloud workflow application,

the executive time of cloud workflow application can be defined as:

$$T = \sum_{t_i \in WCP} (ET(SS(t_i)) + TT(SS(t_i))) \quad (2)$$

As for the total cost of cloud workflow application execution, it can be calculated as follows:

$$C = \sum_{i=1}^{n} Cost(SS(t_i)) \quad (3)$$

In order to meet the time constraint of cloud workflow scheduling, we need to define its earliest start time EST, earliest finish time EFT and latest finish time LFT for each task of cloud workflow application. Due to the earliest start time of task t_i at which t_i can start its computation, EST of t_i can be computed as follows:

$$\mathrm{EST}(t_{\mathrm{entry}}) = 0 \quad (4)$$

$$\mathrm{EST}(t_i) = \max_{t_p \in predecessors(t_i)} \{EST(t_p) + ET(t_p, SS(t_p)) + TT(e_{p,i})\} \quad (5)$$

Accordingly, the earliest finish time of each task t_i is the earliest time at which t_i can finish its computation, the EFT of t_i is computed as follows:

$$\mathrm{EFT}(t_i) = \mathrm{EST}(t_i) + \mathrm{ET}(t_i, \mathrm{SS}(t_i)) \quad (6)$$

The latest finish time of task t_i is the latest time at which t_i can finish its computation, such that the whole cloud workflow can finish before the user defined deadline, D. LFT of t_i can be computed as follows:

$$\mathrm{LFT}(t_{\mathrm{exit}}) = \mathrm{D} \quad (7)$$

$$\mathrm{LFT}(t_i) = \min_{t_c \in successors(t_i)} \{LFT(t_c) - ET(t_c, SS(t_c)) - TT(e_{i,c})\} \quad (8)$$

Definition 3.Critical parent: the critical parent of node t_i is the unscheduled parent of t_i that has the latest data arrival time at t_i, that means it is the parent t_p of t_i, for which EST (t_p) + ET (t_p, SS(t_p)) + TT ($e_{p,i}$) is maximal.

The algorithm obtaining critical path for cloud workflow scheduling is as follows:

Input: w(T,E)
Output: WCP
Begin
Step1: add t_{entry}, t_{exit} and their corresponding dependencies to w;
Step2: for (i=1;i<=n;i++) do
Step3: Select the fastest idle service for each task t_i: SS(t_i);
Step4: end for
Step5: for (i=1;i<=n;i++)
Step6: compute EST(t_i) according to Eq. (5);
Step7: end for
Step8: for (i=n;i<=1;i--)
Step9: compute LFT(t_i) according to Eq. (8);
Step10: end for
Step11: mark t_{entry} and t_{exit} as scheduled nodes;
Step11: t=t_{exit}, WCP=null;
Step12: while (there exists an unscheduled parent of t) do
Step15: add CriticalParent(t) to the beginning of WCP;
Step16: t=CriticalParent(t);
Step17: end while
End.

3 Power consumption model

For the power consumption of any cloud workflow which is composed of n tasks during the execution, we can build the following model:

$$PC = \sum_{i=1}^{n} P(t_i, SS(t_i) \times T(t_i, SS(t_i)) \quad (9)$$

Where PC denotes the total power consumption of cloud workflow execution, SS(t_i) represents the selected service for task t_i, P(t_i, SS(t_i)) indicates the power which is needed to execute task t_i on service SS(t_i), T(t_i, SS(t_i)) denotes the time that is spent to execute task t_i on service SS(t_i).

In this paper, we assume that the task set of cloud computing system is W={w_1,w_2,…,w_m}(m>=n), and the arrival rate for each type of cloud task is denoted as λ_i(i=1,2,…,m).

Each service can establish a queuing model of M/M/1 to process a kind of user task requirement, $s_{i,j}$ means that the service selected for task w_i is deployed on the server h_j. So, the arrival rate $\lambda_{i,j}$ of task w_i allocated to h_j can be represented as follows:

$$\lambda_{i,j} = P_{i,j} \times \lambda_i \tag{10}$$

Where $P_{i,j}$ represents the probability for $s_{i,j}$ deployed on server h_j to execute task t_i. If the service rate for service $s_{i,j}$ to reply task t_i is $\mu_{i,j}$, the average response time ART of executing t_i on $s_{i,j}$ can be calculated as follows:

$$ART = \frac{1}{\mu_{i,j} - \lambda_{i,j}} \tag{11}$$

If the time constraint of task w_i is qt_i, then $\mu_{i,j}$ can be expressed as follows in the case of ART=qt_i:

$$\mu_{i,j} = \lambda_{i,j} + \frac{1}{qt_i} = P_{i,j} \times \lambda_i + \frac{1}{qt_i} \tag{12}$$

The service intensity for server h_j to execute all m types of task (w_1, w_2,…, w_m) can be expressed as follows:

$$\rho_j = \sum_{i=1}^{m} \frac{\lambda_{i,j}}{\mu_{i,j}} = \sum_{i=1}^{m} \frac{P_{i,j} \times \lambda_i}{P_{i,j} \times \lambda_i + \frac{1}{qt_i}} \tag{13}$$

The power of any service $s_{i,j}$ at the moment of t denotes as $Power_{i,j}$ can be calculated as follows:

$$Power_{i,j}(t) = P^c + a_{i,j} \rho_j(t)^{b_{i,j}} \tag{14}$$

Where P^c is constant power consumption of service $s_{i,j}$, $a_{i,j}$ and $b_{i,j}$ are power parameters. Generally, different service intensity corresponds to different power parameters. With Eq. (13), the power of server h_j for $s_{i,j}$ can be computed as follows when the workload of service $s_{i,j}$ tends to stability:

$$Power_j = \begin{cases} P^c + a_{i,j} \times \left(\sum_{i=1}^{m} \frac{P_{i,j} \times \lambda_i}{P_{i,j} \times \lambda_i + \frac{1}{qt_i}}\right)^{b_{i,j}}, 0 < \lambda_{i,j} \le \max(\lambda_{i,j}) \\ 0, \lambda_{i,j} = 0 \end{cases} \tag{15}$$

Therefore, we assume that the start time and the end time for service $s_{i,j}$ are represented as $time_1$ and $time_2$,respectively. Then, the power consumption during the execution of service $s_{i,j}$ can be calculated as follows:

$$PC_{i,j} = \int_{time_1}^{time_2} \left(P^c + a_{i,j} \times \left(\sum_{i=1}^{n} \frac{P_{i,j} \times \lambda_i}{P_{i,j} \times \lambda_i + \frac{1}{qt_i}}\right)^{b_{i,j}}\right) dt \tag{16}$$

4 Optimization of power consumption

4.1 Model of power optimization

According to the model of power consumption in this paper, power consumption optimization of cloud workflow scheduling aims to reduce the total power consumption of cloud workflow execution based on the constraints of time and cost in Service Level Agreement, and its optimization model is represented as follows:

$$\begin{cases} \min \sum_{i=1}^{n} P(SS(t_i), t_i) \\ \sum_{t_i \in WCP} Time(SS(t_i), t_i) \le makspan \\ \sum_{i=1}^{n} C(SS(t_i), t_i) \le Cost \end{cases} \tag{17}$$

Where $P(SS(t_i),t_i)$ denotes the power consumption which is produced by executing task t_i on matched service $SS(t_i)$, $Time(SS(t_i),t_i)$ denotes the time is needed to execute t_i on selected service $SS(t_i)$, makspan represents the total time constraint specified in user's Service Level Agreement, $C(SS(t_i),t_i)$ represents the required cost that is necessary to execute task t_i with allocated service $SS(t_i)$, Cost indicates the total cost defined in user's Service Level Agreement.

Definition 4.Feasible Scheduling Solution: if a scheduling solution corresponding to a services graph can finish the execution of cloud workflow w(T,E) successfully while meeting the required time attribute and cost attribute in user's Service Level

Agreement, we call it a feasible scheduling solution. We assume that Sch_k is a feasible scheduling solution for cloud workflow w(T,E) that can satisfy the time attribute and cost attribute in user's Service Level Agreement, Ω(Sch) is the set of all feasible scheduling solution for cloud workflow w(T,E). For any cloud workflow application, we suppose there should be at least one feasible scheduling solution, and let |Ω(Sch)|=L where L represents the number of feasible scheduling solutions. The power consumption which is needed to execute a feasible scheduling solution can be estimated as follows:

$$PC(FSS_k) = \sum_{s_{i,j} \in S_k} PC_{i,j}, FSS_k = (S_k, D_k) \quad (18)$$

4.2 Algorithm of power consumption optimization

In order to solve the problem of power waste caused by the improper scheduling of cloud workflow, the algorithm idea of power consumption optimization for cloud workflow scheduling is as follows: Firstly, algorithm needs to search for all feasible scheduling solutions among the corresponding service graphs of cloud workflow scheduling. For a scheduling solution of cloud workflow application, if its execution time and execution cost obtained by evaluation are all less than the time constraint and cost constraint in SLA, we can mark the solution as a feasible scheduling solution. Then, the power consumption for each feasible scheduling solution can be computed according to Eq.(18). Finally, we select the scheduling solution with the minimum power consumption as the optimal scheduling solution. The detailed algorithm of power consumption optimization for cloud workflow scheduling (PCOA) is as follows:

Input: DAG of cloud workflow, SG

Output: optimal scheduling solution of cloud workflow with the minimum power consumption

Begin

Step1 :K=0;

Step2 :for each SG_i of cloud workflow do

Step3 : Find out the critical path for the SG_i;

Step4 : if ($\sum_{s_i \in WCP} Time(mapping(s_i, t_i)) \le makspan$ and $\sum_{i=1}^{n} C(s_i, t_i) \le Cost$) then

Step5 : add SG_i to FSS_k(feasible schedule solution set);

Step6 : k++;

Step7 : end if

Step8 :end for

Step9 :compute the power consumption of FSS_0 according to Eq.(16);

Step10:min_PC=PC(FSS_0);

Step11:for (j=1; j<k; j++) do

Step12: compute the power consumption of FSS_j according to Eq.(18);

Step13: if (min_PC>PC(FSS_i)) do

Step14: min_PC= PC(FSS_i);

Step15: set FSS_i as the optimal schedule solution of cloud workflow;

Step16: end if

Step17:end for

End.

5 Experimental results and analysis

To evaluate the effectiveness of power consumption optimization strategy for cloud workflow scheduling proposed in this paper, we employed three workflow scheduling methods including Loss and Gain, Deadline-MDP and PCOA to carry out four scientific workflow applications , such as Montage, Epigenomics, MRI and e-protein in simulated cloud computing environment CloudSim, and compared the time, cost and power consumption for them after Loss and Gain, Deadline-MDP and PCOA finished the execution of Montage, Epigenomics, MRI and e-protein. In experiment, we assigned a time constraint (Montage:400s, Epigenomics:400s, MRI:350s, e-protein:350s) and a cost constraint (Montage:35$, Epigenomics:35$, MRI:30$, e-protein:30$) for each cloud workflow application, which are represented as makspan and cost,

respectively. Moreover, we also provided ten services for each type of task, which are deployed on different servers. All of ten services need to spend different time and cost to process the same task. Generally speaking, a faster service costs more power than a slower one. Environment parameters involved in this experiment and their values are shown in table 1.

After completing ten times execution for each scientific workflow application, the average time, cost and power consumed by three different scheduling methods are shown in fig.3, fig.4 and fig.5, respectively. Fig.3 shows that under the constraints of time and cost in SLA, Loss and Gain method performs each workflow application with the least time, and Deadline-MDP spends the most time to finish the execution of every workflow application, while the run time for PCOA is greater than that of Loss and Gain, and less than that of Deadline-MDP. So, we should adopt Loss and Gain method to schedule various workflow applications if the target of scheduling cloud workflow is to minimize the completion time.

Table 1.Parameters setting of the simulated environment

Parameter	Setting	Description
h	25	Number of servers in simulated cloud environment
λ_i	[5,20]	Average arrival rate for task t_i(i=1,2,…)
$\mu_{i,j}$	[4,15]	Serving rate for server h_j to execute task t_i
P_j	[50w,80w]	Idle power of server h_j
$P_{i,j}$	[200w,600w]	Executive power of server h_j processing task t_i
makspan	[50s-400s]	Time constraint in user's SLA
cost	[10-50]	Cost constraint in user's SLA

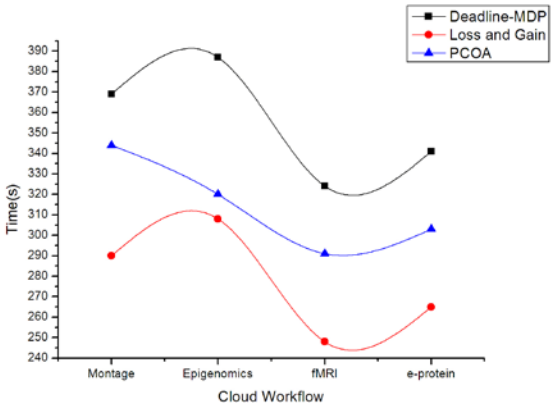

Fig.3 Comparison of average time for performing cloud workflow

Fig.4 indicates that with the constraints of time and cost in SLA, Loss and Gain method needs to spend the highest cost to execute each of workflow applications and Deadline-MDP only spends the lowest cost to perform every workflow application, while the average cost for PCOA to finish the execution of all workflow applications is slightly lower than that of Loss and Gain method. Therefore, we should select Deadline-MDP method if the scheduling aims to minimize the cost of cloud workflow execution.

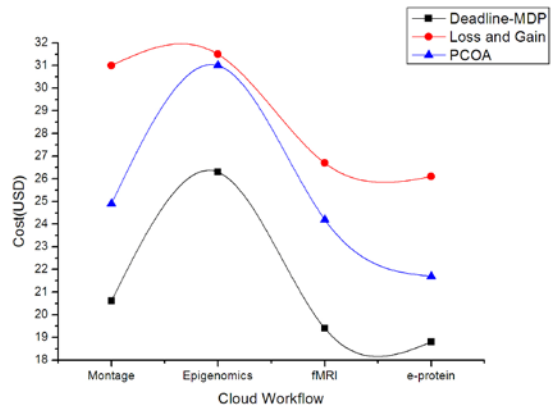

Fig.4 Comparison of average cost for performing cloud workflow

We can know from fig.5 that under the constraints of time and cost in SLA, Deadline-MDP, Loss and Gain spend almost the same power to complete the execution of each workflow application, while the average power consumption produced by PCOA is the optimal among the three scheduling

methods of Loss and Gain, Deadline-MDP and PCOA. The reason why PCOA can reduce the power consumption of cloud workflow application execution is that PCOA can select the optimal scheduling solution for each cloud workflow application through using critical path. The experimental results show that compared with traditional workflow scheduling algorithms based on QoS, the optimization algorithm proposed in this paper not only meets the constraints of time and cost defined in SLA, but also reduces the average power consumption by around 10%. So, we should employ PCOA to dispatch all cloud workflow applications in order to reduce the power consumption of cloud workflow execution caused by improper scheduling algorithm in cloud computing environment.

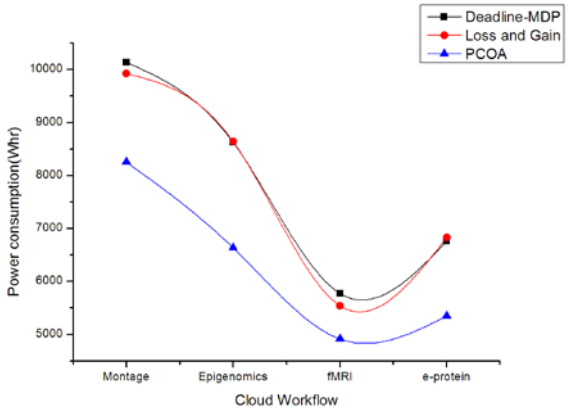

Fig.5 Comparison of average power consumption for performing cloud workflow

6 Conclusions

This paper studied the power consumption optimization of cloud workflow scheduling as energy waste issues in the cloud computing environment have become increasingly prominent. Through analyzing the power computing model for cloud workflow application execution, we have proposed a power consumption algorithm of cloud workflow scheduling under the constraints of time and cost in SLA. Simulated experiments demonstrate that this optimization method is fully effective and feasible. But the power optimization issue isn't implemented in the virtual cloud environment. So, we will further investigate the power consumption optimization of cloud workflow scheduling based on the virtual machines allocation in the future, and carry out experiments in real virtualization cloud platform so as to ensure the correctness and effectiveness of research result.

Acknowledgements

This work was supported by National Natural Science Foundation of China (Grant no. 51277015) and Scientific Research Fund of Hunan Provincial Education Department (Grant no. 13C1003).

References:

[1] Doroshin, A. V., Neri, F, Open research issues on Nonlinear Dynamics, Dynamical Systems and Processes, WSEAS Transactions on Systems, Vol.13, 2014, in press.

[2] Azzouzi, M., Neri, F., An introduction to the special issue on advanced control of energy systems , WSEAS Transactions on Power Systems, Vol.8, No.3 , 2013, p. 103.

[3] Yiming Tan, Guosun Zeng, Wei Wang, Policy of Energy Optimal Management for Cloud Computing Platform with Stochastic Tasks, Journal of software, Vol.23, No.2, 2012, pp.266-278. (In Chinese)

[4] Armbrust M, Fox A, Griffith R, Joseph A D, A view of cloud computing, Communications of the ACM, Vol.53, No.4, 2010, pp.50-58.

[5] Karthikeyan, P., Neri, F, Open research issues on Deregulated Electricity Market: Investigation and Solution Methodologies, WSEAS Transactions on Systems, Vol.13, 2014, in press.

[6] Ciufudean, C., Neri, F, Open research issues on Multi-Models for Complex Technological Systems, WSEAS Transactions on Systems, Vol.13, 2014, in press.

[7] Chuang Lin, Yuan Tian, Min Yao, Green Network and Green Evaluation: Mechanism, Modeling and Evaluation, Chinese Journal of

Computers, Vol.34, No.4, 2011, pp.593-612.
[8] Xuezhi Chai, Jian Cao, Cloud Computing Oriented Workflow Technology, Journal of Chinese Computer Systems, Vol.33, No.1, 2012, pp. 90-95.
[9] Dong Yuan, Yun Yang, Xiao Liu, Gaofeng Zhang, Jinjun Chen, A data dependency based strategy for intermediate data storage in scientific cloud workflow systems, Concurrency and Computation: Practice & Experience, Vol.24, No.9, 2012, pp.956-976.
[10] Panoiu, M., Neri, F, Open research issues on Modeling, Simulation and Optimization in Electrical Systems, WSEAS Transactions on Systems, Vol.13, in press.
[11] Haijun Cao, Hai Jin, Xiaoxin Wu, Song Wu, ServiceFlow: QoS-based hybrid service-oriented grid workflow system, Journal of Supercomputing. Vol.53, No.3, 2010, pp. 371-393.
[12] S. Abrishami,M. Naghibzadeh, Deadline-constrained workflow scheduling in software as a service Cloud, Scientia Iranica, Vol.19, No.3, 2012, pp.680-689.
[13] Neri, F, Open research issues on Advanced Control Methods: Theory and Application, WSEAS Transactions on Systems, Vol.13, 2014, in press.
[14] Jia Yu, Rajkumar Buyya, A Taxonomy of Workflow Management Systems for Grid Computing, Journal of Grid Computing, Vol.3, No.3-4, 2005, pp.171-200.
[15] J. Yu, R. Buyya, Scheduling scientific workflow applications with deadline and budget constraints using genetic algorithms, Scientific Programming Journal, Vol.14, No.1, 2006, pp.217–230.
[16] A. Dogan, F. Özgüner, Biobjective scheduling algorithms for execution time–reliability trade-off in heterogeneous computing systems, The Computer Journal, Vol.48, No.3, 2005, pp. 300–314.
[17] C. Moretti, H. Bui, K. Hollingsworth, B. Rich, All-pairs: an abstraction for data-intensive computing on campus grids, IEEE Transactions on Parallel and Distributed Systems, Vol.21, No.1, 2010, pp. 33–46.
[18] Ewa Deelman, Grids and Clouds: making workflow applications work in heterogeneous distributed environments, International Journal of High Performance Computing Applications, Vol.24, No.3, 2010, pp. 284–298.
[19] Ewa Deelman, Gurmeet Singh, Meihui Su, Pegasus: a framework for mapping complex scientific workflows onto distributed systems, Scientific Programming, Vol.13, No.3, 2005, pp. 219–237 .
[20] Prodan R, Wieczorek M, Bi-criteria scheduling of scientific grid workflows, Automation Science and Engineering, Vol.7, No.2, 2010, pp. 364–376.
[21] Y. C. Lee, A. Y. Zomaya, Rescheduling for reliable job completion with the support of clouds, Future Generation Computer Systems, Vol.26, 2010, pp. 1192-1199.
[22] L. F. Bittencourt, E. R. M. Madeira, HCOC: A Cost Optimization Algorithm for Workflow Scheduling in Hybrid Clouds, Journal of Internet Services and Applications, Vol. 2, 2011, pp. 207-227.
[23] E. Byun, Y. Kee, J. Kim, S. Maeng, Cost optimized provisioning of elastic resources for application workflows, Future Generation Computer Systems, Vol. 27, pp. 1011-1026, 2011.
[24] M. Mezmaz, N. Melab, Y. Kessaci, Y.C. Lee, E.-G. Talbi, A.Y. Zomaya, D. Tuyttens, A parallel bi-objective hybrid meta heuristic for energy-aware scheduling for cloud computing systems, Journal of Parallel and Distributed Computing, Vol.71, 2011, pp. 1497–1508.
[25] P. Lindberg, J. Leingang, D. Lysaker, S.U. Khan, J. Li, Comparison and analysis of eight scheduling heuristics for the optimization of energy consumption and makespan in large-scale distributed systems, Journal of Supercomputing, Vol.59, No.1, 2010, pp. 478–484.
[26] N. Min-Allah, H. Hussain, S.U. Khan, A.Y. Zomaya, Power efficient rate monotonic scheduling for multi-core systems, Journal of Parallel and Distributed Computing, Vol.72, No.1, 2012, pp.48–57.
[27] Peng Xiao, Zhi-Gang Hu, Yan-Ping Zhang, An Energy-Aware Heuristic Scheduling for Data-Intensive Workflows in Virtualized Datacenters, Journal of Computer Science and Technology,

Vol.28, No.6, 2013, pp. 948-961.
[28] Juan J. Durillo, Vlad Nae, Radu Prodan, Multi-objective energy-efficient workflow scheduling using list-based heuristics, Future Generation Computer Systems, Vol.36, 2014, pp.221–236.
[29] Pekař, L., Neri, F, An introduction to the special issue on advanced control methods: Theory and application, WSEAS Transactions on Systems, Vol.12, No.6, 2013, pp. 301-303.
[30] Pekař, L., Neri, F, An introduction to the special issue on time delay systems: Modelling, identification, stability, control and applications, WSEAS Transactions on Systems, Vol.11, No.10, 2012, pp. 539-540.